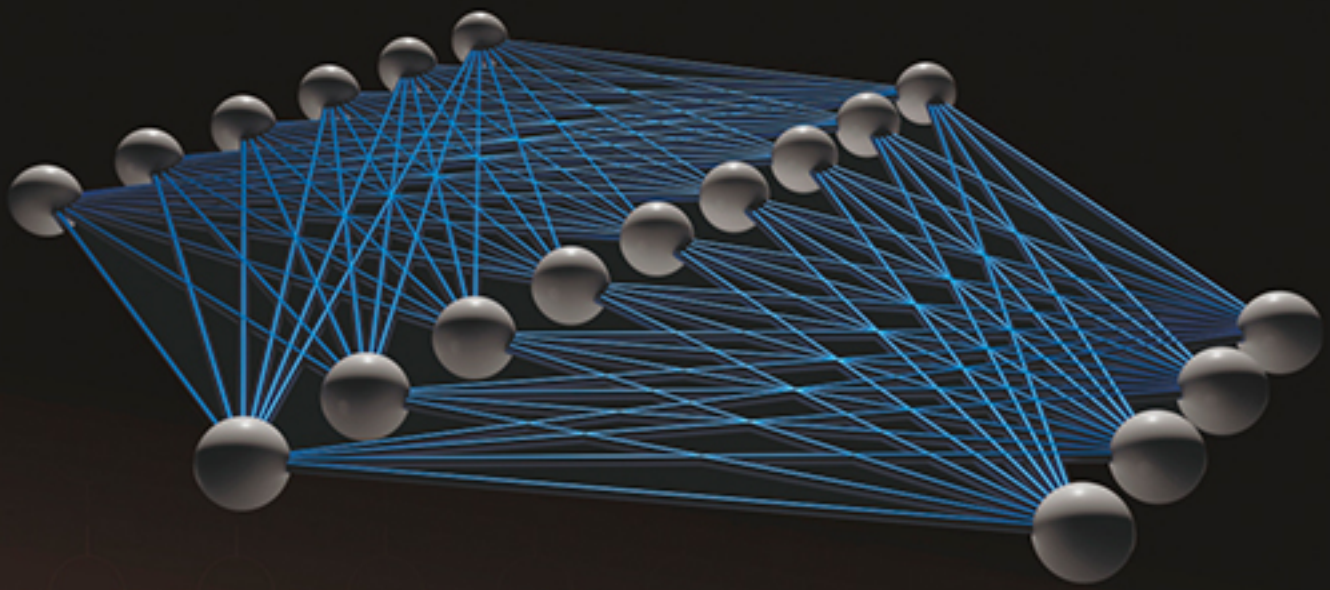
ISSN 1726-5479

SENSORS & TRANSDUCERS

vol. 249
2/21



Artificial Intelligence

Electronic version. Not for distribution.

International Frequency Sensor Association Publishing

IFSA

Sensors & Transducers

International Official Open Access Journal of the International Frequency Sensor Association (IFSA) Devoted to Research and Development of Sensors and Transducers

Volume 249, Issue 2, February 2021

IFSA Publishing: Barcelona • Toronto

Published by International Frequency Sensor Association (IFSA) Publishing. Printed in the USA.

Sensors & Transducers

Volume 249, Issue 2
February 2021

www.sensorsportal.com

e-ISSN 1726-5479
ISSN 2306-8515

Sensors & Transducers journal is an open access, peer review international journal published monthly by International Frequency Sensor Association (IFSA).
Available in both: print and electronic (printable pdf) formats.

Sensors & Transducers Journal

Contents

Volume 249
Issue 2
February 2021

www.sensorsportal.com

ISSN 2306-8515
e-ISSN 1726-5479

Research Articles

Authors are encouraged to submit article in MS Word (doc) and Acrobat (pdf) formats
by e-mail: editor@sensorsportal.com. Please visit journal's webpage with preparation instructions:
http://www.sensorsportal.com/HTML/DIGEST/Submition.htm

International Frequency Sensor Association (IFSA).

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 1-8

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Applying Neural Nets on Sensor Noise to Identify CT Scanner Manufacturers

[1] Markus GERLOFSMA, [1, 2, *] Milan PETKOVIC, [2] Peter van LIESDONK and [1] Vlado MENKOVSKI

[1] Eindhoven University of Technology, Eindhoven, The Netherlands
[2] Royal Philips Research Laboratories, Eindhoven, The Netherlands
[*] E-mail: m.petkovic@tue.nl

Received: 10 October 2020 /Accepted: 10 December 2020 /Published: 28 February 2021

Abstract: The acquisition of Computed Tomography (CT) images has shown to be affected by the scanning device which acquires the image. Specifically, the same subject, scanned by a variety of CT devices, holds different properties depending on the device which acquired the image. This variance consequently has an impact on the medical analysis of the images. Therefore, the ability to determine the acquiring CT device based on the produced CT images may lead to improved diagnoses. In addition, knowledge of the CT device may furthermore provide a means to ensure verification of provenance without the necessity to rely on potentially corrupt or missing DICOM metadata. In this work, we apply a Convolutional Neural Network (CNN) to classify 4 manufacturers of CT scanners, based on the CT images which their devices generate. We apply our experiments on a large, publicly available dataset, and additionally apply previous classification techniques on the same dataset. With an accuracy of up to 93.6 %, our approach significantly outperforms existing work.

Keywords: CT scanners, CT Device Manufacturers, CT Image Classification, Convolutional Neural Networks, Sensor Noise.

1. Introduction

Computed Tomography (CT) has become a vital asset in medical imaging. CT scans are volumetric images which are acquired by capturing a multitude of slices via x-ray beams. These slices are captured at consecutive angles and together form the entirety of a CT volume. Fig. 1 depicts examples of such slices. Once captured, the complete CT images are then predominantly stored as a file in the DICOM imaging format [1]. In addition to the image data itself, this storage format also provides additional metadata about the parameters of the acquisition process of the image and the properties of the capturing CT device. One such device property is the manufacturer of the scanning device.

Such information may be exceptionally valuable as, for example, the estimation of lung density using CT images has shown to vary among CT scanner devices from different device manufacturers, which consequently affects the diagnosis of Emphysema [2], [3]. As such, knowledge of certain DICOM metadata, such as the device manufacturer of the scanner, may improve medical diagnoses themselves.

However, the DICOM metadata which contains this information is predominantly implemented without any security checks and requires only a few mandatory fields to produce a valid file. This leniency causes the metadata to be often incomplete and susceptible to modifications, which may discourage medical professionals to confidently act on the available data. Although the DICOM standard

facilitates additional controls which provide data integrity, these measures are predominantly not implemented to ensure a painless data exchange between parties, such as hospitals.

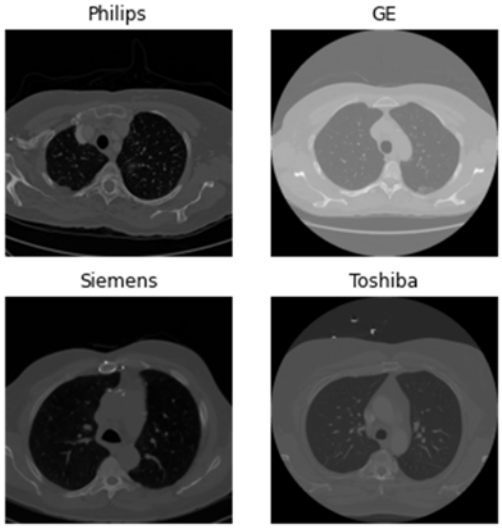


Fig 1. Example of CT slices from the LIDC-IDRI dataset.

To reduce the dependency on DICOM metadata, yet still exploit the benefit of the underlying data, the authors of [4] sought to identify a specific CT scanner device, exclusively based on the image data of a CT scan. The authors extracted the Sensor Pattern Noise (SPN) from the images to obtain distinct device fingerprints for a series of CT scanner devices and used image correlation to perform device identification based on these fingerprints. In addition to precise device identification, these noise patterns may also be leveraged to determine the integrity of the images themselves, which furthermore make them an attractive security control to e.g. detect image tampering [5].

Despite the initial success, the work in [4] exclusively applied an elemental classification approach based on image correlation, which consequently provides an opportunity for potential improvement. Convolutional Neural Networks (CNN) [6] have shown excellent performance for image classification and representation learning in several domains, including applications on medical images, including CT scans [7]. Therefore, they may also prove to be an excellent tool to learn precise device fingerprints and classify CT device manufacturers based on CT images.

As such, different from the related work, we apply a CNN to classify 4 CT manufacturers based on the CT images produced by their respective CT scanners. In our paper we demonstrate the remarkable capability of neural networks to perform CT scanner classification based on CT images as well as sensor noise. We furthermore provide novel insights and show how the extraction of distinct noise patterns has a significantly varying effect on the classification. Finally, we demonstrate the importance of sensible data processing for a task such as CT device classification.

2. Related Work

To attribute images to specific devices, in the seminal work of [8], the authors leveraged the Photo Response Non-Uniformity noise (PRNU) present in the Sensor Pattern Noise (SPN) of pictures taken by photo cameras to attribute images to specific camera models. To extract the noise component of an image, the approach uses a denoising filter based on the work of [9]. This filter is applied on several images from the same, known camera model, and are subsequently averaged to obtain a reference pattern for a specific camera model. This reference pattern then serves as the fingerprint for the device, which enables attribution of future images.

This same denoising and classification procedure has been applied in [4] to identify CT scanners. Therein, the authors consider a total of 3 scanners from 2 manufacturers to perform CT scanner device identification based on the CT images produced by the corresponding scanner. The same authors have furthermore improved this experimental setup by adding an additional correction step during noise extraction [10] and experimented with computing the noise pattern along different dimensions of the original CT image [11].

Aside from specific CT device identification, other works aim to classify a group of devices instead [12], [13]. These works rely on alternative pre-processing steps, supplemented with an SVM [14] to perform the final classification. In [12], an alternative denoising technique is applied to account for the unique image acquisition and reconstruction process of CT images. The authors conducted their experiments on images containing a variety of anatomical objects to classify a total of 17 scanner models produced by 4 manufacturers. Differently, the work in [13] distinguished 3 CT scanner manufacturers by performing a quantitative analysis based on the density distribution of Hounsfield Unit (HU) values within the CT images. The authors fit an SVM on the found distributions and achieve an accuracy of 91.1 %.

One limiting factor we discovered in the related works is that these works are exclusively based on smaller, private datasets, which poses a challenge in making a direct and fair comparison of the achieved results. Furthermore, previous works which applied noise extraction as a preprocessing step, have not performed their classification technique on unprocessed CT images. For this reason, to our knowledge, there also does not yet exist a baseline for the classification of device manufacturers based on unprocessed CT images. This absence of a baseline limits the ability to make confident assertions about the effect of noise extraction on the classification performance.

Different from the related work, this is the first work to perform the classification on both unprocessed CT images and extracted noise patterns, which will indicate the potential benefit of noise extraction when performing device classification. In addition, this work is the first to apply a deep learning method to classify CT scanner manufacturers based on the acquired CT images. Finally, we use a larger, publicly available dataset, containing images from a wider array of devices to provide a better indication for the generalizability of CT scanner classification.

3. Approach

Within our experiments, we first and foremost aim to evaluate the effectiveness of neural nets in identifying CT device manufacturers based on the sensor noise present within a CT image. However, in order to draw sensible conclusions, we also seek to establish a baseline for our experiments. For this reason, we first reproduce and apply the classification technique initially performed in [4], but on a significantly larger dataset than the initial work. We apply this approach on the unprocessed CT slices, as well as on the extracted sensor noise patterns.

Then, for the classification using neural networks, we construct and train two convolutional neural networks (CNN) [6] to classify the slices of CT images. Here, the setup is again twofold. One CNN is trained and tested on unprocessed slices, while the second network is trained to classify the extracted sensor noise of the CT slices.

3.1. Noise Extraction Algorithm

The sensor noise components (Fig. 2) from a CT slice is extracted by applying the approach proposed in [8]. Although this approach has been initially proposed for the identification of photo cameras, the technique has also shown success in identifying CT scanners by the images which they produce [4], [10], [11].

To denoise a slice, a denoising filter *F* is applied. The denoised image is subtracted from the original slice to obtain the desired noise component *w*:

$$\mathrm{w} = \mathrm{s} - F(s \qquad (1)$$

The denoising filter *F* is the same as applied in [8], [9]. Specifically, the work applies a Wiener filter in the wavelet domain, which is applied in two stages as follows. First, the local variance of the image within the wavelet domain is estimated. Then, in the second stage, the estimated variance is applied in a Wiener filter to obtain the denoised image. These two stages are performed using the steps described below:

1) The fourth-level wavelet decomposition is calculated for the slice.

2) For each level, the three high frequency-bands are used for further processing. These are the vertical, horizontal and diagonal sub-bands, respectively.

3) For each wavelet coefficient in the sub-bands, the local variance $\hat{\sigma}^2(i,j)$ is estimated within a $W \times W$ square neighbourhood, for $W \in \{3,5,7,9\}$. The minimum of these 4 variances is chosen as the final estimate for the image variance:

$$\hat{\sigma}_W^2(i,j) = \max\left(0, \frac{1}{W^2} \sum_{(i,j)\in N} h^2(i,j)\, \hat{\sigma}_0^2\right) \qquad (2)$$

$$\hat{\sigma}^2(\mathrm{i},\mathrm{j}) = \min\left(\hat{\sigma}_3^2(i,j),\ \hat{\sigma}_5^2(i,j),\ \hat{\sigma}_7^2(i,j),\ \hat{\sigma}_9^2(i,j)\right), \qquad (3)$$

where $(i,j) \in \mathrm{J}$ denotes the index set for decomposition level J, and $\hat{\sigma}_0^2$ is a manually tuned parameter which has been set to $\hat{\sigma}_0^2 = 5$ to follow the original work [8].

4) After the variance estimation, the Wiener filter is applied to obtain the denoised wavelet coefficients:

$$X_{den}(i,j) = X(i,j) \frac{\hat{\sigma}^2(i,j)}{\hat{\sigma}^2(i,j) + \hat{\sigma}_0^2} \qquad (4)$$

5) The above steps 2−4 are repeated for each wavelet sub-band, on each level of the decomposition.

6) The final, denoised image *F(s)* is then obtained by applying the inverse wavelet transform.

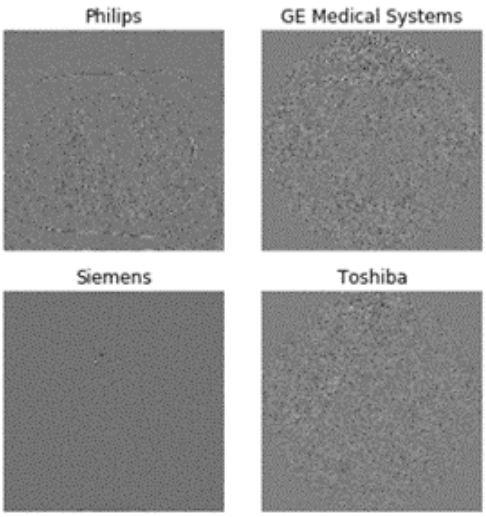


Fig. 2. Example of extracted noise patterns from several CT slices.

3.2. Reference Pattern and Correlation

The baseline classification approach, based on correlation with a reference pattern, is performed by applying the same technique as [4]. This approach is performed in two stages. First, a reference pattern is computed for each class of CT scanners by averaging a random selection of slices from a particular class of scanners. The original process, illustrated in Fig. 3, applies this technique exclusively on the extracted noise patterns. However, since we additionally evaluate this approach on the unprocessed slices, we

skip the noise extraction step in one instance of our experiments. The classification stage, illustrated by Fig. 4, then performs the classification by calculating the correlation of a CT slice, or its extracted noise component *w*, with each of the computed reference patterns *RP*:

$$\mathrm{corr}(\mathrm{s},\mathrm{RP}) = \frac{(s-\bar{s})\left(RP-\overline{RP}\right)}{||s-\bar{s}||\,||RP-\overline{RP}||} \qquad (5)$$

The resulting correlation with a reference pattern indicates the likelihood that the image originates from the same device. Although the original work [4] accomplished this by establishing a certain threshold for the correlation value, we base the decision on the pattern that exhibits the highest correlation.

3.3. Data Split

When performing classification on the 2D slices of a CT image, one may consider two approaches to split the data into their respective training and test sets. The first approach, presented in Fig. 5, accumulates the 2D CT slices from the dataset, and subsequently divides them into separate sets for training and testing. Differently, the second approach, illustrated by Fig. 6, accumulates the slices per volume and subsequently divides the slices of entire volumes into the respective sets.

To our knowledge, the related works [4], [10], [11], [12] have solely considered the first approach. As such, no volume is excluded during the training phase and computation of the reference patterns. However, this first approach may lead to optimistic results as the test set is likely affected by undesired relationships between individual slices which have been generated during the same scanning session of a patient. Specifically, successive, neighboring slices are generally captured within a distance of just 1-3 millimeters. This small difference between the captured slices causes these images to depict nearly identical content. For this reason, the first approach to split the data, shown in Fig. 5, assigns nearly identical slices into both training and test set, and therefore may lead to skewed classification performance.

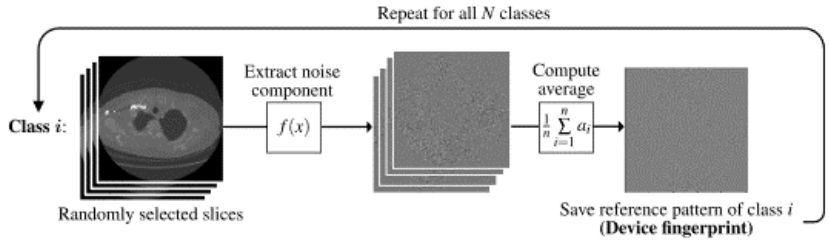


Fig. 3. Creation of reference patterns for a certain amount of devices, given a set of slices from known devices.

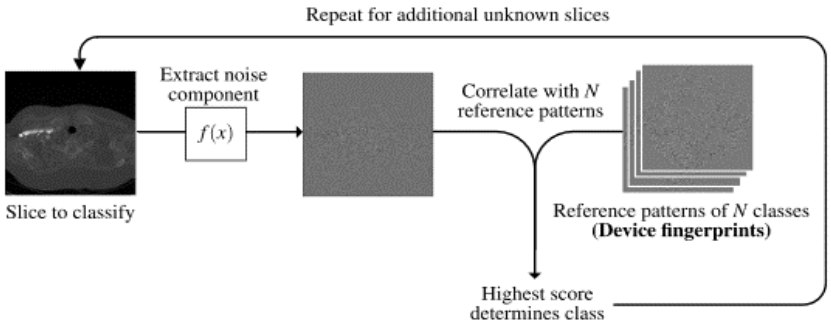


Fig. 4. The process to determine the class of unknown slices, given a precomputed set of reference patterns.

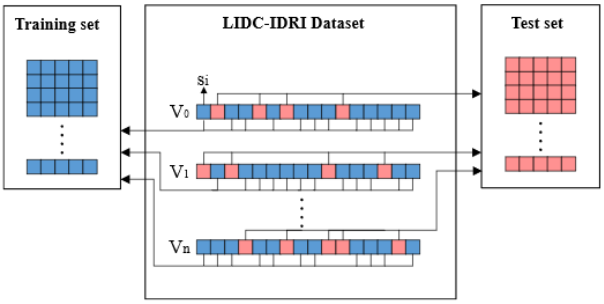


Fig. 5. A split based on individual CT slices. Neighbouring slices may end in either split.

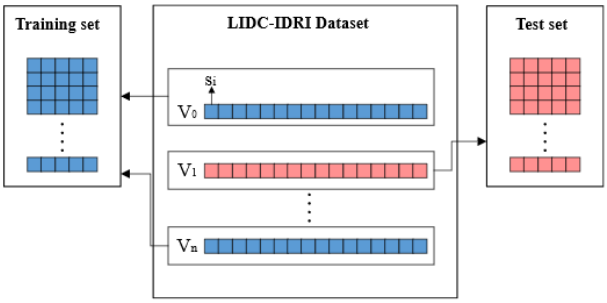


Fig. 6. A split based on CT Volumes. This type of split will keep several potential relationships among slices within a single side of the split.

For this reason, in our experiments, we primarily consider data splits based on the second, volume-based approach illustrated in Fig. 6. However, we also perform the alternative, first approach a single time to highlight the significance of a sensible data split.

4. Dataset

All experiments in our work are performed on samples from the anonymized and publicly accessible collection of CT imagery from the Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) [15]. This dataset contains a total of 1,018 CT volumes, amounting to a total of 244,527 slices. The volumes in this dataset are represented by a total of 4 manufacturers: Philips, GE Healthcare, Siemens, and Toshiba. These manufacturers consequently form the 4 classes for the experiments. Table 1 illustrates the representation of each class within the LIDC-IDRI dataset. The CT scans in this dataset are stored in the 'Digital Imaging and Communications in Medicine' (DICOM) file format[1] which, besides the image data and device manufacturer. In this work, we make use of the metadata within the DICOM format to determine the true label (device manufacturer) of the CT slices.

Table 1. CT volume representation within the LIDC-IDRI dataset.

Manufacturer	No. of Volumes
Philips	75
Siemens	206
GE healthcare	671
Toshiba	70

It is important to note that, besides the manufacturer of a CT scanner, the specific content of the image may be affected by additional circumstances. For example, the images contained within the dataset have been captured by a variety of CT scanner models. The dataset contains images from a total of 15 specific CT scanner models. The specific make of a CT scanner model may also impact the overall performance of a certain class, when for example, a certain model is produced and calibrated differently from other models by the same manufacturer.

However, given the number of images within the dataset, as well as our experimental setup, we hope to eliminate as many potential biases as possible.

5. Experimental Setup

In each experiment, the manufacturer classification is performed on the slices from the LIDC-IDRI [15] dataset. To account for the class imbalance in the LIDC-IDRI dataset, the classes have been balanced by under sampling over-represented classes. As such, the final dataset used in the experiments contained 6,000 slices per class. To extract the noise patterns (Section 3.1) from the CT images, the implementation from [5] has been applied, who have kindly published their source code[2].

To perform the experiments based on the reference pattern and correlation metric (Section 3.2), we continue to primarily follow the work of [4]. Specifically, the reference patterns for the manufacturers are computed using the average of 120 slices per class. However, differently from the initial work, we explicitly split the data as visualized in Fig. 6. As such, slices from volumes which are used to calculate the reference pattern, are entirely separated from the volumes that are classified during the test phase. During classification, each slice from the test set will be compared to each of the reference patterns by calculating the correlation (Equation (5)), where the highest correlation value determines the class.

For the classification using a CNN, a simple, common architecture for image classification has been applied. Specifically, this architecture contained three convolutional layers, with a number of 8, 16 and

[1] https://www.dicomlibrary.com/dicom/

[2] Available at: http://dde.binghamton.edu/download/camerafingerprint/

32 3×3 filters, respectively, with each convolutional layer being followed by a Batch normalization layer [16] and ReLU activation function. The first two convolutional blocks are furthermore followed by a 2×2 max pooling layer, while the last block contains an additional dropout layer to avoid overfitting. Finally, to increase the computational speed, input dimensions have been reduced from the original 512×512 to 299×299. An overview of the architecture is presented in Fig. 7. This architecture performed well on all tasks and has been applied for both networks which classify based on noise patterns and unprocessed slices, respectively. The networks were trained for 10 epochs, on batches of size 128.

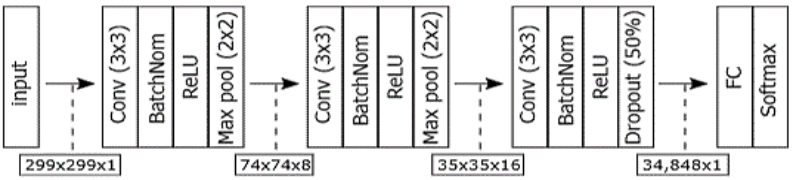


Fig. 7. The CNN architecture used for the classification of CT scanners.

6. Results

6.1. Reference Pattern and Correlation

Table 2 presents the classification accuracies on extracted noise patterns using the reference pattern and correlation metric, initially applied in the work of [4]. The results show that slices from Toshiba and General Electric are classified perfectly, while CT slices from both Siemens and Philips are classified significantly worse. It is also peculiar that Philips and Siemens are most often misclassified amongst each other, which may indicate that these devices share certain software or hardware components or share similarities in the physical acquisition process of the images which consequently affect the device fingerprint.

For the classification on the original, unprocessed slices, the result presented in Table 3 shows that predictions are significantly more distributed compared to the accuracies on noise patterns.

Table 2. Classification accuracy by correlation on extracted noise patterns of CT images.

	Philips	**Siemens**	**GE**	**Toshiba**
Philips	**29.55 %**	33.9 %	22.4 %	14.15 %
Siemens	37.78 %	**35.92 %**	14.58 %	11.72 %
GE	-	-	**100 %**	-
Toshiba	-	-	-	**100 %**
Average	**66.4 % accuracy**			

Table 3. Classification accuracy by correlation on unprocessed CT images.

	Philips	**Siemens**	**GE**	**Toshiba**
Philips	**77.85 %**	22.15 %	-	-
Siemens	29.66 %	**53.91 %**	13.43 %	**3.00 %**
GE	-	-	**76.5 %**	**23.5 %**
Toshiba	-	-	40.97 %	**59.03 %**
Average	**66.8 % accuracy**			

However, the overall accuracy remains similar to the previous result, with a slight increase to 66.8 %, up from the previous 66.4 %. Moreover, predictions for General Electric and Toshiba devices have become split amongst each other and are classified significantly worse. However, Philips and Siemens devices are predicted more accurately on the original images. Interestingly, prediction mistakes for a certain class often fall into just one single, other class.

Nevertheless, the relatively low accuracy on either type of input image indicates that classification based on image correlation with a reference pattern may be insufficient to make accurate predictions on extensive datasets.

6.2. Classification with Neural Networks

As it can be seen in Table 5, the CNN reaches significantly better results and achieves up to 93.6 % classification accuracy on the noise patterns of CT slices. This is a significant improvement over the correlation-based experiments which achieved a maximum accuracy of 66.8 %. In addition, the neural network also outperforms the work of [13] which performed classification on just 3 manufacturers. Moreover, the deep learning classifier reached 1.1 % higher accuracy on the classification of noise patterns as opposed to the classification on original, unprocessed CT slices, presented in Table 4. This result suggests that the extraction of device fingerprints does not interfere with the capability of the neural network to learn its own features. Interestingly, the misclassifications shown in Table 4 also continue to support the apparent similarity between Siemens and Philips devices already highlighted in Section 6.1.

6.3. Effect of Data split

Finally, the CNN has also been evaluated on the naive train/test split in which data is solely separated based on individual slices (Fig. 5). Table 6 shows that this approach indeed significantly favors the classification. The classifier reaches an accuracy of

99.8 % which even outperforms the much smaller setting presented in [4]. This result consequently confirms the hypothesis that the classification of CT scanner manufacturers still requires careful consideration when the data is split into separate sets.

Table 4. Classification accuracy on unprocessed CT images with a neural network.

	Philips	**Siemens**	**GE**	**Toshiba**
Philips	**72.63 %**	27.37 %	-	-
Siemens	1.27 %	**98.73 %**	-	-
GE	-	-	**100 %**	-
Toshiba	-	-	-	**100 %**
Average	**92.5 % accuracy**			

Table 5. Classification accuracy on extracted noise patterns of CT images with a neural network.

	Philips	**Siemens**	**GE**	**Toshiba**
Philips	**98.55 %**	1.45 %	-	-
Siemens	8.40 %	**77.82 %**	-	**13.78 %**
GE	-	-	**100 %**	-
Toshiba	-	-	-	**100 %**
Average	**93.6 % accuracy**			

Table 6. Classification of CT images using a slice-based data split.

	Philips	**Siemens**	**GE**	**Toshiba**
Philips	**99.57 %**	0.43 %	-	-
Siemens	-	**100 %**	-	-
GE	-	0.14 %	**99.64 %**	0.21 %
Toshiba	-	-	-	**100 %**
Average	**99.8 % accuracy**			

7. Conclusions

In this paper, we presented several experiments aimed to classify CT scanner manufacturers based on individual, 2D CT image slices. We evaluated an existing approach, based on the work of [4], on a larger dataset and under stricter conditions. Next, in our approach, we evaluated a CNN on the same task. For both approaches, we performed an additional pre-processing step which extracts the noise component from a CT slice to obtain a representation of the underlying device fingerprint. We compared this pre-processing step to the classification based on the original, unprocessed images. Finally, we showed that this classification task requires careful consideration when splitting the data in respective train and test set.

The results have shown that the classification approach based on reference patterns is not suitable for larger datasets, which naturally carry more variation. In addition, the extraction of the SPN had no significant impact on the performance of this approach. Conversely, the CNN showed outstanding accuracy, and with an accuracy of up to 93.6 %, even out-performed existing work. The classification on the SPN-fingerprint performed only slightly better than the unprocessed images. However, the gained performance is only marginal and may become less significant in fully optimized networks.

8. Discussion and Future Work

The results of the experiments have shown that CNNs are an effective method to classify scanner devices based on CT imagery. Nevertheless, this work primarily serves as a first step towards scanner classification based on neural nets. As such, the methods used in this work may still be optimized. The neural networks applied in the experiments are based on existing architectures and setups, with slight modifications to suit the dimensions and the amount of data. As such, there is still ample opportunity to improve the existing classifiers based on additional domain expertise. In addition to the network optimization, the extracted noise pattern may also be substantially optimized. The current noise pattern to establish the fingerprint was originally intended for ordinary photo cameras yet has also shown applicability for CT scanners. However, in [12], improvements have already been made by applying more sophisticated device fingerprint based on the reconstruction algorithm of CT scanners. In addition, denoising of CT images is an extensively explored topic, and existing methods in this field may also benefit the extraction of more accurate device fingerprints.

Acknowledgements

The work presented in this paper is an extension of the publication featured in the conference proceedings of the international Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI' 2020) [17]. Furthermore, this work is funded by the EC Horizon 2020 research and innovation programme under grant agreement No 780495 (BigMedilytics).

References

[1]. W. D. Bidgood Jr, S. C. Horii, F. W. Prior, D. E. Van Syckle, Understanding and using DICOM, the data interchange standard for biomedical imaging, *Journal of the American Medical Informatics Association*, Vol. 4, Issue 3, 1997, pp. 199-212.

[2]. R. Yuan, J. R. Mayo, J. C. Hogg, P. D. Par´e, A. M. McWilliams, S. Lam, H. O. Coxson, The effects of radiation dose and CT manufacturer on measurements of lung densitometry, *Chest*, Vol. 132, Issue 2, 2007, pp. 617–623.

[3]. B. C. Stoel, M. E. Bakker, J. Stolk, A. Dirksen, R. A. Stockley, E. Piit-ulainen, E. W. Russi, J. H. Reiber, Comparison of the sensitivities of 5 different computed tomography scanners for the assessment of the progression of pulmonary emphysema: a phantom study, *Investigative Radiology*, Vol. 39, Issue 1, 2004, pp. 1–7.

[4]. A. Kharboutly, W. Puech, G. Subsol, D. Hoa, CT-scanner identification based on sensor noise analysis, in *Proceedings of the 5th European Workshop on Visual Information Processing (EUVIP)*, 2014, pp. 1–5.
[5]. M. Chen, J. Fridrich, M. Goljan, J. Lukâs, Determining image origin and integrity using sensor noise, *IEEE Transactions on Information Forensics and Security*, Vol. 3, Issue 1, 2008, pp. 74–90.
[6]. Y. LeCun, Y. Bengio, *et al*., Convolutional networks for images, speech, and time series, *The Handbook of Brain Theory and Neural Networks*, Vol. 3361, Issue 10, 1995, p. 1995.
[7]. J. Kuruvilla, K. Gunavathi, Lung cancer classification using neural networks for CT images, *Computer Methods and Programs in Biomedicine*, Vol. 113, Issue 1, 2014, pp. 202-209.
[8]. J. Lukâs, J. Fridrich, M. Goljan, Digital camera identification from sensor pattern noise, *IEEE Transactions on Information Forensics and Security*, Vol. 1, Issue 2, 2006, pp. 205–214.
[9]. M. K. Mihcak, I. Kozintsev, K. Ramchandran, Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP'99)* (Cat. No. 99CH36258), Vol. 6, 1999, pp. 3253–3256.
[10]. A. Kharboutly, W. Puech, G. Subsol, D. Hoa, Improving sensor noise analysis for ct-scanner identification, in *Proceedings of the IEEE 23rd European Signal Processing Conference (EUSIPCO)*, 2015, pp. 2411–2415.
[11]. A. Kharboutly, W. Puech, G. Subsol, D. Hoa, Advanced sensor noise analysis for CT-scanner identification from its 3D images, in *Proceedings of the IEEE International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2015, pp. 325–330.
[12]. Y. Duan, D. Bouslimi, G. Yang, H. Shu, G. Coatrieux, Computed tomography image origin identification based on original sensor pattern noise and 3D image reconstruction algorithm footprints, *IEEE Journal of Biomedical and Health Informatics*, Vol. 21, Issue 4, 2017, pp. 1039–1048.
[13]. S.-B. Lee, E.-J. Jeong, Y. Son, D.-J. Kim, Classification of computed tomography scanner manufacturer using support vector machine, in *Proceedings of the IEEE 5th International Winter Conference on Brain-Computer Interface (BCI)*, 2017, pp. 85–87.
[14]. C. J. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, Vol. 2, Issue 2, 1998, pp. 121–167.
[15]. S. G. Armato III, G. McLennan, L. Bidaut, M. F. McNitt-Gray, C. R. Meyer, A. P. Reeves, L. P. Clarke, *et al*., Data from LIDC-IDRI. The cancer imaging archive, 2015. [Online]. Available: http://doi.org/10.7937/K9/TCIA.2015.LO9QL9SX
[16]. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, *arXiv preprint arXiv:1502.03167*v3, 2015.
[17]. M. Gerlofsma, M. Petkovic, P. Van Liesdonk, V. Menkovski, CT Scanner Classification with Neural Nets and Sensor Noise, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI'2020)*, Berlin, Germany, 18-20 November 2020, pp. 154-160.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 9-16

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Unsupervised Embedded Gesture Recognition Based on Multi-objective NAS and Capacitive Sensing

[1, 2, *] Juan BORREGO-CARAZO, [1] David CASTELLS-RUFAS, [2] Ernesto BIEMPICA and [1] Jordi CARRABINA

[1] Universitat Autònoma de Barcelona, C. de les Sitges s/n, Bellaterra, 08193, Spain
[2] R+D, Kostal Elétrica S.A., Notari Jesús Led 10, 08181, Senmenat, Spain
[1] Tel.: 93 581 3358
E-mail: juan.borrego@uab.cat

Received: 5 October 2020 /Accepted: 26 November 2020 /Published: 28 February 2021

Abstract: Gesture recognition has become pervasive in many interactive environments. Recognition based on Neural Networks often reaches higher recognition rates than competing methods at a cost of a higher computational complexity that becomes very challenging in low resource computing platforms such as microcontrollers. New optimization methodologies, such as quantization and Neural Architecture Search are steps forward for the development of embeddable networks. In addition, as neural networks are commonly used in a supervised fashion, labeling tends to include bias in the model. Unsupervised methods allow for performing tasks as classification without depending on labeling. In this work, we present an embedded and unsupervised gesture recognition system, composed of a neural network autoencoder and K-Means clustering algorithm and optimized through a state-of-the-art multi-objective NAS. The present method allows for a method to develop, deploy and perform unsupervised classification in low resource embedded devices.

Keywords: Unsupervised learning, Neural networks, Neural architecture search, Capacitive sensing, Embedded electronics.

1. Introduction

Hand gestures are an efficient way of communicating simple concepts. During the last two decades, its usage in Human-Machine Interaction (HMI) has become pervasive thanks to the proliferation of low-cost depth sensors based on structured light, time of flight, and active stereo matching [1].

Depth sensors based on the analysis of light parameters can achieve a high-resolution accuracy, but since they are based on a single point of view that must be distant to the object to recognize, they are mostly used for mid-air gesture recognition [2]. Moreover, they are sensible to illumination conditions. On the other hand, capacitive sensing stands out for its low power, highly sensitive but reliable solution for gesture recognition where close contact is required [3].

Neural Networks (NNs) have achieved outstanding performance in tasks such as image classification or speech translation. In the supervised case, however, their usage requires labeling, which can induce, among other factors, bias in the model and undermine its real-world use [4]. To partially solve this issue, unsupervised methods focus only on the information proportioned by the data to perform the task and avoid annotation and labeling, both error-prone activities which could induce noise and bias. Nevertheless, the bias or noise are not completely removed since the data itself could still be biased and noisy. In the end,

unsupervised methods do not only liberate from the task and effects of labeling but also allow for more freedom in the learning of patterns, by not being constrained to a specific purpose task. Hence, they enable the possibility of pattern discovery and identification outside the constraints of the specific task purpose.

As an added problem, neural networks have required a certain level of computing resources preventing their deployment in low resource platforms like microcontrollers. Lately, this problem has been addressed through optimization techniques like quantization and pruning. However, with the use of such techniques, the network still has to be built manually. In the case of low resource platforms, this fact imposes severe difficulties to develop NNs which are well-performing and compliant with the different memory and latency requirements of the deployment platform. To address this problem, Neural Architecture Search (NAS) [5] was developed, allowing for the automatic building of neural networks. Later, this method was extended to a multi-objective setting [6], to account for the requirements of low resource platforms and deliver functional but also minimal neural networks.

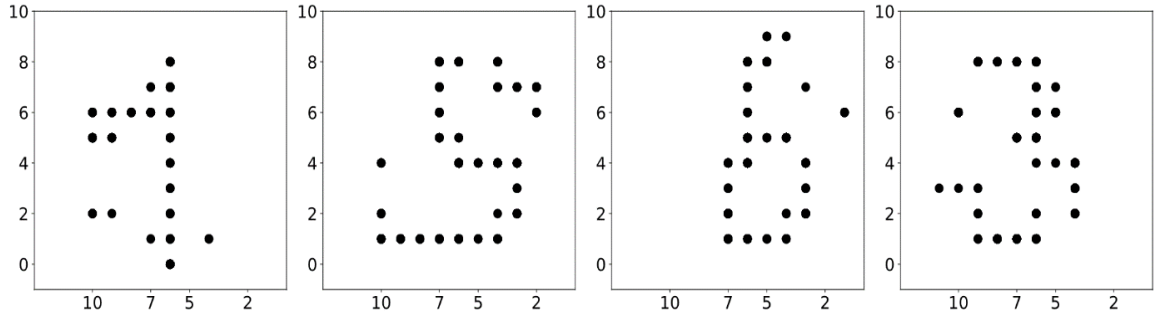


Fig. 1. Examples of numbers drawn at the sensitive surface. From left to right: a one, a five, a six and a three. The black points correspond to the electrode pairs through which the finger has passed. In an image configuration they correspond to 1 s while the background is set to 0.

In this work, we present an auto-encoder neural network that, in junction with K-Means (KM) and optimized through NAS, can perform unsupervised classification in a low resource microcontroller for recognizing gestures. To check the validity of such unsupervised learning and classification, we check our model results in a supervised evaluation setting, which proves the validity of the approach and the possibility of not needing labeled data when using it. Although there have been similar works with other models [7] or with other topics [8], to the best of our knowledge, this is the first work that presents a fully embedded and unsupervised solution for gesture recognition using neural networks and capacitive sensing.

2. Previous Work

Capacitive sensing has been used for multiple application fields including music [9], security [10], or entertainment. It is characterized by offering a low power, only electronic, and cheap alternative to other sensing methods. In the case of gesture recognition, capacitive sensing has also been used extensively; although mostly in touchable surfaces [11], there have been applications based on wearables [12] and on 3D and extended ranges [13]. Different applications require different sensitivities and ranges, capacitive sensing allows for a variety of design parametrizations that change these features. In the present case, we refer to mutual capacitance sensing according to the classification from [3], which allows for a short-range interaction.

Several approaches can be used to classify the sensed signals into recognized gestures. Classic machine-learning methodologies were based on a two-step procedure: first, extracting the relevant features from the input signals; and, second, the classification of such features into the corresponding gestures [14]. Humans associate gestures with meanings. Gestures can be classified as being either static or dynamic. In a static gesture, a certain body part pose that is maintained for a period of time is associated with a meaning (e.g. thumbs-up is associated with OK). In a dynamic gesture, the body part movement is essential to the meaning (e.g. finger-wag is associated with NO). Common algorithms to classify static gestures are support vector machines (SVM) [15], k-nearest neighbors, and random forests, among others. Regarding dynamic gestures, Hidden Markov Models (HMM) [16] or Dynamic Time Warping models are commonly used due to the intrinsic temporal component. In order to feed the data in an appropriate form to the latter algorithms, methods such as Fisher Vectors, PCA, or other feature representations were used.

Nevertheless, with the advent and success, albeit with a higher computational cost, of deep learning this pipeline has changed. In the deep learning setting, the feature extraction step has been deleted in most cases and all the process has been substituted by a neural network. In the case of static classification, methods

based on CNN [17] stand out, differentiating between them with regards to the information they use: camera-based ([18], [19]), or signals ([20], [21]). In the latter case, 1D convolutions stand out as a novel application. In the case of dynamic classification, recurrent models are commonly used ([22], [23]), allowing for a continuous and online gesture classification. Lately, however, and in the case of videos, 3D convolutions have also been used both for continuous and static classification [24].

Deep learning methods, however, usually entail high computational costs that unable their deployment into resource-constrained platforms. To alleviate this problem several methods such as quantization [25], pruning [26], distillation, or NAS [27] have been developed on the software side, while hardware accelerators and deployment frameworks [28] have also implemented in order to speed up inference and allow ease of deployment.

Since then, applications for deploying neural in resource-constrained environments, and more specifically, microcontrollers have flourished. From keyword spotting [29], image classification, and object detection, new applications are being ported to low power and edge devices. Gesture recognition has not been an exception, with new applications and methods appearing continuously; for example, using radar signals [30] or with wearables [31]. In the case of capacitive sensing, most of the applications that use embedded deep learning are oriented to touch detection, prediction, and related tasks in smartphone surfaces [32], while works developed in the microcontroller environment are less frequent.

In most cases, the development of such embedded gesture recognition applications with neural networks requires the annotation of data [33]. There are few cases, where unsupervised methods have been used for gesture recognition with neural networks have been used. In [7] the authors employ several unsupervised methods, such as K-means, Self Organizing Maps or Hierarchical Clustering for classifying gestures sensed through a gyroscope and a geomagnetic sensor. However, they do not embed the models in the microcontroller. In [8], the authors use an approach similar to our proposal based on an autoencoder plus K-Means to classify images. However, they do not perform any kind of optimization in order to allow the model to be run in resource-constrained devices, reason why our methods stand out.

Precisely and, to the best of our knowledge, this is the first work using unsupervised gesture recognition in a microcontroller using neural networks and capacitive sensing.

3. Sensing Method and Data

3.1. Capacitive Sensing Platform

As depicted in Fig. 2, the sensing platform consists of a surface touch module with three elements: the plastic cover, the capacitive foil, and the embedded board. The touch interface is the upper part of the plastic cover, which is immediately after the capacitive foil and glued to it. The embedded board is in the lower part and connected to the capacitive foil. The sensing component, which is the capacitive foil, consists of a central sensitive surface and 24 electrodes which run through it: 9 horizontally and 13 vertically, plus a global shield electrode.

The sensing mechanism is based on the transference of charge between two capacitors: integration and measurement condensers. More details of the sensing methodology can be found in [23].

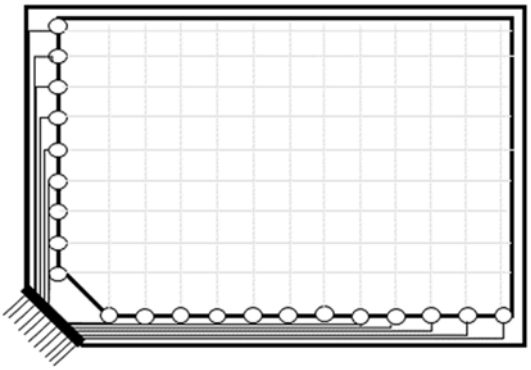


Fig. 2. Illustration of the capacitive foil. Each of the dots represents an electrode, emitting each one a distinctive measurement of its row/column in the surface. The foil is pasted underneath the plastic cover and connected to the microcontroller through the lower-left pins.

The final result of a measurement is a set of voltages sensed at the measurement capacitor, one for each electrode. Each voltage is then converted through a 10-bit ADC to obtain a final raw value for each electrode at a predefined sampling rate (in the present case, 500 Hz).

3.2. Data Collection and Preparation

The user enters the gestures by touching the sensing surface and, without withdrawing the finger, draws the gesture on it. We detect the touch by the method described in [34], and then we begin to collect the raw values of the electrodes as described in the previous section. By examining which electrodes, horizontally and vertically, have the highest value we can determine in which spatial coordinates the finger is placed. When the user withdraws the finger, we stop collecting data. It has to be noted, that the gestures described do not contain segmentation and thus we do not assess that problem. That is, we consider a gesture all the signals from a touch to a withdrawal, and force users to perform the whole gesture in such a manner.

By integrating all the coordinates through which the finger has passed we construct a greyscale image of 9×13 pixels of the figure drawn. That is, we mark the points through which the user has passed with a 1,

while the rest of the untouched points remain as background with a value of 0. Hence, the orientation of the strokes is deleted and only the final form of the gesture drawn is recorded. That is we produce a static version of the gesture. To ease the deployment in a microcontroller and due to deployment framework limitations, we end up squaring the image to 13×13 by padding it. An example of such a method can be seen in Fig. 1, where different numbers drawn on the sensitive surface can be visualized as the points through which the finger has passed.

To collect the dataset, numbers from 1 to 9 were drawn a total of 100 times per number on the sensing surface. The dataset is divided in a stratified manner in training, validation, and test, with 720, 90 and 90 samples respectively.

4. Models and NAS Framework

4.1. Models

We consider two auto-encoder (AE) model types, that is, networks that are trained to replicate the input. The first a Convolutional Auto-encoder (CAE), and the second a Dense Auto-encoder (DAE), made only of fully connected layers.

In general, the architecture consists of two differentiated parts: the first, the encoder, is in charge of extracting the information and compress it in a central vector, and the second, reconstructing the input. In the CAE case, compression is performed by a succession of convolutional or pooling layers followed by ReLU activations, while the input reconstruction is performed by deconvolution layers and final upsampling. In the DAE case, all the layers are fully connected followed by ReLU activations, except for the final layer which reconstructs the input by reordering the output of the network. In both cases, there is a central fully connected layer that produces the latent feature vector. An important note is that to reconstruct the input, as it is made of 0s and 1s, we have used a Binary Cross-entropy Loss for training the AEs.

To provide unsupervised classification, a KM algorithm is in charge of grouping the compressed feature vectors into clusters. The choice of KM is central since it needs the number of clusters to be specified. In our case, this is precisely the number of classes.

4.2. Training and Model Optimization

To train the whole unsupervised method we divide the training into three separate steps. First, we train the AE to replicate the input. The loss used is an L2 norm based reconstruction loss. The networks are trained for a maximum of 50 epochs and the model with the best validation loss is used as the final model; the number of epochs is chosen based upon observation of previous training.

Second, once it has stopped training, we forward the entire training dataset through it and collect the latent feature vector for each input. The latent vector corresponds to the central chosen vector, which can be seen in Fig. 3. In the CAE case, it has 12 elements, while in the DAE it has 117. Hence, the feature extraction through the autoencoder acts as a form of data compression algorithm with an effective compression ratio close to 10.

Third, we train the KM algorithm with all the collected feature vectors. That is, we have a feature vector per image and each of those vectors correspond to the central vector obtained by forwarding the corresponding image through the trained autoencoder. For example, in the CAE case, we have 720 vectors of size 12. As we are classifying 9 different numbers (from digit 1 to 9, excluding 0), we establish 9 clusters for the K-Means algorithms. K-Means algorithm is initialized following the k-means ++ strategy [35] of similarly distanced cluster centers and the distance used is the L-2 norm[1]. Finally, to obtain test or validation results, we forward the test or validation sets through the AE, obtain the feature vectors, and predict to which cluster they belong.

Regarding the performance measure, we have selected a clustering supervised measure, V-Measure [36], for checking the actual classification capabilities of the framework. Choosing an actual unsupervised clustering metric, such as Silhouette Score would not provide direct insight into the accuracy of the predictions. The training, however, remains fully unsupervised.

As detailed as the first step, we have to build and train the network. However, building and tuning neural networks to be well-performing but also to comply with hardware requirements is a difficult task. Hence, we employ an extended implementation of a NAS framework [6] oriented towards optimizing accuracy, model size, maximum feature map (working memory), and latency. An important note is that, as the search space is composed of conditional variables, we use an implementation of the Arc Kernel [37], which is especially suited for conditional spaces. By decomposing the space using and using a cylindrical embedding, it is able to assess the conditionality among variables. The embedding , $g_i: \mathcal{X}_i \rightarrow \Re^2$, where $\mathcal{X}_i$ is the original dimension *i* space, can be used with any distance and covariance method:

$$g_i(\boldsymbol{x}) = \begin{cases} [0,0]^T & if\ \delta_i(\boldsymbol{x}) = False \\ w_i\left[\sin\left(\pi\rho_i\frac{x_i}{u_i - l_i}\right), \cos\left(\pi\rho_i\frac{x_i}{u_i - l_i}\right)\right] & otherwise, \end{cases}$$

[1] For more details, the interested reader can check the implementation at https://scikit-learn.org/stable/modules/clustering.html#k-means

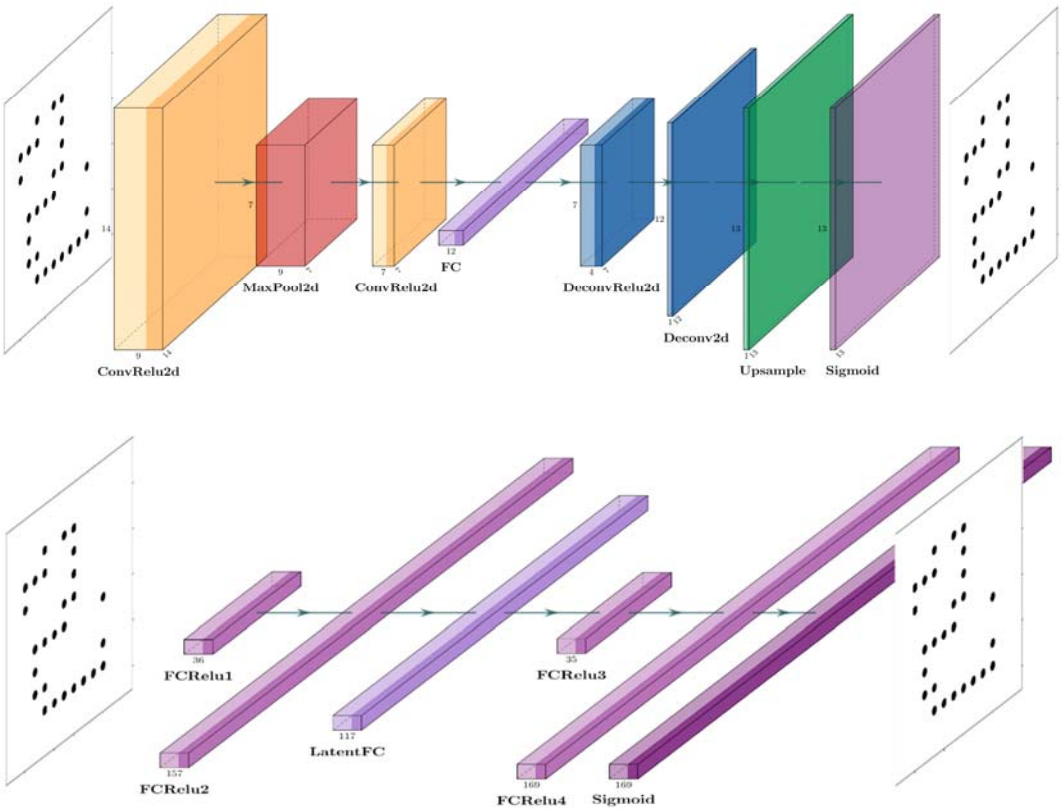


Fig. 3. Best architectures resulting from the optimization procedures for both the CNN and FC autoencoders. The elements in the image represent the feature maps. In the convolution case, the size of the feature map is indicated by the vertical and depth numbers, while the horizontal number indicates the number of feature maps. In the FC case, the number indicates the number of neurons in each step and the images are flattened at the beginning and reshaped at the end.

where $w_i \in \Re^+$ and $\rho_i \in [0,1]$ are the radius and angle factor of the cylindrical space, $u_i - l_i$ establishes the length scale of the dimension i, and δ_i is a delta function establishing the existence of coupling among dimensions x.

The framework works, in each architecture case, CA or DE, with a configuration space composed of both training hyperparameters and architectural parameters, including post-training linear static quantization and L1-Norm based pruning. Once the search space has been defined, the framework sequentially searches for solutions that minimize the following objectives

$$Error\ (\Omega_i) = 1 - Accuracy_{Validation}(\Omega_i)$$
$$ModelSize(\Omega_i) = \sum_l \|w_l\|_0$$
$$WorkingMemory(\Omega_i) = \max_l(\|x_l\|_0 + \|w_l\|_0 + \|y_l\|_0$$
$$Latency(\Omega_i) = \frac{InferenceFLOPS}{Platform\ Frequency},$$

where Ω_i is the specific configuration of the search space, l is the specific layer of the network, w_l the weights of layer l, y_l the output of that layer and x_l the input. In our present case, the accuracy corresponds to the correct classification of each pixel of the output with regards to the input.

Once we have the best model found and trained by our NAS framework, we can proceed with the next steps of training the KM model and obtaining test results.

5. Results

The models established, CAE and DAE, are both developed in PyTorch and the NAS procedure is developed under Ax [38] and BoTorch [39].

First, we illustrate the optimization procedure for the CAE with SpArseMod in Fig. 4. As seen, while the optimization procedure advances there is a decrease in the model size. Also, one can observe the trade-off between the model size and the error: while we continue to obtain smaller architectures the error worsens, obtaining in some cases a compromise.

The final result is an election of the best points of the search: the Pareto frontier. With the best-chosen model, we train and test the K-Means unsupervised clustering and classification. We present the optimization results for the AEs, which correspond to the results for model size, maximum feature map, and latency, and detailed in Table 1, as well as the test results for the K-Means unsupervised classification. As seen, the CAE outperforms the DAE in almost all metrics, except for latency where DAE is a little bit faster. The results for the model size and maximum feature map sizes are low enough to be able to embed the model in a low resource microcontroller.

After the optimization result and after training the KM model with the feature maps obtained, we obtain the test results for the unsupervised classification. In this case, the CAE model performs much better than the DAE, achieving a good result regarding the V-Measure. This indicates the feasibility of this method to provide unsupervised classification.

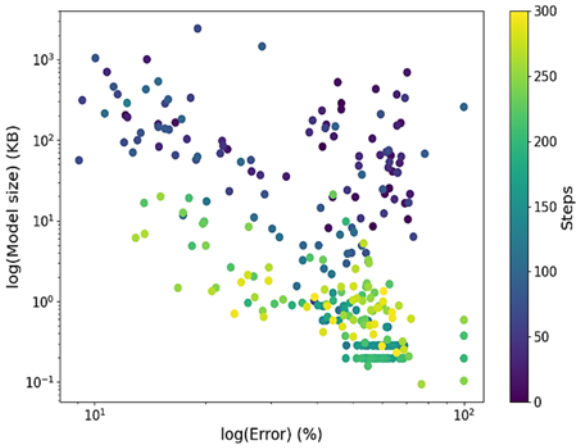


Fig. 4. Optimization procedure for CAE. The y axis represents the Model size and the x-axis the error (per pixel classification). Both axes are in logarithmic scale. The color illustrates the evolution procedure as each of the steps of the Bayesian optimization procedure.

Table 1. Results for the optimization procedure with SpArSeMod. MF corresponds to maximum feature map size, MS to model size, L to latency, and V-M to the V-Measure.

Model	MF (kB)	MS (kB)	L (ms)	V-M (%)
CNN	2.69	6.13	5.53	87.18
FC	18.32	39.67	4.08	58.63

As the final step of our development, we further embed the best model with CMSIS-NN in an NXP-S32K142 microcontroller to perform inference. To perform inference and obtain test results in the embedded implementation, we only need the encoder part of the CAE model and the centroids of KM. We proceed in the following manner: first and for each input, we obtain the feature vector by forwarding the input through the encoder, and second, we measure the distance to all the centroids and choose the nearest one to assign a class.

It is important to note that, in CMSIS-NN legacy API, the quantization used is the power of two based and in the case of PyTorch is linear, enforcing thus a loss of precision. This changes the accuracy of the model but not the size since in both cases we use 8-bit integers. We obtain a V-Measure of 84.08 %, hence resulting in a loss of around 3.10 % in the V-measure, but also validating our embedded deployment.

Finally, to have a visual representation of the clusters, we embed with t-SNE [40] all the encoded features from the training set and also the centroids. As can be seen in Fig. 5, the clusters are well separated identifying each one as a group of gestures corresponding to a specific number.

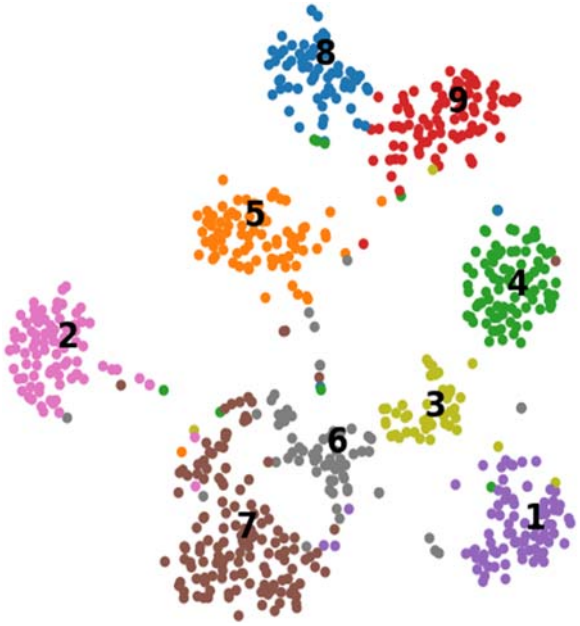


Fig. 5. Cluster visualization of the different gesture numbers through t-SNE.

6. Conclusions

We have shown a full pipeline of work for unsupervised classification of gestures. We have employed a CAE plus KM as a model, and a multi-objective NAS to find a proper embeddable model. Finally, we have been able to embed that model into an embedded platform with CMSIS-NN, obtaining good performance results, and thus validating our approach. There are two next important steps to be able to improve the gesture recognition system. First, to change the quantization to linear instead of power of two based to reduce the drop in performance and also to improve. Second, to improve the clustering by adding a Kullback-Leibler divergence component in the loss, hence separating more the feature vectors corresponding to different numbers [41].

Acknowledgments

This project is supported by the Spanish Ministry of Science, Innovation, and Universities under grant RTI2018-095209-B-C22 and the Catalan Government industrial Ph.D. program under grant 2018-DI-3.

References

[1]. S. Giancola, M. Valenti, R. Sala, A survey on 3D cameras: Metrological comparison of time-of-flight,

structured-light and active stereoscopy technologies, *Springer*, 2018.

[2]. F. M. Caputo, P. Prebianca, A. Carcangiu, L. D. Spano, A. Giachetti, Comparing 3D trajectories for simple mid-air gesture recognition, *Computers and Graphics*, Vol. 73, 2018, pp. 17–25.

[3]. T. Grosse-Puppendahl, *et al.*, Finding common ground: A survey of capacitive sensing in human-computer interaction, in *Proceedings of the Conference on Human Factors in Computing Systems (ACM CHI'17),* 6-11 May 2017, pp. 3293–3316.

[4]. R. Binns, M. Veale, M. Van Kleek, N. Shadbolt, Like Trainer, Like Bot? Inheritance of Bias in Algorithmic Content Moderation, *Social Informatics*, 2017, pp. 405–415.

[5]. D. Stamoulis, *et al.*, Single-path nas: Designing hardware-efficient convnets in less than 4 hours, in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019, pp. 481–497.

[6]. I. Fedorov, R. P. Adams, M. Mattina, P. N. Whatmough, SpArSe: Sparse Architecture Search for CNNs on Resource-Constrained Microcontrollers, in *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, Canada, 2019, pp. 1–26.

[7]. A. Moschetti, L. Fiorini, D. Esposito, P. Dario, F. Cavallo, Toward an Unsupervised Approach for Daily Gesture Recognition in Assisted Living Applications, *IEEE Sens. J.*, Vol. 17, Issue 24, 2017. pp. 8395–8403.

[8]. C. Song, F. Liu, Y. Huang, L. Wang, T. Tan, Auto-encoder based data clustering, *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, Vol. 8258 LNCS, Issue PART 1, 2013, pp. 117–124.

[9]. J. A. Paradiso, N. Gershenfeld, Musical applications of electric field sensing, *Comput. Music J.*, Vol. 21, Issue 2, 1997, pp. 69–89.

[10]. M. Huynh, P. Nguyen, M. Gruteser, T. Vu, POSTER: Mobile Device Identification by Leveraging Built-in Capacitive Signature, in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1635–1637.

[11]. K. Hinckley, M. Sinclair, Touch-Sensing Input Devices, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1999, pp. 223–230.

[12]. A. Pouryazdan, R. J. Prance, H. Prance, D. Roggen, Wearable Electric Potential Sensing: A New Modality Sensing Hair Touch and Restless Leg Movement, in *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, 2016, pp. 846–850.

[13]. A. Nelson, G. Singh, R. Robucci, C. Patel, N. Banerjee, Adaptive and Personalized Gesture Recognition Using Textile Capacitive Sensor Arrays, *IEEE Trans. Multi-Scale Comput. Syst.*, Vol. 1, Issue 2, 2015, pp. 62–75.

[14]. S. Escalera, V. Athitsos, I. Guyon, Challenges in multi-modal gesture recognition, *Gesture Recognit.*, 2017, pp. 1–60.

[15]. D.-Y. Huang, W.-C. Hu, S.-H. Chang, Vision-based hand gesture recognition using PCA+Gabor filters and SVM, in *Proceedings of the Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2009, pp. 1–4.

[16]. M. A. Moni, A. B. M. S. Ali, HMM based hand gesture recognition: A review on techniques and approaches, in *Proceedings of the 2nd IEEE International Conference on Computer Science and Information Technology*, 2009, pp. 433–437.

[17]. J. Nagi, *et al.*, Max-pooling convolutional neural networks for vision-based hand gesture recognition, in *Proceedings of the IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2011, pp. 342–347.

[18]. X. Liu, K. Fujimura, Hand gesture recognition using depth data, in *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition,* 2004, pp. 529–534.

[19]. H.-I. Lin, M.-H. Hsu, W.-K. Chen, Human hand gesture recognition using a convolution neural network, in *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE)*, 2014, pp. 1038–1043.

[20]. V. Shanmuganathan, H. R. Yesudhas, M. S. Khan, M. Khari, A. H. Gandomi, R-CNN and wavelet feature extraction for hand gesture recognition with EMG signals, *Neural Comput. Appl.*, Vol. 32, Issue 21, 2020, pp. 16723–16736.

[21]. S. Y. Kim, H. G. Han, J. W. Kim, S. Lee, T. W. Kim, A hand gesture recognition sensor using reflected impulses, *IEEE Sens. J.*, Vol. 17, Issue 10, 2017, pp. 2975–2976.

[22]. P. Wang, W. Li, S. Liu, Y. Zhang, Z. Gao, P. Ogunbona, Large-scale continuous gesture recognition using convolutional neural networks, in *Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 13–18.

[23]. D. Castells-Rufas, J. Borrego-Carazo, J. Carrabina, J. Naqui, E. Biempica, Continuous touch gesture recognition based on RNNs for capacitive proximity sensors, *Personal and Ubiquitous Computing*, 2020.

[24]. P. Molchanov, S. Gupta, K. Kim, J. Kautz, Hand gesture recognition with 3D convolutional neural networks, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 1–7.

[25]. B. Jacob, *et al.*, Quantization and training of neural networks for efficient integer-arithmetic-only inference, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2704–2713.

[26]. S. Anwar, K. Hwang, W. Sung, Structured pruning of deep convolutional neural networks, *ACM J. Emerg. Technol. Comput. Syst.*, Vol. 13, Issue 3, 2017, pp. 1–18.

[27]. E. Liberis, Ł. Dudziak, N. D. Lane, μNAS: Constrained Neural Architecture Search for Microcontrollers, *Computer Science,* 2020.

[28]. L. Lai, N. Suda, V. Chandra, CMSIS-NN: Efficient neural network kernels for arm Cortex-M CPUs, *arXiv Prepr. arXiv1801.06601*, 2018.

[29]. Y. Zhang, N. Suda, L. Lai, V. Chandra, Hello edge: Keyword spotting on microcontrollers, *arXiv Prepr. arXiv1711.07128*, 2017.

[30]. M. Scherer, M. Magno, J. Erb, P. Mayer, M. Eggimann, L. Benini, TinyRadarNN: Combining Spatial and Temporal Convolutional Neural Networks for Embedded Gesture Recognition with Short Range Radars, *arXiv Prepr. arXiv2006.16281*, 2020.

[31]. E. Torti, *et al.*, Embedded real-time fall detection with deep learning on wearable devices, in *Proceedings of the 21st Euromicro Conference on Digital System Design (DSD)*, 2018, pp. 405–412.

[32]. H. V. Le, S. Mayer, N. Henze, Investigating the feasibility of finger identification on capacitive

touchscreens using deep learning, in *Proceedings of the 24th International Conference on Intelligent User Interfaces*, 2019, pp. 637–649.
[33]. F. Sakr, F. Bellotti, R. Berta, A. De Gloria, Machine learning on mainstream microcontrollers, *Sensors*, Vol. 20, Issue 9, 2020, p. 2638.
[34]. U. Borgmann, Method for Detecting Contact on a Capacitive Sensor Element, US Patent App. 16/354, 699, March 2019.
[35]. D. Arthur, S. Vassilvitskii, k-means++: The advantages of careful seeding, *Stanford*, 2006.
[36]. A. Rosenberg, J. Hirschberg, V-measure: A conditional entropy-based external cluster evaluation measure, in *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007, pp. 410–420.
[37]. K. Swersky, D. Duvenaud, J. Snoek, F. Hutter, M. A. Osborne, Raiders of the lost architecture: Kernels for Bayesian optimization in conditional parameter spaces, *arXiv Prepr. arXiv1409.4011*, 2014.
[38]. E. Bakshy, *et al.*, AE: A domain-agnostic platform for adaptive experimentation, *SemanticScholar*, 2018.
[39]. M. Balandat, *et al.*, BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization, in *Advances in Neural Information Processing Systems*, Vol. 33, 2020.
[40]. L. van der Maaten, G. Hinton, Visualizing data using t-SNE, *J. Mach. Learn. Res.*, Vol. 9, Issue 86, 2008, pp. 2579–2605.
[41]. V. Prokhorov, E. Shareghi, Y. Li, M. T. Pilehvar, N. Collier, On the importance of the Kullback-Leibler divergence term in variational autoencoders for text generation, *arXiv Prepr. arXiv1909.13668*, 2019.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 17-24

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Real Time Object Detection with Noisy Sensors Using Deep Learning

[1, 2] Aditya Singh, [1] Pratyush Kumar, [1] Vedansh Priyadarshi, [1] Yash More, [1] Aishwarya Praveen Das, [1] Bertrand Kwibuka and [1, *] Debayan Gupta

[1] Ashoka University, Rajiv Gandhi Education City, 131029, Sonepat, India
[2] Carnegie Mellon University, 5000 Forbes Ave, 15213, Pittsburgh, PA United States
Tel.:+91 9909869367
E-mail: debayan.gupta@ashoka.edu.in

Received: 10 December 2020 /Accepted: 15 January 2020 /Published: 28 February 2021

Abstract: In this paper, we introduce a first of its kind, radio-signal based object detection system for controlled environments, which substitutes complex signal processing and expensive hardware with deep learning networks to detect patterns from low-quality, inexpensive sensors. Our system operates in the less crowded low-frequency range of 433 MHz in contrast to existing RF-based sensing methods and uses mini-Doppler maps generated from raw I/Q data, thereby allowing us to use cheap, off-the-shelf software defined radios. We demonstrate that our system is versatile enough to handle occlusions and is also sensitive to multiple objects; additionally, it does not use visual data and hence is not hampered by bad lighting. The core of our system is a VGG-16 based CNN architecture trained on the mini-Doppler maps. We achieve an accuracy of 0.96 on a binary classification task of detecting the presence or absence of an object in an enclosed space. Furthermore, we observe that our system shows promise for more complicated detection algorithms as it is able to successfully differentiate between the presence of a single object and two identical objects placed together. Our results indicate that convolutional networks can learn features important enough from spectrograms that enable it to distinguish the presence of objects, thereby eliminating the need of sophisticated signal processing methods to do the same.

Keywords: Deep Learning, CNN, Object detection, Software defined radio (SDR).

1. Introduction

Object detection has attracted high research attention in the past few years due to the remarkable results in the field of deep learning. Neural networks have proven to be good feature extractors and are used in several applications such as human-computer interaction, traffic-monitoring, healthcare and autonomous driving. Deep learning models have achieved high accuracy in diagnostic tasks across multiple medical modalities, and object detection techniques have helped flag organ damage via access to just its corresponding images [12]. These applications often rely on deep learning agents to supervise decision making, perception and understanding of user behavior in different scenarios/environments.

Advancements in the field of deep learning have also led to various applications in robotic systems such as Spot Mini and Atlas by Boston Dynamics. Deep learning combined with video surveillance can be used to perform crowd scene analysis, crowd density estimation, crowd segmentation and detection, which can be critical for disaster management [27].

Estimation and monitoring of crowds can also help ensure practice of social distancing norms amid the current coronavirus pandemic[1]

Additionally, smart cities and buildings have been able to optimize energy consumption based on the number of people in a certain area. An example of this is Google DeepMind[2]. DeepMind has been able to cut down Google's data center cooling bill by 40 % by using an ensemble of neural networks to predict future temperature and pressure of the data center and recommend actions to maintain optimal system settings [13].

However, most of these detection and monitoring approaches have several inherent limitations. They:

1) Require multiple cameras to be installed in an area to achieve adequate performance and coverage which results in high deployment cost,

2) Suffer from the same limitations as humans, i.e, cameras cannot perceive visual information in the dark or see through walls and occlusions. Additionally, in the age of comprehensive facial recognition technology, visual surveillance approaches pose major questions challenging the right to privacy of the public. Fortunately, visible light is just one end of the frequency spectrum. Recent advances in wireless research have shown that certain radio frequency signals can traverse through walls/occlusions and span across dark and smoky environments which are difficult to monitor [2]. Radio waves are also reflected off of the human body [35] (and any other conducting material). Furthermore, since typical WiFi systems operate in the radio wave channel, their ubiquity is an added advantage for building RF-based detection systems. These properties together make radio waves a better alternative to visible light for human-body detection. Unfortunately, most existing research projects utilising RF-waves for human body detection focus on device-based active methods that require significant instrumentation of the person/object and the environment. This limits their scalability and robustness. Furthermore, such design also impinges on the privacy of the users. Thus, there has been a push in research to build device-free methods that can overcome these restrictions. Recent advances in device-free methods have been promising. Although they rely on using expensive, state-of-the-art hardware operating in the overcrowded high frequency WiFi channel [1, 18, 36].

Here, we present a Deep Learning-based portable object-detection system, which uses low-frequency radio waves (433 MHz, via inexpensive software radios), designed to operate through occlusions [6].

We perform the following experiments to test its versatility:

- Detect an object placed in line-of-sight of Receiver and Transceiver;
- Differentiate between the presence of single object and two identical objects kept together.

Our experiments suggest that deep neural networks can be successfully trained to carry out the otherwise intractable tasks of both single and multi-object detection using low-frequency radio waves. Our approach is partly inspired by the mechanisms used by certain owls who can use very low frequency waves (around 20 kHz) to localise their prey during night-time and through occlusions. Due to its portability and cheap deployment costs, our system can be a crucial step in the direction of building scalable, robust and cost-efficient surveillance systems.

2. Related Works

Recent applications of ANNs to signal processing show that CNNs are able to outperform decade old feature search algorithms in radio modulation recognition.

Timothy, *et al.* [19] achieved state of the art accuracy for radio modulation recognition by training a CNN on raw I/Q samples (WiFi enabled SDR operating at 900 MHz)collected across 11 different modulations (8 digital and 3 analog), and normalized to unit variance. Even for high signal to noise ratio, the CNN achieved a better accuracy over a large coverage area. They were able to improve the accuracy by leveraging deep residual networks and LSTM [14] which performed better on visual and time-series data [20].

Zhao, *et al*. [36] introduced RF-pose 3D which provides a significant improvement in RF-based sensing by incorporating ANNs. RF-pose 3D takes the 4D RF signal captured by an FMCW radio (window of 3 seconds) as input, similar to the radio used by RF-capture and WiTrack (5.4-7.2 GHz) [3, 4]. The model operates using a regular softmax loss and is able to predict the location of each key point in space as the voxel with the highest score. Their system is capable of accurately tracking the motion and presence of a single person. To scale this to multiple people, the authors operate on the output of the CNN (in the horizontal plane) from an intermediate layer (feature map).The resulting network is able to accurately detect the 3D skeleton of multiple people and simple actions (walking, sitting, standing) over a range of 40 feet in multiple environments.

Li, *et al*. [18] further augmented this system to detect multiple actions by performing multimodal training using the obtained 3D skeleton from RF-pose 3D and existing vision-based action recognition data sets. The spatial-temporal attention module trained on the multimodal data is able to perform action recognition on visual and RF-based data. Further, they found that transferring knowledge related to action recognition across modalities is able to empirically improve performance regardless of whether the skeletons are generated from RF or vision-based data. This system, called RF-action is able to accurately

[1]https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public

[2]https://deepmind.com/blog/article/deepmind-ai-reduces-google-data-centre-cooling-bill-40

recognise actions and interactions (29 single actions and 6 inter-actions) of multiple humans through walls and in the dark (up to 40 feet) and represents a significant improvement inaction recognition capabilities.

Wang, *et al.* [30] presented WiHear which enables to hear any "talks" without using any specific devices. WiHear detects and analyzes radio reflections due to mouth movements. WiHear solves the problem of recognising micro-movement of lips by using partial multipath effects and wavelet packet transformation. WiHear is also able to listen to multiple people talking by leveraging MIMO technology. They tested this on USRP N210 and commercial WiFi router. They achieve detection accuracy of 91 % on average for a single individual speaking no more than six words and up to 74 % for no more than three people talking simultaneously. Wang, *et al*. [29] also showed that a wearable device can be exploited to understand a user's fine-grained hand movements at millimeter level, which enables attackers to reveal users' secret keys entries.

Al-qaness, *et al.* [5] proposed a device-free CSI-based human activity recognition system which exploits CSI of an indoor ubiquitous wireless device. They described recent research in device-free human sensing technologies and proposed an effective method of CSI filtering, pattern segmentation using machine learning methods, and micro-activity classification. They evaluated the proposed method in an indoor environment and took useless noise into consideration. Their model was able to classify nine human micro-activities. They evaluated their model in various scenarios with ten people and their model reached high accuracy. There are some limitations to their methods as well. It can't track movements of two or more people.

Bahl, *et al.* [8] used RSSI for pose estimation. Presence Of human body within wireless range causes signal attenuation, leading to a spike in RSSI measurement. Thus, RSSI has been widely used in various human-based activity recognition, for instance, device-free indoor localisation [32], density estimation of crowd in a room [31].The ease of calculating RSSI makes it more useful across devices. But RSS measurements are not effective in indoor environment due to several reasons [21]. As RSS measurements, being calculated over a multi-path channel, are prone to effects of multipath fading, and presence of reflecting objects in the environment surrounding leads to the receiver seeing multiple copies of the transmitted signal superimposed on each other leading to difference in attenuation. Moreover, RSSI can only detect limited types of human activities due to single path loss.

Zhao, *et al.* [34] proposed a wireless system called EQ-Radio that infers people emotions from the radio signals that bounce off their bodies. EQ-Radio tried to segment these RF reflections to the individual heartbeats. It had no idea how the heartbeats look like in reflected signals. There are two additional issues that they faced while classifying various human emotions. First, the reflected signals from human bodies were noisy. Second, they were operating in a low Signal to Interference and Noise Ratio (SINR) where there was too much interference due to breathing. EQ-Radio gave results very close to the ECG-based models.

Pu, *et al*. [24] proposed a Doppler shift based gesture recognition system, WiSee which is able to recognise 9 hand/leg gestures based on the unique Doppler shift profiles extracted from wireless signals. Due to different body movements with respect to the SDR and transmitter, there is a unique positive and negative Doppler shift pattern for different activities. WiSee has achieved the highest accuracy of 94 %.

Although the above systems show promising results, they use expensive hardware which operate on high frequency radio waves (2.0 GHz ≤ f ≤ 7.2 GHz). In the following sections we explain our approach presenting relevant differences since we deploy a low-cost, low-frequency setup which is capable of high accuracy detection.

3. Convolutional Neural Network

Convolutional Neural Networks (CNNs) have shown outstanding performance in understanding images [10], videos [17] and audios [7] related tasks not limited to object detection, colorization, image synthesis, pose recognition. Below we briefly describe the important layer types in CNN that are relevant to this paper. Deep Neural Networks (DNNs): DNNs are known to compute any continuous function to a desired level of precision by adding sufficient numbers of neurons in a hidden layer. Each unit of neuron in deep neural networks computes the weighted sum of inputs and the result is passed through an activation function (like softmax, ReLU etc.) to introduce non-linearity. Mathematically, the value of a neuron ${a_i}^n$ at the n-th layer can be calculated using $\sigma(\sum_j {w_{ij}}^n {a_j}^{n-1})$, where ${a_j}^{n-1}$ are the neurons from the previous layer, ${w^n}_{ij}$ are the weights and σ(·) introduces non-linearity. CNNs: In case of RGB images, CNNs work on 3D tensors consisting of two spatial dimensions, height and width, and a channel dimension. There is a chain of nonlinear filters which extract the abstract features from a low level RGB image like edges, boundaries, and finally the actual object. Unlike vanilla neural networks, neurons are partially connected to neurons in the previous layer. Due to this property of CNNs, they are easily able to learn sparse features present in the data. CNN consist of sequence of layers $f_1, f_2, \ldots$, and f_l. Each layer acts as a function which takes x_{l-1} tensor as an input and gives x_l tensor as output. For layer l, we can write this function as $x_l = f^l(x_{l-1})$. The functions $f^l s$ are non-linear local and translation invariant operators. The value of neurons in any layer l can be calculated using convolution operation of the weights kernels and tensors value in the previous layer, that is $x^l = \sigma(f^l * x_{l-1})$, where $*$ is the convolution

operator, f^l refers to the weight kernel at layer l, and x^l is the computed value of neurons in layer l. The CNN training can be summarised using the Algorithm 1.

Algorithm 1: CNN Training

Input: Tensors: $c, x_0, w_1, w_2, \ldots, w_{l-1}$
Output: Minimize the value of L, where *loss* is a loss function
for $k \leftarrow 1$ to l **do**
$$x_k \leftarrow f_k(x_{k-1}, w_k)$$
end for
$$L = loss(x_l, c)$$

Below we briefly explain the building blocks of CNNs:

1. Convolution Layer: If the input in l^{th} layer is an order 3 tensor of size $H^l \times W^l \times D^l$, then convolution kernel tensor of size $H \times W \times D^l$ overlay on top of the input tensor at a location. We compute the product of corresponding elements in all the D^l channels and sum the HWD^l product to get the convolution result at that location. The kernel is moved from left-to-right and top-to-bottom to complete the convolution operation.

2. Rectified Linear Unit (ReLU) Layer: The ReLU function performs truncation operation individually for each element in the input tensor:

$$y_{i,j,d} = max(0, x^l{}_{i,j,d}),$$

where $0 < i < H^l = H^{l+1}$, $0 < j < W^l = W^{l+1}$, and $0 < d < D^l = D^{l+1}$.

3. Batch Normalization (BN) Layer [16]: BN smooths the input layer by recentering and rescaling the input layer and can be represented mathematically as,

$$BN_{\gamma^k \beta^k}(x^k) = \gamma^k x^k + \beta^k,$$

where x^k is the input tensor, and γ^k and β^k are learnt during the training process. It also helps in mitigating the problem of internal covariate shift.

4. Dropout Layer: Dropout is a regularization technique used to ignore a percentage of randomly selected neurons during training to improve generalisation performance of the network. Dropout means temporarily removing hidden and visible units from the network. If the model performance is not improved after a new epoch, then the dropped units are retained.

5. Softmax Layer: This layer is used to convert the vectors of actual scores of classes to a vector of probability values. One more important property of the softmax operator is that it is shift invariant. The softmax loss is used to measure the difference between the class scores $s = \{s_C\}^k{}_{C=1}$ and the target label C^*as follow:

$$L_{softmax}(s, C^*) = -log \frac{e^{s_{C^*}}}{\sum_C e^{s_C}},$$

where s_{C^*} is the score prediction of the target class. Finally, the CNN computes class scores for new input and predicts it as the class with the highest score, as shown in Fig. 1.

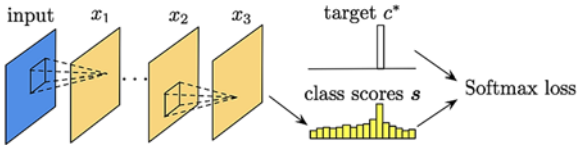


Fig. 1. CNN architecture for classification task, where x_1, x_2 and x_3 are feature maps, s shows the class scores, and c^* refers to the label. The Softmax loss function compares the score vector with the label.

4. System Setup and Data Collection

Our setup comprises a hardware platform (to send and receive signals), and software (to control the transmission and run detection algorithms). Our system uses a Transmitter (Tx) made out of a Raspberry Pi 3 and a receiver (Rx) based on a Realtek RTL SDR along with an off-the-shelf dipole antenna. We use the RTLSDR and SCIPY Python libraries for interfacing and transmitting narrowband FM waves at 433 MHz. We perform three basic experiments to examine whether our system can detect a single object and scale to multiple such objects. For the first experiment we train a Convolutional Neural Network (CNN) to detect the presence and absence of an object at close range. For the second experiment we double the range of our system setup and examine the impact on detection. Finally, for the third experiment we examine whether our setup is capable of distinguishing the presence of two similar objects from the presence of a single one. Three datasets, namely D_A, D_B and D_C are collected for each of the three experiments correspondingly.

D_A, D_B and D_C are collected in a closed room (8.5 m×5.5 m). For D_A, Tx and Rx are placed 1 m apart; the distance is doubled for D_B. In both cases the object (a 1 L water bottle) is placed exactly midway. We sample the received signal in raw I/Q form, with each recorded sample being 0.5 s. D_A is sampled at 2.6 MS/s, whereas D_B is recorded at a reduced 2 MS/s to decrease computational overhead. 2000 samples are collected in DA, and 6000 in DB. For DC, Tx and Rx are placed 2 m apart (as in DB. Data is collected for two cases, first the object is placed exactly midway as done in D_A and DB, for the second case two of the same bottles are placed 20 cm apart in line of sight of the antenna. For each of these cases 4500 samples are collected with each recorded sample being 0.5 s and a

sample rate of 2 MS/s. The setup used for collecting D_C has been shown in Fig. 2.

For D_C, Tx and Rx are placed 2 m apart (as in D_B. Data is collected for two cases, first the object is placed exactly midway as done in D_A and D_B, for the second case two of the same bottles are placed 20 cm apart in line of sight of the antenna. For each of these cases 4500 samples are collected with each recorded sample being 0.5 s and a sample rate of 2 MS/s. The setup used for collecting D_C has been shown in Fig. 2.

Fig. 2. System setup for detection of 2 identical bottled kept together, Transceiver and Receiver are placed 2 m apart.

5. Preprocessing and Model

After the three sets of data are collected, the data is operated on in three different representations. The first is a raw time series radio signal, and the second is a power spectral density function generated for our samples as in [22]. For the final representation we apply Short-Time Fourier Transform (STFT) to each sample to convert it into a spectrogram representation in the frequency-time domain (50 % overlap in each chunk) [18, 36]. We obtain 1,000 and 3,000 samples for each case (object, no object) in D_A and D_B respectively. In D_c we obtain 4,500 samples for each case (one object, two objects 20 cm apart).

We start with a baseline architecture as shown in Fig. 3 consisting of two convolution layers and two dense layers, then progressively vary the hyperparameters to analyze their effect on performance and arrive at the final model(s). The selected CNN model is based on the VGG-16 [26] architecture which has shown remarkable success in image recognition. It contains four convolutional layers, each followed by a pooling layer using SAME padding. ReLU activation functions are used in the convolutional layers to introduce non-linearity and a Glorot uniform kernel initializer is used to initialise the convolution layers. Two fully connected layers follow the stack of convolutional layers. The final dense layer uses a softmax activation function and the model is trained using binary cross-entropy loss via Stochastic Gradient Descent. For the model used on D_A, Batch Normalisation is done before the inputs are fed into the second convolutional layer and 40 % of the neurons are dropped after the final convolutional block to prevent overfitting. In contrast, for D_B, Batch Normalisation is done after every convolutional layer and the first dense layer, and 50 % of the neurons are dropped after the second and fourth convolutional layer.

The model trained on D_c is slightly different, it uses four convolutional layers with SAME padding and ReLU activations, with a pooling layer after the second and fourth convolutional layer. Two fully connected layers follow the stack of convolutional layers, Batch Normalisation is done after each convolutional layer and the first dense layer. The final dense layer uses a softmax activation function and 50 % of the neurons are dropped after the second and fourth convolutional layer and the first dense layer. We evaluate the performance of our CNN classifiers using 5-fold Cross Validation technique[3]

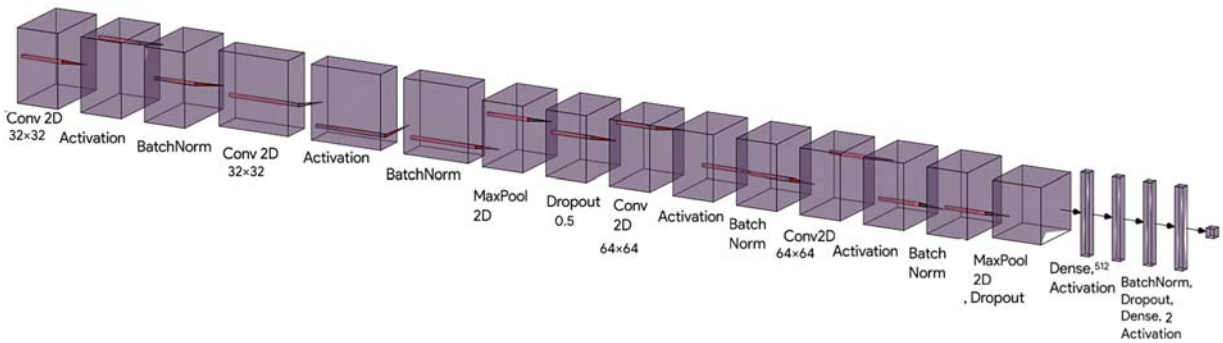


Fig. 3. Visualisation of the CNN architecture used for D_C.

[3]Our models are built using Tensorflow and trained on an NVIDIA RTX 2070

6. Observations

Training a three layered shallow network using the raw I/Q Data and PSD data did not yield useful results as the classifier was unable to generalize well on the test set. This Suggests that the input data was not an adequate feature for detecting the object (or was too noisy). However, we observed that training the CNN described above using the heatmaps of the reflected signals as input yielded a high accuracy on the test set. After training the CNN on D_A for 10 epochs, the network is able to achieve a test accuracy of 0.96 (average over 10 runs). Furthermore, an accuracy of 0.87 was observed on the test set after training and evaluating the CNN on D_B for 20 epochs; there is a decrease in inaccuracy as the distance between T_x and R_x (and the distance of the object from T_x and R_x) is doubled. Still, our system is able to perform high-accuracy object detection even after the range is increased.

Finally, training the described CNN on D_C resulted in a high testing accuracy of 0.99, suggesting that our system is not only capable of simple object detection, but can also be scaled to detect the presence of multiple similar objects. Given this initial success, it would be interesting to analyze the robustness of our low-frequency detection through further experiments. The direct next steps would be determining:

a) The exact functioning range of such a setup,

b) Whether the position of objects be localised to what precision? and

c) How many different/identical objects can be tracked simultaneously and whether our system can track human bodies.

The accuracy results are recorded in Table 1 and the confusion matrix for D_B and D_C is shown in Fig. 4 and Fig. 5 respectively. Furthermore, loss and accuracy with respect to D_A are plotted against the epochs and shown in Fig. 6.

This demonstrates that CNNs are an effective model for identifying and extracting relevant features to carry out such classification tasks.

Table 1. Comparison of performance of CNN on D_A, D_B (as the distance between T_x and R_x is doubled from 1 m to 2 m) and D_C.

Dataset	Validation Accuracy	Testing Accuracy
D_A	0.94	0.96
D_B	0.91	0.87
D_C	0.98	0.99

7. Conclusions

This paper proposes that it is possible to perform simple object detection with high accuracy using extremely cheap radio hardware by leveraging CNNs. We demonstrate this by developing a proof-of-concept system to detect the presence of a water bottle based on spectrograms of the received signal. Additionally, we are also able to use our system to detect the presence of multiple identical objects. Critically, we do almost no explicit signal processing or feature engineering, relying on the neural network to do the heavy lifting. This indicates the possibility of doing complex, customized object detection without expense on human capital or classical signal processing.

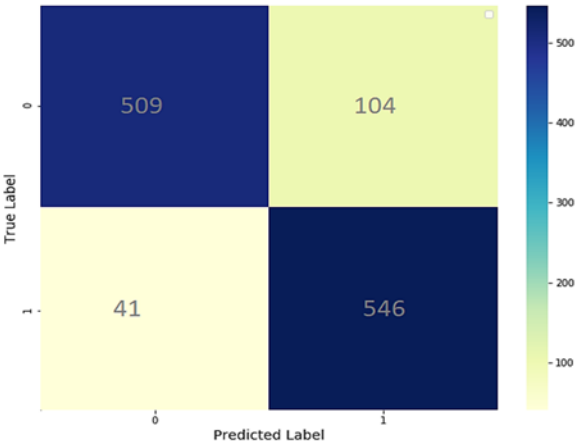


Fig. 4. Confusion matrix for D_B.

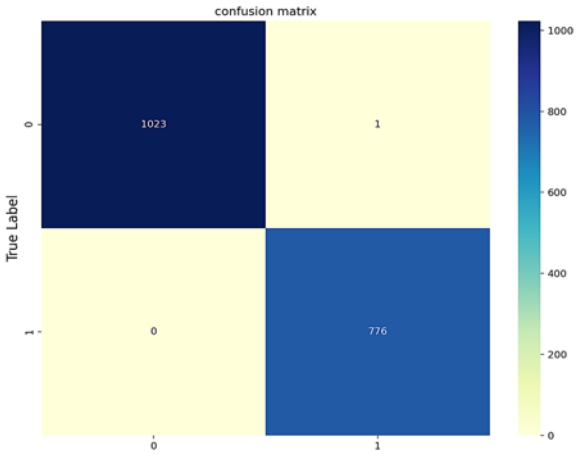


Fig. 5. Confusion matrix for D_C.

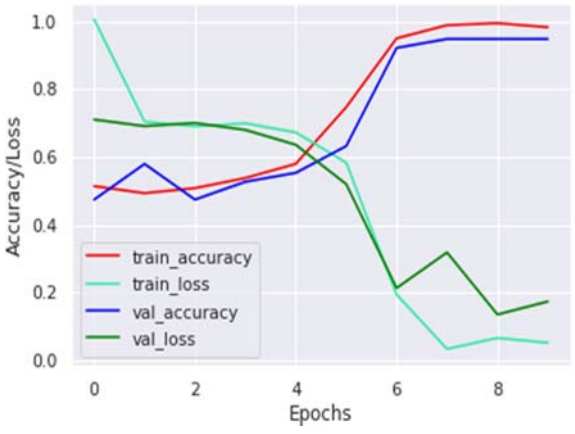


Fig. 6. Accuracy/Loss vs Epochs for D_A.

In the future, we plan to scale the current system by using multiple SDRs to emit an FMCW wave, to determine how the system works in terms of detecting and localizing multiple objects (and human bodies) when it is fed more information about the surroundings. Furthermore, we want to extend our work to perform material detection and classification tasks; additionally, we also want to test whether our low-frequency system can be used to localize objects in an indoor environment. We also want to perform far more complex classifications, reporting precise location of objects, calibrating item sizes, handling more noise, and adapting our system for new environments.

References

[1]. F. Adib, D. Katabi, See through walls with WiFi!, in *Proceedings of the Association for Computing Machinery Conference*, 2013, pp. 75-86.

[2]. F. Adib, Z. Kabelac, D. Katabi, R. C. Miller, 3D tracking via body radio reflections, in *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14),* Seattle, WA, 2014, pp. 317–329.

[3]. F. Adib, C.-Y. Hsu, H. Mao, D. Katabi, F. Durand, Capturing the human figure through a wall, *Association for Computing Machinery Transactions on Graphics*, Vol. 34, Issue 6, Oct. 2015, pp. 1-13.

[4]. F. Adib, Z. Kabelac, D. Katabi, Multi-person localization via RF body reflections, in *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, Oakland, CA, pp. 279–292, May 2015.

[5]. M. A. A. Al-qaness, Device-free human micro-activity recognition method using WiFi signals, *Geospatial Information Science*, Vol. 22, Issue 2, 2019, pp. 128–137.

[6]. P. Ali-Rantala, L. Ukkonen, L. Sydanheimo, M. Keskilammi, M. Kivikoski, Different kinds of walls and their effect on the attenuation of radiowaves indoors, in *Proceedings of the IEEE Antennas and Propagation Society International Symposium (IEEE-APS)*, Vol. 3, 2003, pp. 1020–1023.

[7]. Y. Aytar, C. Vondrick, A. Torralba, SoundNet: Learning sound representations from unlabeled video, in *Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16),* Red Hook, NY, USA, 2016, pp. 892–900.

[8]. P. Bahl, V. N. Padmanabhan, Radar: an in-building RF-based user location and tracking system, in *Proceedings of the IEEE Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'2000),* Vol. 2, 2000, pp. 775–784.

[9]. G. Benton, M. Finzi, P. Izmailov, A. G. Wilson, Learning invariances in neural networks, in *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020),* Vancouver, Canada, 2020.

[10]. J. Deng, W. Dong, R. Socher, L. Li, Kai Li, Li Fei-Fei, ImageNet: A large-scale hierarchical image database, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[11]. J. Ehrich, E. Molloy, M. Mantovani, Conceptual design of future children's hospitals in europe: Planning, building, merging, and closing hospitals, *The Journal of Pediatrics*, Vol. 182, Issue 12, 2016, pp. 411-412.

[12]. A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, J. Dean, A guide to deep learning in healthcare. *Nature Medicine*, Vol. 25, Issue 1, January 2019, pp. 24–29.

[13]. R. Evans, J. Gao, DeepMind AI reduces google data centre cooling bill by 40 %, Jul 2016. https://deepmind.com/blog/article/deepmind-ai-reduces-google-data-centre-cooling-bill-40.

[14]. S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural Computation*, Vol. 9, Issue 8, Nov. 1997, pp. 1735–1780.

[15]. F. Hu, Y. Deng, W. Saad, M. Bennis, A. H. Aghvami, Cellular-connected wireless virtual reality: Requirements, challenges, and solutions, *IEEE Communications Magazine*, Vol. 58, Issue 5, 2020, pp. 105–111.

[16]. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in *Proceedings of the 32nd International Conference on Machine Learning (ICML'15),* Vol. 37, 2015, pp. 448–456.

[17]. K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang, W. Ouyang, T-CNN: Tubelets with convolutional neural networks for object detection from videos, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 28, Issue 10, 2018, pp. 2896–2907.

[18]. T. Li, L. Fan, M. Zhao, Y. Liu, D. Katabi, Making the invisible visible: Action recognition through walls and occlusions, in *Proceedings of the Institute of Electrical and Electronics Engineers- International Conference on Computer Vision (IEEE/CVF- ICCV)*, 2019, pp. 872–881.

[19]. T. J. O'Shea, J. Corgan, T. C. Clancy, Convolutional radio modulation recognition networks, in Jayne C., Iliadis L. (eds) Engineering Applications of Neural Networks. EANN 2016. Communications in Computer and Information Science, Vol. 629. *Springer,* 2016.

[20]. T. J. O'Shea, T. Roy, T. C. Clancy, Over the air deep learning based radio signal classification, *IEEE Journal of Selected Topics in Signal Processing*, Vol. 12, Issue 1, Feb. 2018, pp. 168 – 179.

[21]. A. Parameswaran, M. Iftekhar Husain, S. Upadhyaya, Is RSSI a reliable parameter in sensor localization algorithms: an experimental study, *Computer Science*, 2009.

[22]. A. Pérez-Zapata, A. F. Cardona-Escobar, J. A. Jaramillo-Garzón, G. M. Díaz, Deep convolutional neural networks and power spectral density features for motor imagery classification of EEG signals, in *Proceedings of the International Conference on Augmented Cognition (AC)*, *Springer International Publishing*, 2018, pp. 158–169.

[23]. A. F. Pérez-Zapata, A. F. Cardona-Escobar, J. A. Jaramillo-Garzón, G. M. Díaz, Deep convolutional neural networks and power spectral density features for motor imagery classification of EEG signals, in D. D. Schmorrow and C. M. Fidopiastis, editors, *Augmented Cognition: Intelligent Technologies*, *Springer International Publishing*, 2018, pp. 158-169.

[24]. Q. Pu, S. Gupta, S. Gollakota, S. Patel, Whole-home gesture recognition using wireless signals, in *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking (MobiCom'13)*, New York, NY, USA, 2013, pp. 27–38.

[25]. K. Shibata, H. Yamamoto, People crowd density estimation system using deep learning for radio wave sensing of cellular communication, in *Proceedings of the International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2019, pp. 143–148.

[26]. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in *Proceedings of the International Conference on Learning Representations*, 2015.

[27]. G. Sreenu, M. A. S. Durai, Intelligent video surveillance: a review through deep learning techniques for crowd analysis, *Journal of Big Data*, Vol. 6, Issue 1, June 2019.

[28]. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, *J. Mach. Learn. Res.*, Vol. 15, Issue 1, Jan. 2014, pp. 1929–1958.

[29]. C. Wang, X. Guo, Y. Wang, Y. Chen, B. Liu, Friend or foe?: your wearable devices reveal your personal PIN, in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security (ASIA CCS'16)*, New York, NY, USA, 2016, pp. 189–200. https://doi.org/10. 1145/2897845.2897847.

[30]. G. Wang, Y. Zou, Z. Zhou, K. Wu, L. M. Ni, We can hear you with Wi-Fi!, *IEEE Transactions on Mobile Computing*, Vol. 15, Issue 11, 2016, pp. 2907–2920.

[31]. C. Xu, B. Firner, R. S. Moore, Y. Zhang, W. Trappe, R. Howard, F. Zhang, N. An, SCPL: Indoor device-free multi-subject counting and localization using radio signal strength, in *Proceedings of the 12th International Conference on Information Processing in Sensor Networks (IPSN'13)*, New York, NY, USA, 2013, pp. 79–90.

[32]. D. Zhang, Y. Liu, L. M. Ni, RASS: A real-time, accurate and scalable system for tracking transceiver-free objects, in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2011, pp. 197–204.

[33]. M. Zhao, S. Yue, D. Katabi, T. S. Jaakkola, M. T. Bianchi, Learning sleep stages from radio signals: A conditional adversarial architecture, in *Proceedings of the 34th International Conference on Machine Learning (ICML'2017),* Vol. 70, 2017, pp. 4100–4109.

[34]. M. Zhao, F. Adib, D. Katabi, Emotion recognition using wireless signals, *Commun. ACM*, Vol. 61, Issue 9, Aug. 2018, pp. 91–100.

[35]. M. Zhao, T. Li, M. A. Alsheikh, Y. Tian, H. Zhao, A. Torralba, D. Katabi, Through-wall human pose estimation using radio signals, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7356–7365.

[36]. M. Zhao, Y. Tian, H. Zhao, M. A. Alsheikh, T. Li, R. Hristov, Z. Kabelac, D. Katabi, A. Torralba, Rf-based 3D skeletons, in *Proceedings of the Association for Computing Machinery Conference: Special Interest Group on Data Communication (ACM-SIGCOMM)*, 2018, pp. 267–281.

[37]. Z. Zou, Q. Chen, I. Uysal, L. Zheng, Radio frequency identification enabled wireless sensing for intelligent food logistics, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 372, Issue 2017, June 2014, 20130313.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 25-35

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Uncertainty Estimation for Deep Learning-based Thermographic Imaging

[1, *] Bernhard LEHNER, [2] Thomas GALLIEN, [1, 4] Péter KOVÁCS, [5] Gregor THUMMERER, [5] Günther MAYR, [6] Peter BURGHOLZER and [1, 3] Mario HUEMER

[1] Silicon Austria Labs (SAL), JKU LIT SAL eSPML Lab, Altenbergerstr. 69, 4040, Austria
[2] Silicon Austria Labs (SAL), Inffeldgasse. 33, 8010, Austria
[3] Johannes Kepler University Linz, Institute of Signal Processing, JKU LIT SAL eSPML Lab, Altenbergerstr. 69, 4040, Austria
[4] Department of Numerical Analysis, Eötvös Loránd University, 1117, Hungary
[5] Josef Ressel Centre of Thermal NDE of Composites, University of Applied Sciences Upper Austria, Roseggerstraße 15, 4600, Austria
[6] Research Center for Non Destructive Testing, Altenberger Straße 69, 4040, Austria
[1] Tel.: +43 5 0317
E-mail: bernhard.lehner@silicon-austria.com

Received: 1 October 2020 /Accepted: 15 January 2021 /Published: 28 February 2021

Abstract: Thermographic imaging is a contactless and nondestructive way to detect defects inside the specimen. The current state-of-the-art approach combines model- and deep learning-based reconstructions in a hybrid fashion. The recently developed virtual wave concept (VWC) provides a framework to develop such hybrid solutions, and allows to utilize physical priors, such as non-negativity and/or sparsity. In combination with the superiority of deep learning approaches over hand-crafted features and heuristics, improved reconstruction accuracy compared to previous methods was achieved. However, the reconstruction results still deteriorate under low SNR conditions caused by defects located deeper underneath the surface. Therefore, it would be useful to have an uncertainty estimate that reflects the reliability of the reconstructions. This would enable the automatic identification of results that are likely to be inaccurate and require a closer inspection. In this paper, we propose two computationally very cheap methods to estimate the uncertainty of thermographic imaging results. In order to show the generalization capability of our approach, we thoroughly evaluate it under different conditions and even with different deep model architectures.

Keywords: Thermal tomography, Virtual wave, Nondestructive testing, Regularization, Sparsity, Deep learning.

1. Introduction

Deep learning-based approaches for thermographic imaging have been shown to significantly outperform pure numerical, model-based methods even under real-world conditions [1]. However, the thermographic reconstruction results still show signs of deterioration under low SNR conditions which are caused by deeper lying subsurface defects.

This phenomenon is to a large degree a manifestation of information loss, and can only be

partially compensated by priors in the form of non-negativity and sparsity. Therefore, it would be helpful to at least being able to automatically identify highly deteriorated thermographic reconstructions.

In our specific setting, uncertainty estimation should provide a single-valued indication of the reliability of thermographic reconstructions, and thus be highly correlated with the actual loss. Furthermore, it should be applicable in real-time scenarios and on embedded systems with restricted computational capabilities. Consequently, it is important that the method is computationally light-weight and fast. So, existing methods that require several forward passes, like e.g., dropout-based uncertainty estimation [2] or Bayesian ensembling [3] cannot be used.

Our previous work on uncertainty estimation [4] is the starting point of this paper, and we extend it in several aspects. First, we present novel results in a different setting that demonstrates the generalization capability of our approach. Second, we provide more detailed insights to the results so far, and also address possible extensions. Third, we discuss the potential limitations of our method and point to possible future work in order to further improve our current approach.

In the following sections, we first introduce the basic principles of thermographic imaging along with some practical applications for it. Afterwards, we briefly discuss the challenges that are posed by thermodynamics and entropy for the task at hand. We then present two techniques designed to alleviate the deterioration of the results related to these challenges. First, we discuss model-based reconstruction techniques that require modelling the physical aspects of the problem along with regularization. Second, we review deep learning-based reconstruction techniques, where deep learning is either applied in an end-to-end fashion, or along with domain knowledge.

1.1. Thermographic Image Reconstruction

Non-destructive evaluation (NDE) has become an integral part in multiple industry processes in which a product failure might result in an accident or body injury. NDE allows to identify defects before they become critical and thus helps to avoid serious damage of components [5]. Due to rapid developments in the field of infrared (IR) detectors and advanced approaches in signal processing, the emerging NDE technique of active IR thermography (IRT) has become a powerful tool [6]. Contactless and non-intrusive, this technique provides a rapid means for characterizing thermal properties and for locating and sizing of subsurface defects. Since 2017, active IRT has been qualified for testing safety-critical components in the aviation industry. Due to the non-ionizing thermal radiation, active IRT also has emerging applications in medicine [7].

In most thermal NDE applications, one-dimensional (1D) physic-based forward models are used for image enhancement and depth estimation [8]. The 1D models become inaccurate for the reconstruction of finite sized defects, taking into account that the anisotropic heat conduction, e.g., in composite materials, amplifies these effects [9]. To increase the depth resolution, radar inspired signal processing techniques, as matched filtering, were implemented to detect a pre-known signal waveform in a highly noisy channel [10].

Image reconstruction can be seen as the solution of a mathematical inverse problem in which the cause (measured surface temperature) is inferred from the effect (internal heat sources or reflectors). In general, inversion of thermal diffusion is severely ill-posed due to the diffusive nature of the heat propagation. A physical model describes the relationship between the surface temperature and the heat distribution inside the material. Assuming a particular source shape, a parameter estimation problem has to be solved by the minimization of an objective function for a set of parameters (e.g., diameter, depth, thickness). If the shape is not known a priori, regularization procedures can be used for the reconstruction of any energy distribution in a predefined mesh grid [11]. In particular, pulse-echo photo-thermal imaging is the most widely implemented method because the sample is often only accessible from one side. A uniformly spatially distributed pulse, e.g., flashlights, launches a package of plane diffusion waves into the material which propagates normal to the surface. When the scattered waves return to the surface, it affects the surface temperature decay. Holland and Schiefelbein proposed a full 3D image reconstruction technique, which represents the surface temperature as a linear combination of heat contributions [12]. All these methods are based on a large-scale linear inversion that considers (admittedly imperfectly) lateral heat flows. Furthermore, the surfaces have to be segmented into overlapping tiles due to the high computational costs.

A completely new imaging approach is to employ photoacoustic techniques for photothermal signals, the so-called virtual wave concept (VWC) [13]. The reconstruction process is a two-stage process. In the first step, a virtual wave signal is calculated for every location of the measured surface temperature signal. As the virtual wave obeys the wave equation, it can be reversed in time in a second step to reconstruct the internal heat sources and reflectors. This computationally cheap two-stage process for imaging can be used for 1D, 2D and 3D heat conduction problems.

However, the resolution limit of the thermographic reconstruction decays linearly with depth [14]. This diffusion-based blurring can be compensated by incorporating a priori information (e.g., sparsity, non-negativity, etc.) for the computation of the virtual wave field. Additionally, the signal-to-noise ratio (SNR) can be significantly increased by using well-known acoustic reconstruction methods, such as synthetic aperture focusing techniques (SAFTs). Here, the heat flow in all directions is taken into account for reconstruction.

However, handcrafted features and heuristics can be outperformed by deep learning approaches, as was demonstrated in [1]. Here, two training strategies were applied. The first was carried out in an end-to-end fashion, where the surface temperature data was fed directly to a neural network. The second approach incorporated domain knowledge, where the VWC [13] served as a mid-level representation that was fed to the neural network.

1.2. Practical Applications

The model- and deep learning-based reconstruction techniques mentioned in the last sections can be utilized in a wide range of NDE applications [14]. As a case-study, we considered the problem of thickness estimation in 1D [15], and active thermographic computed tomography in 2D [1] and 3D [13]. To this end, we built up various test specimens, such as steel step wedges, and physical phantoms made of epoxy resin, which includes steel beads and graphite bars. With the help of these specimen, the positive impact of prior knowledge (e.g., non-negativity, sparsity) in the reconstruction process could be demonstrated.

Furthermore, we could test the generalization ability of the deep learning-based reconstruction methods. This is extremely important, as the deep neural networks for thermographic image reconstruction were trained on simulated data only. It was shown in [1] that the neural networks can successfully generalize the learned reconstruction algorithm to real-world data even under previously unseen conditions (e.g., noise, rotation). In order to foster the development of other reconstruction techniques, we released both the synthetic and the real-world datasets along with the implementation of the deep learning-based methods to the research community[1].

1.3. Principles of Thermography

Thermography uses heat diffusion to characterize thermal properties and to estimate the location and size of subsurface defects. In active IRT, the sample is heated up by absorption of light, by eddy-currents for electrically conductive internal structures, or by other energy sources, such as ultrasound or microwave absorption. The surface temperature development caused by this is observed with an IR camera. Depending on the direction of the heat diffusion, there are two scenarios. First, the subsurface structures to be imaged can be heated directly, e.g., by eddy-currents. Here, the heat diffuses to the surface, and we consider this a one-way diffusion. Second, the sample surface is heated up, e.g., by a short laser pulse. In that case, the heat diffuses into the sample and the subsurface structures influence the measured surface temperature evolution. We consider this *a two-way diffusion*.

In Fig. 1, we can see an example of a measurement setup, where the specimen is heated up by eddy-currents. All investigations in this paper refer to such a *one-way diffusion* scenario.

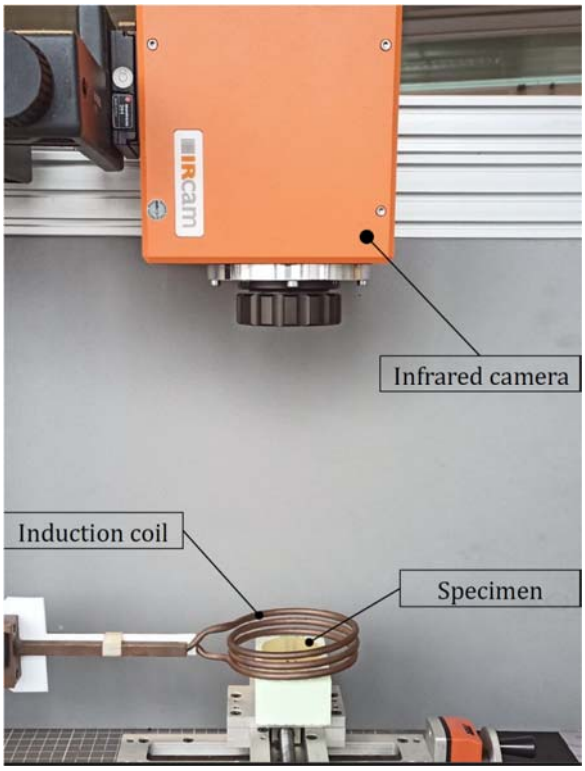


Fig. 1. Measurement setup where the specimen is heated up by eddy-currents.

1.4. Challenges of Thermographic Imaging

Only ideal processes in which dissipative effects such as friction are neglected can be reversed in time. An example of such an ideal process would be a pendulum where damping is ignored, hence oscillating forever. However, real processes like heat conduction do not exhibit a rewind button. From the 2nd law of thermodynamics, we know that heat always flows from the hot to the cold and never in the other direction due to entropy production. Since Szilard [16] and Landauer [17] with his aphorism "Information is physical" we know that the acquisition of information, such as the measuring process in thermography, must obey the laws of (non-equilibrium) thermodynamics. Heat diffusion in active thermography causes entropy production which turns out to be equal to information loss, and in imaging processes less information always corresponds to a limited spatial resolution.

In the VWC we link the measured thermal signal to an ideal solution of the wave equation, the so-called virtual wave, which is reversible in time. Due to the

[1] Available at: https://git.silicon-austria.com/pub/confine/ThermUNet

information loss for the thermal signal, the calculation of the virtual wave is an ill-posed inverse problem that requires regularization. In frequency domain, the loss of information due to entropy production during heat diffusion results in a limited bandwidth [14]. For all frequency components above a truncation frequency the distribution density for the signal does not differ significantly from the equilibrium density. Therefore, these high frequency components cannot contribute to the reconstructed virtual wave. The spatial resolution limit is then diffraction limited to half of the wavelength at the truncation frequency. It turns out that this information related criterion results in the same truncation frequency as when the signal amplitude in frequency domain gets less than the noise level, described by the SNR. For thermographic imaging with its high entropy production from heat diffusion, it is essential to use additional information such as positivity and sparsity by implementing iterative algorithms, e.g., the alternating direction method of multipliers (ADMM) to get a better resolution.

1.5. Model-based Reconstruction Techniques

Thanks to the VWC, the problem of thermographic imaging can be addressed by the two-stage reconstruction process that we already discussed in Section 1.1. Mathematically, each stage is an ill-posed inverse problem. Therefore, regularization is inevitable in order to obtain feasible solutions. Penalized least-squares methods provide a general framework to solve such problems:

$$\tilde{\boldsymbol{x}} = \mathrm{argmin}_{\boldsymbol{x}}\{||\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}||_2^2 + \lambda^2\Omega(\boldsymbol{x})\}, \qquad (1)$$

where $\boldsymbol{b}$ represents the measurement data, $\boldsymbol{A}$ is a matrix that models the physical phenomenon (e.g., heat diffusion, wave propagation), and $\tilde{\boldsymbol{x}}$ is an approximate solution to the virtual waves or to the initial temperature profile of the specimen. The penalty function $\Omega(\cdot)$ is of particular importance that allows to utilize domain knowledge, such as smoothness, non-negativity, sparsity, joint or group sparsity [18]. There are various ways to incorporate such constraints into the reconstruction process, e.g., by using spherical projections [19], Abel- or curvelet transformations [20].

Besides its numerical challenges, there are many computational issues in the reconstruction process. One of these is related to the proper choice of $\lambda > 0$, which is a crucial parameter of the regularization that controls the trade-off between overfitting and underfitting. Several state-of-the-art algorithms can be adopted to estimate the optimal value of λ, such as the L-curve method [21], and the Picard condition [22] that takes the special structure of $\boldsymbol{A}$ into account. This matrix has a block diagonal structure that allows the efficient implementation of the regularization by using the ADMM [23] in conjunction with the L-curve method.

However, this does not apply to the second stage, where the forward operator $\boldsymbol{A}$ cannot be explicitly formulated. An alternative to resolve this problem would be to use time- and frequency-domain SAFT [24, 25]. Another approach would be to utilize deep learning, as will be described in the next section.

1.6. Deep Learning-based Reconstruction Techniques

Besides their rigorous mathematical background, the performance of the model based reconstruction methods is limited. For instance, convex relaxation is one of the most common techniques for solving sparse approximation problems [26, 27]. These iterative regularization algorithms usually have high computational complexity. Hence, their convergence and thus the reconstruction process, which is often recalculated multiple times (e.g., for each cross-sectional data of a 3D measurement), may be slow. Additionally, modeling assumptions such as linearity, convexity, and normality restrict the eligible class of prior knowledge. These handcrafted features and heuristics are often incorporated into Eq. (1) via mathematical constraints and penalty functions Ω, which can lead to suboptimal solutions.

Deep learning is a promising alternative to resolve some of the remaining issues, as was shown recently [18, 1]. However, supervised deep learning methods typically require large amounts of labeled data, which is seldom easy to obtain. In thermography, that would involve the production of physical, so-called phantoms, with varying material properties, e.g., defects at various positions, sizes, and shapes. Fortunately, relatively simple analytic models of the heat diffusion process can be used to synthesize arbitrarily large amounts of data. In order to ensure proper generalization capabilities of a deep learning approach trained with such data, an evaluation under previously unseen, even real-world conditions is of utmost importance.

In [1], the superiority of deep learning approaches over handcrafted features and heuristics was demonstrated via two training strategies. The first was carried out in an end-to-end fashion. That is, the surface temperature data was fed directly to a neural network. The second approach incorporated domain knowledge. Here, the virtual wave concept [13] was involved as a feature extraction step before the neural network was applied.

Unfortunately, limitations of the physical world as discussed in Section 1.4 cannot simply be overcome by just applying deep learning. For that reason, the reconstruction results still deteriorate under low SNR conditions, and for defects located deeper underneath the surface, albeit to a much lesser extent compared to the numerical model-based methods. Therefore, it would be useful to have an uncertainty estimate that reflects the reliability of any given output. This would in turn enable the identification of results that are likely to be inaccurate, hence requiring a closer

investigation. In [4], we introduced two uncertainty estimation methods specifically targeting outputs from the deep learning approaches previously mentioned. However, due to the limitations of a short paper, some details and questions about these methods had to be left out. The remainder of this paper is therefore dedicated to fill in those gaps.

2. Uncertainty Estimation

In this section, we propose two different methods for uncertainty estimation. In order to keep the methods lightweight, we directly infer the uncertainty by analyzing the predicted image containing the defects in a single forward pass. This is a clear advantage over methods that require several forward propagations, like e.g., dropout-based uncertainty estimation [2] or Bayesian ensembling [3].

In the following section, we start by describing the neural network architectures and the data that we use throughout our experiments. With these networks and the data in mind, we developed our uncertainty estimation methods, which will be discussed afterwards.

2.1. Network Architectures

We propose to use a u-net variant that was introduced for various medical image segmentation tasks [28, 29]. This architecture won the International Symposium on Biomedical Imaging (ISBI) cell tracking challenge in 2015, and is known to work well even with relatively little training data. It is basically an autoencoder with skip-connections from the contracting path to the expansive path, enabling the network to localize. The networks are trained to minimize the mean squared error (MSE) between the predicted and the actual target masks. This loss will also serve as a reference to assess the quality of the uncertainty estimation later on. For a more detailed description we refer to the original paper [28].

In this paper, we focus on the two specific architectures that we introduced in [18]. One was designed to be extremely compact in terms of number of parameters, while still delivering satisfying results. It consists of just 16 filters in the first (single channel) layer, and has three layers in each the contracting and the expansive path, resulting in only 109 thousand weights. The second, larger architecture was designed to maximize the performance on our development data, and no restrictions in terms of network capacity were considered. Surprisingly, a very similar architecture as the compact one turned out to work best. It has also just 16 filters in the first (single channel) layer, five layers in each the contracting and the expansive path, and about 1.8 million weights.

Interestingly, the specific training setup did not make a difference to the outcome of the architecture search. That is, for both the end-to-end and the hybrid approach the exact same architectures were found to be most useful. The difference between the end-to-end and the hybrid approach lies in the input data for the network, which we describe in the following section.

2.2. Data

Since the vast amounts of data that is typically required for deep learning approaches is not available, we created a data set comprising simulated data for 2D. We started by randomly placing up to five square-shaped defects with side lengths between two and six pixels in order to end up with a binary target mask. The corresponding surface temperature measurements were then simulated by assuming adiabatic boundary conditions (i.e. there is no heat flow into or out from the specimen) [1]. These surface temperature measurements were later used as input to the end-to-end approach. For our hybrid method, the virtual waves from the temperature measurements were computed by ADMM in conjunction with the Abel transformation [9]. As a result, each sample is represented by three images: the target mask, the temperature measurements, and the virtual waves. All of them are single channel images with a dimensionality of 256 by 64 pixels. In Fig. 2, we can see an example of this data in the form of an initial temperature signal (representing the 2D target), a simulated surface temperature signal (corresponding to 2D thermographic data) and a simulated ideal virtual wave signal computed from it.

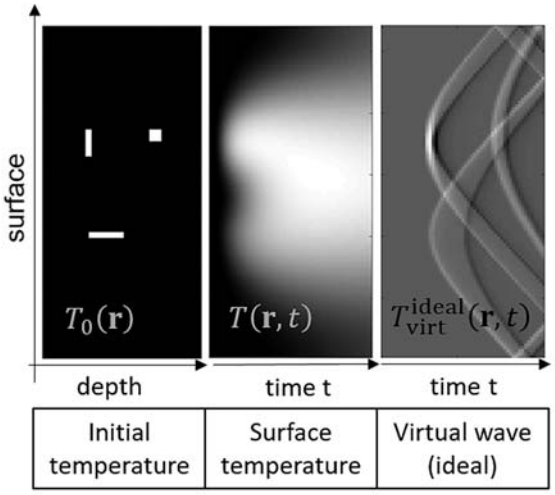


Fig. 2. Example of a target mask, the corresponding temperature measurement and virtual waves.

Additionally, we computed ten different versions of each sample, representing SNRs (i.e., initial temperature vs. noise) from -20 dB to 70 dB in 10 dB steps. The noise was added to the temperature before calculating the virtual waves. This is supposed to increase robustness against changes in the level of SNR during training. Furthermore, having multiple versions per sample is useful for a more detailed

evaluation. This data is available to the public, and for a more detailed description of it we like to refer to [1].

We divided this data into three non-overlapping subsets as follows: Our **training data** consisted of 8,000 samples in ten versions for different SNRs, thus comprising a total of 80,000 samples. We normalized the samples to have zero mean and unit standard deviation using only the training data. For development and validation purposes, we always used the same 1,000 samples. Considering the ten different versions, we had 10,000 samples in our **validation data**. A **test dataset** was unseen in every regard and also normalized based solely on the training data. It also consisted of 1,000 samples in ten versions for different SNRs, comprising 10,000 samples in total.

2.3. Uncertainty Estimation Methods

While developing our methods, we considered several requirements that are relevant for real-world use-cases. First and foremost, the estimate should highly correlate with the true MSE. Furthermore, it should be suitable for real-time scenarios and deployment on systems with restricted computational capabilities, e.g., edge devices.

Our first intuition was to compute the MSE on an estimation of the ground truth (i.e., the defects). The better this estimate, the higher the correlation to the true MSE, i.e. computed with respect to the actual ground truth. For this, we started by defining a mask operator $\mathbf{g}_\theta : \mathbf{X} \in \mathbb{R}^{M\times N} \rightarrow \{0,1\}^{M\times N}$ that will be applied on the output of the neural network, where

$$[\mathbf{g}_\theta(\mathbf{X})]_{ij} = \begin{cases} 1 & \mathbf{X}_{ij} > \theta,\ 0 < \theta \leq 1 \\ 0 & \text{else} \end{cases} \qquad (2)$$

With this mask operator, we then compute two binarized predictions $\widehat{\mathbf{X}}_{\text{sens}} = \mathbf{g}_\alpha(\widehat{\mathbf{X}})$ and $\widehat{\mathbf{X}}_{\text{conf}} = \mathbf{g}_\beta(\widehat{\mathbf{X}})$, where α and β are set to 0.01 and 0.95, respectively. This leads to $\widehat{\mathbf{X}}_{\text{sens}}$ being highly sensitive with respect to the magnitude of each output pixel. $\widehat{\mathbf{X}}_{\text{conf}}$ on the other hand involves only high confident pixels, i.e. with high magnitude. In Fig. 3, we can see some examples of this binarization step. These two binarized masks then serve as the basis for our uncertainty estimates.

For our 1st method, we start by evaluating the MSE of the predicted image $\widehat{\mathbf{X}}$ on only the non-zero pixels of a masked version of $\widehat{\mathbf{X}}$. This is the numerator in the uncertainty estimate that we define as

$$UCRT_1 = \frac{\text{mse}\left(\widehat{\mathbf{X}} \circ \widehat{\mathbf{X}}_{\text{sens}}, \widehat{\mathbf{X}}_{\text{sens}}\right)}{1 + \frac{||\widehat{\mathbf{X}}_{\text{conf}}||_0}{||\widehat{\mathbf{X}}_{\text{sens}}||_0}}, \qquad (3)$$

where ∘ defines the Hadamard product used to apply the mask pixel per pixel. The L^0-norm is used to denote the number of non-zero pixels of the corresponding mask. We use it to compute the ratio of the number of non-zero pixels of the confident to the sensitive binary masks. This ratio $\in \mathbb{R}[0,1]$ tends towards 1 as the predictions become more accurate, since every pixel in the sensitive mask is also an element of the confident mask. In contrast, as the SNR decreases and the predictions become ever more deteriorated, this ratio tends towards 0. As a result, the denominator $\in \mathbb{R}[1,2]$ scales the initial estimate of the MSE.

For our 2nd method, we take the relative estimated area of the defects into account as follows

$$UCRT_2 = \left(1 - \frac{||\widehat{\mathbf{X}}_{\text{conf}}||_0}{||\widehat{\mathbf{X}}_{sens}||_0}\right)\frac{||\widehat{\mathbf{X}}_{\text{sens}}||_0}{n}, \qquad (4)$$

where n denotes the number of pixels of the output image. Notice, that the ratio of the number of non-zero pixels of the confident to the sensitive masks is again utilized as a scaler.

In order to be able to assess the performance of our approaches relative to another common approach, we introduce our baseline in the following Section.

2.4. Uncertainty Estimation Baseline

Where appropriate, we compare our method with an ensemble based approach similar to the one presented by [30]. Here, they model the prediction by means of a Gaussian mixture model. The uncertainty measure is derived by calculating the mean of the Kullback-Leibler divergence of the individual mixture components with respect to the ensemble distribution. However, since thermographic imaging differs significantly from the proposed regression setting, we do not explicitly model the predictions to be Gaussian. Instead, we derive the uncertainty measure rather from an empirical estimate of the ensemble variance. That is, we take the mean of the pixelwise variances as our uncertainty baseline, and refer to it as UCRT_ENS for the remainder of this paper. The MSE of the ensemble – which consists of five models in our scenario – is computed on the averaged predictions of it.

3. Experiments

In this section, we present the results of three conducted experiments, each designed to shed light on a specific aspect of our uncertainty estimations. First, we will assess how our uncertainty estimates correlate with the actual loss for different architectures and both the end-to-end and the hybrid approach. Second, we will demonstrate that our uncertainty estimates generalize to unseen conditions in terms of SNR. Third, we consider a practical use-case scenario and utilize the uncertainty estimations for rejection.

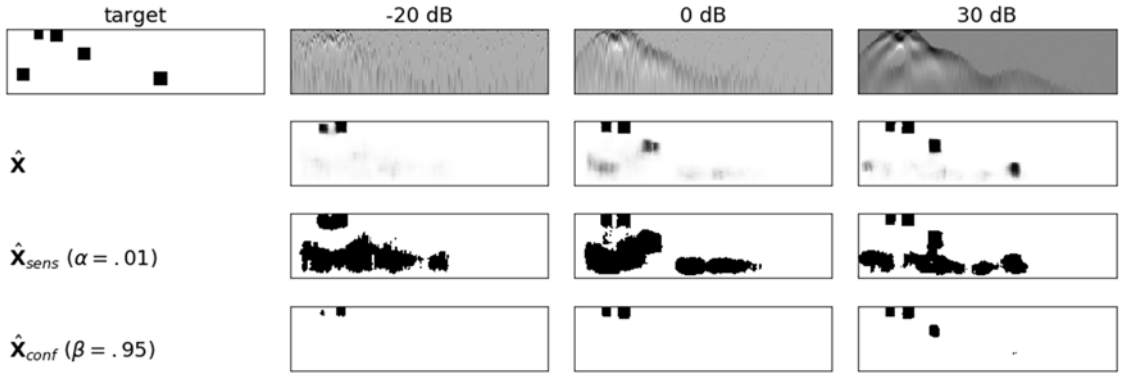


Fig. 3. 1st row: target (representing defects) and input images (virtual waves) for the neural network under several SNR conditions. 2nd row: raw output of the neural network. It can be seen how the quality of the output suffers as the SNR decreases. In the 30 dB scenario, only the most left defect deep inside the specimen is missed, whereas under the -20 dB condition only the two defects closest to the surface appear in the output of the neural network. 3rd row: the highly sensitive binarized mask used to compute the MSE. 4th row: another binarized mask based on high confident pixels.

3.1. Correlation with Loss

With this experiment, we want to show the high correlation between the actual loss (MSE) and the uncertainty estimates. For this, we take the two different approaches introduced in [1]. The first approach is end-to-end, where we feed the temperature measurements directly to the neural networks. This approach has the least computational load, but at the same time the highest task complexity. The second approach is hybrid, where we use virtual waves instead of temperature measurements as input. Compared to the end-to-end approach, some of the computational burden is taken away from the neural network, which yields better results. By combining them with each the compact and the large architecture (see Section 2.1), we get results from four different setups in total.

In Fig. 4, we can see a comparison between our uncertainty estimates and the actual loss for the four different setups.

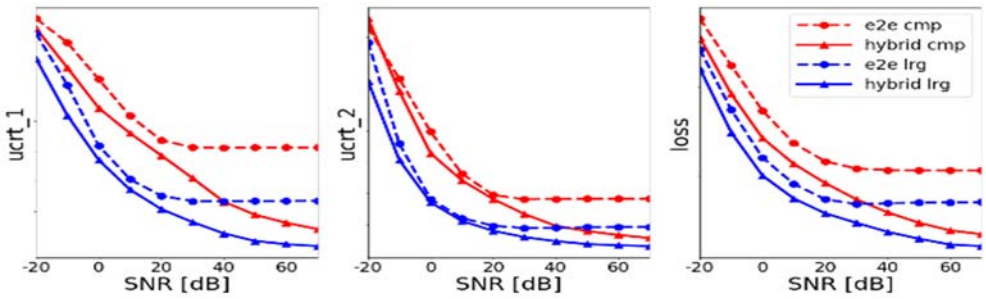


Fig. 4. Uncertainty estimations according to the two approaches compared to actual loss (MSE). The monotonic decrease of the loss corresponding to increased SNRs is well reflected in all scenarios.

This was done by computing the average loss and uncertainties for all SNR conditions on the unseen test data (see Section 2.2). The large and compact architectures are referred to as *lrg* and *cmp*, respectively. As can be seen, there is a high correlation between the uncertainty estimates and the MSE. Also the relative performance in terms of loss between each setup is represented by the uncertainty estimates, at least to a certain extent. Even the fact that the end-to-end models (see *e2e cmp*, *e2e lrg*) are not producing ever better results from approximately 30 dB SNR upwards is well reflected in both uncertainty estimations.

Intuitively, both uncertainty estimates seem to give satisfying results, and UCRT_1 seems to better reflect the relative performance between the setups.

3.2. Performance on Unseen SNR Conditions

With this experiment, we want to investigate the behavior of our uncertainty estimation methods under previously unseen SNR conditions compared to our baseline (see Section 2.4). In order to do so, we cannot re-use the models from the previous experiment, but have to create a new setup. Therefore, we train two neural networks, a full SNR range model and a high SNR range model using the large architecture (see Section 2.1). The high SNR range model is expected to perform worse on data corresponding to untrained SNR conditions, and will be helpful to determine the capabilities of the methods. We do the same thing for the baseline (see Section 2.4), except that we need to train five models for each ensemble.

The specific training regime for the two networks and the baseline is as follows. For training and validating the full SNR range models, we use all available ten different versions of each sample, representing SNRs from -20 dB to 70 dB in 10 dB steps. All in all, we use 27,000 and 10,000 samples for training and validation of the full range models, respectively. For training and validating the high SNR range models, we use just seven SNRs, ranging from 10 dB to 70 dB in 10 dB steps. All in all, we use 18,900 and 7,000 samples for training and validation of the high range model, respectively.

However, the unseen test data is identical to the original setting (see Section 2.2), and is normalized according to the training data respectively. As with the previous experiment, we compute the average loss and uncertainties for each SNR condition on the test data.

We start by showing a weakness of the baseline (see Section 2.4) in Fig. 5, where we plot the loss and uncertainty of the high SNR and the full SNR range models. For better comparison, we normalize the loss and uncertainty each to be between 0 and 1. As can be seen, the decreased loss corresponding to increased SNRs is well reflected for the high range model, but not for the full range model. This result seems counterintuitive at first, since the baseline performs well on unseen, and fails on previously seen conditions. The reason for that is, that even though the predictions are more corrupted under very low SNRs, they are getting more similar within the ensemble. This in turn leads to a relatively low variance of pixel values, which is reflected in the erroneous outcome of the baseline.

In Fig. 6, we can see that contrary to the baseline, both our uncertainty methods yield results that better reflect the actual loss curves.

3.3. Utilizing Uncertainty to Reject

In this experiment, we utilize the uncertainty estimates to implement a reject option. That is, we want to be able to reject samples with an expected loss above a specific threshold. To this end, we formulate a 2-class classification problem to distinguish *reject* and *no reject* categories. In order to end up with these categories, we fixed the threshold for the loss so that 30 % of the validation data results fall into the category *reject*.

The corresponding uncertainty threshold needs then to be fixed for each method and each model separately. From a practical point of view, different requirements may need to be fulfilled. Therefore, we decided to evaluate according to several uncertainty thresholds as follows. The uncertainty-based rejection should give maximum accuracy, equal error rate, a false positive rate of 5 %, and a false positive rate of 1 %. These four requirements lead to four different uncertainty thresholds, hence four evaluation results.

We present in Table 1 the results of the different scenarios on the validation data for all three methods. For each metric (precision, recall, f1-measure, and accuracy), the results on the full range model and the high range model are listed on the left and right column, respectively.

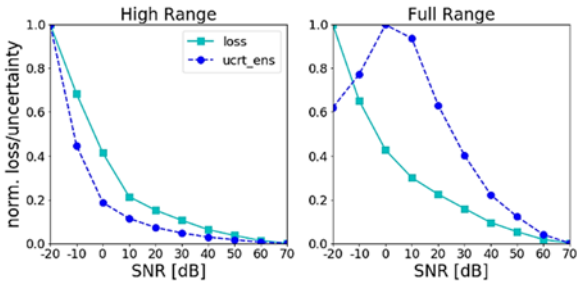


Fig. 5. Loss and uncertainty of the ensemble approach. Full Range: trained with all SNR conditions from [-20 to 70] dB; High Range: trained only with a subset ranging from [10 to 70] dB; This method reflects the decreased loss corresponding to increased SNRs only for the high range model.

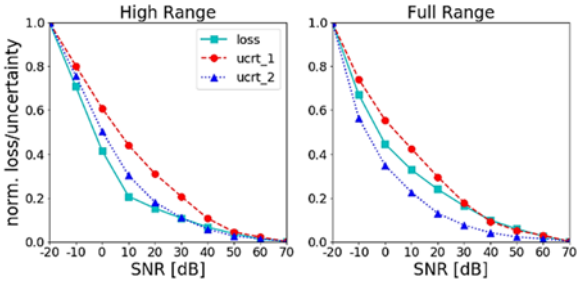


Fig. 6. Loss and uncertainty of our approaches. Compared to the ensemble approach in Fig. 5, both methods better reflect the decreased loss corresponding to increased SNRs.

Table 1. Results of the uncertainty based rejection on the validation set. UCRT_2 consistently outperforms the other methods.

	Precision		Recall		F1-Measure		Accuracy	
	full	high	full	high	full	high	full	high
Evaluation @Maximum Accuracy								
UCRT_ENS	.70	.84	.68	.73	.69	.78	.82	.88
UCRT_1	.73	.79	.60	.71	.66	.75	.81	.86
UCRT_2	.82	.85	.78	.80	.80	.82	.88	.90
Evaluation @Equal Error Rate								
UCRT_ENS	.64	.72	.80	.86	.71	.78	.80	.86
UCRT_1	.63	.69	.80	.84	.70	.76	.80	.84
UCRT_2	.75	.76	.87	.88	.80	.82	.87	.88
Evaluation @5% False Positives								
UCRT_ENS	.80	.86	.45	.70	.58	.77	.80	.88
UCRT_1	.79	.83	.44	.59	.57	.69	.80	.84
UCRT_2	.86	.87	.71	.76	.78	.81	.88	.89
Evaluation @1% False Positives								
UCRT_ENS	.89	.95	.19	.42	.31	.58	.75	.82
UCRT_1	.84	.88	.13	.17	.22	.29	.73	.75
UCRT_2	.95	.96	.46	.54	.62	.69	.83	.86

Interestingly, the full SNR range scenario seems to be more challenging for all uncertainty based rejection methods. This is due to the fact that the models trained only with high SNR ranges produce higher losses under the unseen, low SNR conditions (-20, -10 and 0 dB) compared to the models trained with the full

SNR range. Therefore, the average of the actual loss (MSE) of the 30 % that we categorize as reject tends to be higher compared to the full range scenario. This increased distance makes it easier to identify the samples to reject. The rejection based on method UCRT_2 seems to be the least affected by this phenomenon.

Additionally, the rejection based on UCRT_2 consistently outperforms the other methods in all regards, and on both the full and high range models. This suggests that for our specific task at hand, there is a correlation between the area of defects and the loss, since UCRT_2 factors this in.

The rejection based on method UCRT_1 gives results that are comparable to the baseline method UCRT_ENS when evaluated at maximum accuracy and equal error rate. When evaluated at a limited false positive rate, UCRT_1 gives inferior results compared to UCRT_ENS, especially for the high range model.

Finally, we keep the uncertainty thresholds and evaluate on the unseen test data set (see Section 2.1). In Table 2, the results of the different scenarios are listed for all three methods. In general, the test data results are very similar to the validation data results, suggesting good generalization capabilities. Consistently, the rejection based on UCRT_2 outperforms the other methods in all regards. Notice, that since we took the loss thresholds for the *reject* categorization from the validation data, the relative size of the class of interest *reject* is not exactly 30 %, but very close with 31.5 % and 30.6 % for the full and high SNR range scenarios, respectively.

Table 2. Results of the uncertainty based rejection on the unseen test set. In general, the results are very similar to the validation data results, and UCRT_2 consistently outperforms the other methods.

	Precision		Recall		F1-Measure		Accuracy	
	full	high	full	high	full	high	full	high
Evaluation @Maximum Accuracy								
UCRT_ENS	.69	.85	.69	.73	.69	.79	.80	.88
UCRT_1	.75	.80	.60	.71	.67	.75	.81	.86
UCRT_2	.84	.85	.79	.81	.81	.83	.89	.90
Evaluation @Equal Error Rate								
UCRT_ENS	.63	.73	.80	.86	.71	.79	.79	.86
UCRT_1	.65	.70	.80	.85	.72	.77	.80	.84
UCRT_2	.76	.77	.88	.90	.82	.83	.87	.89
Evaluation @5% False Positives								
UCRT_ENS	.79	.87	.47	.71	.59	.78	.79	.88
UCRT_1	.80	.84	.45	.59	.58	.70	.79	.84
UCRT_2	.87	.88	.71	.78	.78	.82	.88	.90
Evaluation @1% False Positives								
UCRT_ENS	.92	.96	.19	.43	.32	.59	.74	.82
UCRT_1	.86	.89	.12	.19	.22	.31	.72	.74
UCRT_2	.96	.97	.46	.55	.62	.70	.82	.86

4. Discussion

The consistency of the results on unseen data, unseen SNR conditions, different network architectures, and input data suggests good generalization capabilities. A possible direction of future work would be to assess our methods in different contexts. We think it could be useful for tasks in general, where the output can be converted into one or several binary masks, like semantic- or instance segmentation. Although, we consider it necessary to calibrate the α and β and hyperparameters to maximize the correlation with the loss first. The scope of application could even be extended towards a more general quality measurement that can also be applied to the results of the purely numerical methods.

Another interesting research direction would be dynamic model selection, where the best out of at least two models is dynamically selected based on their uncertainty estimates. Nevertheless, first experiments in that direction suggest that the difference in model performance needs to be substantial in order to reliably select the actual best model for each sample. Again, calibrating α and β might allow for a more fine-grained decision making. Therefore, implementing and evaluating an automatic calibration seems to be the logical next step to move our research forward.

5. Conclusion

In this paper, we presented two light-weight methods for uncertainty estimation of thermographic imaging results, and reported on three experiments. First, we demonstrated the high correlation of both methods with the actual loss (MSE) of different architectures (compact and large) and approaches (end-to-end and hybrid). Second, we intentionally limited the range of data in terms of SNRs during training in order to assess the performance on unseen, very low SNR conditions. While the ensemble-based baseline failed to give acceptable results on very low SNR conditions, both our methods exhibited the desirable behavior. Third, we showed how to use uncertainty to reject predictions that are expected to exceed a specific loss threshold.

In all conducted experiments, the ensemble based baseline was consistently outperformed by UCRT_2, and most often by UCRT_1. Another advantage of the proposed methods is, that just the result of a single forward pass is required, making them computationally less expensive than the ensemble-based baseline. Therefore, our uncertainty estimates are best suited for applications at the edge, even in real-time scenarios.

Acknowledgements

This work has been supported by Silicon Austria Labs (SAL), owned by the Republic of Austria, the Styrian Business Promotion Agency (SFG), the federal state of Carinthia, the Upper Austrian Research (UAR), and the Austrian Association for the Electric and Electronics Industry (FEEI); and by the University SAL Labs initiative of Silicon Austria Labs (SAL) and its Austrian partner universities for applied fundamental research for electronic based systems.

The financial support by the Austrian Federal Ministry of Science, Research and Economy and the National Foundation for Research, Technology and Development is gratefully acknowledged. Financial support was also provided by the Austrian research funding association (FFG) under the scope of the COMET program within the research project Photonic Sensing for Smarter Processes (PSSP) (contract number 871974). Parts of this work have been supported by the Austrian Science Fund (FWF), projects P 30747-N32 and P 33019-N.

References

[1]. P. Kovács, B. Lehner, G. Thummerer, G. Mayr, P. Burgholzer, M. Huemer, Deep learning approaches for thermographic imaging, *Journal of Applied Physics*, Vol. 128, Issue 15, 2020, 155103.

[2]. Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, in *Proceedings of the 33rd Int. Conf. on Machine Learning*, New York, NY, USA: PMLR, Vol. 48, 2016, pp.1050–1059.

[3]. T. Pearce, F. Leibfried, A. Brintrup, M. Zaki, A. Neely, Uncertainty in neural networks: Approximately Bayesian ensembling, in *Proceedings of the 23rd Int. Conf. on Artificial Intelligence and Statistics*, Vol. 108, 2020, pp. 234-243.

[4]. B. Lehner, T. Gallien, Uncertainty estimation for non-destructive detection of material defects with u-nets, in *Proceedings of the 2nd Int. Conf. on Advances in Signal Processing and AI (ASPAI)*, 2020.

[5]. N. Ida, N. Meyendorf, Eds. Handbook of advanced nondestructive evaluation, *Cham, Switzerland: Springer*, 2019.

[6]. V. Vavilov, D. Burleigh, Infrared Thermography and Thermal Nondestructive Testing, *Cham: Springer Int. Publishing*, 2020.

[7]. S. Roointan, P. Tavakolian, K. S. Sivagurunathan, M. Floryan, A. Mandelis, S. H. Abrams, 3D dental subsurface imaging using enhanced truncated correlation-photothermal coherence tomography, *Scientific Reports*, Vol. 9, Issue 1, 2019, p. 16788.

[8]. V. P. Vavilov, D. D. Burleigh, Review of pulsed thermal NDT: Physical principles, theory and data processing, *NDT & E Int.*, Vol. 73, 2015, pp. 28–52.

[9]. G. Thummerer, G. Mayr, P. D. Hirsch, M. Ziegler, P. Burgholzer, Photothermal image reconstruction in opaque media with virtual wave backpropagation, *NDT & E International.*, Vol. 112, 2020, p. 102239.

[10]. P. Tavakolian, K. Sivagurunathan, A. Mandelis, Enhanced truncated-correlation photothermal coherence tomography with application to deep subsurface defect imaging and 3-dimensional reconstructions, *Journal of Applied Physics*, Vol. 122, Issue 2, 2017, p. 023103.

[11]. A. Mendioroz, K. Martínez, R. Celorrio, A. Salazar, Characterizing the shape and heat production of open vertical cracks in burst vibrothermography experiments, *NDT & E International.*, Vol. 102, 2019, pp. 234–243.

[12]. S. D. Holland, B. Schiefelbein, Model-based inversion for pulse thermography, *Experimental Mechanics*, Vol. 59, Issue 4, 2019, pp. 413–426.

[13]. P. Burgholzer, M. Thor, J. Gruber, G. Mayr, Three-dimensional thermographic imaging using a virtual wave concept, *Journal of Applied Physics*, Vol. 121, Issue 10, 2017, 105102.

[14]. P. Burgholzer, G. Mayr, G. Thummerer, M. Haltmeier, Linking information theory and thermodynamics to spatial resolution in photothermal and photoacoustic imaging, *Journal of Applied Physics*, Vol. 128, Issue 17, 2020, p. 171102.

[15]. G. Mayr, G. Stockner, H. Plasser, G. Hendorfer, P. Burgholzer, Parameter estimation from pulsed thermography data using the virtual wave concept, *NDT & E International*., Vol. 100, 2018, pp. 101–107.

[16]. L. Szilard, Über die Entropieverminderung in einem thermodynamischen System bei Eingriffen intelligenter Wesen, *Zeitschrift für Physik*, Vol. 53, Issue 11-12, 1929, pp. 840–856 (in German).

[17]. R. Landauer, Information is physical, *Physics Today*, Vol. 44, Issue 5, 1991, pp. 23–29.

[18]. P. Kovács, B. Lehner, G. Thummerer, M. Günther, P. Burgholzer, M. Huemer, A hybrid approach for thermographic imaging with deep learning, in *Proceedings of the 45th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020, pp. 4277–4281.

[19]. G. Thummerer, G. Mayr, M. Haltmeier, P. Burgholzer, Photoacoustic reconstruction from photothermal measurements including prior information, *Photoacoustics*, Vol. 19, 2020, 100175.

[20]. B. Pan, S. R. Arridge, F. Lucka, B. T. Cox, N. Huynh, P. C. Beard, E. Z. Zhang, M. M. Betcke, Photoacoustic reconstruction using sparsity in curvelet frame, *arXiv arXiv:2011.13080*, 2020.

[21]. P. C. Hansen, Analysis of discrete ill-posed problems by means of the L-curve, *SIAM Review*, Vol. 34, Issue 4, 1992, pp. 561–580.

[22]. P. C. Hansen, Rank-deficient and discrete ill-posed inverse problems: Numerical aspects of linear inversion. Philadelphia, PA, USA: *SIAM Monographs on Mathematical Modeling and Computation*, 1998.

[23]. J. Eckstein, P. D. Bertsekas, On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators, *Mathematical Programming*, Vol. 55, Issue 1–3, 1992, pp. 293–318.

[24]. L. J. Busse, Three-dimensional imaging using a frequency-domain synthetic aperture focusing technique, *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, Vol. 39, Issue 2, 1992, pp. 174–179.

[25]. F. Lingvall, T. Olofsson, S. T., Synthetic aperture imaging using sources with finite aperture: Deconvolution of the spatial impulse response, *The Journal of the Acoustical Society of America*, Vol. 114, Issue 1, 2003, pp. 225–234.

[26]. J. A. Tropp, Just Relax: Convex Programming Methods for Identifying Sparse Signals in Noise, *IEEE Transactions on Information Theory*, Vol. 52, Issue 3, 2004, pp. 1030–1051.

[27]. J. A. Tropp, Greed is good: algorithmic results for sparse approximation, *IEEE Transactions on Information Theory*, Vol. 50, Issue 10, 2004, pp. 2231–2242.

[28]. O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, *in Proceedings of the Medical Image Computing and Computer-Assisted Intervention (MICCAI'2015)*, 2015, pp. 234–241.

[29]. M. D. Jenkins, T. A. Carr, M. I. Iglesias, T. Buggy, G. Morison, A deep convolutional neural network for semantic pixel-wise segmentation of road and pavement surface cracks, in *Proceedings of the 26th*

European Signal Processing Conference (EUSIPCO), Sep. 2018, pp. 2120–2124.

[30]. B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, *in Advances in Neural Information Processing Systems 30*, (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds.), *Curran Associates, Inc.*, 2017, pp. 6402–6413.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 36-44

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Image Enhancement in the LIP Framework and Noise Reduction with Deep Convolutional Neural Networks to Produce High Quality Images from Low Light Acquisitions

[1, *] Maxime CARRE and [2] Michel JOURLIN

[1] NT2I Company, 10 rue Jean Servanton, 42000 Saint-Etienne, France
[2] Hubert Curien Laboratory, 18 rue du Pr Lauras, 42000 Saint-Etienne, France
E-mail: m.carre@nt2i.fr

Received: 15 October 2020 /Accepted: 31 January 2021 /Published: 28 February 2021

Abstract: The LIP (Logarithmic Image Processing) model is recognized as an efficient framework to process images acquired in transmitted and reflected light, and to take into account the human visual system. Several image enhancement algorithms have been developed in the LIP framework. Some of them exploit an important property of the LIP model consisting in simulating exposure time variations. Applied to very low light images, our LIP algorithms enhance not only the signal, but also the noise and lead to quantized grey levels. Noise reduction based on Deep Convolutional Neural Networks is performed on these enhanced noisy images. The combination of LIP enhancement and DCNN denoising transforms low light images into well-balanced images with low noise level. In addition to classical PSNR evaluation, we propose an estimation of image noise based on LIP contrasts computed at a local scale. The use of LIP contrast results in a noise evaluation consistent with human vision.

Keywords: LIP model, Exposure time simulation, Image enhancement, Low light images, Noise reduction, Deep convolutional neural networks.

1. Introduction

This paper is an extended version of work presented in [1].

For the reader not familiar with the LIP framework, we propose the following references: [2-4], and more recently an overview book [6]. The LIP Model was first dedicated to images acquired in transmission (Fig. 1) when the observed object is located between the source and the sensor and its opacity and thickness may be variable. Such a situation is very common in the biomedical field (microscopy, tomography, scanner…).

We will see later why the Model is also useful for images acquired in reflection. The applications described in this paper are mainly about reflected light acquisition, but the proposed solution can be used in both configurations: transmitted and reflected light imaging.

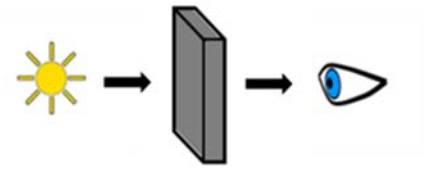


Fig. 1. Image acquired in transmittance: the source, the semi-transparent obstacle, and the sensor.

2. Basis of the LIP Model

Let us define D as the spatial domain. If f is an image defined on D with values in the grey scale $[0, M[$, the transmittance $T_{f(x)}$of f at $x \in D$ is the ratio of the transmitted flux at x by the incoming flux (intensity of the source):

$$T_{f(x)} = \frac{I_t(x)}{I_i(x)} \tag{1}$$

Note that within the LIP model, the grey scale is inverted: 0 corresponds to the "white" extremity of the grey scale, which means to the source intensity, i.e. when no obstacle (object) is placed between the source and the sensor. M corresponds to the "black" extremity.

In a mathematical formulation, the transmittance $T_{f(x)}$ is the probability, for a particle of the source incident at x, to pass through the obstacle that is to say to be seen by the sensor.

The addition $f \triangleq g$ of two images f and g is associated to the superposition of the objects generating f and g (Fig. 2).

Fig. 2. Addition of two images.

In this case, the transmittance law can be written:

$$T_{f \triangleq g} = T_f . T_g \tag{2}$$

A link has been established ([4]) between the grey level $f(x)$ and the transmittance $T_{f(x)}$ according to:

$$T_{f(x)} = 1 - \frac{f(x)}{M} \tag{3}$$

By replacing such transmittances values in (2), the following addition formula is obtained:

$$f \triangleq g = f + g - \frac{f.g}{M} \tag{4}$$

From (4), the product $\lambda \triangleq f$ by a real number λ is deducted:

$$\lambda \triangleq f = M - M(1 - \frac{f}{M})^{\lambda} \tag{5}$$

Such laws (4) and (5) possess interesting properties: at a physical level, they refer to the Transmittance Law and at a mathematical level, they give a structure of Vector Space to the set $I(D, [0, M[)$ of images defined on D with values in the grey scale $[0, M[$. This is due, in particular, to the possibility to define a subtraction operator $\triangle$:

$$f \triangle g = \frac{f - g}{1 - \frac{g}{M}} \tag{6}$$

Note that when f is greater (darker) than g for each point x of D, $f \triangle g$ takes its values in the grey scale and is then an image.

Plenty of new concepts and tools have been defined in the LIP framework, for example new notions of contrast (Logarithmic Additive Contrast, Logarithmic Multiplicative Contrast) and new metric tools (cf. [5] and [6] for more details and various applications).

Considering two points x and y of D and a grey level function f, the following notions of contrast are commonly used:

- The simplest one consists of computing the absolute value $|f(x) - f(y)|$ of the difference between $f(x)$ and $f(y)$.

- Knowing that the Human Visual System is not linear but of logarithmic nature, such a contrast is not adapted to model a Human Visual interpretation of an image. To give a higher weight to contrasts between dark points, a "physician" approach has been introduced, the Michelson contrast. If R is a subset of D, this contrast of f inside R, $C_R^{Mi}(f)$ is estimated as:

$$\frac{Max_{x \in R}\left(f(x)\right) - Min_{x \in R}\left(f(x)\right)}{Max_{x \in R}\left(f(x)\right) + Min_{x \in R}\left(f(x)\right)} \tag{7}$$

If the region R is restricted to a pair of neighboring points (x, y), $C_{(x,y)}^{Mi}(f)$ becomes:

$$\begin{gathered} \frac{|f(x) - f(y)|}{f(x) + f(y)} \text{ if } f(x) \neq 0 \text{ or } f(y) \neq 0 \\ 0 \text{ if } f(x) = f(y) = 0 \end{gathered} \tag{8}$$

Quotient in Formula (8) may be interpreted as the coefficient of variation of the pair $(f(x), f(y))$, i.e. the standard deviation divided by the mean value.

Inside the LIP framework, a similar notion, namely the Logarithmic Additive Contrast (LAC) between $f(x)$ and $f(y)$. $C_{(x,y)}^{\triangleq}(f)$ or $C_{(x,y)}^{\triangleq}$ when no confusion is possible, is obtained according to:

$$\begin{gathered} Max\big(f(x), f(y)\big) \triangle Min\big(f(x), f(y)\big) \\ = \frac{|f(x) - f(y)|}{1 - \frac{Min(f(x), f(y))}{M}} \end{gathered} \tag{9}$$

Remark: If $f(x)$ is noted f, and $f(y)$ becomes $f + \delta f$, the definition of Formula (9) shows that the LAC is expressed as:

$$M \frac{\delta f}{M - f} = MW \tag{10}$$

This result is comparable to the equality established in [3] between the Weber constant W and the quotient $\delta f\ /(\ M - f)$. Thus, we can assert that the LAC follows the Weber - Fechner law. This point justifies the use of the Logarithmic Additive Contrast to evaluate the visual quality of a denoising step.

To complete the previous remark, Brailean ([7]) demonstrated that the Model is consistent with Human Vision, which means that the LIP tools are successfully applicable to images acquired in reflection, particularly when we want to interpret those images as a human eye would do.

Physical interpretation of Formula (9): the LAC between two grey levels appears obviously as the grey level which must be added (i.e., superposed in transmitted light) to the brightest in order to obtain the darkest one.

The Michelson's contrast is explicitly linked to the LAC one according to the Formula (cf. [6]):

$$M.C^{Mi}(f(x), f(x) + 2k) = C^{\triangle}(M - f(x), M - f(x) - k) \quad (11)$$

Such a formula assigns a physical meaning to the Michelson's contrast. Moreover, the difference between the two considered grey levels must be doubled for Michelson: this explains why in our first experimentations the LAC appeared more sensitive than Michelson.

A useful application of the LAC consists of defining it locally. If N is a neighborhood of x, it is possible to compute the value of $C^{\triangle}_{(x,y)}$ for each y of N and then to take the maximum or the average of these values:

$$\max_{y \in N} C^{\triangle}_{(x,y)} \text{ or } \frac{1}{n} \sum_{y_i \in N} C^{\triangle}_{(x,y_i)} \quad (12)$$

This application can be used for contour detection purposes, or in the present work, for noise evaluation. By computing the average value of every local contrasts, we get an estimation of the global noise level inside the image.

3. Image Enhancement in the LIP Framework

One of the most significant properties of the LIP model consists of simulating exposure time variations ([8]). Indeed, it has been demonstrated that LIP addition and subtraction can be used to simulate variations of source intensity or sensor's sensitivity.

Physically, a LIP addition of an image by a constant value is equivalent to placing a homogenous absorbing medium (in addition to the observed object) between the source and the sensor and thus reducing the incoming intensity of the source. It may be interpreted also as a reduction of sensor's sensitivity. The exposure time is an evident camera setting able to modify this sensitivity.

In both interpretations, the effect will be the same on the resulting image: it will be darkened compared to the image generated without this added absorbing medium.

At the opposite, a LIP subtraction of an image by a constant value is equivalent to remove a homogenous part of the observed object. In analogy with the LIP addition, this removal increases the source intensity or the sensor's sensitivity. Then the resulting image will be a brightened version of the image acquired without this removal (Fig. 3).

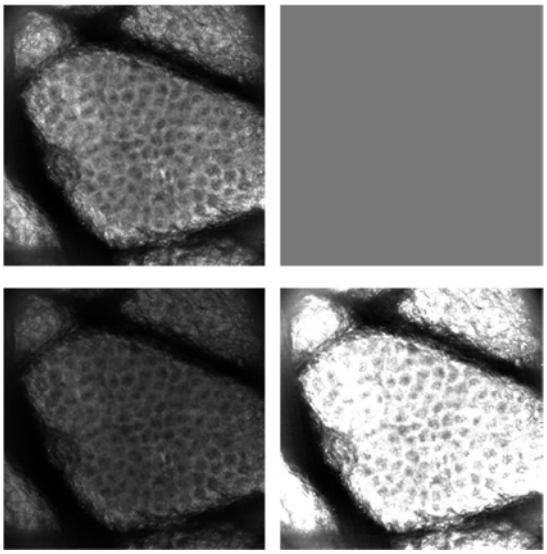


Fig. 3. LIP addition and subtraction. Top left: image acquired in transmitted light, top right: uniform image with grey level 128, bottom left: LIP addition of the original image and the uniform image, bottom right: LIP subtraction of the original image by the uniform image.

These principles are defined in a transmitted light configuration. In a reflected light configuration, it has been demonstrated [8] that the simulation of sensor sensitivity, exposure time for instance, by LIP addition/subtraction is also well adapted (Fig. 4).

Let us return in transmitted light imaging to understand the effect of a LIP multiplication of an image.

Physically, the LIP product of an image by a scalar λ is a homothety which equates to modify the thickness of the observed object. Using $\lambda > 1$ permits to increase the thickness while using $\lambda < 1$ decreases it. Obviously, a reduced thickness increases the brightness of the resulted image (the incoming flux is less filtered) and a larger thickness darkens the resulted image (the incoming flux is more filtered).

In reflected light conditions, this operation has no evident interpretation. But the non-linearity of the LIP multiplication and its isomorphism property (it applies the grey scale [0,255] on itself: the output values cannot exceed the grey level range) can be efficient for image enhancement (Fig. 5). It can be used as a tone mapping operation to improve the display of high dynamic images.

Fig. 4. Simulation of exposure time variation. Left: image acquired at 10 ms, right: simulation of 100 ms exposition applied to the image acquired at 10 ms.

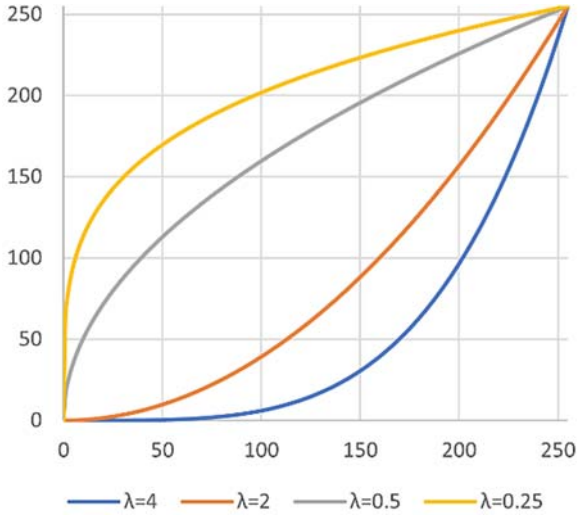


Fig. 5. Look Up Tables between input and output grey levels for multiple λ values.

Some algorithms based on LIP multiplication have been developed. They aim at computing λ automatically to optimize an image parameter, for example, maximizing the standard deviation of the image histogram ([6], chapter IV) or stabilizing the brightness to a specific value.

An example of image enhancement using LIP multiplication is proposed in Fig. 6, for a color image. To process color images, the same operations are applied on each channel R, G, B to guarantee the conservation of real colors (same λ for each channel). Color images can also be processed with the LIPC model [9] considering more specifically human vision.

Another LIP enhancement algorithm directed to local tone mapping has been developed ([10]) using LIP subtraction. The principle of this algorithm is to simulate local variations of exposure time and aims at stabilizing spatially the brightness of the image to a minimum value. Dark regions are enhanced while bright regions correctly balanced are kept untouched. The local brightness is modelized using a Bilateral Filtering. In this filtering, the classical grey level distance has been replaced by a LIP additive contrast producing a constant filtering, independent of lighting variations.

The use of LIP subtraction, simulating exposure time variation aims at producing a realistic image enhancement. An example is proposed in Fig. 7.

Fig. 6. LIP multiplication. Left: night street image, right: image enhanced by LIP $\triangle$ to maintain an average of 100 (grey level) on the luminance plane.

Fig. 7. Spatial stabilization. Left: image of a street with a strong shadow, right: image enhanced with local exposition variations.

4. Problem: Noise and Quantization

Let us remember that whatever the enhancement method, image noise is amplified as well as image information. Strong noise can be a discomfort for a human observer and source of difficulties for many image analysis algorithms.

In particular, such a problem arises when applying the exposure time simulation to low-light images in order to enhance them.

On Fig. 9, we apply on a sequence of images acquired at different exposure times (Fig. 8) a brightness stabilization algorithm based on LIP exposure time simulation ([10]). In this example, each image is stabilized to a reference brightness corresponding to the image acquired with the higher exposure time, here 50 ms. Obviously, the noise level increases as the exposure time decreases.

Such a problem damages the quality of the enhanced image and necessitates the application of a noise filtering more or less adapted to the considered images.

Another issue concerns the quantization of the grey level scale. Since the original dark image presents a compressed histogram, our enhancement leads to a quantized histogram (Fig. 9). In the case of strong exposure time correction, this quantization can be visible by a human observer.

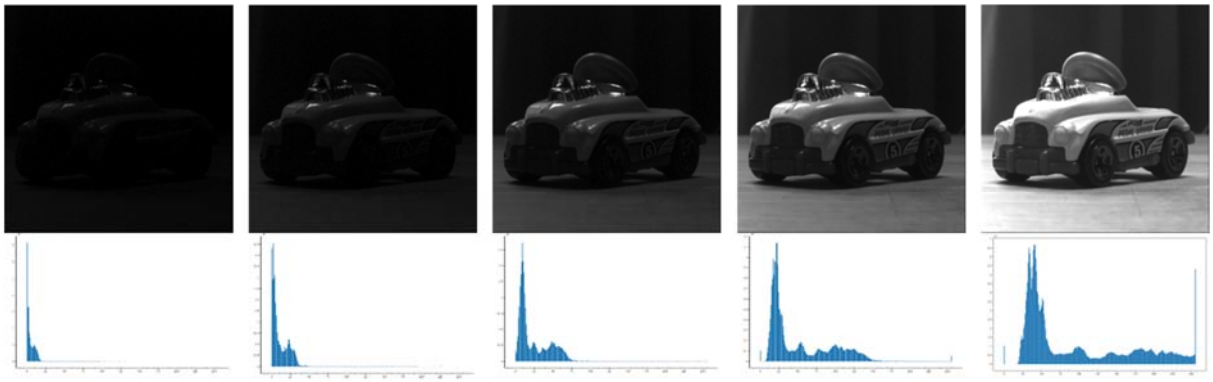


Fig. 8. Images acquired at different exposure times, from left to right: 0.75 ms, 2.22 ms, 6.67 ms, 20 ms, 50 ms.

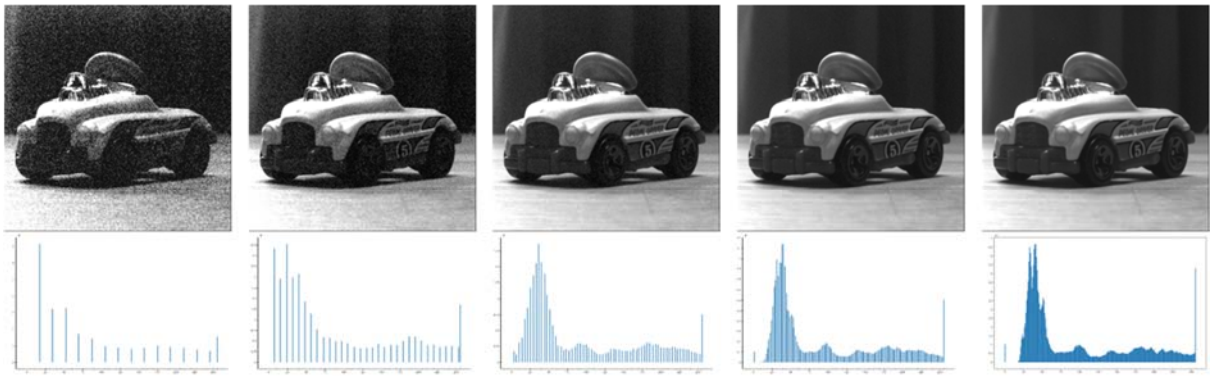


Fig. 9. LIP stabilization applied on Fig. 1 images.

5. Proposed Solution

To overcome those drawbacks, we decided to apply a Deep Learning approach, according to [11]. This approach consists of restoring noisy images by training a convolutional neural network. The proposed network is based on a U-network shape ([12]) and the particularity of [11] lies in the use of only corrupted couples of images in the training phase, instead of classical couples constituted of a noisy image and a clean one.

In our situation, this particularity greatly facilitates the preparation of images databases required in machine learning methods. Generating thousands of noisy/clean images would require an acquisition system able to switch quickly between two exposure times (a low exposure time for the noisy image, and a higher one for the clean target). Creating sequences of noisy images acquired in the same conditions is much easier and makes the solution adaptable to many acquisition systems.

In order to process enhanced images presenting different noise intensities as described in Fig. 9, we train the network with images acquired under different brightness conditions and variable exposure times.

6. Results

Fig. 10 and Fig. 10 (b) show the application of the trained network on the images of Fig. 9. In each situation, the noise has been significantly reduced. This visual estimation is confirmed by Table 1, presenting the evolution of PSNR computed for each exposure time on images processed with LIP enhancement only and processed with the combination of LIP enhancement and DCNN (Deep Convolutional Neural Networks) denoising.

Table 1. PSNR according to exposure level.

Expo. (ms)	**0.75**	**2.22**	**6.67**	**20**	**50**
LIP	19.3	23.4	28.8	31.5	clean
LIP denois.	25.4	27.2	33.3	37.6	clean

In addition to the PSNR evaluation, we propose to estimate the noise level of images using LIP tools. As explained above in Section 2, the average value of LIP

contrasts computed in each pixel with its neighboring points represents an estimation of the intensity of image noise. In comparison to classical grey levels distance, using LIP additive contrast aims at interpreting grey levels closer to human sensitivity.

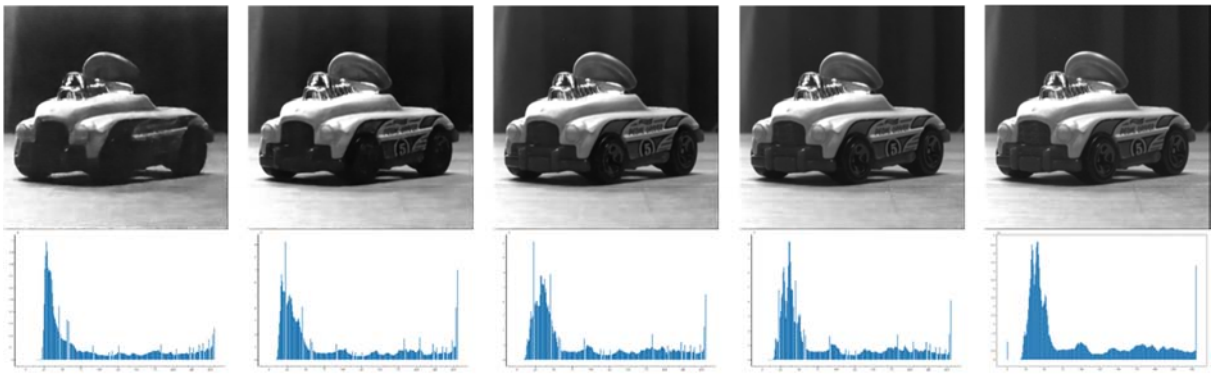
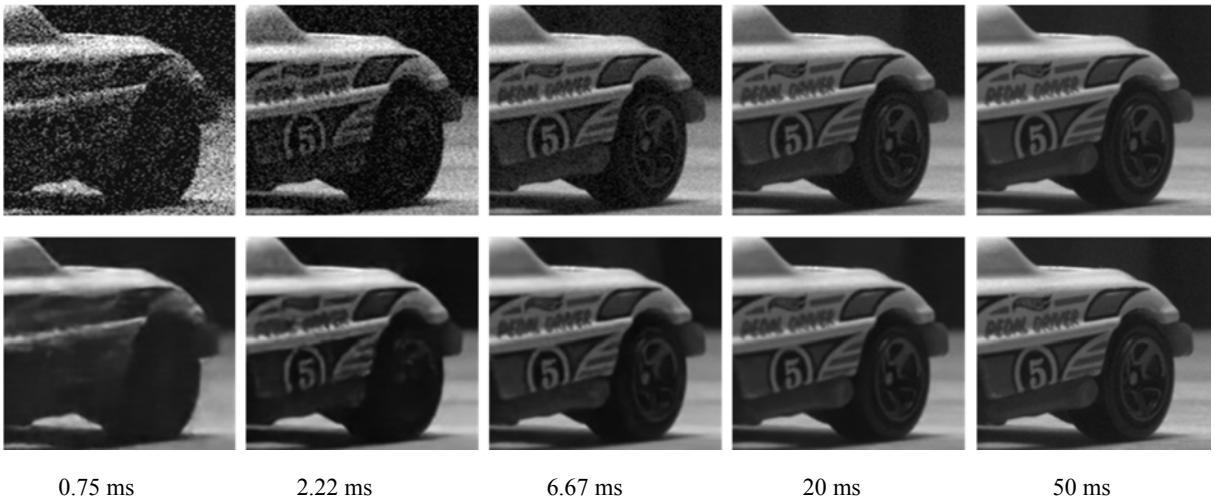


Fig. 10. Noise and quantization correction on Fig. 2 images.



0.75 ms 2.22 ms 6.67 ms 20 ms 50 ms

Fig. 10 (b). Top: Zoom on images acquired at different exposure times processed with LIP stabilization, bottom: zoom on images acquired at different exposure times processed with LIP stabilization and noise reduction.

Table 2 presents LIP noise estimations based on this average of LIP contrasts computed at a local scale.

Unlike the PSNR evaluation comparing images to a clean reference, each image gets an absolute noise level (without reference). That is why a noise level is also proposed for the clean target in Table 2.

Table 2. Average LIP contrast according to exposure level, values in the grey level scale [0, 255].

Expo. (ms)	**0.75**	**2.22**	**6.67**	**20**	**50**
LIP	65.5	58.5	29.7	14.6	10.5
LIP denois.	7.2	8.8	8.4	7.7	7.4

Each denoised image gets a noise intensity inferior to 10 grey levels, while original images get values between 65.5 and 10.5.

It is interesting to note that the denoised version of the originally noisiest image (0.75 ms exposure time) gets a lower noise value than the clean target. This is due to the loss of details of this extreme case, resulting in a "simplified" image. Some information could not be recovered from the original very dark image (Fig. 10 (b)).

We can notice also that the image corresponding to the clean target (without the application of the denoising step) gets a noise level superior to all the denoised ones. By looking closely to its details, it can be explained by the "image grain": even the clean target gets a minimum noise level, which can be reduced by the denoising network.

Visually, starting from the 6.67 ms exposure time, the resulting images are very close to the clean target despite a slightly blurred rendering. This blur effect is a frequent issue in denoising problematics (see zoomed images in Fig. 10 (b)).

In our case, the benefits of the network are twofold: as expected the noisy signal is cleaner, and in addition

the quantization of the signal is corrected. Indeed, due to the last layer of the neural network presenting a linear activation, the output signals get continuous values, while every trained one were quantized. This point appears clearly by comparing the histograms of Fig. 9 and Fig. 10.

7. Applications

The proposed solution, LIP brightness stabilization and noise reduction with DCNN, can be directly applied to color images by learning RGB noisy images. An example is shown on Fig. 11 and Fig. 11 (b).

The adaptation of the network to different noise intensities allows its application to images enhanced by another LIP algorithm performing brightness spatial stabilization ([10]), described in Section 3. This algorithm consists of simulating local variations of exposure time to process high dynamic images. The brightness is locally adjusted to enhance only the dark areas. This local correction leads to different noise intensities within a single image. Since the network has been trained under different brightness conditions, its application in this situation is well adapted.

An example is proposed in Fig. 12 and Fig. 12 (b). Regions initially bright and with a very low level of noise are almost untouched: the brightness is not modified but low noise can be slightly corrected as described in Section 6. Dark regions are much more modified: the brightness is enhanced, increasing the noise, which is significantly reduced by the denoising network. Finally, the resulting image is well balanced in every region with low levels of noise. A limit of the solution is related to too dark areas in which information cannot be retrieved.

Fig. 11. Application on a color image. From left to right: low exposure acquisition (2 ms), LIP stabilization applied on the low exposure image, noise reduction applied on the enhanced image, high exposure acquisition (clean target, 50 ms).

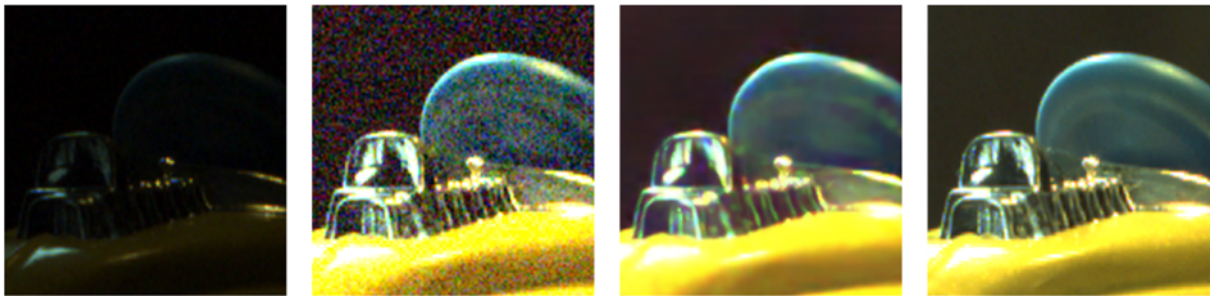


Fig. 11 (b). Zoom on Fig. 11.

Fig. 12. Brightness spatial stabilization. From left to right, low exposure acquisition (5 ms), higher exposure exposition (50 ms) leading to burned areas, LIP local stabilization applied on the low exposure image, noise reduction applied on the enhanced image.

Fig. 12 (b). Zoom on Fig. 12.

8. Discussion

In the present paper, we propose a novel solution that focuses on enhancing very low-light images in the LIP framework. By adding a denoising process based on DCNN to LIP enhancement algorithms, very dark images or complex images with dark and bright regions can be corrected to stabilize the brightness and reduce drastically the noise level.

Using some optimizations [13], the denoising network, based on a U-Net architecture can be run in real time, as well as most of the LIP enhancement algorithms.

Other low light imaging applications can be processed by the proposed solution. For example, image acquisition of moving objects using standard exposure times lead to blur results. This is commonly the case for sports meetings, traffic surveillance and for industrial control when objects are transported by means of a conveyor. In such situations, the solution to acquire clear images consists of drastically reducing the exposure time, which produces low-light images.

We have already mentioned ([14]) the ability of the LIP Model to enhance such images. Nevertheless, at this time we had not applied the denoising step.

We plan also to develop the combination LIP tone mapping algorithm and DCNN denoising to compare HDR (High Dynamic Range) image generated from multiple exposure images and HDR image generated from a single exposure image enhanced and denoised.

References

[1]. M. Carre, M. Jourlin, Exposure Time Simulation thanks to the LIP (Logarithmic Image Processing) Model with Noise Reduction by Deep Convolutional Neural Networks, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI'2020),* 2020, pp. 46-49.

[2]. M. Jourlin, J.-C. Pinoli, Logarithmic Image Processing, *Acta Stereologica*, Vol. 6, 1987, pp. 651-656.

[3]. M. Jourlin, J.-C. Pinoli, A model for Logarithmic Image Processing, *Journal of Microscopy*, Vol. 149, Issue 1, 1988, pp. 21-35.

[4]. M. Jourlin, J.-C. Pinoli, Logarithmic Image Processing: The mathematical and physical framework for the representation and processing of transmitted images, *Advances in Imaging and Electron Physics*, Vol. 115, 2001, pp. 129-196.

[5]. M. Jourlin, M. Carre, J. Breugnot, M. Bouabdellah, Logarithmic image processing: additive contrast, multiplicative contrast, and associated metrics, *Advances in Imaging and Electron Physics*, Vol. 171, 2012, pp. 357-406.

[6]. M. Jourlin, Logarithmic Image Processing: Theory and Applications, *Advances in Imaging and Electron Physics*, Vol. 195, 2016, pp. 2-259.

[7]. J. Brailean, B. Sullivan, C. Chen, M. Giger, Evaluating the EM algorithm for image processing using a human visual fidelity criterion, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP),* Vol. 4, 1991, pp. 2957–2960.

[8]. M. Carre, M. Jourlin, LIP operators: Simulating exposure variations to perform algorithms independent of lighting conditions, in *Proceedings of the International Conference on Multimedia Computing and Systems (ICMCS)*, 2014, pp. 122–126.

[9]. M. Jourlin, J. Breugnot, F. Itthirad, M. Bouabdellah, B. Closs, Logarithmic Image Processing for Color Images, *Advances in Imaging and Electron Physics*, Vol. 168, 2011, pp. 65-108.

[10]. M. Carre, M. Jourlin, Brightness spatial stabilization in the LIP framework, in *Proceedings of the 14th International Congress for Stereology and Image Analysis (ICSIA)*, *Acta Stereologica,* 2015.

[11]. J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, T. Aila, Noise2Noise: Learning Image Restoration without CleanData, in *Proceedings of the 35th International Conference on Machine Learning,* Stockholm, Sweden, PMLR 80, 2018, pp. 2965-2974.

[12]. O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, *Medical Image Computing and Computer-Assisted Intervention (MICCAI*), Vol. 9351, 2015, pp. 234–241.

[13]. D. Brittain, Creating performant DL models for use in client applications, in *Proceedings of the GPU Technology Conference (GTC 2020)*, 2020.

[14]. V. Deshayes, P. Guilbert, M. Jourlin, How simulating exposure time variations in the LIP model. Application: moving objects acquisition, in *Proceedings of the 14th International Congress for Stereology and Image Analysis (ICSIA)*, *Acta Stereologica,* 2015.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 45-53

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Lessons Learned on Conducting Dwelling Detection on VHR Satellite Imagery for the Management of Humanitarian Operations

[1, *] Lorenz WICKERT, [1] Manfred BOGEN and [1] Marvin RICHTER

[1] Fraunhofer IAIS, Schloss Birlinghoven 1, 53757 Sankt Augustin, Germany
[1] Tel.: +49 2241 14-3415, Fax: +49 2241 14-2342
E-mail: lorenz.wickert@iais.fraunhofer.de

Received: 27 November 2020 /Accepted: 29 December 2020 /Published: 28 February 2021

Abstract: The management of humanitarian operations in highly intense situations like migration movements happening at borders often lack current and sufficient information. Satellites do provide large-scale information fast. When dealing with a migration situation, satellite images now can give information about where refugees are before they arrive at a border, giving first responders urgently needed lead time for contingency and capacity planning. Dwelling Detection, a method conducted on satellite images of refugee camps, is able to count the dwellings in a camp. From that, the number of inhabitants in a camp can be derived for forecasting purposes. To count the dwellings, object detection machine learning methods can be used. In Wickert et al. [ASPAI' 2020, **1**, 2020], a dwelling detection workflow using a Faster R-CNN is described. The workflow contains a newly developed annotation method, an inhabitant estimate for analyzed camps and a global confidence factor indicating the quality of the analysis of the image and the estimate of the inhabitants. In this actual extension of Wickert et. al. [ASPAI' 2020, **1**, 2020], lessons learned from multiple training and testing runs are documented, following a detailed analysis of those tests and validations in Wickert et. al. [ISPRS 2020, **2**, 2020]. In this extended article we conclude that the workflow produces results that can be used in humanitarian operations. We further document our lessons learned in developing a dwelling detection workflow and we provide recommendations for training a dwelling detection classifier. We advise humanitarian operators to build a dwelling detection classifier following our recommendations and use satellite images in actual humanitarian operations. This approach can reduce stress for all people involved in a humanitarian (crisis) situation and lead to better decisions in intense migration situations.

Keywords: Artificial Intelligence (AI), Machine Learning (ML), Deep Learning, Convolutional Neural Network (CNN), Remote Sensing (RS), Dwelling Detection, Object Detection, Lessons Learned.

1. Introduction

Humanitarian operations, with migration management in particular can be complicated and time-critical tasks. Often migration movements allow only a short reaction time due to their high intensity. To ease the work of humanitarian operators, a dwelling detection workflow was developed in [1] to automate parts of the information gathering process. In the paper, the development of the dwelling detection workflow was motivated as following:

"Forced Migration is one of the most pressing issues of society today. The United Nations High Commissioner for Refugees reported that in 2018 'the worlds forcibly displaced population remained yet again at a record high' [3]. This leads to highly intense humanitarian operations like the migration movements in Europe in 2015/16, where humanitarian

operations were challenging to all people involved because of a short reaction time and a lack of sufficient information. Satellite data allows the monitoring of large areas of the earth in a short time. Together with machine learning technology, satellite data can be used to yield large-scale information almost immediately, supporting decision making in humanitarian operations. In this paper, advances in satellite sensors [4, p. 187ff] and object detection using machine learning [5] are used to speed up the task of 'dwelling detection' [6]. This is a fast method to get information on the number of inhabitants of camps when this information is hard to get otherwise, either because those numbers are not available or there is no communication among the different parties, authorities and organization engaged in a humanitarian situation."

After developing the initial dwelling detection workflow, extensive tests were conducted on the workflow's core component, the object detection model. More precisely, different configurations of the annotation workflow, different representations of the input images into the object detection model and the generalization of the workflow were tested. From the results of these tests, recommendations for training a dwelling detection classifier were derived. A detailed analysis of the results is reported in [2]. In this article, we extend the description of the initial dwelling detection workflow by reporting what was tested, the results of the tests as well as the recommendations derived from those tests.

The research described in this paper was conducted in the context of the HUMAN+[1] project. In the HUMAN+ project, a real-time situational awareness system for efficient management of migration movements was developed. Besides the dwelling detection module, camera streams from cameras located at borders were analyzed, social media platforms were monitored and evaluated concerning migration movement and reports from operators in the field were collected and included in the HUMAN+ situational awareness system.

2. Methodology

In the following chapter, the methodology for building the dwelling detection classifier is documented in Sections 2.1 – 2.5. These sections are cited from [1]. In Section 2.6, the methodology used for testing the annotation workflow and the dwelling detection classifier is described, following [2].

2.1. Dwelling Detection

Due to the fact that individual humans cannot be seen on commercially available satellite images, a method for extracting valuable information about humans from satellite imagery had to be defined. dwelling detection offers a fitting method. It is conducted on very high resolution (VHR) satellite images of refugee camps. Those satellite images are available as result of regular satellite operations and recordings. Dwellings are counted and multiplied with an average, size-based factor of how many people live in one dwelling, giving an estimate over how many people may live in a camp and may eventually arrive at a border. This information can be used in medium-term planning of humanitarian operations during migration situations. Reasons why these information are not available for humanitarian operators: camps are run by private companies [7], camps do not develop as planned because of a rapid growth of inhabitants [8] or because camps are makeshift camps which are created arbitrarily, sometimes called "jungles" [7], [9].

2.2. Deep Learning Object Detection

Dwelling detection is traditionally done by experts by hand, taking a lot of time which is not available in crisis situations. Recent advantages in image classification and object detection on images using Convolutional Neural Networks (CNN) allow first approaches to automate this task [10], [11]. A Faster R-CNN [12] offers a state-of-the-art CNN architecture for object detection to eventually develop a dwelling detection workflow. To do so, a complete machine learning workflow (see Fig. 1) was created, following a framework of preparing input data, defining the expected output data and building a core network which constructs the intrinsic and natural relationship of the input-output pair [13].

The training workflow consists of three main steps. In the first step, satellite images annotated by hand are used to prepare ground truth training data consisting of a train- and a validation-set.

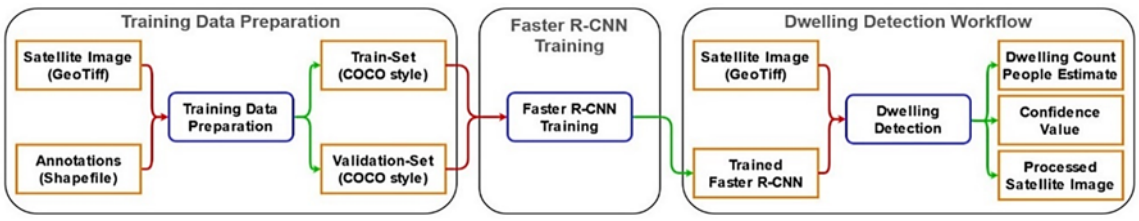


Fig. 1. The machine learning workflow of dwelling detection on remote sensing satellite images. Figure taken from [1].

[1] https://giscience.zgis.at/human/ (13N14716-13N14723)

Those sets are used to train a Faster R-CNN in the second step. In the last step, the trained Faster R-CNN is used to analyze a new satellite image of a camp. The classifier outputs a dwelling count and a people estimate, a confidence value concerning the Faster R-CNN analysis and the processed input satellite image with found dwellings marked.

2.3. Training Data Preparation

The input data in a dwelling detection task is VHR satellite imagery of refugee camps identified at this point in time by a human being.

In this work, images were taken from Google Earth (Google Inc.)[1] as examples and due to the fact that free satellite images were not available. On those, the dwellings forming a camp have to be found by an object detection algorithm like Faster R-CNN and marked with a bounding box.

A Faster R-CNN is trained using supervised learning. The ground truth data required to train the network consists of bounding boxes around each dwelling. Further, the size of the input images needs to be small enough to be efficiently handled by a Faster R-CNN. Therefore, the input satellite image is split up into 300 pixel × 300 pixel tiles. For speeding up the annotation process, a workflow building up on a seeded region growing algorithm [14], was developed (see Fig. 2). Each dwelling needs to be point-annotated by a human; then each annotation is used by the region growing algorithm as a seed to estimate the extent of one dwelling. The results of the region growing algorithm are then filtered to eliminate bad regions. Around each region, a bounding box is defined and stored in the MS COCO[2]-annotation format [15].

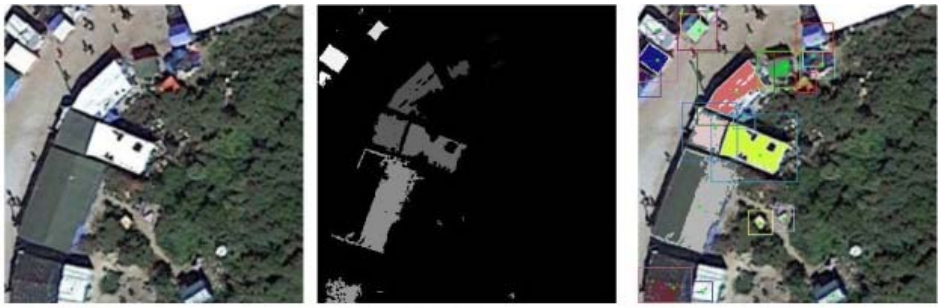


Fig. 2. To create training data, the initial tile containing point annotations (a) is analyzed using a region fill algorithm (b) resulting in ground truth boxes (c) around dwellings. Figure taken from [1].

2.4. Faster R-CNN Training

Refugee settlements inhabit a huge degree of variety due to their characteristics discussed in Section 2.1. To achieve a network generalizing well, this variety has to be represented in the training data [10]. Therefore, annotations for the training set were made on satellite images of nine different refugee camps. The images have a huge variation in the landscape surrounding the camps, the organization form of the camps, the size of the camps and the quality of the images as a result of the recording distance from the satellite and the satellites optical sensors. On each image, annotations on the upper 50 % of the input image were added to the training set, annotations on the lower 50 % of the input image were added to the validation set.

For implementation and training of the Faster R-CNN an open source implementation was used[3]. A Faster R-CNN consists of a pre-trained backbone network for feature extraction, a Regional Proposal Network (RPN) for generating object proposals, and a classification network outputting bounding boxes around found objects and a class vector for each bounding box. As a backbone, ResNet50 [16] was used. The network was trained on one GPU for 36,000 iterations.

2.5. Dwelling Detection Workflow

To analyze a satellite image using the trained Faster R-CNN, a copy of the input image, which is cut in various 300 pixel × 300 pixel tiles to handle its size, is made. Each tile is analyzed separately. Found objects are accepted as a dwelling if their class security for being a dwelling is higher than a defined threshold.

The dwelling detection workflow has two outputs: The first output is a copy of the input satellite image with all found dwellings marked with a bounding box (see Fig. 3). The second output consists of the number of found dwellings and the estimated number of inhabitants in a camp. The number of inhabitants is estimated with regard to the size of the found bounding

[1] https://www.google.com/intl/de_de/earth/
[2] https://cocodataset.org/#home
[3] https://github.com/facebookresearch/Detectron

boxes. Problematic is that the spatial resolution of one pixel can change from image to image and is not always known. Therefore, the estimation algorithm has to be independent of this information. This is achieved by assigning a fixed number of inhabitants to the smallest and the largest dwelling found by the Faster R-CNN. For the dwellings with sizes between those anchor points, the number of inhabitants in a tent is interpolated linearly.

Further, a confidence metric indicating the certainty of the network about the results of the analysis is calculated. The metric is the ratio of the number of found objects $O_{cls \geq 0.85}$ with a class security higher or equal than 0.85 divided by the number of objects $O_{cls \geq 0.5}$ with a class security higher or equal than 0.5:

$$\text{Confidence} = \frac{O_{cls \geq 0.85}}{O_{cls \geq 0.5}} \quad (1)$$

Fig. 3. The output of the Faster R-CNN. The input satellite image of a refugee camp located in Idomeni (Greece) was analyzed. The red bounding boxes mark the dwellings found by the Faster R-CNN. Figure taken from [1].

2.6. Test and Validation Workflow

The main idea of the test and validation workflow is to train various instances of the Faster R-CNN using different configurations. The resulting Faster R-CNNs are evaluate on uniquely configured evaluation sets. The performance of the resulting Faster R-CNNs on the different evaluation sets is used to derive an assertion about the performance of the training configurations. The main information derived is how well the individual networks learn the training distribution as well as how well they generalize their detection capabilities to unseen satellite imagery.

While training the Faster R-CNNs, three parameters were varied:

1. The color-representation of the images in the training sets with possible values “color”, “grayscale” and “color & gray”;

2. The color-representation of the images which were given to the annotation workflow as input with the possible values “color” and “grayscale”;

3. And the number of iterations the Faster R-CNNs were trained.

Table 1 shows the resulting eight trained networks, where the networks with id 1-5 were discussed in [2] and the networks with id 6-8 have not been discussed before.

Table 1. Training configurations of the networks tested. Adapted table taken from [2].

Id	Conf	Training on	Annotations on	Train Time
1	n_cc_36k	Color	Color	36.000 (Iter)
2	n_gc_27k	Grayscale	Color	27.000 (Iter)
3	n_gc_36k	Grayscale	Color	36.000 (Iter)
4	n_gg_27k	Grayscale	Grayscale	27.000 (Iter)
5	n_gg_36k	Grayscale	Grayscale	36.000 (Iter)
6	n_cg_27k	Color	Grayscale	27.000 (Iter)
7	n_mix_27k	Color & Gray	Color	27.000 (Iter)
8	n_mixg_27k	Color & Gray	Grayscale	27.000 (Iter)

The different networks were tested on two sets of validation sets with four validation sets each. In the first set of validation sets, the images contained are those used to train the networks. In the second set of validation sets, new images not seen by the networks before, but of the same camps the networks were trained on, are contained. The four individual validation sets inside one set differentiate in the color-representation of the images inside each set and the color-representation of the input-images in the annotation algorithm. The complete configuration of each validation set is documented in Table 2. Validation sets with id 1-3 & 5-7 were discussed in [2], validation sets with id 4 & 8 have not been discussed before.

Table 2. Configurations for the Validation Sets the different networks were trained on. Adapted table taken from [2].

Id	Conf	Type	Images	Annotations on
1	v_orig_cc	Trained on	Color	Color
2	v_orig_gc		Grayscale	Color
3	v_orig_gg		Grayscale	Grayscale
4	v_orig_cg		Color	Grayscale
5	v_new_cc	Not Seen	Color	Color
6	v_new_gc		Grayscale	Color
7	v_new_gg		Grayscale	Grayscale
8	v_new_cg		Color	Grayscale

As performance metrics, Average Precision (AP) and Average Recall (AR) were calculated using the MS COCO Detection evaluation metrics [17]. Further the F1-score [18] was calculated for each network on all eight validation sets.

3. Results

In [1] it was outlined, that "a complete machine learning workflow was implemented. For training data creation an annotation method using point annotations which are processed using a seeded region growing algorithm to generate ground-truth data was developed. This method allows creating a sufficient amount of ground-truth bounding boxes in a short amount of time".

The results for the initial network used in the conference paper, corresponding to network *n_cc_36k*, are reported in Section 3.1 and the results regarding the dwelling detection performance are discussed in Section 3.2, both citing [1]. Section 3.3 discloses the results of the tests and validations described in Section 2.6, following [2].

3.1. Machine Learning Results

The Faster R-CNN *n_cc_36k* was trained and tested on nine satellite images of refugee camps with different sizes from different parts of the world on different landforms and with varying image quality. On the validation set, comprising of ground truth data from all of the nine used satellite images, a mean Average Precision (mAP) of 68.2 % and a mean Average Recall (mAR) of 72.0 %, determined using the MS COCO detection evaluation metrics [17], was achieved. It can be concluded that the network was able to construct the relationship between the input-satellite image and the dwellings appearing on the image. This holds true for the huge variety of camps the network was trained on.

3.2. Dwelling Detection Results

The trained Faster R-CNN finds objects on the analyzed image. To estimate the number of inhabitants in a camp the workflow described in Section 2.5, which is independent of the spatial resolution of the input satellite image and therefore works without image-specific configurations, is used. For the estimation, the number of inhabitants in the smallest dwelling was set to three and the number of inhabitants in the largest dwelling was set to twelve. These limits were set based on an information from the HUMAN+ project partner Johanniter Austria, saying that around ten people live in a standard tent in a transit camp and around six people live in a standard tent in a permanent camp. The threshold for a found object being accepted as a dwelling was set to a class security of 0.85, following [11].

To evaluate the estimate numbers, real-world numbers of inhabitants for each camp, measured at around the time the satellite image was shot, were researched in newspaper articles. It has to be noted that these numbers are not official numbers and are therefore fuzzy. Further, the estimations are based on no concrete knowledge of an individual camp. Therefore, the absolute numbers cannot be assumed as the real numbers but function as an early warning system, stating an order of magnitude of the number of inhabitants.

Consequences from the result of the workflow have to be taken in accordance with the computed confidence metric and the visual output of the network. A look on the annotated satellite image produced by the workflow can give a first impression on how successful the analysis was. Further, the confidence metric makes a statement about how well the Faster R-CNN was able to work with the input image. A low confidence means, that there were a lot of objects the network did not discard in the first place but is also not sure about the objects being a dwelling. In that case, we recommend to humanitarian operators to gather more information from different sources concerning the camp and the area around it. A higher confidence indicates that humanitarian operators can include the order of magnitude of displaced peoples in their medium-term planning.

3.3. Test and Validations Results

Table 3 shows the results for the tests on images used for training the networks, Table 4 the results for the tests on new images.

Looking at Table 3, the most expectable result is, that the best-performing network for a validation set is the network that was trained on training data constructed using the same configurations as the validation set. This shows, that the networks learn the distribution of the input data they are trained with as expected [19].

The second parameter influencing the performance of a network is the training time. When comparing the networks which have the same configuration concerning the input images and the annotation-mode but were trained for a different number of iterations (**_gc_** and **_gg_**), it can be seen in Table 3 that the longer trained networks only perform better on the validation sets having the same distribution as the training set the networks were trained on. On the other validation sets, the performance is about the same between the longer (36.000 iterations) and the shorter (27.000 iterations) trained networks. In Table 4, which shows the generalization capabilities of the networks, it can even be seen that the shorter-trained networks outperform the longer trained networks throughout all validation sets. This is a clear indicator that the longer trained networks struggle with overfitting [20].

Table 3. Validation Results of the individually trained networks (*n_**) on validation sets built from images the networks were trained on (*v_orig_**). Adapted table taken from [2].

Val.-Set	v_orig_cc			v_orig_gc			v_orig_gg			v_orig_cg		
Metric	**AP**	**AR**	**F1**	**AP**	**AR**	**F1**	**AP**	**AR**	**F1**	**AP**	**AR**	**F1**
Network												
n_cc_36k	68.2 %	72.0 %	70.0 %	55.9 %	61.1 %	58.4 %	27.6 %	35.0 %	30.9 %	22.1 %	29.5 %	25.3 %
n_gc_27k	57.9 %	63.4 %	60.5 %	60.9 %	65.8 %	63.3 %	23.7 %	30.4 %	26.6 %	19.9 %	25.2 %	25.2 %
n_gc_36k	61.6 %	66.2 %	63.8 %	65.2 %	69.1 %	67.1 %	23.6 %	30.9 %	26.8 %	19.7 %	25.5 %	22.2 %
n_gg_27k	36.0 %	48.2 %	41.2 %	35.4 %	47.7 %	40.6 %	42.6 %	48.1 %	45.2 %	14.1 %	21.3 %	17.0 %
n_gg_36k	35.5 %	47.8 %	40.7 %	35.5 %	47.9 %	40.8 %	46.2 %	50.6 %	48.3 %	14.6 %	21.6 %	17.4 %
n_cg_27k	43.6 %	54.7 %	48.5 %	42.6 %	52.2 %	46.9 %	19.5 %	29.2 %	23.4 %	29.6 %	34.6 %	31.9 %
n_mix_27k	59.3 %	64.9 %	62.0 %	58.0 %	63.6 %	60.7 %	23.4 %	30.0 %	26.3 %	19.9 %	25.4 %	22.3 %
n_mixg_27k	42.2 %	53.4 %	47.1 %	42.1 %	53.4 %	47.1 %	18.7 %	28.3 %	22.5 %	29.0 %	34.1 %	31.3 %

Table 4. Validation Results of the individually trained networks (*n_**) on validation sets built from images not seen by the networks while training (*v_new_**). Adapted table taken from [2].

Val.-Set	v_new_cc			v_new_gc			v_new_gg			v_new_cg		
Metric	**AP**	**AR**	**F1**	**AP**	**AR**	**F1**	**AP**	**AR**	**F1**	**AP**	**AR**	**F1**
Network												
n_cc_36k	6.6 %	13.0 %	8.8 %	5.3 %	11.7 %	7.3 %	9.2 %	14.9 %	11.4 %	10.4 %	16.5 %	12.8 %
n_gc_27k	8.4 %	12.8 %	10.1 %	8.8 %	13.3 %	10.6 %	10.1 %	14.4 %	11.9 %	9.7 %	13.9 %	11.4 %
n_gc_36k	8.2 %	12.5 %	9.9 %	8.3 %	12.8 %	10.1 %	9.8 %	13.7 %	11.4 %	9.6 %	13.5 %	11.2 %
n_gg_27k	7.9 %	13.9 %	10.1 %	8.4 %	14.6 %	10.7 %	11.1 %	17.2 %	13.5 %	10.4 %	16.2 %	12.7 %
n_gg_36k	7.3 %	12.7 %	9.3 %	7.7 %	13.4 %	9.8 %	9.7 %	15.4 %	11.9 %	9.0 %	14.5 %	11.1 %
n_cg_27k	9.6 %	17.5 %	12.4 %	6.5 %	14.2 %	8.9 %	9.4 %	17.1 %	12.1 %	12.4 %	20.0 %	15.3 %
n_mix_27k	9.3 %	13.8 %	11.1 %	8.9 %	13.2 %	10.6 %	10.2 %	14.4 %	11.9 %	10.7 %	15.0 %	12.5 %
n_mixg_27k	10.7 %	18.0 %	13.4 %	9.9 %	17.4 %	12.6 %	12.9 %	20.3 %	15.8 %	13.4 %	20.5 %	16.2 %

A highly interesting behavior of the networks becomes evident when looking at the color representation of the images in the training set used to learn the networks models. Table 3 shows that the models learned on color-images achieve the highest overall performance over all conducted tests. This indicates that color images enable more detailed feature learning. However, the networks trained on grayscale images generalize better on unseen data compared to those networks trained on color images.

A possible explanation for that can be found in [21]: The authors show that CNNs tend to learn object textures instead of object shapes when building a model of the input data. Inputting grayscale images in a CNN, and thus making textures more abstract, could force the CNN to take the object shapes more into consideration when learning the input data.

From that observation, the hypothesis was made that for best generalization a network should be trained with a set of training images that contains color- and grayscale-images. The hypothesis was tested by training the networks *n_mixg_27k* and *n_mix_27k*. To evaluate and compare the generalization capabilities, we examined the performance on unknown images in Table 4. Therefore, the average F1-score over all validation sets containing unknown images (*v_new_**) was aggregated in Table 5.

It is observed that network *n_mixg_27k* has a more than 2 % higher average F1 score on the *v_new_** validation sets than the other trained networks. The median of all average F1-scores is 11.5 % and the standard deviation of all networks average F1 scores is 1.3 %. Network *n_mixg_27k* deviation from the mean is 2.3 times higher than the standard deviation of all averaged F1 scores over all networks. It can therefore be argued that *n_mixg_27k* generalizes better than the other networks.

Table 5. Average F1 over all validation sets containing images not seen by the networks before.

Network	Average F1 on v_new*
n_cc_36k	10.0 %
n_gc_27k	11.0 %
n_gc_36k	10.7 %
n_mix_27k	11.5 %
n_gg_27k	11.7 %
n_gg_36k	10.5 %
n_cg_27k	12.2 %
n_mixg_27k	14.5 %

Regarding the best annotation method, only observations and no precise conclusions concerning the best mode are possible. Table 3 shows that networks trained on annotations made on grayscale images generally achieve a lower maximum AP and AR but are more stable on validation sets with annotations made on color images than networks trained on annotations made on color images are on validation sets with annotations made on grayscale images. This higher generalization capability is further indicated by the tests run on the *v_new_** validation

sets, where the networks trained on annotations made on grayscale images (*n_gg_27k*, *n_cg_27k*) perform better than network *n_mix_27k* trained on mixed-representation input data with annotations made on color images.

4. Discussion

The created workflow can be discussed from two perspectives: On the one hand, Section 4.1 take a broad look on the whole workflow, discussing possible improvements of the workflow in terms of updating technology and methodology citing [1]. Further, the overall real-world applicability is discussed in this section. On the other hand, Section 4.2 zooms into the building and training of the dwelling detection classifier, the Faster R-CNN model itself, following [2]. There, recommendations for training an object detection classifier are derived from the observations made in Section 3.3.

4.1. Dwelling Detection Workflow

The workflow was developed with the goal of a fast applicability. This goal was reached successfully by using open software and data, speeding up the slow and cumbersome process of ground-truth generation and annotation and building convenient workarounds when information about the data was missing. Nevertheless, trade-offs between accuracy and applicability had to be made:

The seeded region growing algorithm used for speeding up annotating ground truth data is rather simple. Plenty of improved image segmentation algorithms that are existing today [22] could improve the ground truth generation. A more accurate ground truth segmentation would also allow switching the deep learning architecture from a Faster R-CNN to a Mask R-CNN [23]. Mask R-CNN builds up on the Faster R-CNN architecture and allows image segmentation on top of object detection in images. Using segmentations of dwellings can yield more accurate estimates for the inhabitant estimation. Further accuracy can be achieved by using georeferenced satellite imagery where the spatial resolution of a pixel is known. Combining the extend or the footprint of a dwelling with the spatial resolution of a pixel allows to accurately determine the size of a dwelling which can be used for inhabitant estimation based on a parameter taking the actual size of a tent into account.

If accuracy or fast applicability is more important is nevertheless context dependent. For operators managing one specific camp, more accurate results are essential. For humanitarian operators working on broader migration situations, fast information giving insights about the magnitude of the situation are important. In this context, the presented workflow is a good enhancement: We presented actual workflow results in September 2019 to professionals in migration management like the German Red Cross[1] as part of an international exercise carried out in the HUMAN+ project. The additional information about people in the refugee camps generated through dwelling detection was appreciated unanimously. Though those professionals wanted to have absolute numbers, this is not doable due to the uncertainties described in this paper. The number of people in a refugee camp has to be seen in context with the confidence value generated. Those numbers must be combined and merged with other information gathered in the HUMAN+ project. Only then a meaningful picture of the situation can be achieved to help first responders to make the right decisions in crisis management and refugees to be taken care of.

4.2. Training Recommendations

The tests and validations described in Section 3.3 reveal some interesting properties indicating measures to increase the performance of a (dwelling detection) object detection classifier. From those observed properties, recommendations become available for future projects developing dwelling detection classifiers.

First, we recommend having a high variance in the training data. Variance should be present concerning technical parameters, meaning that the training set should contain (satellite) imagery of different image quality, different spatial resolutions as well as different environment conditions. Further variance should be present concerning semantic parameters of the images. In the case of dwelling detection that means having different types of camps and landscape-types in the training set.

Highly important for generalization purposes is the color-representation of the images in the training set. As shown in Section 3.3, color-images enable a higher maximum performance in a trained network and grayscale images enable better generalization capabilities in networks. In this article, it is shown that having both, color- and grayscale-images, in the training set improves the generalization capabilities of a trained dwelling detection network. We therefore highly recommend converting some training images to grayscale when working with a training set consisting of color-images.

Another problem that we found during the tests and validations was overfitting. Two measures can be taken to prevent overfitting. On the one hand, it is important to have a high variance in the training set. We propose as a rule of thumb that it is better to annotate few dwellings on many satellite images than to annotate many dwellings on only a few satellite images. On the other hand we suggest to use well

[1] https://www.drk.de/en/

established deep learning approaches to tackle overfitting like data augmentation techniques [24] and early stopping [25].

Concerning the annotation algorithm, no direct recommendations can be given from a technical point of view. For a general discussion on the fast annotation approach see Section 4.1.

5. Conclusions

Concerning the dwelling detection workflow we concluded in [1]: "A machine learning based dwelling detection classifier was built using modern object detection techniques, a classifier which yields results almost immediately helpful in humanitarian operations. An important step for doing so is the developed training data preparation workflow, which is fast and convenient for annotations.

The workflow outputs a result in the magnitude of the real numbers of inhabitants in a migrant camp. It works on the image without the need of adjustments by the user, thus generating rapid and useful information. The generated information is best combined with other information about the migration situation from different sources. The visual representation of the results with bounding boxes drawn on the input image as well as the calculated confidence factor increase the interpretability of the results. The results were deemed to be useful by experts in the field."

Additionally, we carried out extensive tests and validations. An exhaustive analysis of these tests can be found in [2]. In this article, the hypothesis created in [2] that a mixture of color- and grayscale-images in the training data set improves the generalization capabilities of an object detection classifier, was backed up by training a new network which shows a higher generalization capability compared to the networks trained in [2]. Finally, as lessons learned, recommendations for training a dwelling detection classifier, derived from the tests and validations conducted were specified.

Acknowledgements

The authors acknowledge the financial support by the German Federal Ministry of Education and Research (BMBF) and the Federal Ministry Republic of Austria Climate Action, Environment, Energy, Mobility, Innovation and Technology (BMVIT/BMK) in the framework of the HUMAN+ project (funding code 13N14716 to 13N14723)

References

[1]. L. Wickert, M. Bogen, M. Richter, Dwelling Detection on VHR satellite imagery of Refugee camps using a Faster R-CNN, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI'2020)*, Berlin, November 2020, pp. 7–11

[2]. L. Wickert, M. Bogen, M. Richter, Supporting the Management of Humanitarian Operations Concerning Migration Movements with Remote Sensing, in *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XLIII-B3-2020, August 2020, pp. 233–240.

[3]. Global Trends - Forced Displacement in 2018, UNHCR, Switzerland, 2018. Accessed: July 30, 2019. [Online]. Available: https://www.unhcr.org/statistics/unhcrstats/5d08d7ee7/unhcr-global-trends-2018.html.

[4]. J. B. Campbell, R. H. Wynne, Introduction to Remote Sensing, Fifth Edition, *Guilford Press*, 2011.

[5]. Z.-Q. Zhao, P. Zheng, S.-T. Xu, X. Wu, Object Detection with Deep Learning: A Review, *IEEE Transactions on Neural Networks and Learning Systems*, 2019, pp. 1–21.

[6]. K. Spröhnle, D. Tiede, E. Schoepfer, P. Füreder, A. Svanberg, T. Rost, Earth Observation-Based Dwelling Detection Approaches in a Highly Complex Refugee Camp Environment — A Comparative Study, *Remote Sensing*, Vol. 6, Issue 10, September 2014, pp. 9277–9297.

[7]. I. Katz, A network of camps on the way to Europe, *Forced Migration Review*, Issue 51, 2016, pp. 17-19.

[8]. A. Dalal, A. Darweesh, P. Misselwitz, A. Steigemann, Planning the Ideal Refugee Camp? A Critical Interrogation of Recent Planning Innovations in Jordan and Germany, *UP*, Vol. 3, Issue 4, December 2018, pp. 64-78.

[9]. B. Beznec, M. Speer, M. S. Mitrović, Governing the Balkan route: Macedonia, Serbia and the European border regime, *Rosa Luxemburg Stiftung Southeast Europe*, 2016.

[10]. J. A. Quinn, M. M. Nyhan, C. Navarro, D. Coluccia, L. Bromley, M. Luengo-Oroz, Humanitarian applications of machine learning with remote-sensing data: review and case study in refugee settlement mapping, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 376, Issue 2128, September 2018, p. 20170363.

[11]. O. Ghorbanzadeh, D. Tiede, Z. Dabiri, M. Sudmanns, S. Lang, Dwelling Extraction in Refugee Camps Using CNN – First Experiences and Lessons Learnt, *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XLII–1, September 2018, pp. 161–166.

[12]. S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, Issue 6, June 2017, pp. 1137–1149.

[13]. L. Zhang, L. Zhang, B. Du, Deep Learning for Remote Sensing Data: A Technical Tutorial on the State of the Art, *IEEE Geoscience and Remote Sensing Magazine*, Vol. 4, Issue 2, June 2016, pp. 22–40.

[14]. R. Adams, L. Bischof, Seeded region growing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, Issue 6, June 1994, pp. 641–647.

[15]. T.-Y. Lin, *et al.*, Microsoft COCO: Common Objects in Context, in *Proceedings of the Computer Vision – ECCV 2014*, Vol. 8693, (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds.), *Springer International Publishing*, 2014, pp. 740–755.

[16]. K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, in *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016, pp. 770 - 778.
[17]. The COCO Consortium, Common Objects in Context, *Detection Evaluation*, 2019. http://cocodataset.org/#detection-eval
[18]. N. Chinchor, MUC-4 evaluation metrics, in *Proceedings of the 4th Conference on Message Understanding (MUC4'92)*, McLean, Virginia, 1992, pp. 22-29.
[19]. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature*, Vol. 521, Issue 7553, May 2015, pp. 436–444.
[20]. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research*, Vol. 15, Issue 1, 2014, pp. 1929–1958.
[21]. R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, W. Brendel, ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness, *in Proceedings of the Seventh International Conference on Learning Representations (ICLR' 2019)*, New Orleans, Louisiana, USA, 2019.
[22]. H. Zhu, F. Meng, J. Cai, S. Lu, Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation, *Journal of Visual Communication and Image Representation*, Vol. 34, January 2016, pp. 12–27.
[23]. K. He, G. Gkioxari, P. Dollar, R. Girshick, Mask R-CNN, 2017, Accessed: 31 Jul. 2019, pp. 2961–2969. [Online]. Available: http://openaccess.thecvf.com/content_iccv_2017/html/He_Mask_R-CNN_ICCV_2017_paper.html
[24]. A. Mikołajczyk, M. Grochowski, Data augmentation for improving deep learning in image classification problem, in *International Interdisciplinary PhD Workshop (IIPhDW)*, 2018, pp. 117–122.
[25]. Y. Yao, L. Rosasco, A. Caponnetto, On Early Stopping in Gradient Descent Learning, *Constructive Approximation*, Vol. 26, Issue 2, August 2007, pp. 289–315.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 54-64

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

AI-assisted Distributed Applications at Edge for Industrial Field Autonomous Systems

[1, *] Yannick FOURASTIER, [2] Claude BARON, [3] Hakima CHAOUCHI, [4] Carsten THOMAS, [5] Thomas BOURGEAU, [6] Jean-Paul SMET, [7] Viktor YEHOROV and [8] Philippe ESTEBAN

[1] CodEUrope SAS, 244 Route de Seysses, 3100 Toulouse, France
[2] Université Toulouse, INSA, LAAS-CNRS, ISAE-Supaéro, 7 Av. Colonel Roche, 31200 France
[3] Institut Polytechnique de Paris, Telecom SudParis, 9, rue Charles Fourier, 91011, Evry, France
[4] HTW Berlin, Treskowalee 8, 10318, Berlin, Germany
[5] KB Digital, 1291 Commugny, Switzerland
[6] Nexedi GmbH, Technology Centre Munich, Agnes-Pockels-Bogen 1, 80992 München, Germany
[7] University of Odessa, Lab Mironaft, Kanatna St, 112-114, 65000 Odessa, Ukraine
[8] Université de Toulouse III, LAAS-CNRS, 7 Av. Colonel Roche, 31200 Toulouse, France
E-mail: yannick.fourastier@codeurope.eu

Received: 7 October 2020 /Accepted: 30 November 2020 /Published: 28 February 2021

Abstract: Complex industrial systems are increasingly software-defined, rapidly evolving into autonomous, self-adaptive processing at industrial fields. They are constituting worlds of things, wherein the internet of things is heavily developing with local semantic features. Also, artificial intelligence technologies are spreading fast in the industrial context, improving the operations efficiency. However, Edge computing systems involving artificial intelligence must also continuously ensure reliable and safe operations. This paper assesses the state of the art of cognitive technologies with their relevance for artificial intelligence implementation in industrial cyber-physical systems. It then introduces a state of research on artificial intelligence safety assurance practices. Implementation through three use cases at the software stack is illustrated with a virtual operating system that operates the data space in the industrial layout. More precisely, the industrial development is illustrated with a real case of swarm computing for advanced robotics powered with SlapOS[1]. Although still at an early stage, it shows experimentally how Edge operating systems are evolving fast as a kind of AI intensive software.

Keywords: Artificial intelligence, Cyber-physical system, Industrial edge, Swarm processing, Virtual operating system, Digital twin, Safety of intended functionality.

1. Introduction

Digitization is deploying fast in the industrial field. Complex industrial systems are increasingly software defined, as it also follows the digitization by softwarisation wave. They're rapidly evolving into autonomous and to some extent, self-adaptive processing relying on the artificial intelligence (AI)

[1] SlapOS is a general purpose overlay operating system for distributed POSIX infrastructures (Linux, xBSD, etc.) with a strong focus on service management. It is published as Free Software under fairly permissive licensing

algorithms developments and the fast and reliable distributed communication technologies offered. In the context of this paper, the "cognitive technologies" expression groups those technologies which gather intelligence from their data generation, and proceed them thanks to artificial intelligence techniques, in order to produce a learnt model that can autonomously outcome a result. From a similar approach, they may articulate some programmable mechanisms to self-adapt also to their context. These cognitive technologies are spreading fast at the so-called industrial edge; the processing connected nodes that controls the industrial Cyber Physical Systems (CPS). While enabling these systems with adaptive functioning, the deployed cognitive techniques are possibly self-evolving too in order to optimize the performance efficiency of such cyber-physical systems. When doing that, they also carry concerns, which can even turn into issues, regarding the safety assurance and also the performing stability. Our work addresses AI-assisted distributed applications at Edge for autonomous cyber-physical systems which are operating in the industrial field with dependable performance and critical safety. This paper is an extended version of our ASPAI' 2020 one [1].

This paper summarizes first the industrial related cognitive technologies; also called Industrial Internet of Things (IIoT). The industrial internet of things composes a virtual platform in the field. Such a world of industrial objects liaises semantically each with all, and also with distributed software codes and patterns. As a semantic web of things, it is sometimes named as WoT in the literature. It composes an industrial data space with methods, resulting with a software technology producing a digital twin. Standardized by the ISO, the industrial IoT as well as the digital twin, interact with the business orchestration, such as the Manufacturing Execution System (MES) in the production context, the signalling systems at railway, the urban traffic regulation at the smart city, or the building control and automation, etc. Such a digitized industrial platform brings more automatic decisions locally and actions; meaning as close as possible to the CPS. We study a software stack, namely SlapOS, able to virtually power safely the autonomous intelligent industrial system with Edge Computing. It is to build up a technology able to virtually power and control safely the autonomous industrial Cyber Physical Systems (ACPS) with distributed Edge embedded AI.

The Edge embedded AI indeed supports local decisions and actions which often influence safety critical operations at field level. This paper discusses the safety assurance practices for the design and development of dependable cyber-physical systems which are AI assisted.

The paper illustrates three actual implementations of industrial cases following our described vision. One aims at wind turbine health monitoring, the other is a virtualized control (virtual PLCs' interconnecting IoTs' at a production process), and the third is about guidance and navigation of a drone swarm.

2. Cognitive Technologies in the Industrial Field

Different sorts of algorithms support signal analysis, information characterization, memorizing and retrieving, processes automation, anomaly detection, operations predictions, and decision making. They can also emerge models from some perceived situation. This latter is factually a time series of input data, which produces the machine to learn to produce a model with some probability of correctness. What a supervisor will validate or not, then the machine will progress faster its learning.

In this section, cognitive technologies and their relevance for artificial intelligence implementation in the industrial field are succinctly introduced. Distinguishing from genetic algorithms (GA) and robotized process automation (RPA), we structure four families of techniques: machine learning (ML), deep neural networks (DNN), hybrid neural networks (HNN), and generative adversarial networks (GAN).

2.1. Automation Techniques: RPA

Robotized process automation (RPA) [2] mimics actions in sequences that are usually performed by a human operator. RPA uses structured data and kind of wired logic to function. Contrary to AI functions that use unstructured input data and develop its own logic to produce wished results. RPA in the industrial field, relates to the replacement of legacy hardwired automation by software executing on a generic processing platform.

2.2. Evolutionary Computing: Genetic Algorithms (GA)

In decision making processes, Genetic algorithms (GA) [3] develop heuristics and meta-heuristics from candidate solutions, to look for a best fit solution. GAs' functions on natural selection principles, with derivating secondary solutions from the candidates characteristics, mutated, altered, recombined or crossed. Proceeding iteratively, new generations of candidate solutions are produced. A fitness function evaluates the new set against the problem to solve. GA's are well fitting optimization problems, also to sort best decisions for specific situations as they are often in the industrial field, especially in complex process chains.

2.3. Machine Learning (ML)

Machine learning is essentially a form of applied statistics, at which complicated functions estimations are emphasized, while the importance of proving confidence intervals around these functions is decreased [4]. ML is driven from two main

approaches, frequentist estimators and Bayesian inferences. Learning algorithms can be divided into supervised and unsupervised categories [5], in order to develop performant generalization abilities. These techniques are useful for pattern recognition applications, predictive models, and also model prediction with deep learning (signal processing, time series, speech recognition, objects, situations, etc.).

2.4. Artificial Neural Networks (ANN)

ANN are densely interconnected adaptive processing units (perceptrons) [6]. They enable fine-grain parallel computation of non-linear static or dynamic systems. ANN can be considered as parallel computational models. A major feature of such technique is its adaptive nature. Traditional programming is replaced with learning by example prior to solve problems. Artificial neural network results are linkable to applications at which there are little or incomplete data to solve a problem, but available training data. Also, ANNs' are relevant for function approximation, associative memory, non-linear system modeling, combination, optimization, information clustering, forecasting and prediction, and control.

2.5. Deep Neural Networks (DNN)

Multilayer perceptrons (MLP) implement feedforward neural networks. They aim to approximate a function with a mapping $y=f(x;)$, and to record the resulting value for the best approximation [5]. With data flowing through the assessed function from x, through intermediate computations in charge of defining this function, and finally to output y, these learning models are feedforward. There aren't feedback connections, unless the implementation of recurrent neural networks with back propagation. While ones are efficient for image or complex analysis, the others power many natural language applications. Architecturing ML techniques with MLP, deep neural networks (DNN) have considerably simplified the modeling process that was a painful bottleneck to processing complex heterogeneous data. By incorporating both, feature engineering and conceptual learning, DNN directly processes raw data then generates the final, conclusive results [7]. Such techniques have better capability to generalize unseen combinations of features when solving large scale regression, classification problems, etc. Their applications can be answer selection, recommender system, predictions.

2.6. Hybrid Neural Networks (HNN)

ANNs' requires previously defined architecture for the neural net. Supervised, unsupervised or reinforcement learning are then usually achieved by adjusting the connection weights iteratively using a gradient descent-based optimization algorithm [8]. Design results difficult since the sensitivity to initial conditions of training and local character, leads to hybrid connectionist and symbolic computation at ANN and hidden layers behaviour weighting [6].

At hybrid neural networks (HNN), symbolic representations (probability distributions over the kinds of data encountered) provide explicit, fast initial coding, direct control, dynamic variable binding and knowledge abstraction, while the architecture (MLPs' net) provides robust learning, generalization to similar input, and fault-tolerant processing. HNN results with high adaptability and resistance to external noise, what's relevant for monitoring models, and the forecasting of computer networks states, which edge processing at industrial edge typically [9].

2.7. Adversarial Learning, Generative Nets (GAN)

GANs' develop beyond HNN to generate more efficient ANN models (architecture and embedded symbolic, which hyperparameters). A generative model somehow collaborates with a discriminative one, evaluating generated candidates. In control theory, adversarial learning based on neural nets can train robust controllers in a game theoretic sense, by alternating the iterations between a minimizer policy, the controller, and a maximizer one, the disturbance. Adversarial cognitive techniques are relevant to orchestrate the resilience in control systems.

2.8. Usages

Production systems automation is progressing since industry exists, as far as pure automation is considered one of the key factors for implementing cost effective production. In such a paradigm, the less the production system requires human energy, the more is presumably effective. The last fifty years of literature in industry engineering techniques have investigated that lead times and quality levels may exceed by far those of human workers. Such a machine-oriented assertion also leads to the artificial cognition principle in production systems and to function them software-defined. Fig. 1 summarizes the cognitive system architecture introduced by Bannat and *et al.*, with [10].

Such a cognitive cyber-physical architecture is concerned with its performance specifics which relates to processes timing, planning and coordination, including accordingly to discrete events. Hence, to some extent, the system components are often real time constrained. Additionally, they can have to deal with human safety and also environmental considerations. Therefore, they have to mitigate hazards accordingly.

With cognitive techniques integrated into the software defined industrial architecture, they are required to fulfil highly demanding requirements as other system components, in particular the safety ones. The Table 1 below proposes a summary of the AI techniques and the industrial safety. Although obviously depending on their allocation in the system and this latter's business specifics, the adherence of the AI techniques to real time constraints in the industrial system depends on their usage and purpose (what action in the CPS powering the industrial field).

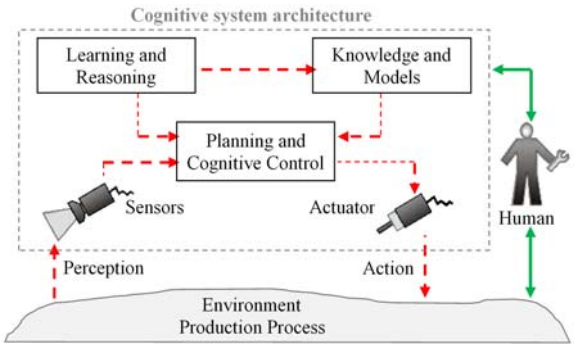


Fig. 1. Cognitive system architecture with closed perception-action loop [10].

3. Virtually Distributed Computing at Industrial Field

The concept of digital twins is at the limit of artificial intelligence technologies. A digital twin reproduces identically, but only digitally, the functioning of a real physical system. By extracting perceived behaviour from physical functioning, this technology produces a Model Order Reduction (MOR). It enables multiple numerical simulations to be carried out, in particular, in order to decide before any changes are made, which configuration variations to apply.

By working as close as possible to the physical system, distributed in the edge computing layer, the software does not necessarily need a computer to carry out numerical simulations, which are very costly in terms of computing time when all the layers are dissociated. In fact, it can give results in real time. This much shorter calculation time makes it possible to predict the behaviour of a system and control it in real time. Such an implementation between the digital twin and its real system constitutes "closed-loop digital twins". Reproducing the functioning of the real object, it also combines AI techniques to anticipate certain actions, and adjust certain behaviours of the digital twin. Visioning mechanisms allows human agents to interact or act remotely on real objects.

The ISO/IEC30141 [11] standardized the industrial internet of things into six domains as shown with Fig. 2: user, operations and management, application and services, access and communication, sensing and controlling, and physical entity.

Table 1. Cognitive techniques and industrial safety.

Cognitive techniques	Industrial Field action	Real Time adherence / safety
RPA	Automation of repetitive at data flows processing	Slack adherence
GA	Decision making Optimal solution computing	Slack adherence
ML	Predicting from time series or pattern sets, involved in machine self-deciding	Can be relevant
ANN	Problem solving Classification and sorting Function approximation Incomplete data sets linearizing	Can be relevant
DNN	Sorting Large scale regression Identification of combinations Pattern recognition Problems classification Generalization	Can be relevant depending on the architecture of the powered AI
HNN	Monitoring Forecasting	Can be tight adherence
GAN	Robust controllers training Control systems resilience	Tight adherence

Such a standardized IoT connects cyber-physical systems to remote processing and storage resources made available in the network. The main challenge being to ensure reliability, safety and security of these connected cyber-physical systems, bringing those processing and storage resources closer to these systems, is the way to go; it is named Edge computing or fog by opposition to the cloud, that is central.

Edge Computing moves the industrial internet of things as virtually distributed swarm computing, in near proximity to the field where CPS are orchestrated, while performing their tasks. In addition, AI algorithms perform better, they allow us to build cognitive techniques at the edge in order to better control, predict and automate decisions locally, thus requiring less human intervention. Fig. 3 illustrates possible placements of cognitive technologies closer to the industrial field. The industrial cloud is beyond the field digital fog where edge embedded cognitive algorithms are running. This cognitive capacity is increasing fast, following the 4.0 digitization pace.

This distributed Edge Computing offers both CPS control and data collection, storage and processing. It requires a highly reliable operating system and a CPS functioning safety, by design architecture. While data processing and storage capacities are increasing at such a fog (“digital cloud” at the edge), this one can also provide available resources opportunistically. Assisted with artificial intelligence techniques, the Edge virtual operating system organizes dynamically the sharing of resources to implement concretely a fog. It groups pseudo-available resources for functions or

services to execute virtually at the cyber-physical system or close neighbourhood.

In order to build successfully this solution, and ensure reliable and safe behavior of the overall connected and cognitive based systems, it is necessary:

1. To connect the right physical objects to the network in line with the semantic web of IoTs'.
2. To build a correct representation of the physical objects using the virtual objects of that semantic web. Then these virtual objects bind to their physical platform.
3. To design and use efficient and well architectured distributed processing.
4. To build a robust, resilient, and secure, data persistence management.

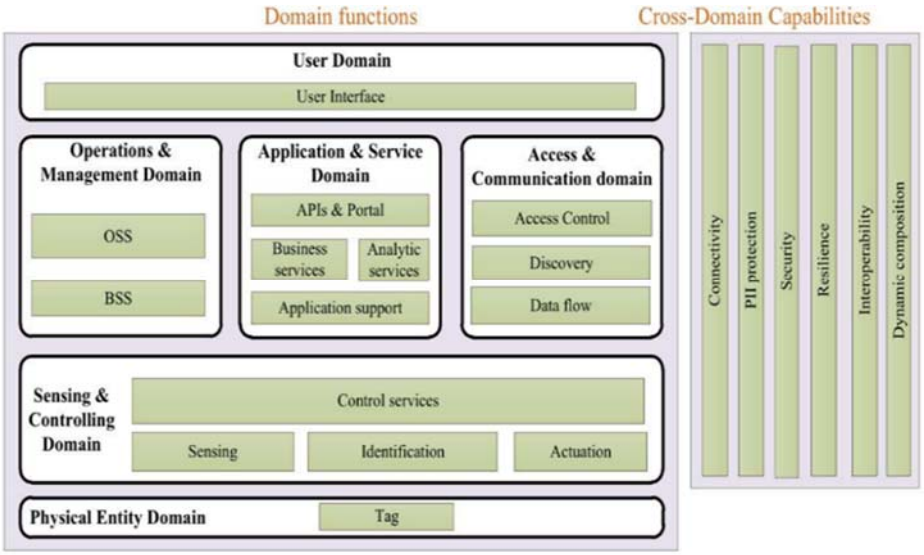


Fig. 2. ISO/IEC 30141 functional view [11].

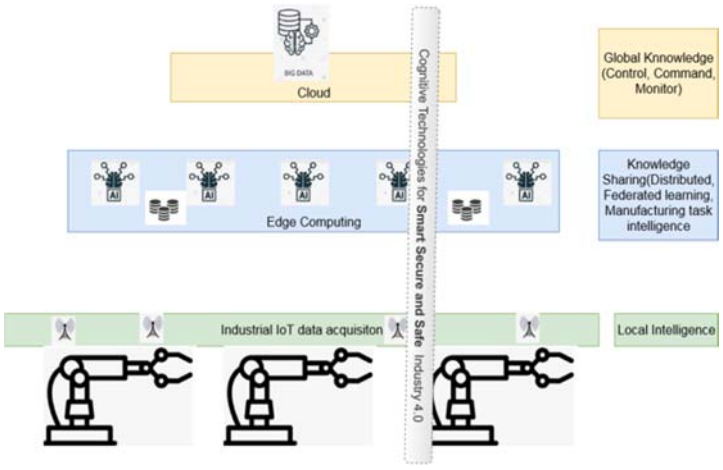


Fig. 3. IIoT and Cognitive technologies over the edge and the cloud in Future Smart Secure and Safe (3S) Industry 4.0 systems.

3.1. The World of Things with Multi-level Intelligence: the IoT 2.0

The internet of things connecting devices (sensors, actuators), enables software-defined command and control to interact with the physical world. That logical internetworking of things, forms the IoT 1.0, is quite the same as the general Internet has been in the late 90s'. With the technology heterogeneity challenging the machine-to-machine connectivity, the industrial internet of things (IIoT) has provided a software abstraction layer. That IIoT ended up by building a multilevel connectivity from the device, then to some gateway, up to the cloud to process the IoT device collected data.

With the fast growing of AI algorithms, IoT data analytics is getting us to a second phase named IoT 2.0. Building intelligent objects, able to interact autonomously with each other, and not anymore only connected objects, is a real challenge. IoT 2.0 is more

about interfacing between machines and humans in a very intuitive fashion. It is about using AI intensively to understand people in the industrial field, to run autonomously processes with no human intervention. It is about getting these intelligent objects to become collaborators of humans (e.g. the cobots), multimodal with human interactions, including voice and gestures.

Combining connected things with virtual experiences, made possible by virtual reality (VR) and augmented (AR), has opened up new possibilities in the industrial cognitive system. With VR, the users are entirely removed from their real settings, and can interact with the digital twin through immersive experience. This software generated technology, re-creates the running scenario, and cancels the need to access the physical process. With AR, the user's current environment can be digitally overplayed with image, 3D model, sound, touch or haptic experience, and smell.

3.2. Distributed Physical and Virtual Objects Brings Semantic: Digital Twins

Digital twin is nothing but a virtual replica of an actual physical asset [13]. Back in the 1960s', NASA scientists created a replica of the physical asset on the ground and built a technology to match the processes and systems in space. Even though they were not directly connected with some data link, but the parameters manually propagated from one to the other, they were kind an ancestor of digital twins.

The ISO23247 [14] standardizes the digital twin reference architecture and technology. Its framework is layered on the IEC/ISO30141 one, as illustrated with Fig. 4, although the schema groups the six domains of IoT into four blocks at the digital twin framework. The six domains are organized in a way they can distribute in any Edge computing architecture, with cloud uplink or not, depending on the needs of computation.

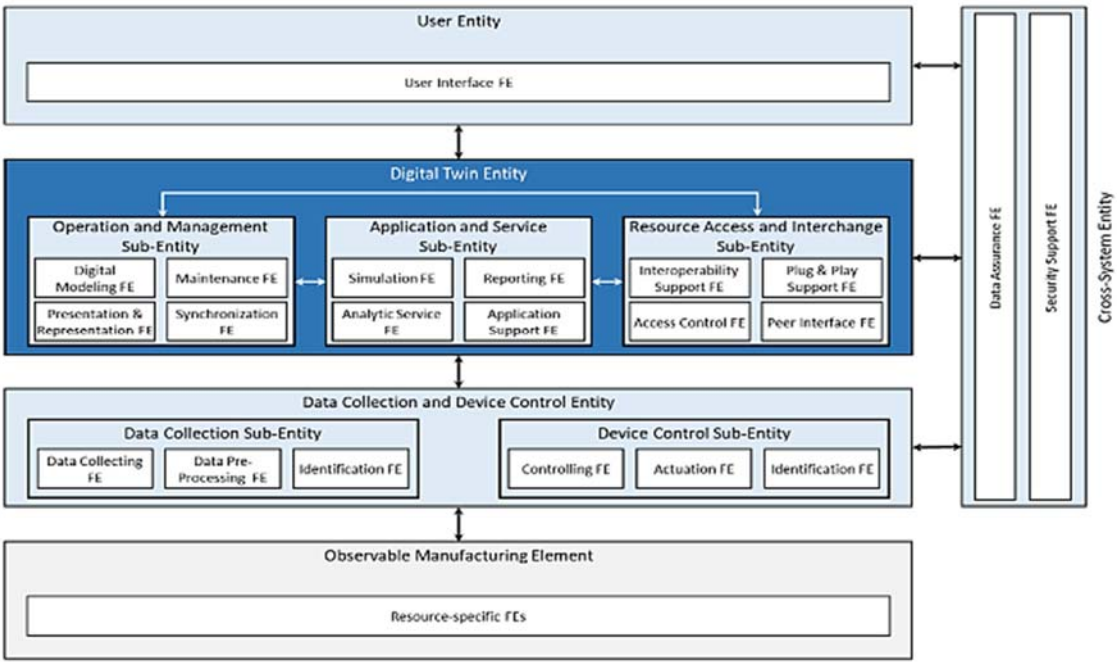


Fig. 4. Digital Twin framework ISO23247 [14].

3.3. Distributed Intelligent Processing

A major challenge of that standardized digital twin [14] is to really take advantage of the IoT technology in industrial decisions. IoT generates a myriad of data by the possibly huge amount of connected devices, including sensors and actuators, that are usually aggregated and stored at fog virtual platform, and up centrally to the cloud.

The interaction and the management of IoT devices has enabled new service opportunities including predictive maintenance, continuous monitoring of the parts that are more subjected to degradation and ageing, scheduling and remote running of maintenance interventions, product quality prediction and autonomous adjustment for correction or automatic tagging of non-quality.

Standardized digital twin (in fact most of previous IIoTs') exhibits several characteristics that can be sector specific (e.g. manufacturing, smart cities, smart buildings, etc.) in terms of communication bandwidth, real time adherence, latency improvements, robust connectivity, together with limited costs on large scale deployment scenarios. An integrated infrastructure is deployed to collect information from heterogeneous sensors, to process it in the cloudlets (herein kind of edge clouds), to transmit to the cloud only what complies with the digital twin policies and applied

parameters to update the related in the form of a closed-loop system. Edge computing results to be a technology of distributed cloud that moves the processing functionality from centralized datacenters, to the edge of the network, in a virtualized field.

Generally speaking, such layout virtualization migrates core capabilities such as networking, computing, storage, and applications closer to the devices, and in particular to the IoT endpoints. There are already interesting examples in the related literature [14] about intelligent services close to manufacturing units, e.g., to meet key requirements such as agile connection, data analytics via edge nodes, highly responsive cloud services, personalized enforcement of privacy policy strategies.

In such a way, the distribution of the digital twins over cloudlets that architecture the industrial data space, composes hybrid twins which compose distributed intelligent processing with the industrial IoT, Edge nodes, and digital twin enabled services running at cloud as illustrated with Fig. 5.

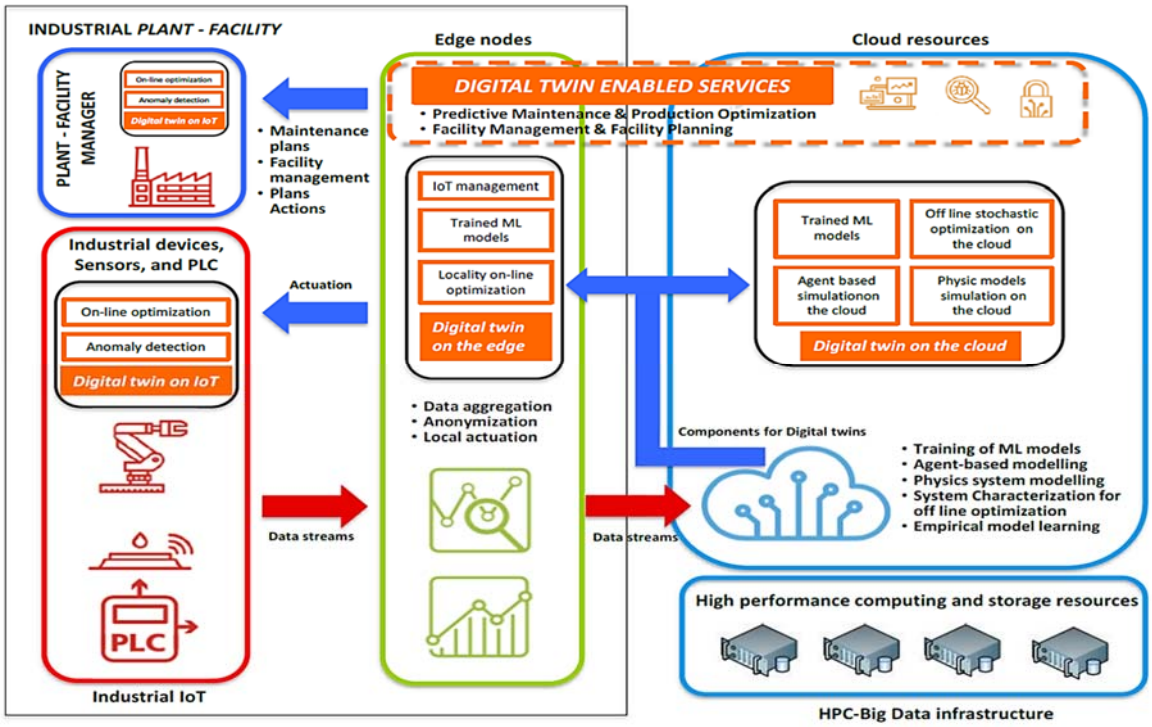


Fig. 5. Concept vision of IoTwins distributed hybrid twins [14].

With reducing the need of uploading data to the cloud, a kind of distributed intelligent processing architecture enables to proceed larger volumes of data, for example to proceed time series for AI treatment at the Edge. Nevertheless, it supposes reliable, secure and resilient data persistence mechanisms.

3.4. Data Persistence Management

From industrial technological realities such as the digital twins and its underlying industrial internet of things, robotics, additive manufacturing, sensors, or augmented reality capacities, to the development of innovative services within an ecosystem approach, the challenge is to well design the data system. This latter is to process the data, and also to store it, but virtually.

Data processing and management (DPM) needs to define the industrial data lifecycle and how to develop the business value. The defined DPM enables the architecture of data persistence management. ITU Technical Specification D2.1 has standardised a list of 23 capabilities to proceed from the data creation to its use and disposal [15]. From a data point of view, the listed capabilities might affect the state and structure of data, its location, the combination with other data, its transformation, use and disposal. TS-D2.1 also specifies processes to accumulate data for future processing and use. Duration of storage depends on the application, security, privacy and governance rules.

Data Collection and storage is one of the steps in the data lifecycle. Persistence is “the continuance of an effect after its cause is removed" [Datastax]. In the context of storing data in a computer system, this means that the data survives after the process with which it was created has ended. In other words, for a data store to be considered persistent, it must write to non-volatile storage, else a virtual storage that provides equivalence of non-volatility. The investigated EdgeDB [16] sounds a promising solution in front of other solutions we experimented to result in such non-volatility equivalence (namely BerkeleyDB, ObjectBox, Realm).

4. Safety Assurance for AI at Critical CPS

Embedded systems as they are designed for a fixed and known environment, lack events uncertainty which is faced by cyber physical systems. Building distributed AI based CPS framework, increases the need to build a strong safety approach that covers predicted but also unpredicted events. In the context of software defined physical systems, as typically are industry 4.0 facilities, the successful deployment of cyber-physical systems depends on the resilience functionality that should avoid as much as possible the safety risks. Additionally these CPS should have the correct behaviour recovery on a real time, using cognitive technologies at the edge.

Dependability of a system reflects the degree of trust into that system, because it demonstrates [17] (Fig. 6):

- Availability: The probability that the system will be up, running, and able to deliver the expected services.
- Reliability: The probability that the system will correctly deliver services as expected.
- Safety: A judgment of how likely it is that the system will not do what it should not do (typically to cause damage to people or its environment).
- Security: A judgment of how likely it is that the system can resist accidental or deliberate faults.
- Resilience: A judgment of how well a system can maintain the continuity of its critical services in the presence of disruptive events such as failure or fault injection.

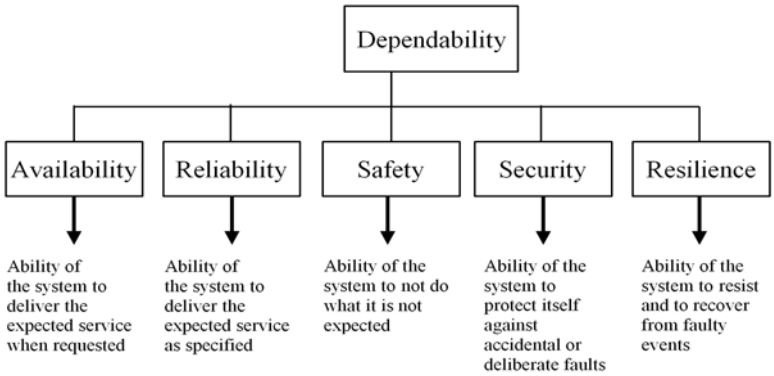


Fig. 6. Dependability attributes [17].

5. Cybersecurity at Virtual Swarm

With the virtualisation of the swarm processing into a virtual operating system, it raises cybersecurity issues. Actually, with the controllers which were previously hardwired and physically partitioned, even though the programmable logic, the cyber-physical system was relatively less exposed. But that's only relatively: the accidentology has already shown since long ago, that cyber-attacks on industrial systems are common [24], [25]. With the virtualization of the whole process control, closely the execution, the cyber risk likelihood may increase with widening the access, standards protocols and multiplication of the objects in the industrial WoT.

Industrial safety remains a concrete reality at stake. Therefore, the distribution system which deploys the swarm processing has also to deploy simultaneously a security architecture, with security functions, mechanisms and their parameters. It could be a strategy of security in the uncertainty, but discussion of such an option is out of current scope. Industrial cyber-physical systems are presumably composed with known components. The IoTs' on which the virtual system deploys shall natively embed a root of trust (as it is commonly the case with ARM-based microcontrollers). Then, the local micro-kernel can boot a safe image and procure a trusted local environment for further execution. Hosted local cybersecurity features are protected by the root of trust mechanisms. Hence the IoT is able to partition securely the device architecture, while interfacing with the deployed virtual operating system. Multiple trusted domains are then in execution on such an IoT platform. What means the IoT device can optimize the usage of its available resources accordingly with some policy (e.g. energy saving, safety critical operation, etc.) [26]. Nevertheless this cloudlet behavior, the primary function of the installation is to run a mission critical process, which shall comply with safety requirements. As explained by Hang and Wu [27], virtualization and isolation are mandatory. Hardware assisted virtualization and the virtual operating system layer (Open vSwitch in their example), build up such a cloudlet element, an edge computing / cloud networking system. Access control mechanisms, resource management, and cryptographic functions implement the cyber-secure environment that enable a proper functioning of the virtual swarm, with hardening against object requests or kind illegal codes.

Critical cyber and cyber-physical systems (CPS) are more and more using predictive AI models. These models help to expand, customize, and optimize the capabilities of the systems, but they are also vulnerable to a new and imminent class of attacks.

Although our current understanding of AI models is very limited, they are already identified to embed vulnerabilities. For instance, so-called "adversarial examples" can be generated algorithmically for defeating neural network models while appearing indistinguishable to human senses [18]. This can cause an autonomous vehicle to crash, illegal messages to bypass filters, while attacks may remain impossible to detect. A second type of vulnerability arises when the adversary provides malicious training samples that result in introducing biases into the cognition model. A third vulnerability is the potential violation of security monitors which provide the training data. More generally, the space of vulnerabilities and their impact on the overall control system are still to well-understand in order to overcome the safety risks that could result from security impacts.

6. Industrial Applications

In this section, we present three different industrial experimental use cases that empower AI-assisted distributed applications at the Edge. Each use case has been implemented thanks to SlapOS, an open source edge computing framework which provides, among others, autonomous intelligence, cognitive technologies and distributed swarm computing features.

Reproducing the IoTwins hybrid digital twin architecture previously described at section 3, the first case has demonstrated the use of GPUs at the edge to support real-time autonomic anomaly detection of the structural health of wind turbines with avoiding unnecessary service downtime, as well as extensive damage to the entire system. Based on models [28] calculated on a cloud-hosted data-lake using Keras deep learning, each edge node was equipped with a smart sensor. As wind turbines are generally deployed in low network coverage zones with limited uplink throughput, it was important to leverage IoT sensors in the wind turbines with powerful edge nodes handling most of the AI logic and reducing its network dependence. Thus, GPU based edge nodes directly connected to the wind turbines sensor were measuring continuous vibration to detect immediate icing, to optimize parallel processing, out-of-core processing, predictive algorithms, and to process data in parallel on each server (map reduce, batch processing, etc.) for higher reliability. It has shown higher effectiveness than older technologies deployed in the field with less network throughput.

The second case demonstrates Nexedi's industrial control implementation based on virtualized PLCs' interconnected with IoTs at industrial layout and business orchestration in the cloud. In the context of food-processing industry, a fruit selection workflow has been proposed combining control of the production line (conveyors, valves, motors, etc.) with a software defined PLC, image recognition based on real time video capture and fault diagnosis process with ML toolkit at the edge.

Such an implementation virtualizes the production line with that software defined PLC powered by SlapOS that is an Edge operating system. Completed with part of the stack running at cloud, it builds up a digital twin which complies with the ISO23247 standard. The software defined PLC layer implements a distributed application, with an optical inspection and detection function, so the conveyor is "trained" to blow away any odd fruit the image recognition detects in the flow. For that purpose, the used ML toolkit was scikit, and openCV for image recognition. The management system interface in the cloud is then replicated as a software element at the edge being closer to the field sensors and actuators while providing an equivalent level of process control.

The third case studies a drone guidance implementation at a fleet, based on artificial intelligence deployed locally at IoT, with navigation at Edge and path planning in the cloud. A fleet of twenty drones were deployed, running autonomous software distributed at the swarm, to detect animals equipped with radio tag. Reporting services were to communicate position, map and sensed metrics in intermittent network conditions. The service provisioning handled the intermittent communication of the control station with the virtual drone system. It was to allocate to each drone a specific flight plan, whenever they were having connectivity access. The flight plan service was deployed at every drone to let the drone fly autonomously. Service operation can execute then, offline, totally autonomous, and independent from cloud interaction.

A mesh communication service was also deployed on all the drones. It was enabling drone to drone to datacenter communication, using a chain of transient links. With this approach, drones were able to deploy in any area with poor radio communication, yet achieving a defined flight plan to scan the area in a minimum amount of time.

7. Conclusions and Future Works

The emergence of Industry 4.0 has accelerated an increasing shift toward industrial fields functioning with software defined processing infrastructures embedding artificial intelligence techniques to perform the industrial operations, autonomous and efficient. Different AI techniques are supporting the operating functions which can deploy at the various domains of the ISO 30141 standardized industrial internet of things. Additionally the virtualization of the devices with software, the Edge Computing technologies have also introduced a next stage of virtualization of the whole physical layer into a software abstraction. Hence, elaborating upon that standard IoT architecture for industrial applications, the digital twin provides a cognitive system for such a virtualized world of industrial things, standardized with ISO23247.

Beyond just virtualizing the devices, the Edge computing technologies are coming with specific

virtual operating systems which enable distributed computation accordingly the resources availability, and also data persistence. This latter emerges kind of a virtual storage in the fog that is a sort of cloudlet. It opens the possibility to cancel the need of uploading amounts of data to the cloud, but to compute locally the industrial processing. Only relevant data are then uploaded to the cloud. Hybrid digital twinning implements then a distribution of a new sort of industrial operating system software that deploys partly at cloud, and partly at fog. That way the AI-based cyber-physical system can function autonomously at the field.

Nevertheless, safety is a high concern for such industrial cyber physical systems. Running with potentially evolving configuration, the intelligent autonomous system requires specific attention for its design. The ISO26262 standard and complementary ISO/PAS 21448 "Safety of the intended functionality", provide an appropriate framework for the development of safe systems which run powered with AI. Tailored safety architecture and subsystems enable to conceive, autonomous intelligent CPS which can be eligible for the functioning of industrial applications and safety critical systems. Cybersecurity at the virtual swarm has to implement the prevention, alerting, and protection of the swarm components. The safety risks that could result from security impacts, are to overcome in order to ensure the safe performance of the AI-based CPS.

Experimental use cases at industrial environments or applications, have explored scenarios of execution with solutions. Although the validation of assumptions, these experiments also showed us limits and gaps, that lead currently to limitations of the technology. Future work on these technology aspects will have to develop software components we are missing for implementing correct Edge computing operating system. Appropriate Edge artificial intelligence can elaborate then on that basis, from a continuation of the industrial use cases.

Acknowledgment

We gratefully thank Dr Oleg Illiashenko and Department 503 of XAI, Ukraine National University of Aerospace in Kharkiv, for the collaboration and valuable recent results about methods and technologies for designing and implementing dependable systems based on the internet of things.

References

[1]. Y. Fourastier, C. Baron, H. Chaouchi, *et al*., Industrial Field Autonomous Systems: AI-assisted Distributed Applications at Edge, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI'2020)*, Berlin, Germany, November 2020, pp. 201-203.

[2]. S. Madakam, R. M. Holmukhe, D. K. Jaiswal, The future digital work force: robotic process automation (RPA), *Journal of Information Systems and Technology Management – Jistem USP*, Vol. 16, 2019.

[3]. J. Shapiro, Genetic Algorithms in Machine Learning, *Machine Learning and Its Applications, Advanced Course on Artificial Intelligence (ACAI'1999)*, *Springer*, 1999, pp. 146-168.

[4]. T. Mitchell, Machine Learning, *McGraw Hill*, 1997.

[5]. I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, *MIT Press*, 2016.

[6]. M. H. Hassoun, Fundamentals of Artificial Neural Networks, *MIT Press*, 1995.

[7]. P. A. Gutierrez, C. Hervas-Martinez, Hybrid Artificial Neural Networks: Models, Algorithms and Data, in *Cabestany J., Rojas I., Joya G. (eds.) Advances in Computational Intelligence IWANN 2011. Lecture Notes in Computer Science*, Vol. 6692. Springer, Berlin, Heidelberg, 2011.

[8]. H. Tian, S. C. Chen, M. L. Shyuy, S. H. Rubin, Automated Neural Network Construction with Similarity Sensitive Evolutionary Algorithms, in *Proceedings of the 20th IEEE Int. Conf. on Information Reuse and Integration for Data Science*, Los Angeles, CA, USA, July 30-August 1, 2019, pp. 283-290.

[9]. I. Saenko, F. Skorik, I. Kotenko, Application of Hybrid Neural Networks for Monitoring and Forecasting Computer Networks States, in Cheng L., Liu Q., Ronzhin A. (eds), Advances in Neural Networks – ISNN 2016, ISNN 2016, Lecture Notes in Computer Science, Vol. 9719, *Springer*, Cham. 2016.

[10]. A. Bannat, T. Bautze, *et al*., Artificial Cognition in Production Systems, *IEEE Transactions on Automation Science and Engineering* Vol. 8, Issue 1, January 2011, pp. 148-174.

[11]. ISO/IEC 30141:2018, Internet of Things (loT) — Reference Architecture, ISO, 2018.

[12]. H. Hinduja, S. Kekkar, S. Chourasia, H. B. Chakrapani, Industry 4.0: digital twin and its industrial applications, *RIET-IJSET International Journal of Science Engineering and Technology*, Vol. 8, Issue 4, August 2020.

[13]. ISO/DIS 23247-1 Automation systems and integration — Digital Twin framework for manufacturing — Part 1: Overview and general principles, *ISO*, 2018.

[14]. P. Bellavista, A. Mora, Edge Cloud as an Enabler for Distributed AI in Industrial IoT Applications: the Experience of the IoTwins Project, in *Proceedings of the 1st Workshop on Artificial Intelligence and Internet of Things (AI&IOT'2019)*, CEUR Workshop, 2019, Rende, Italy, November 2019.

[15]. ITU-T Focus Group on Data Processing and Management to support IoT and Smart Cities & Communities, ITU-T, Technical Specification D2.1-Data processing and management framework for IoT and smart cities and communities, July 2019.

[16]. Y. Yang, Q. Cao, H. Jiang, EdgeDB: An Efficient Time-Series Database for Edge Computing, *IEEE Access*, Vol. 7, September 2019, pp. 142295-142307.

[17]. A. Avižienis, J.-C. Laprie, B. Randell, Fundamental Concepts of Dependability, LAAS Report no. 01-145, *LAAS-CNRS*, Toulouse, France, 1992.

[18]. Y. Fourastier, C. Baron, C. Thomas, *et al*., Assurance levels for decision making in autonomous intelligent systems and their safety, in *Proceedings of the IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, May 2020.

[19]. J. McDermid, K. Daffey, Safety of artificial intelligence and its role in autonomy: A Maritime Perspective, in *Proceedings of the Safety Critical Systems Symposium*, 2018.
[20]. McDermid, Y. Jia, I. Habli, Towards a Framework for Safety Assurance of Autonomous Systems, in *Proceedings of AISafety 2019*, Macao, China, 2019.
[21]. R. Salay, R. Queiroz, K. Czarnecki, An analysis of ISO 26262: Using machine learning safely in automotive software, *SAE Technical Paper*, *WCX World Congress Experience*, 2018.
[22]. ISO 26262 series - Road vehicles – Functional safety - Part 9: Automotive safety integrity level (ASIL) - Oriented and safety oriented analysis, *ISO*, December 2018.
[23]. ISO/PAS 21448:2019 - Road vehicles — Safety of the intended functionality, *ISO*, January 2019.
[24]. Y. Fourastier, L. Pietre-Cambacedes, *et al*., Cybersécurité des installations industrielles, *Cépaduès*, 2015.
[25]. D. Kapp, L'industrie doit panser le numérique, Dossier Menaces sur l'industrie, *Face Au Risque*, Issue 562, Mai 2020.
[26]. S. Simanta, G. A. Lewis, E. Morris, K. Ha, M. Satyanarayanan, A reference architecture for mobile code offload in hostile environments, in *Proceedings of the Joint Workshop IEEE/IFIP Conference on Software Architecture*, August 2012, pp. 282–286.
[27]. D. Huang, H. Wu, Mobile Cloud Computing: Foundations and Service Models, Chap. 8, *Elsevier Morgan Kaufmann*, 2018.
[28]. O. Kramer, F. Gieseke, B. Satzger, Wind energy prediction and monitoring with neural computation, *Neurocomputing*, Vol. 109, 2013, pp. 84–93.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 65-71

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

The Novel Deep Learning Algorithm and Computer Aided System for Early Prediction of Delirium

[1, *]Jong-ha Lee and [2] Sangwoo Cho

[1] Dept. of the Biomedical Engineering, Keimyung University, Daegu, South Korea
[2] The Center for Advanced Technical Usability and Technologies, Keimyung University, Daegu, South Korea
[*] E-mail: Segeberg@gmail.com

Received: 20 January 2021 /Accepted: 22 February 2021 /Published: 28 February 2021

Abstract: Delirium is generally known to be reversible but can in fact lead to permanent cognitive dysfunction. Despite this, the early detection and reporting rate of delirium in clinical practice is less than 30%. Although there are multiple tools for the early diagnosis of delirium, they require advanced training and are not widely used, which limits their use in real-world practice. This study aimed to use machine learning to classify delirium patients to assess the ease and accuracy of this technique in clinical practice.

Keywords: Artificial Intelligence, Data Processing, Sensor, Mobile Health, Delirium.

1. Introduction

Delirium, also known as acute confusion, is a psychopathological condition that generally occurs over several hours to a few days. The incidence of delirium is thought to be between 22 %–87 %, and varies depending on the population, with the highest incidence among patients in Intensive Care Units (ICU). Delirium occurs suddenly through complicated and variable mechanisms. Fluctuations in consciousness and orientation are characteristics of delirium, but it can also be accompanied by compromised cognition such as decreased concentration and verbal capacity [1].

Both delirium and dementia present as a similar decline in cognitive function, and when assessed by symptoms only, the two conditions are difficult to differentiate based only on the degree of cognitive decline [2]. Nevertheless, delirium and dementia are two independent conditions with different prognoses, and, therefore, it is important to differentiate between the two. The biggest difference between delirium and dementia is the continuity of symptoms. Symptoms of delirium generally occur acutely within hours to days and are corrected within days once the cause has been established and corrected [3]. Symptoms of delirium also show large fluctuations over the course of a day, while symptoms of dementia manifest over months, and their severity tends to be uniform without significant fluctuation. Delirium is a relatively common condition that occurs in 10 %–15 % of all inpatients, particularly in post-operative or elderly patients. The incidence rate of delirium also increases with sudden changes in the environment, such as hospitalization. Indeed, 7 %–47 % of elderly inpatients develop delirium, particularly immediately after admission. Delirium can serve as a warning for physical illnesses and can have a high mortality rate; hence, it is considered a medical emergency. If the underlying pathology is corrected, symptoms can improve within days, but if the cause is uncertain or difficult to correct, it can lead to long-term manifestation. Delirium is generally known to be reversible but can in fact lead to permanent cognitive

dysfunction. Despite this, the early detection and reporting rate of delirium in clinical practice is less than 30 %. Although there are multiple tools for the early diagnosis of delirium, they require advanced training and are not widely used, which limits their use in real-world practice. This study aimed to use machine learning to classify delirium patients to assess the ease and accuracy of this technique in clinical practice [4].

There are multiple tools to assess delirium. In South Korea, a translated version of Gaudreau's delirium assessment tool (2005) is available for nurses to apply easily in clinical settings. This tool is based on the Confusion Rating Scale (CRS), with an added category of psychomotor retardation to assess low-activity delirium, which was a limitation of previously available assessment tools. Gaudreau's delirium assessment tool comprises five categories, including inappropriate behavior, disorientation, illusions/hallucinations, psychomotor retardation, and inappropriate behavior, and provides examples of signs or symptoms that represent each category. Nurses can refer to the said examples to assess the severity of the patient's signs or symptoms and assign a score between 0 and 2. The tool is well utilized because it only takes around 1 min to apply and can be used after minimal training to reduce the time and concentration required for history taking. Based on the Korean version of NU-DESC delirium diagnostic criteria, this tool shows specificity of 97 % and sensitivity of 81 %. However, it has limited capacity to predict delirium in post-operative patients due to its low predictability and lack of accurate risk factors. Another delirium assessment tool, the Confusion Assessment Method for the Intensive Care Unit (CAM-ICU), can be used for patients who are intubated or on machine ventilation and have difficulty communicating. The credibility and feasibility of the CAM-ICU has been shown in multiple studies, with sensitivity and specificity of 77.4 % and 72.4 %, respectively. Although the CAM-ICU is a useful tool for the assessment of delirium, the subject needs to be awake in order to answer the questions, which limits its use in the ICU setting in South Korea where patients are often on mechanical ventilation or sedatives [5-7].

Delirium is characterized by a fluctuation in consciousness and disorientation that is accompanied by cognitive dysfunction such as decreased concentration or verbal acumen and psychiatric symptoms. The incidence rate and healthcare burden for delirium can be significantly reduced with early detection, prevention, and intervention. However, due to lack of knowledge or technology, delirium is often not recognized or confused with other conditions such as dementia. Various modalities such as the brain wave test, computed tomography (CT), magnetic resonance imaging (MRI), and delirium assessment tools are being used to diagnose delirium, but the difficulty of applying these tools in a clinical setting leads to an increased demand for healthcare workforce and an increased cost.

As shown in Fig. 1, the recent Korean study on 256 patients with delirium showed that 101 post-operative patients developed dementia within 6 months of surgery, and that elderly patients with history of delirium had a 9-fold increase in the risk of dementia and an increased risk of permanent cognitive dysfunction. Furthermore, the same study showed that, if prolonged, 40 %–50 % of patients expired within 1 year of the onset of delirium (Asan Medical Center). This study, as a precedent study to diagnose and predict delirium symptoms, used RF, SVM, and Random forest machine learning models to analyze the correlation among risk factors that can distinguish delirium patients early. Through this approach, we ultimately sought to assess the feasibility of diagnosing delirium through machine learning [8-9].

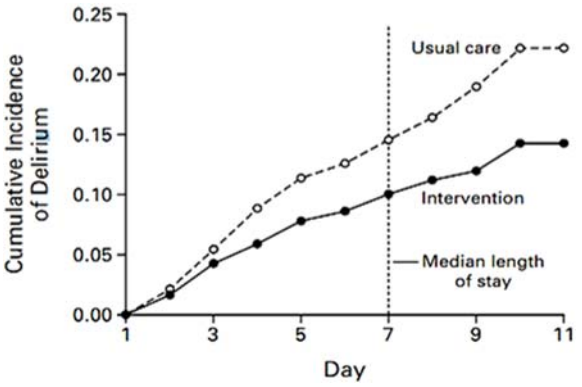


Fig. 1. Cumulative incidence of delirium by group.

2. Materials and Methods

The purpose of this experiment was to gather useful clinical knowledge to assist with the early detection of delirium and to offer an objective capacity index. Prospective cohort study data were used for inpatients at two long-term care facilities in Daegu and Kyeongbuk, with 120 and 100 beds, respectively. The 321 subjects who participated in the 6-month prospective cohort study between October 2016 and March 2017 were categorized into four groups based on the presence of delirium or dementia: patients with no delirium or dementia, those with dementia, those with delirium, and those with delirium leading to dementia. Of the total 321 study subjects, 148 patients who were transferred from a long-term care facility within 12 h of admission, expired during the study period, met the exclusion criteria, or refused to participate were excluded, and the remaining 173 were included in the analysis.

In order to suggest clinical classification criteria for delirium, we re-categorized cohort study patients with no delirium or dementia, or with dementia only to the no delirium group, and those with delirium, or delirium leading to dementia to the delirium group.

Delirium rarely has a single causative factor; rather, it occurs due to a combination of various factors, including age, disease, and environment.

Inouye proposed a multifactorial model that explains the relationship among risk factors for delirium in elderly patients. In this model, the risk factors for delirium were largely divided into predisposing factors and precipitating factors to illustrate the relationship between the two groups. According to this model, patients with multiple predisposing factors such as dementia, comorbidities, and depression showed a high incidence of delirium with only a few precipitating factors such as pain and surgery. Similarly, patients with many precipitating factors showed a high incidence of delirium with only a few predisposing factors. Based on this model, many studies have been conducted on the risk factors for delirium. The major predisposing factors identified from these studies include old age, psychiatric comorbidity, depression, dementia, sleep disorder, nutritional imbalance, alcohol, smoking, drug abuse, and history of delirium. The major precipitating factors identified from these studies include pain, use of restraint, surgery, dehydration, bed sores, and infection.

We aimed to conduct a detailed analysis of the clinical risk factors associated with delirium by analyzing the relationship between each predisposing and precipitating factor.

2.1. Supprot Vector Machine (SVM)

The SVM is a machine learning method that was first proposed by Vapnik (1996) for pattern recognition and data analysis, usually used for classification or regression problems. This model creates a non-probabilistic binary linear classifier model that decides new data categories based on a given data set. The basic principle is to measure the distance between the data from two groups to establish the middle point, and then obtain the optimal hyperplane to learn how to categorize each group. This model defines the decision boundary to classify data and to predict the class of given data by confirming the boundary within which unclassified data fall.

There are countless lines available if data are separated linearly. However, one must determine the optimal line that minimizes the categorization error for data other than the training data set.

In a 3-dimensional space or above, the optimal plane, rather than the line, must be found. In other words, generalizing to the nth dimension, the optimal hyperplane must be found, which in turn becomes the decision boundary. If data cannot be separated linearly, non-linear mapping can be used to convert to a higher dimension to find the optimal linear separation that separates the hyperplane and identify the optimal decision boundary. Mapping in 3-dimension makes it linearly separable. Therefore, using appropriate non-linear mapping in a sufficiently higher dimension will always allow data with two classes to be separated on a hyperplane. The SVM is highly accurate as it can model complicated non-linear decision boundaries, and is also less likely to overfit compared to other models.

2.2. Random Forest (RF)

RF is an ensemble learning model developed by Breiman (2001) that uses a bootstrap method to combine multiple decision trees. In traditional decision trees, the maximum height of the tree can be set by data pruning, but this does not adequately correct for overfitting. RF forms multiple decision trees and processes new data points through each tree simultaneously. Next, it conducts voting within the results categorized by the trees and adopts the result with the most votes as the final categorization result. A tree comprises a hierarchical group of nodes and edges. Nodes are further categorized into internal nodes and terminal nodes (leaf nodes). Decision trees, true to their name, are used for decision making, which divides the decision process into a hierarchy of simple questions. For simple questions, the user can set the parameter, but, for more complicated questions, the model automatically learns both the tree structure and parameters from the training data set. Some trees formed by RF can be overfitted, but this model has the advantage of preventing variability in prediction by forming many trees. The logistic model allows the dependent variable or result value to remain between 0 and 1, regardless of the independent variable. This is obtained by performing odds to logit conversion.

2.3. Logistic Regression

Logistic regression is a probability model that was first proposed in 1958 by a British statistician, D. R. Cox. Logistic regression is a statistical technique that is used to predict the probability of an event by using a linear combination of independent variables. The purpose of logistic regression is similar to that of general regression analysis, in that it aims to represent the relationship between independent and dependent variables in a specific function that can be used for future prediction models.

However, logistic regression differs from linear regression model. The linear regression model can be seen as a classification method, since the dependent variable is categorical, and the result of data input is categorized. The logistic regression model is often used when the dependent variable addresses a binominal problem. If a problem with more than two categories is to be addressed, multinominal logistic regression or polytomous logistic regression is used, and if there are multiple ordinal categories, ordinal logistic regression is used. Logistic regression analysis is widely used for classification and prediction in many fields, including medicine, telecommunication, and data mining.

2.4. K-fold Cross Validation

K-fold cross validation is a method of model analysis in statistics that uses held-out validation (using part of the data set as a validation set) to evaluate the capacity of a model. The limitation of this method is that the small data set compromises the credibility of capacity evaluation for the test set. If the capacity varies by test set, bias occurs in the model evaluation criteria due to coincidence. In order to correct this problem, K-fold cross validation forces all data to be used in at least one test set. This is more time-consuming than normal methods but allows for more learning data sets by classifying data into training and test sets rather than training, validation, and test sets. As a result, it can improve the accuracy for data sets with a smaller number of data points. This study used 10-fold cross validation by dividing the available data into training and test sets and then by dividing training into 10-folds as shown in Fig. 2.

3. Results

3.1. Comparison of Predisposing Factors Between Delirium and Non-delirium Groups

Table 1 shows the comparison of predisposing factors between delirium and non-delirium groups based on the re-classification of earlier cohort study data. The mean age of all patients was 76.9 years, 81.4 years in the delirium group, and 72.7 years in the non-delirium group; thus, the delirium group showed a higher mean age by 8.7 years. In total, 128 of the patients (74 %) were female, and the proportion of female patients was 78.3 %. The Charlson Comorbidities Index (CCI) was higher by 3.5 points in the delirium group, and cognitive impairment (25 %) and medication use (0.8-fold) were also more common in the delirium group compared to the non-delirium group. There was no significant difference in alcohol or BMI between the two groups. Smoking was more common in the non-delirium group. In addition, 46.2 % of patients were transferred from hospital, followed by home and another care facility, showing similar characteristics to the delirium group.

3.2. Correlation Analysis of Risk Factors

The correlation between risk factors identified in previous studies, including age (≥ 65 years), CCI, surgery, sleep disorder, and history of brain damage, was analyzed. The CCI was highly correlated with age and nutritional deficiency, while age ≥ 65 years showed a high correlation coefficient of ≥ 0.5 with medication and cognitive impairment. Bed sores showed high correlation with nutritional deficiency and diaper use. Other factors had relatively low correlation coefficients of ≤ 0.25. Fig. 3 show the results.

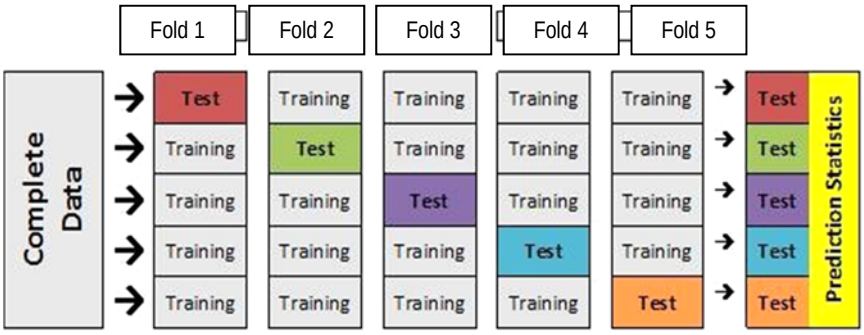


Fig. 2. K-fold cross validation process.

Table 1. Background demographics of participants.

Variable	No delirium	Delirium	Total
Age	72.7 ± 12.1	81.4 ± 8.5	76.9 ± 11.4
Female	63 (70.0)	65 (78.3)	128 (74.0)
BMI (kg/m^2)	21.4 ± 3.4	20.1 ± 3.8	20.8 ± 3.6
CCI	3.1 ± 3.1	6.6 ± 2.9	4.8 ± 3.5
Medication	6.8 ± 3.2	7.6 ± 3.3	7.2 ± 3.2
Smoking	18 (20.0)	11 (13.3)	29 (16.8)
Drinking	8 (8.9)	10 (12.0)	18 (10.4)
Cognitive dysfunction	32 (35.6)	50 (60.2)	82 (47.4)
Home	34 (37.8)	21 (25.3)	55 (31.8)
Hospital	33 (36.7)	47 (56.6)	80 (46.2)
Other facilities	22 (24.4)	15 (18.1)	37 (21.4)
Others	1 (1.1)	-	1 (0.6)

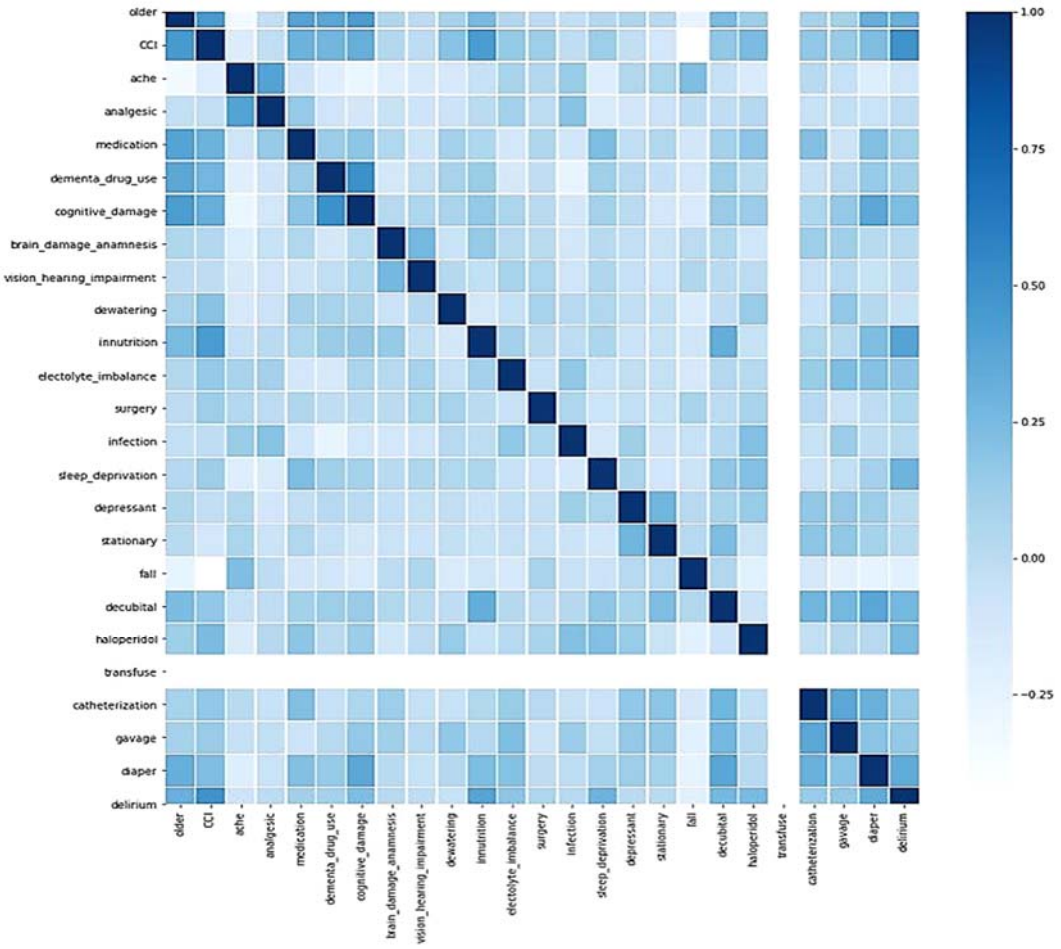


Fig. 3. Correlation analysis of risk factors.

3.3. Risk Factor Analysis of Delirium and Non-delirium Groups

Table 2 shows the results of risk factor analysis comparing the predisposing factors and precipitating factors as selected by previous studies between delirium and non-delirium groups. Predisposing factors that showed higher frequency in the delirium group compared to the non-delirium group included nutritional deficiency, fluid imbalance, sleep disturbance, and delirium medication. Precipitating factors that showed higher frequency in the delirium group included surgery, bed sores, foley catheter, nasogastric feeding, and diaper use. The fall and pain score was greater in the non-delirium group compared to the delirium group. There was no significant difference between the two groups in history of brain damage, infection, dehydration, restraints, etc.

3.4. Model Performance Index Analysis

Fig. 4 and Fig. 5 shows the performance index of each model as assessed by 10-fold cross validation. Of the three models, the RF showed a classification capacity of 0.802 for the delirium group and 0.825 for the non-delirium group. The RF showed the highest performance of the three models, with an average classification capacity as follows: TP rate, 0.813; recall, 0.813; accuracy, 0.813; and ROC area, 0.885. Moreover, the SVM showed an average classification capacity as follows: TP rate, 0.726; recall, 0.724; accuracy, 0.724; and ROC area, 0.724. Logistic regression showed a TP rate of 0.725, recall of 0.724, accuracy of 0.724, and ROC area of 0.791, showing similar results to SVM.

4. Conclusions

This study used machine learning technology to develop a predictive tool for delirium that can be used in a clinical setting to detect and prevent delirium early. Correlation among risk factors categorized as predisposing or precipitating factors in earlier studies was analyzed with group risk factors that were highly correlated. The selected risk factors were than analyzed for their associated with delirium and non-delirium groups to identify clinical factors that can distinguish delirium patients. However, further studies are needed to be able to diagnose delirium based on the identified variables in a real-world clinical setting.

Table 2. Analysis of risk factors for delirium.

Variable	No delirium (n = 90)	Delirium (n = 83)	Total (n = 173)
History of brain damage	8 (8.7)	8 (9.3)	16 (9.1)
Malnutrition	10 (11.9)	37 (46.0)	49 (27.9)
Fluid imbalance	1 (1.3)	9 (8.8)	9 (5.1)
Dehydration	3 (3.1)	3 (3.7)	6 (3.5)
Surgery	3 (3.2)	9 (8.8)	11 (6.0)
Infection	11 (12.2)	12 (13.5)	22 (12.9)
Sleep deprivation	11 (11.8)	33 (37.1)	42 (24.0)
Restraint	1 (1.1)	1 (1.2)	2 (1.2)
Immobilization	3 (3.3)	3 (3.6)	6 (3.5)
Fall	49 (54.4)	26 (31.3)	75 (43.4)
Bedsore	9 (10.1)	26 (29.8)	33 (19.6)
Medication use for delirium	3 (3.1)	16 (18.3)	18 (10.7)
Foley catheter	4 (4.2)	10 (12.2)	14 (8.3)
Nasogastric feeding	4 (4.1)	11 (13.3)	15 (8.7)
Diaper	23 (24.8)	51 (61.2)	74 (42.1)
Pain score	3.2 ± 1.2	2.9 ± 1.3	3.0 ± 1.3

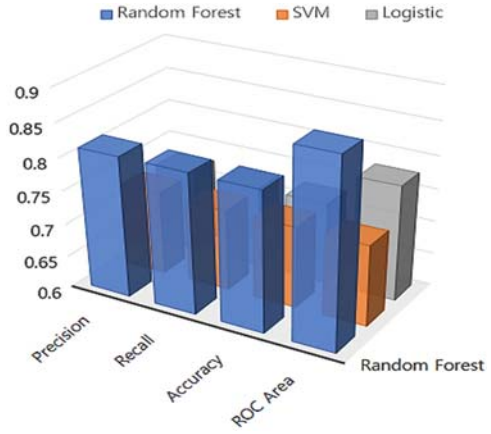


Fig. 4. Performance index analysis.

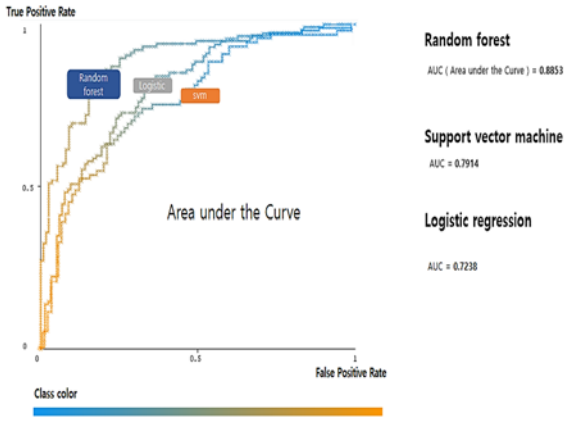


Fig. 5. Comparison of ROC curve.

Three models of machine learning technique that show high performance, RF, SVM, and logistic regression, were used to assess the selection accuracy of machine learning for delirium. A limitation of this study is that the total sample size was small at 173 patients: 83 patients in the delirium group and 90 in the non-delirium group. The inclusion of larger volumes of data would greatly improve the accuracy of the model. Future studies should acquire more data to improve accuracy, and to select classification criteria of the risk factors derived in this study and compare them to other detection tools to assess the clinical applicability of the model. If the capacity of this model is increased, and the accuracy of clinical categorization criteria is improved, healthcare cost and healthcare work force demand for the management and prevention of delirium can be greatly reduced, and treatment efficacy can be improved.

Acknowledgements

This work was supported by the Technology Innovation Program (Development of a novel imaging platform for the screening of cervical cancer; fast speed digital scanner and A.I. engined 3D image reconstruction algorism, B20180275) funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea).

References

[1]. Yu Ki Dong, *et al*., Delirium in acute elderly care unit; prevalence, clinical characteristics, risk factors and prognostic significance, *Journal of the Korean Geriatrics Society*, Vol. 9, Issue 3, 2005, pp. 182-189.

[2]. Lee Eun-Joon, *et al*., Risk factors related to delirium development in patients in surgical intensive care unit, *Journal of Korean Critical Care Nursing*, Vol. 3, Issue 2, 2010, pp. 37-48.

[3]. Saczynski J. S., Marcantonio E. R., Quach L., Fong T. G., Gross A., Inouye S. K., Jones R. N., Cognitive trajectories after postoperative delirium, *New England Journal of Medicine*, Vol. 367, Issue 1, 2012, pp. 30-39.

[4]. Maldonado J. R., Pathoetiological model of delirium: a comprehensive understanding of the neurobiology of delirium and an evidence-based approach to prevention and treatment, *Critical Care Clinics*, Vol. 24, Issue 4,

2008, pp. 789-856.
[5]. Lin S. M., Liu C. Y., Wang C. H., Lin H. C., Huang C. D., Huang P. Y., Kuo H. P., The impact of delirium on the survival of mechanically ventilated patients, *Critical Care Medicine*, Vol. 32, Issue 11, 2004, pp. 2254-2259.
[6]. Bergeron N., Dubois M. J., Dumont M., Dial S., Skrobik Y., Intensive Care Delirium Screening Checklist: evaluation of a new screening tool, *Intensive Care Medicine*, Vol. 27, Issue 5, 2001, pp. 859-864.
[7]. Maldonado J. R., Pathoetiological model of delirium: a comprehensive understanding of the neurobiology of delirium and an evidence-based approach to prevention and treatment, *Critical Care Clinics*, Vol. 24, Issue 4, 2008, pp. 789-856.
[8]. Ouimet S., Riker R., Bergeon N., Cossette M., Kavanagh B., Skrobik Y., Subsyndromal delirium in the ICU: evidence for a disease spectrum, *Intensive Care Medicine*, Vol. 33, Issue 6, 2007, pp. 1007-1013.
[9].Breitbart William, Christopher Gibson, Annie Tremblay, The delirium experience: delirium recall and delirium-related distress in hospitalized patients with cancer, their spouses/caregivers, and their nurses, *Psychosomatics*, Vol. 43, Issue 3, 2002, pp. 183-194.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 72-92

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Performance Analysis and Comparison of Sequence Identification Algorithms in IoT Context

[1, 2] P. -S. GREAU-HAMARD, [1, 2] M. DJOKO-KOUAM and [2] Y. LOUET

[1] Informatics and Telecommunications Laboratory, ECAM Rennes Louis de Broglie, Campus de Ker-Lann, 2 Contour Antoine de Saint-Exupéry, 35170 Bruz, France

[2] Signal, Communication, and Embedded Electronics (SCEE) team, Institute of Electronic and Telecommunications of Rennes (IETR) – UMR CNRS 6164, CentraleSupélec, Campus de Rennes, Avenue de la Boulaie, 35510 Cesson-Sévigné, France

E-mail: pierre-samuel.greau-hamard@ecam-rennes.com, moise.djoko-kouam@ecam-rennes.com, Yves.Louet@centralesupelec.fr

Received: 15 October 2020 /Accepted: 15 December 2020 /Published: 28 February 2021

Abstract: In the fast developing world of telecommunications, it may prove useful to be able to analyse any protocol one comes across, even if it is unknown. To that end, one needs to get the state machine and the frame format of the protocol. These can be extracted from network and/or execution traces via Protocol Reverse Engineering (PRE). In this paper, we aim to evaluate and compare the performance of three algorithms used as part of three different PRE systems of the literature: Aho-Corasick (AC), Variance of the Distribution of Variances (VDV), and Latent Dirichlet Allocation (LDA). In order to do so, we suggest a new meaningful metric complementary to precision and recall: the fields detection ratio. We implemented and simulated these algorithms in an Internet of Things (IoT) context, and more precisely on ZigBee Data Link Layer frames. The results obtained clearly show that the LDA algorithm outperforms AC and VDV.

Keywords: Protocol reverse engineering, AC, VDV, LDA, Performance comparison.

1. Introduction

With the ever growing development of telecommunications, and especially Internet of Things (IoT), a lot of new protocols are constantly appearing. In order to know what they are used for, we need to understand how they work.

In this paper, we place ourselves in the context of a communicating object coming into an unknown environment and wanting to establish a communication with the existing networks. To that end, the object needs to have 'generic', or 'multi-standard' behavior, i. e. to be able to adapt itself to whichever standard is used in the target environment. It is the same goal as the one pursued by Software Defined Radio, except that in our case, we propose to learn the unknown protocol of the environment, and not just identify it from a database.

This is the goal of Protocol Reverse Engineering (PRE), a family of techniques which aims at reconstructing the frame formats and/or the state machine of a target unknown protocol through analyzing execution traces and/or network traces.

There is no precisely defined procedure to perform PRE, but the most encountered one [1] is a five-step process.

1) Firstly, the radio traffic is intercepted and the frames issued by the targeted protocol are isolated.

2) Next, the meaningful binary sequences (features) of these frames are identified.

3) Then the frames are grouped by format via the use of these features.

4) Within each group, sequence alignment is performed, and, finally.

5) The frame formats and/or the state machine of the targeted protocol are reconstructed.

In this paper, we focus solely on the second step, the identification of remarkable sequences. This step aims at reducing the quantity of information needed to label a frame. This is achieved by identifying the remarkable sections of the frames, i. e. in our case by spotting the recurring sequences and their positions. Such sequences are most probably keywords. Our goal is to evaluate and compare the performance of different techniques achieving this, in order to obtain useful data for choosing a technique or a family of techniques to be used in a real-life system. To this end, we selected the Variance of the Distribution of Variances (VDV) [2], Aho-Corasick (AC) [3], and Latent Dirichlet Allocation (LDA) [4] techniques.

The simulation context in which we will simulate the performance of these techniques lies in the analysis of Data Link Layer (DLL) frames of the ZigBee protocol.

Most of the surveys in the PRE domain involve comparing a rather narrow range of tools and their approaches without delving into the exact mechanics or presenting their performance, like in [5]. However, some of them are more exhaustive, and present in detail the techniques used by the tools [6] and the protocols they are able to reverse engineer [7].

Nevertheless, these surveys do not refer to the performance of the different tools in a quantifiable way, and they also do not present the individual performance of the techniques used in each tool. Such an approach is legitimate, as they browse a wide range of PRE tools, but this is where the particularity of our paper stands. We select only three techniques as opposed to the dozens present in the previous surveys, and we evaluate their performance through simulations, which has not been done in the previous papers.

This article is an extended version of the paper presented at the ASPAI'2020 conference [8], with more detailed explanations of the compared techniques, and complementary simulation results.

The rest of this paper is organized as follows: Section 2 presents the theory related to the three techniques studied; in Section 3, we simulate and compare them; and finally, we conclude in Section 4.

2. State of the Art of the Three Techniques

In this section, we present the principle and mechanisms of each of the sequence identification techniques, as well as the practical algorithms designed from them to fit our context.

2.1. Variance of the Distribution of Variances

This technique aims at statistically identifying in a population the parts which offer the least variability. The following presentation is based on the approach proposed by A. Trifilò, *et al.* [2].

The VDV technique considers a population formed of groups of individuals. The latter are themselves composed of elements which can take different numerical values. Fig. 1 illustrates this assumed data structuring.

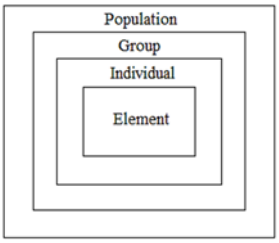


Fig. 1. Structuring of the data (=population) considered by the VDV technique.

The technique unfolds in five steps:

- For each group, calculating the average, then the variance of the value of each element across all the individuals in a given group.
- Across groups, calculating the average, then the variance of these variances.
- Retaining the elements whose variance of the variances is less than a given filtration threshold.

Algorithm 1 in Fig. 2 presents this process under the form of a pseudo-code algorithm.

Algorithm 1: VDV algorithm

```
Data: Data, filtration threshold
Result: Invariant elements positions
for each group do
    for each element do
        Compute the considered element average value
        across individuals within the considered group;
    end
end
for each group do
    for each element do
        Compute the considered element values variance
        across individuals within the considered group;
    end
end
for each element do
    Compute the considered element average variance
    across groups;
end
for each element do
    Compute the considered element variance of the variances
    across groups;
    if Inferior to filtration threshold then
        Select element;
    end
end
```

Fig. 2. VDV pseudo-code algorithm.

The actual algorithm used in our context derives from the technique above, with some modifications.

The groups of individuals previously considered are now replaced by flows composed of DLL frames to be analysed, and the base unit is switched from element to 'token', an n-bit long position slot on the frames which can assume different sequences of n consecutive bits. These equivalences are summarized in Table 1.

Table 1. Summary of equivalent terms for the VDV algorithm.

General term used in the technique explanation	Equivalent term used in our study case
Population	Data Link layer trace
Group	Flow
Individual	Frame
Element	Token

To be able to detect fields regardless of their position, all the possible tokens obtainable from a frame are created.

The filtration threshold actually used for filtering (FT_{VDV}) is not a fixed value, but a value proportional to the average variance of the variances. The proportionality coefficient is called filtration threshold coefficient (FT^c_{VDV}), and the formula linking FT_{VDV} and FT^c_{VDV} is:

$$FT_{VDV} = \overline{V_V(i)} \times FT^c_{VDV}, \quad (1)$$

with $V_V(i)$ the variance of the variances of token i.

The frames being collected on a radio link, it is not possible to clearly identify flows, so we create them by randomly attributing frames to flows following a discrete uniform law.

To enable the detection of fields of different lengths, the algorithm is executed many times, for different values of n, i.e. token lengths, and all the single sequences extracted from these runs are kept for metrics computation.

The successive steps of the process are illustrated in Fig. 3.

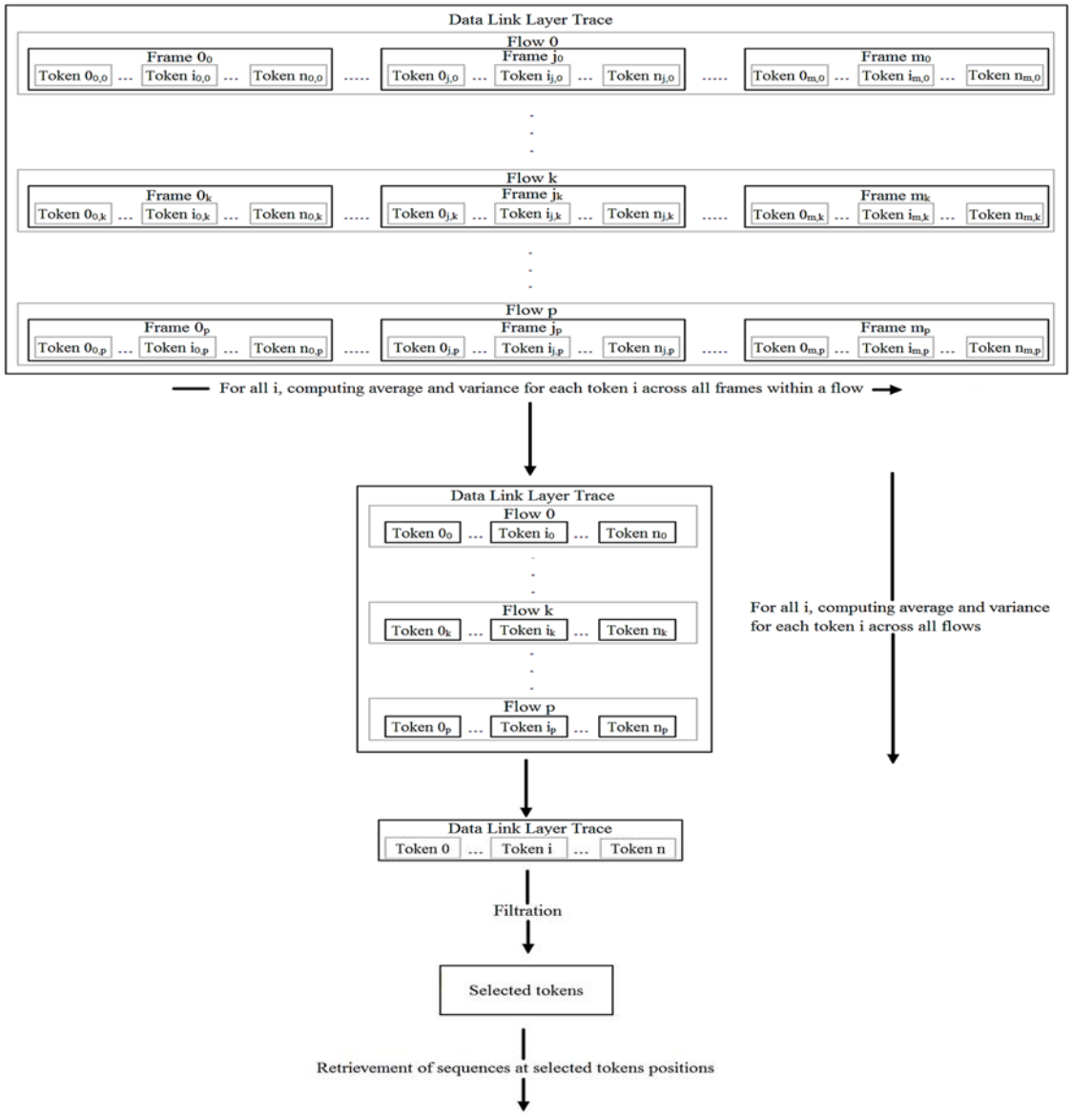


Fig. 3. Illustration of the VDV algorithm applied to our study case.

The parameters of the algorithm are as follows:

- Maximal length of the sequences searched (token length).
- Filtration threshold value.
- Number of generated flows.

2.2. Aho-Corasick

This technique was designed by A. Aho and M. Corasick in order to identify a string of characters in a text [3]. However, its use can be extended to identify any pattern composed of a sequence of elements taking values from a discrete finite space. The search is then run on a sequence of these elements whose length is superior or equal to the targeted pattern. The following presentation is based on the approach proposed by Y. Wang, *et al.* [9].

The particularity of AC is that it is based on a state machine to optimize the processing speed.

This machine is represented as a tree, with each of its states described by the character to be read to access this particular state. This automaton is ruled by three functions:

- A transition function g, defining the next state of the machine, depending on the current state and the character read, when the latter is relevant in regards to the strings searched.
- A failure function f, defining the next state of the machine, depending on the current state, when the character read is not relevant in regards to the strings searched.
- An activity function o, defining the word(s) finishing on a state of the machine, depending on the current state.

The technique operates in two major steps:

- Constructing the state machine based on the strings to search in the text.
- Scanning the whole text character by character, and notifying, for each character, the strings ending on that character.

Algorithm 2 in Fig. 4 presents this process under the form of a pseudo-code algorithm.

Fig. 5 illustrates a sample state machine, built for searching the following strings: pit, it, pity, and paw.

Each circle represents a state. The letter written on each state corresponds to the one to be read to access this state. The empty state is the root of the tree. The thick straight lines materialize the transitions ruled by function g. They are travelled through only in the root to leaves direction. The thin curved lines materialize the transitions ruled by function f. They are travelled through only in the leaves to root direction. The thicker circles on some states mean that one or multiple strings, specified next to the states, end on these states.

The actual algorithm used in our context derives from the one above, with some modifications.

The text considered in the AC technique is now replaced by the DLL trace to be analysed, and the base unit is switched from character to bit. Moreover, the strings to be identified are now all the possible *n*-bit sequences.

Algorithm 2: AC algorithm

```
Data: text,strings
Result: searched strings positions
All elements of o are initialized to 0;
All elements of g and f are initialized to -1;
for every string do
    Generate the corresponding branch of the tree with g;
    Associate the state correspondig to last character
    of the string to the completed string with o;
end
for each character of the dictionary do
    With g, link the root with itself
    for all characters not starting a string;
    for each state associated to this character do
        Insert in a queue the state
        if it corresponds to a string begining;
        if the state do not have any next step according to g then
            if no other state corresponds to this character then
                Link the state to root with f;
            else
                Link with f the state to one of the other
                states corresponding to this character;
                Repeat the linking with f until all states
                corresponding to this character forms a chain,
                and finally link it to root with f;
            end
        end
    end
end

while the queue is not empty do
    Take first element of the queue;
    for each character of the dictionary do
        if there is a corresponding state on a tree branch then
            Find the deepest state which associated string
            is a substring of the string associated to the current state;
            Associate to the found state, in o,
            the string completed on the next state of the branch;
            Introduce the next state of the branch in the queue;
        end
    end
end
Start at the root;
for each character of the text do
    while there is no next state with g do
        Use f to move to next state;
    end
    Use g to move to next state;
    if a string finishes on the current state then
        Get its ending position in the text,
        and deduce its starting position;
    end
end
```

Fig. 4. AC pseudo-code algorithm.

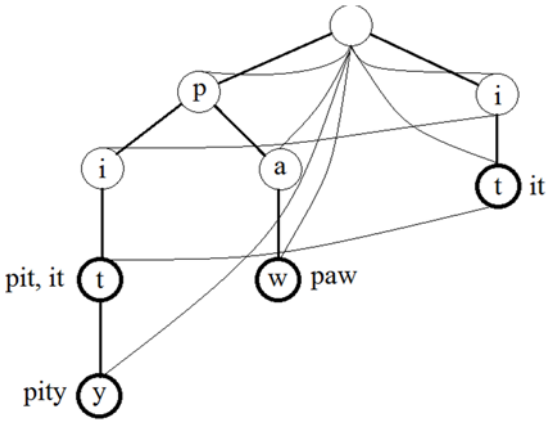


Fig. 5. Illustration of an AC technique example state machine.

These equivalences are summarized in Table 2.

Table 2. Summary of equivalent terms for the AC algorithm.

General term used in the technique explanation	Equivalent term used in our study case
Text	Data Link layer trace
String	Flow
Character	Frame

An occurrence counter of the sequences was added, in order to perform filtering. The sequences under a threshold FT_{AC} proportional to the average number of appearances of any sequence considering a uniform distribution are filtered out. FT_{AC} is given by:

$$FT_{AC} = \frac{n-L+1}{2^L} \times FT_{AC}^{c}, \quad (2)$$

with n the number of bits of the trace, L the length of the sequences searched, and FT_{AC}^{c} the proportionality coefficient called filtration threshold coefficient. This filtering is done to keep only the frequent enough sequences.

The sequences with a similarity level superior to a given threshold are fused. By fusion, we mean that in a group of similar enough sequences, we retain the one best representing all the other ones. We achieve that through unsupervised ascendant hierarchical clustering of the sequences, using the similarity as the distance metric, defined by:

$$\mathrm{Sim}(\mathrm{X},\mathrm{Y}) = \frac{\mathrm{l}(\mathrm{X},\mathrm{Y})-\mathrm{ed}(\mathrm{X},\mathrm{Y})}{\mathrm{l}(\mathrm{X},\mathrm{Y})}, \quad (3)$$

with X and Y representing any two sequences, $l(X,Y)$ the average length of X and Y, and $ed(X,Y)$ the minimal edition distance between X and Y, i. e. the minimal number of operations to apply on one of the sequences to obtain the other one. All the sequences being the same length, the average length of X and Y, $l(X,Y)$, equals those of X and Y.

To enable the detection of fields of different lengths, the algorithm is executed many times, for different values of n, i.e. lengths of sequences, and all the single sequences extracted from these runs are kept for metrics computation.

The parameters of the algorithm are as follows:

- Maximal length of the sequences searched.
- Filtration threshold value.
- Fusion threshold value.

2.3. Latent Dirichlet Allocation

This technique comes from the machine learning domain of Information Retrieval (IR), which aims at modeling a text mathematically, in order to extract its meaning. It was designed with the objective to identify latent topics from a document corpus, and to associate terms coming from a dictionary to them. However, its use can be extended to regrouping sequences of single elements taking values in a finite discrete space, from a collection of data. The following presentation is based on the approach proposed by Y. Wang, *et al*. [10].

The name Latent Dirichlet Allocation stems from the fact that the model uses Dirichlet as the a priori law of the latent variables (topics) allocated to the observed variables (words). Practically, the Dirichlet law is used to generate a probability vector characteristic of a multinomial law, from a vector of concentration parameters. These parameters have values varying from 0 to +∞, 0 meaning that probabilities of the generated probability vector will be 0 or 1 (before normalization) and +∞ meaning probabilities will be equal to 0.5 (before normalization). 1 means that the a priori is in fact that there is no a priori, i. e. probabilities can take any value between 0 and 1 with equal probability. The Dirichlet law was chosen for its conjugacy property with the multinomial law which is used for classification (like LDA). Conjugacy of Dirichlet law and multinomial law means that if the latent variables follow a Dirichlet a priori law, their a posteriori law will be multinomial.

The LDA technique is first and foremost a generative model for a corpus based on a Bayesian network; the actual implemented algorithm is deduced from it upon inference.

Let us introduce the necessary notions and parameters needed to understand the generative model and the inference based on it:

- A word w is an element taking value from a dictionary v gathering all the known vocabulary.
- A document m is a set of words w, modeled by a vector.
- A corpus W is a set of documents m, modeled by a vector.
- A term t is the base element of the vocabulary.
- V is the set of terms of the dictionary or its cardinal.
- K is the set of topics desired or its cardinal.
- M is the set of documents in the corpus or its cardinal.
- α and β are the Dirichlet prior parameters of the topics over documents and the words over topics distributions, respectively.
- ξ is the parameter of the Poisson law determining the number of words in each document.
- $\overrightarrow{\theta_m}$ is the vector characterizing the topics distribution for the document m. $\Theta = \{\overrightarrow{\theta_m}\}_{m=1}^{M}$ is the matrix $M \times K$ characterizing the topics distribution over the documents.
- $\overrightarrow{\varphi_k}$ is the vector characterizing the terms of v distribution for the topic k. $\Phi = \{\overrightarrow{\varphi_k}\}_{k=1}^{K}$ is the matrix $K \times V$ characterizing the terms distribution over the topics.
- N_m represents the number of words in document m.
- $w_{m,n}$ represents the n^{th} word of document m. The vector $\overrightarrow{w_m}$ represents the words of document m.

The vector of vectors $M \times N_m$ $\overrightarrow{w}$, represents the words of the corpus.

- $z_{m,n}$ represents the topic associated to the n^{th} word of document m. The vector $\overrightarrow{z_m}$ represents the topics respectively attibuted to each word of document m. The vector of vectors $M \times N_m$ $\overrightarrow{z}$, represents the topics respectively attributed to each word of the corpus.

Let us illustrate the structures of the vectors of vectors $\overrightarrow{w}$ and $\overrightarrow{z}$ with a random example corpus and three latent subjects considered. We precise that $\overrightarrow{z}$ is filled-in with random topics ids which do not reflect what LDA would really do (outside random initialization). We consider the three following documents in the example corpus:

- Document 1: This is an example.
- Document 2: The documents do not have all the same number of words.
- Document 3: The matrices are vectors of vectors.

$$\overrightarrow{w} = \begin{bmatrix} \text{This} \\ \text{is} \\ \text{an} \\ \text{example} \end{bmatrix} \begin{bmatrix} \text{The} \\ \text{documents} \\ \text{do} \\ \text{not} \\ \text{have} \\ \text{all} \\ \text{the} \\ \text{same} \\ \text{number} \\ \text{of} \\ \text{words} \end{bmatrix} \begin{bmatrix} \text{The} \\ \text{matrices} \\ \text{are} \\ \text{vectors} \\ \text{of} \\ \text{vectors} \end{bmatrix}$$

$$\overrightarrow{z} = \begin{bmatrix} 1 \\ 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \\ 2 \\ 1 \\ 3 \\ 2 \\ 3 \\ 1 \\ 1 \\ 2 \\ 3 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \\ 3 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

In LDA, we consider that a corpus is a set of documents, each of those being composed of a random number of words, where the number is drawn following a Poisson law of parameter ξ. Each of the words takes a value within the dictionary.

In the generative model, to begin, Θ and Φ are randomly generated following a Dirichlet law of parameters α and β, respectively.

Firstly, for each word to be generated of each document to be generated, the topic associated to it is randomly drawn following a multinomial law parameterized by Θ, knowing the document the word is in. Next, the value of the word is drawn from the dictionary, following a multinomial law parameterized by Φ, knowing the previously drawn topic associated with the word.

The generative process of the documents according to LDA is described in Algorithm 3 in Fig. 6, and summarized on the plate diagram in Fig. 7.

The arrows represent the causality links (probability laws) linking the random variables, represented as circles. The zones delimited by dashed rectangles materialize the different sets of elements composing the corpus:

- The topics k;
- The documents m;
- The words n.

The dashed circles correspond to the latent variables, or hidden variables, in opposition to the observed and known variables. The hatched circle is the final, observed variables generated by the model: the words present in the corpus.

It is to be noted that α and β are matrices, and ξ is a vector, but in the LDA generative algorithm, they are expressed as scalars, as we consider all the elements within each of them to be equal. When the actual α, β, and ξ are used, they are generated by replicating the scalar values as many times as needed.

Algorithm 3: LDA generative model

Data: M,K,v,α,β,ξ
Result: $\vec{w}$

for $k = 1$ *to* K **do**
  Generate $\vec{\phi_k}$ according to a Dirichlet law of parameter $\vec{\beta}$:
  $\vec{\phi_k} \sim$ Dirichlet($\vec{\beta}$);
end
for $m = 1$ *to* M **do**
  Generate $\vec{\theta_m}$ according to a Dirichlet law of parameter $\vec{\alpha}$:
  $\vec{\theta_m} \sim$ Dirichlet($\vec{\alpha}$);
  Randomly draw a number of words according to a Poisson law
  of parameter ξ : $N_m \sim$ Poisson(ξ);
  for $n = 1$ *to* N_m **do**
    Randomly draw a topic $z_{m,n}$ according to a multinomial law
    of parameter $\vec{\theta_m}$: $z_{m,n} \sim$ multinomial($\vec{\theta_m}$);
    Randomly draw a word $w_{m,n}$ in v according to a multinomial law
    of parameter $\vec{\phi_{z_{m,n}}}$: $w_{m,n} \sim$ multinomial($\vec{\phi_{z_{m,n}}}$);
  end
end

Fig. 6. LDA pseudo-code generative algorithm.

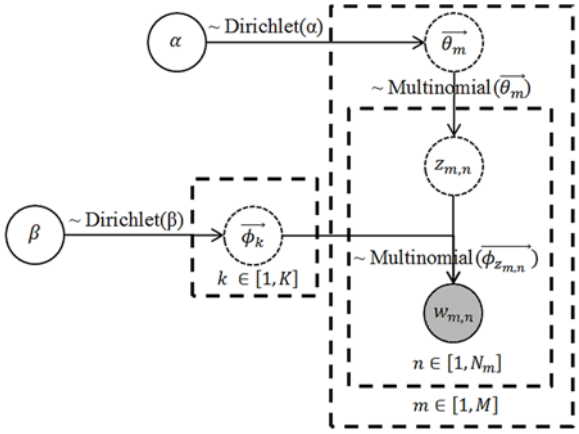


Fig. 7. Plate diagram of the LDA generative model.

Moreover, ξ does not have any use outside the generative algorithm, so it will not be mentioned anymore.

The goal of the LDA technique is to infer the terms over topics and topics over documents distributions, i. e. the matrices Φ and Θ, from the corpus of documents.

Given the parameters α and β, and a document m, the joint probability of all the observed variables ($\overrightarrow{w_m}$) and hidden variables ($\overrightarrow{z_m}$, $\overrightarrow{\theta_m}$, Φ) concerned by m is given by:

$$p(\overrightarrow{w_m}, \overrightarrow{z_m}, \overrightarrow{\theta_m}, \Phi|\alpha, \beta) = \prod_{n=1}^{N_m} p(w_{m,n}|\overrightarrow{\Phi_{z_{m,n}}}) p(z_{m,n}|\overrightarrow{\theta_m}) p(\overrightarrow{\theta_m}|\alpha) p(\Phi|\beta), \quad (4)$$

where $\overrightarrow{\Phi_{z_{m,n}}}$ represents the probability vector of the matrix Φ associated to $z_{m,n}$, the topic associated to the nth word of document m.

The joint probability of $\overrightarrow{w_m}$ and $\overrightarrow{z_m}$ is obtained by integrating over $\overrightarrow{\theta_m}$ and Φ:

$$p(\overrightarrow{w_m}, \overrightarrow{z_m}|\alpha, \beta) = \iint p(\overrightarrow{\theta_m}|\alpha) p(\Phi|\beta) \prod_{n=1}^{N_m} p(w_{m,n}|\overrightarrow{\Phi_{z_{m,n}}}) p(z_{m,n}|\overrightarrow{\theta_m}) \, d\Phi d\overrightarrow{\theta_m} \quad (5)$$

The marginal distribution of $\overrightarrow{w_m}$ is obtained by summing $p(\overrightarrow{w_m}, \overrightarrow{z_m}|\alpha, \beta)$ over all the possible values of $z_{m,n}$, i. e. all the topics K:

$$\begin{aligned} p(\overrightarrow{w_m}|\alpha, \beta) &= \iint p(\overrightarrow{\theta_m}|\alpha) p(\Phi|\beta) \prod_{n=1}^{N_m} \sum_{z_{m,n}} p(w_{m,n}|\overrightarrow{\Phi_{z_{m,n}}}) p(z_{m,n}|\overrightarrow{\theta_m}) \, d\Phi d\overrightarrow{\theta_m} \\ &= \iint p(\overrightarrow{\theta_m}|\alpha) p(\Phi|\beta) \prod_{n=1}^{N_m} \sum_{z_{m,n}} p(w_{m,n}|\overrightarrow{\theta_m}, \Phi) \, d\Phi d\overrightarrow{\theta_m} \end{aligned} \quad (6)$$

The joint probability of all words in the corpus and their associated topics is given by:

$$p(\vec{w}, \vec{z}|\alpha, \beta) = \prod_{m=1}^{M} p(\overrightarrow{w_m}, \overrightarrow{z_m}|\alpha, \beta) \quad (7)$$

The marginal probability of all the words in the corpus is given by:

$$p(\vec{w}|\alpha, \beta) = \prod_{m=1}^{M} p(\overrightarrow{w_m}|\alpha, \beta) \quad (8)$$

We aim at computing the a posteriori distributions of $\vec{z}$, Φ, and Θ, given the words $\vec{w}$ observed. However, Φ and Θ can be computed directly from the distribution of $\vec{z}$, so we consider only the latter. The target distribution is consequently as follows:

$$\begin{aligned} p(\vec{z}|\vec{w}, \alpha, \beta) &= \frac{p(\vec{w}, \vec{z}|\alpha, \beta)}{p(\vec{w}|\alpha, \beta)} \\ &= \frac{\prod_{m=1}^{M} p(\overrightarrow{w_m}, \overrightarrow{z_m}|\alpha, \beta)}{\prod_{m=1}^{M} \sum_{z_{m,n}} p(\overrightarrow{w_m}, \overrightarrow{z_m}|\alpha, \beta)} \end{aligned} \quad (9)$$

This distribution is intractable because of its denominator being a sum over K^M terms. It will therefore be estimated through Gibbs sampling [11].

The Gibbs sampling technique comes from observing that it is impossible to simultaneously infer

all the latent variables of the model (i. e. the topics). Random initialization. So, instead, one at a time, their distributions are inferred conditionally to all the other ones, then a new realization of the inferred distribution is drawn. When repeating this operation over all the variables a large number of times, theory shows that the realizations drawn (i. e. the sample) eventually converge towards what would be sampled from the target distribution. Then, the properties of the distribution can be statistically computed from the sample.

Applied to the LDA, Gibbs sampling unfolds in the following two steps:

- The initialization, where the words are randomly given associated topics.

- An iterative process during which, for each word $w_{m,n}$, the probability to get each of the topics for this word given the topics attributions to all the other words of the corpus is computed. The obtained probability vector parameterizes a multinomial law, which is used to draw a new topic for the word $w_{m,n}$ This step is repeated until a stopping criterion is reached.

We can then compute the terms over topics and topics over documents distributions, i. e. the matrices Φ and Θ, from the vectors of vectors $\overrightarrow{w}$ and $\overrightarrow{z}$. We can finally make a decision on which words to associate to which topics and which topics to associate to which documents based on Φ and Θ.

Fig. 8 illustrates the principle of Gibbs sampling applied to LDA. To that end, we define the following terms:

- The vector of vectors of the topics associated to the words, $\overrightarrow{z} = \{z_{i,j}\}_{1\leq i\leq M, 1\leq j\leq n(i)}$, with M the number of documents and n(i) the number of words of document i.

- The vector of vectors of the topics associated to the words to which the word $w_{i,j}$ is excluded, $\overrightarrow{z_{-ij}} = \{z_{m,p}\}_{1\leq m\leq M, 1\leq p\leq n(i), m\neq i, p\neq j}$.

- The vector of vectors representing the corpus, $\overrightarrow{w} = \{w_{i,j}\}_{1\leq i\leq M, 1\leq j\leq n(i)}$.

- The vector parameterizing the multinomial distribution of the topic $z_{i,j}$, associated to the word $w_{i,j}$, $\overrightarrow{P_{z_{i,j}}} = \{P^k_{z_{i,j}}\}_{1\leq k\leq K}$.

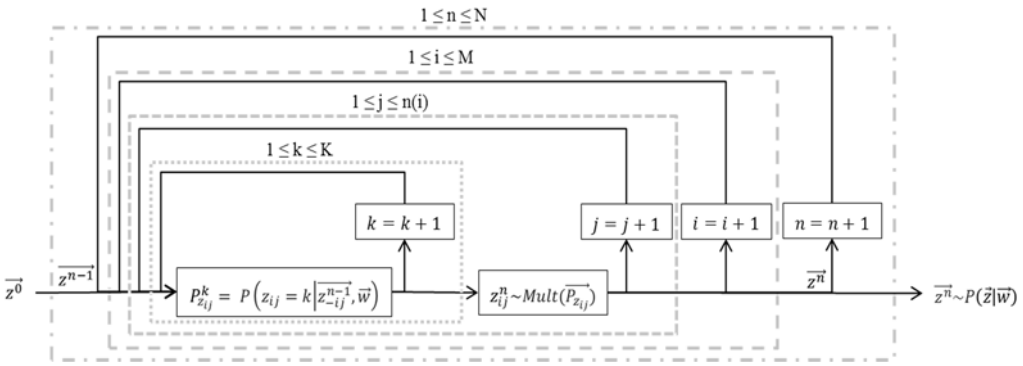


Fig. 8. Algorigram of the Gibbs sampler procedure.

- N, the number of iterations of the algorithm (a possible stopping criterion).

The perplexity [12] expresses the ability of a model to generalize to unknown data. It is defined as the reciprocal of the geometric mean by element of a test corpus likelihood, given the model with its current parameters (which change at each learning iteration).

More practically, in the case of the LDA, it is the inverse of the average likelihood by word of the test corpus given the current state of the matrices Φ and Θ. The perplexity is expressed as follows:

$$perplexity(W_{test}) = \prod_{m\in M} p(\overrightarrow{w_m})^{-\frac{1}{N}} \qquad (10)$$

The information conveyed by the perplexity on the quality of the learned model can be interpreted as follows: for a given word slot, the probability to obtain with this model the word actually present in the test corpus will be the inverse of the perplexity. This probability is to be compared to that of a draw from a uniform law over all the words of the dictionary.

We will now present in Algorithm 4 in Fig. 9 the learning procedure of the LDA model via Gibbs sampling, leading to the obtention of $\overrightarrow{z}$, Θ, and Φ, given the observed corpus W, the number of keywords assumed K, the dictionary v, and the Dirichlet a priori parameters α and β.

The notation –i means all elements of the considered ensemble, except the one in position i.

The actual algorithm used in our context derives from the above, with some modifications.

The documents considered in the LDA technique are replaced by the DLL frames to be analysed, so the corpus consequently becomes the DLL trace. The topics are replaced by keywords of the protocol, and the words by n-grams, groups of n consecutive bits. The n-grams having no natural delimiters, like spaces for words, all the possible n-grams obtainable from a frame are created. The terms become the different possible sequences of n bits. These equivalences are summarized in Table 3.

Algorithm 4: LDA learning algorithm via Gibbs sampling

Data: W,K,v,α,β
Result: $\vec{z}$, Φ, and Θ

Initialize to 0 $n_m^{(k)}$, representing the number of time
a word of document m is assigned to topic k;
Initialize to 0 n_m, representing the number of words in document m;
Initialize to 0 $n_k^{(t)}$, representing the number of time
the term t is assigned to topic k;
Initialize to 0 n_k, representing the number of time
a word is assigned to topic k;
for $m = 1$ *to* M **do**
  for $n = 1$ *to* N_m **do**
    $z_{m,n}$ = k ~ multinomial(1/K);
    $n_m^{(k)} + 1$, $n_m + 1$, $n_k^{(t)} + 1$, $n_k + 1$;
  end
end

while *stopping criterion not verified* **do**
  for $m = 1$ *to* M **do**
    for $n = 1$ *to* N_m **do**
      The word $w_{m,n}$ is a term currently assigned to topic k;
      $n_m^{(k)} = n_m^{(k)}$ - 1;
      $n_m = n_m$ - 1;
      $n_k^{(t)} = n_k^{(t)}$ - 1;
      $n_k = n_k$ - 1;
      for $i = 1$ *to* K **do**
        Compute probability

$$p(z_i|\vec{z_{-i}}, \vec{w}) \propto \frac{n_{k,-i}^{(t)} + \beta}{\sum_{t=1}^{V} n_{k,-i}^{(t)} + V \times \beta} \times \frac{n_{m,-i}^{(k)} + \alpha}{\sum_{k=1}^{K} n_{m,-i}^{(k)} + K \times \alpha};$$

      end
      Normalise these probabilities : $p(z_i|\vec{z_{-i}}, \vec{w}) = \frac{p(z_i|\vec{z_{-i}}, \vec{w})}{\sum_{i=1}^{K} p(z_i|\vec{z_{-i}}, \vec{w})}$;
      Draw randomly a new topic : k' ~ multinomial(p($z_i|\vec{z_{-i}}$,$\vec{w}$));
      Assign the word $w_{m,n}$ to topic k';
      $n_m^{(k')} = n_m^{(k')} + 1$;
      $n_m = n_m + 1$;
      $n_{k'}^{(t)} = n_{k'}^{(t)} + 1$;
      $n_{k'} = n_{k'} + 1$;
    end
  end
end

Compute Φ, with $\phi_{k,t} = \frac{n_k^{(t)} + \beta}{\sum_{t=1}^{V} n_k^{(t)} + V \times \beta}$;

Compute Θ, with $\theta_{m,k} = \frac{n_m^{(k)} + \alpha}{\sum_{k=1}^{K} n_m^{(k)} + K \times \alpha}$;

Fig. 9. LDA pseudo-code learning algorithm via Gibbs sampling.

Table 3. Summary of equivalent terms for the LDA algorithm.

General term used in the technique explanation	Equivalent term used in our study case
Corpus	Data Link layer trace
Document	Frame
Topic	Keyword
Word	N-gram
Term	N-bits sequence

The dictionary is composed of all the possible n-grams with n bits.

The gradient of perplexity between two iterations is used as the stopping criterion of the Gibbs sampler. The perplexity P of a learning corpus W is practically calculated as follows:

$$P(W) = e^{-\frac{\sum_{m \in M} \ln p(\overrightarrow{w_m})}{\sum_{m \in M} N_m}}, \quad (11)$$

with M the set of frames from the learning DLL trace, N_m the number of n-grams in the DLL frame m, and

$$p(\overrightarrow{w_m}) = \prod_{t \in V} (\sum_{k \in K} \theta_{m,k} \varphi_{k,t})^{N_m^t}, \quad (12)$$

with *V* the set of the single *n*-grams, *K* the set of the keywords, and N_m^t the number of occurrences in document *m* of the single *n*-gram *t*.

We calculate the perplexity gradient ∇P between two consecutive time indexes *n-1* and *n* as follows:

$$\nabla P(n, n-1) = \frac{|P(n) - P(n-1)|}{P(n)} \quad (13)$$

The sampling continues as long as the perplexity gradient is above a threshold defined as the maximal perplexity gradient divided by the number of frames in the DLL trace.

Once the matrices Θ and Φ are calculated, for each keyword, the *n*-grams with the highest appearance probabilities are selected. For that purpose, the *n*-grams are ordered by descending probabilities, then iterated through, calculating the gradient within each pair of consecutive *n*-gram probabilities. Mathematically, if we consider a keyword *k*, and its associated *n*-gram distribution vector, $\overrightarrow{\varphi_k}$, in descending order, the probability gradient ∇p_k of a *n*-gram in position $n \in [2, V]$ is:

$$\nabla p_k(n) = \frac{|\varphi_k(n) - \varphi_k(n-1)|}{\varphi_k(n)} \quad (14)$$

The first *n*-gram is always selected, and the others are selected if their probability gradient is under the threshold defined as the maximal probability gradient.

To enable the detection of fields of different lengths, the algorithm is executed many times, for different values of *n*, i.e. *n*-gram lengths, and all the single sequences extracted from these runs are kept for metrics computation.

The parameters of the algorithm are as follows:

- Maximal length of the sequences searched (n-grams length);
- Number of latent keywords assumed;
- Maximal perplexity gradient value;
- Maximal probability gradient value;
- α a priori Dirichlet parameter;
- β a priori Dirichlet parameter.

3. Comparative Simulations

In this section, we present the metrics (including new ones proposed in this paper) used to quantify the performance of the algorithms, as well as the parameterization for the simulations. We then discuss the results produced.

3.1. Performance Metrics

In order to define the metrics quantifying the performance of the algorithms, we introduce the notions of sequence, field, and matching condition as follows:

- **Sequence**: a sequence *s* is characterized by its length *l*, value *v*, and the set of its positions $p = \{p_i\}_{i \in \mathbb{N}}$. A sequence is then represented by $s(l, v, p)$. The list of the detected sequences $S = \{s\}$ is given by the identification algorithms.

- **Field**: a field *c* is characterized by its possible lengths *L*, characteristic values *V* (null if absent), and possible positions $P = \{P_i\}_{i \in \mathbb{N}}$. A field is then represented by $c(L, V, P)$. The list of the fields $C = \{c\}$ is obtained from the ZigBee specification. A field can be either detectable ($V \neq \emptyset$) or not detectable ($V = \emptyset$).

- **Matching condition**: the sequence $s(l, v, p)$ and the field $c(L, V, P)$ match if the following property is verified:

$$\begin{cases} l \in L, v \in V, p \cap P \neq \emptyset, if\ V \neq \emptyset \\ l \in L, p \cap P \neq \emptyset, if\ V = \emptyset \end{cases} \quad (15)$$

As metrics, we first use a trade-off between precision and recall, the F score [13] which is the harmonic mean of the two. This metric quantifies the quality of the sequence detection, i. e. the performance of the algorithm. It is given by:

$$F = \frac{2 \times P \times R}{P + R}, \quad (16)$$

with P the precision, quantifying the part of what was correctly detected from the totality of what was detected, given by:

$$P = \frac{T_p}{T_p + F_p}, \quad (17)$$

with T_P the true positives, which are the sequences correctly detected, and F_P the false positives, which are the sequences incorrectly detected.

R is the recall, quantifying the part of what was correctly detected from what was supposed to be detected, given by:

$$R = \frac{T_p}{T_p + F_n}, \quad (18)$$

with F_n the false negatives, which are the sequences incorrectly not detected.

Each sequence counts as one true positive if its properties match at least one detectable field, it counts as one false positive in all other cases.

The number of false negatives is equal to the difference between the number of remarkable sequences across all fields and the number of true positives.

In addition to the quality of the sequence detection, we want to quantify the quality of the field detection, as it is more representative of the usefulness of the algorithm.

To the best of our knowledge, a metric serving that purpose does not exist, so we introduce our own, the **fields detection ratio**. It quantifies the detected information of a field, and comes in three different forms: q_l for lengths, q_v for values, and q_p for positions.

Let us consider $S' = \{s \in S | eqn(15)\}$ the set of sequences matching at least one field, and $c_i(L_i, V_i, P_i)_{i \in I}$, a generic field, with $L_i = \{L_{ij}\}_{j \in \mathbb{N}}$, $V_i = \{V_{ij}\}_{j \in \mathbb{N}}$, $P_i = \{P_{ij}\}_{j \in \mathbb{N}}$, and I the set of all existing fields, and $s_k(l_k, v_k, p_k)$ a generic sequence.

$$q_{l_i} = \frac{Card(L_i')}{Card(L_i)}, q_{v_i} = \frac{Card(V_i')}{Card(V_i)}, \quad q_{p_i} = \frac{Card(P_i')}{Card(P_i)}, \tag{19}$$

are, respectively, the ratios between the number of possible lengths, values, and positions of c_i detected and the total number of possible lengths, values, and positions of c_i, with $L_i' = \{L_{ij} | \exists s_k \in S' | l_k = L_{ij}\}$, $V_i' = \{V_{ij} | \exists s_k \in S' | v_k = V_{ij}\}$, $P_i' = \{P_{ij} | \exists s_k \in S' | \exists p_k | P_{ij} \in p_k \cap P_i\}$.

For each of the previous ratios, the average over all fields is calculated as follows:

$$\mu_q = \sum_{i \in I} \frac{q_i}{Card(I)}, \tag{20}$$

with q_i representing the different ratios presented in (19).

These averages are the metrics we use for our simulations.

3.2. Simulation Context

In order to evaluate the performance of the algorithms VDV, AC, and LDA, we simulate them with a DLL trace of a protocol widespread in the IoT: ZigBee [14]. This protocol is based on the standard IEEE 802.15.4 [15], widely used in the IoT for physical and data link layers.

The DLL traces to be analysed are generated by randomly creating frames from a data frame formats base we created according to ZigBee specification. The trace generation process is run at each single simulation, so the traces analysed are never the same (although they respect the same statistical properties). The traces are then processed by the algorithms in an offline procedure.

The comparative simulation was done in two steps.

1) Observing the influence of each of the algorithms parameters by simulating them over an arbitrary but wisely chosen parameters set domain.

2) Choosing the best performing parameter set among all the ones simulated, and comparing the performance achieved.

This method allowed us to get parameter sets yielding good performance, but not the best that could be achieved by the algorithms. This is due to the fact that we used a simple optimization approach, considering that all the parameters influence the algorithms in an independent manner.

We chose to simulate all the algorithms with a number of frames varying from 1 to 1000 to see the impact of the trace size on the performances, with a logarithmic increment to cover the domain with fewer points. We limited ourselves to 1000 frame traces for processing power and memory usage limitations of our simulating hardware.

We know that most of the sequences to be found in the test datasets have a length inferior or equal to 8 bits. We also observed a steep increase in the processing time when looking for up to 16-bit sequences, the longest actually present in the datasets. Therefore, the maximal length of the sequences searched will vary from 1 to 8 bits by power of two increments.

Note that each curve point is calculated by averaging over 100 runs of the simulation corresponding to this point parameter set.

The metrics presented in the results graphs will be the following ones:

- The precision of the detected sequences.
- The recall of the detected sequences.
- The F score of the detected sequences.
- The average fields lengths detection ratio (only for the comparison of the best performances).
- The average fields values detection ratio (only for the comparison of the best performances).
- The average fields positions detection ratio (only for the comparison of the best performances).

3.3. Study of VDV Parameters Influence

We first run our implementation of the VDV algorithm with a series of parameter sets to study their influence.

Table 4 summarizes the parameters used for the simulations of the VDV algorithm.

Table 4. Summary of the VDV algorithm simulation parameters.

Parameters	Sequences length influence	Flows number influence	Filtration threshold influence
Number of frames	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment
Sequences length	1 to 8, power of 2 increment	8	8
Number of flows	10	5 to 30, increment of 5	10
Filtration threshold	1	1	0.0001 to 1, power of 10 increment

In the graph Fig. 10, we can see first of all that the precision of the VDV algorithm is inversely proportional to the maximal length of the sequences searched. This can be explained by the fact that the longer the sequences searched are, the more different

sequences can be detected, and therefore the lower their respective numbers of appearances are, so the statistical thresholding is less efficient, hence the increase in false positives.

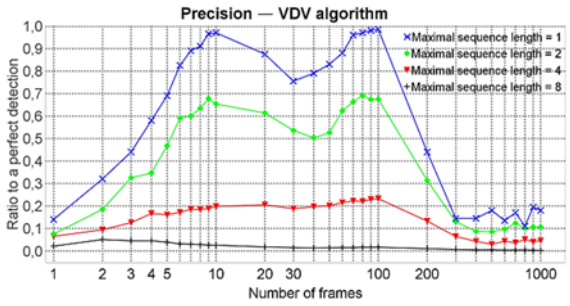


Fig. 10. Precision of the VDV algorithm relative to the trace size and parameterized by maximal sequence length.

We can also a notice a rapid increase in precision with the number of frames in the DLL trace, when the latter is low (below 10 frames). This behaviour is to be expected, as the algorithm uses statistical filtering, which only makes sense on large samples. However, there is also a collapse in precision when the trace becomes large (above 100 frames). This is due to the fact that as the size of the trace increases, the variance decreases, so it becomes more difficult to discriminate the sequences to be detected with the filtration threshold, resulting in an increase in false positives and a decrease in true positives.

It can be seen from the graph in Fig. 11 that the recall of the VDV algorithm is proportional to the maximal length of the sequences searched. This seems logical, since as the length increases, it becomes possible to detect more remarkable sequences present in the DLL trace.

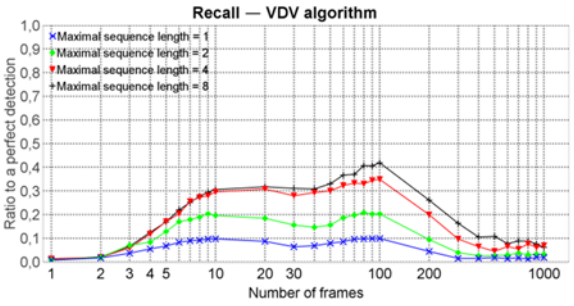


Fig. 11. Recall of the VDV algorithm relative to the trace size and parameterized by maximal sequence length.

There is also a similar behaviour to that of the precision in relation to the number of frames in the trace, which can be explained by the same reasons.

From the graph in Fig. 12, we can conclude that for 10 flows and a filtation threshold of 1, the maximal length of the sequences searched maximising the performance of the VDV algorithm is 4 bits, as we favor the large DLL traces.

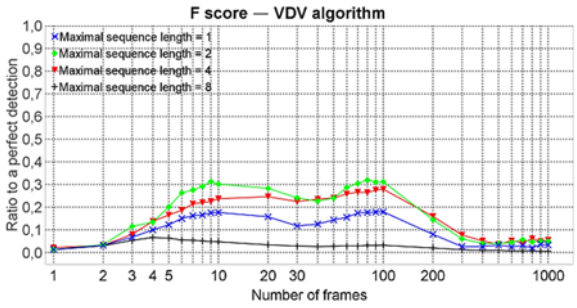


Fig. 12. F score of the VDV algorithm relative to the trace size and parameterized by maximal sequence length.

The precision curves of the VDV algorithm parameterised by the number of flows generated within it will not be presented as this parameter seems to have no influence on this metric.

The graph in Fig. 13 can be split into three parts along the axis of the number of frames of the DLL trace. Below 20 frames, the recall of the VDV algorithm is inversely proportional to the number of frames. This is due to the fact that the variance calculations performed within each flow lose their meaning if the flows contain too few frames, resulting in better coverage for a lower number of flows.

From 20 to 100 frames, the performances are approximately equivalent, as this is an intermediate area between small and large traces.

From 100 to 1000 frames, the recall is proportional to the number of flows, because by increasing the diversity of the flows, full advantage is taken of the two-level variance calculation performed.

We also notice a behaviour relative to the trace size identical to that of the previous curves in Fig. 10 and Fig. 11, which can be explained by the same reasons.

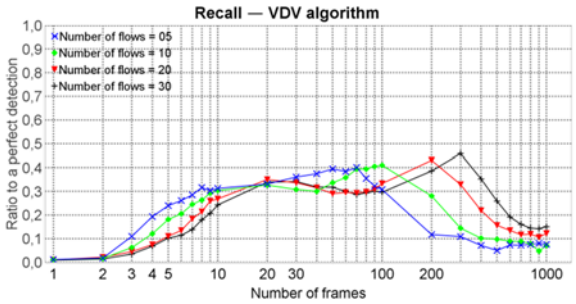


Fig. 13. Recall of the VDV algorithm relative to the trace size and parameterized by the number of flows.

The F score curves being no more interesting than the precision curves, we can say that for a maximal length of the sequences searched of 8 and a filtation threshold of 1, the best number of flows is 30 or more, as we favor the large DLL traces.

The graph in Fig. 14 shows that the precision of the VDV algorithm is inversely proportional to its filtration threshold. This is due to the fact that as the filtration threshold decreases, only those sequences

with the lowest variance of the variances are retained, and this is usually a detectable sequence.

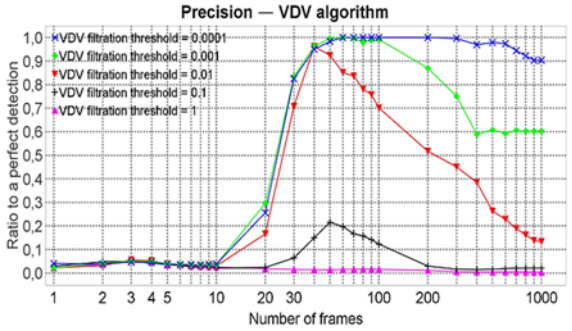


Fig. 14. Precision of the VDV algorithm relative to the trace size and parameterized by the filtration threshold value.

The same behaviour with respect to the number of frames is observed, similar to all other performance curves of the VDV algorithm, but it can be seen that the value of the filtration threshold is indeed the parameter influencing the critical size of the DLL trace above which performance collapses. It would probably have been more appropriate to tie this threshold to the number of frames in the trace.

The identical precision regardless of the filtration threshold for a trace of less than 10 frames is related to the fact that the variance is so big that a large proportion of the tokens are detected, and therefore there are a lot of false positives.

We can observe on the graph in Fig. 15 the collapse in performance when the filtration threshold is too high, characteristic of our implementation of VDV. However, we can also confirm that a threshold independent of the trace size is not at all relevant; indeed, before 7 frames, all curves display the same rapid growth, but afterwards, we can see that each of them seems to have approximately the same behaviour: a phase of decrease, then growth, and finally decrease, but shifted on the axis of the number of frames of the trace. We can therefore assume that for each trace size there exists a corresponding optimal filtration threshold.

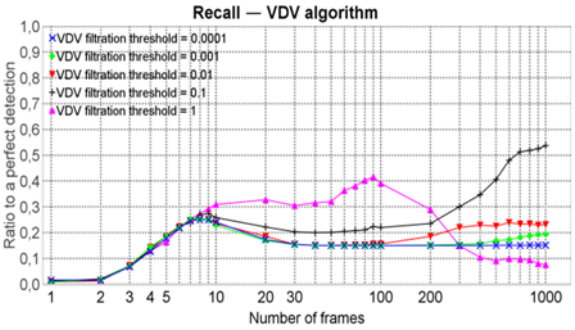


Fig. 15. Recall of the VDV algorithm relative to the trace size and parameterized by the filtration threshold value.

From the graph in Fig. 16, we can conclude that for a maximal length of the sequences searched of 8 and 10 flows, the filtration threshold maximising the performance of the VDV algorithm over the simulated interval is 0.001, which is only just beginning its growth phase. However, if we were to consider a larger trace, 0.0001 would probably be a better alternative, although ideally the filtration threshold should be indexed to the size of the DLL trace.

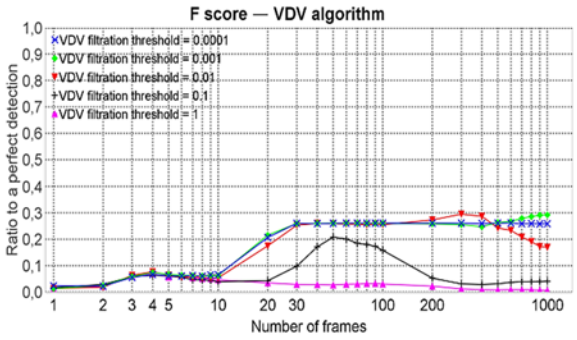


Fig. 16. F score of the VDV algorithm relative to the trace size and parameterized by the filtration threshold value.

3.4. Study of AC Parameters Influence

We then run our implementation of the AC algorithm with a series of parameter sets to study their influence.

Table 5 summarizes the parameters used for the simulations of the AC algorithm.

It can be seen from the graph in Fig. 17 that the precision of the AC algorithm is inversely proportional to the maximal length of the sequences searched, similar to the VDV algorithm, and for the same reasons.

Table 5. Summary of the AC algorithm simulation parameters.

Parameters	Sequences length influence	Filtration threshold influence	Fusion threshold influence
Number of frames	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment
Sequences length	1 to 8, power of 2 increment	8	8
Filtration threshold	1	0.7 to 1.5, increment of 0.1	1
Fusion threshold	1	1	0.1 to 1, increment of 0.1

The missing points for small trace sizes of the curve corresponding to a maximal sequence length of 1 bit means that no sequences were detected, which is not surprising as applying statistics to very short

sequences on a small number of frames does not tend to lead to any tendencies, so filtration may result in nothing being selected.

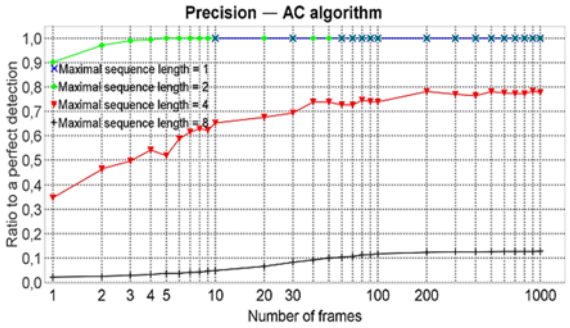


Fig. 17. Precision of the AC algorithm relative to the trace size and parameterized by maximal sequence length.

It can also be seen that the precision increases logarithmically with the size of the DLL trace, but unlike VDV, the precision does not eventually collapse when the number of frames increases. This can be explained by comparing the formulas for computing the filtration thresholds of the two algorithms: VDV's filtration threshold does not depend on the trace size, whereas AC's does. As we previously assumed, such a threshold is more relevant.

In the graph Fig. 18, we can see that the recall of the AC algorithm is proportional to the maximal length of the sequences searched, similar to the VDV algorithm, and for the same reasons.

We also observe the same logarithmic growth similar to the previous curves in Fig. 17, and for the same reasons.

From the graph in Fig. 19, we can conclude that for a filtration and fusion threshold of 1, the maximal length of the sequences searched maximising the performance of the AC algorithm is 4 bits.

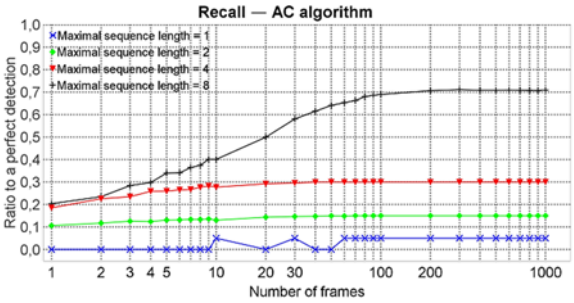


Fig. 18. Recall of the AC algorithm relative to the trace size and parameterized by maximal sequence length.

The graph in Fig. 20 shows us that the precision of the AC algorithm increases with the filtration threshold at first, then, beyond 1.3, it decreases. The increase of the filtration threshold means that the filtering is more restrictive, so that, initially, the previously retained sequences that are no longer retained are mainly false positives, hence the improvement in precision. However, after a certain stage, the filtering becomes too hard, and removes more true positives than false positives, which explains the presence of this extremum.

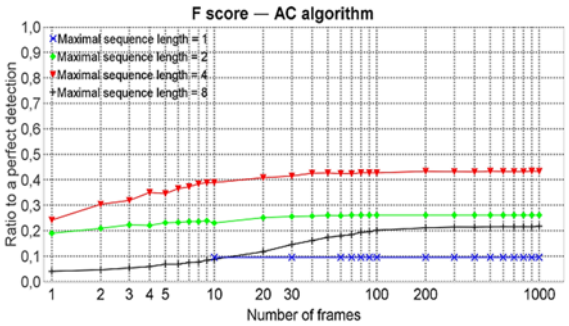


Fig. 19. F score of the AC algorithm relative to the trace size and parameterized by maximal sequence length.

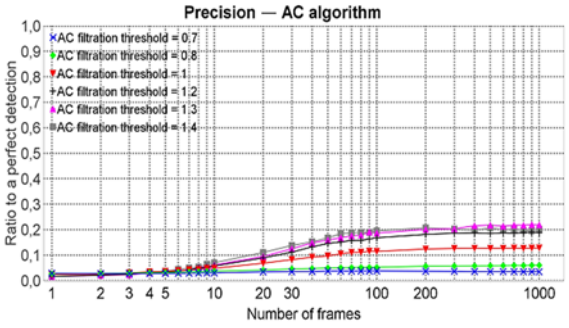


Fig. 20. Precision of the AC algorithm relative to the trace size and parameterized by the filtration threshold value.

The same logarithmic growth is also observed, similar to all AC performance curves, and for the same reasons.

It can be seen on the graph in Fig. 21 that the recall of the AC algorithm is inversely proportional to the value of the filtration threshold, but only from a value of 0.8, because the recall stops improving below this value. This simply means that all detectable sequences with a length less than or equal to 8 bits have been detected. The missing 15 % is caused by the 3 16-bit sequences present in the DLL trace.

From the graph in Fig. 22, we can conclude that for a maximal length of the sequences searched of 8 and a fusion threshold of 1, the filtration threshold value maximising the performance of the AC algorithm is 1.2 or 1.3.

The graph in Fig. 23 shows us that the precision of the AC algorithm is inversely proportional to the fusion threshold between 0.2 and 0.9, and does not vary outside these limits. A low fusion threshold means that relatively different sequences are merged, so that the total number of detected sequences decreases. It appears, according to the graph that the

number of false positives decreases faster than the number of true positives, which means that false positives are assimilated with true positives, causing an increase in precision. However, when the threshold reaches 0.2, for each sequence length, all the sequences are merged into one, making it impossible to gain anything more by merging. On the contrary, when the threshold reaches 0.9, it becomes impossible to merge sequences, hence a non-existent impact of the merging procedure.

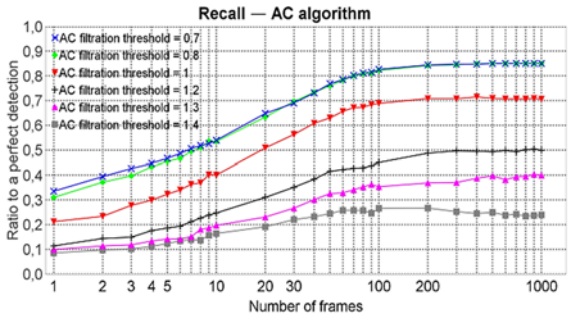


Fig. 21. Recall of the AC algorithm relative to the trace size and parameterized by the filtration threshold value.

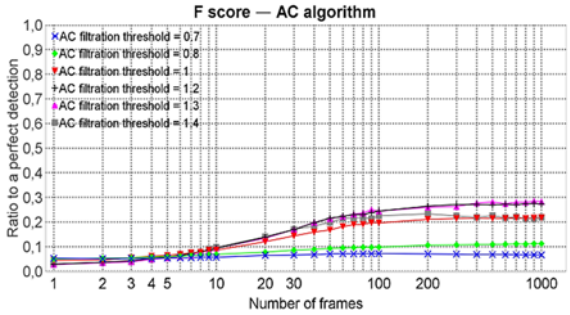


Fig. 22. Fscore of the AC algorithm relative to the trace size and parameterized by the filtration threshold value.

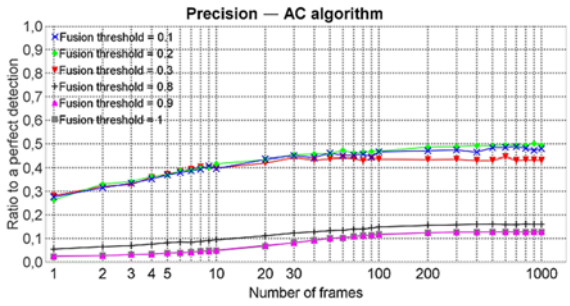


Fig. 23. Precision of the AC algorithm relative to the trace size and parameterized by the fusion threshold value.

We can see in the graph Fig. 24 that the recall of the AC algorithm is proportional to the fusion threshold between 0.2 and 0.9, and does not vary outside these boundaries. This seems logical, because a higher fusion threshold implies less sequence merging, and therefore greater diversity, leading to better recall. The visible boundaries are due to the same reasons as for the graph in Fig. 23.

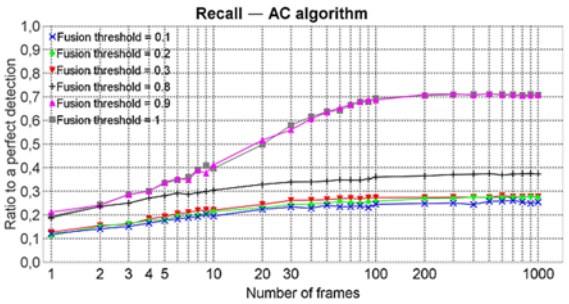


Fig. 24. Recall of the AC algorithm relative to the trace size and parameterized by the fusion threshold value.

From the graph Fig. 25, we can conclude that for a maximal length of 8 of the sequences searched and a filtration threshold of 1, the fusion threshold of the sequences maximising the performance of the AC algorithm is 0.2.

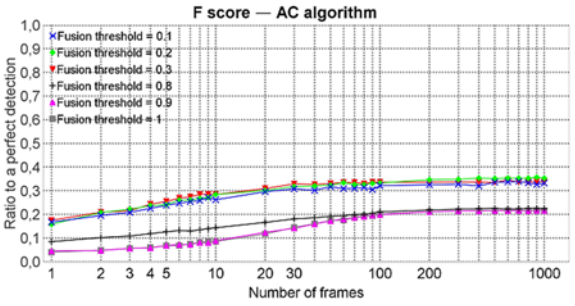


Fig. 25. Precision of the AC algorithm relative to the trace size and parameterized by the fusion threshold value.

3.5. Study of LDA Parameters Influence

We finally run our implementation of the LDA algorithm with a series of parameter sets to study their influence.

We do not present the performance curves parameterized by Beta because this parameter has virtually no influence given the parameter values we simulated. Beta influences the a priori on the concentration of the multinomial distribution of the words over topics. We simulated it for values inferior to 1, so it means it just influenced the gap between high and low probability values.

However, with the maximal probability gradient value we used, the selected keywords would always be the ones above the probability gap, whatever the gap width.

Table 6 summarizes the parameters used for the simulations of the LDA algorithm.

Table 6. Summary of the LDA algorithm simulation parameters.

Parameters	Sequences length influence	Keywords number influence	Maximal perplexity gradient influence	Maximal probability gradient influence	Alpha influence
Number of frames	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment	1 to 1000, logarithmic increment
Sequences length	1 to 8, power of 2 increment	8	8	8	8
Keywords number	10	3 5 10 20 30	10	10	10
Maximal perplexity gradient	1	1	0.1 to 10, logarithmic increment	1	1
Maximal probability gradient	0.1	0.1	0.1	0.01 to 1, logarithmic increment	0.1
Alpha	1	1	1	1	0.01 to 100, power of 10 increment
Beta	0.0001	0.0001	0.0001	0.0001	0.0001

It can be seen on the graph in Fig. 26 that the precision of the LDA algorithm is inversely proportional to the maximal length of the sequences searched, similar to the VDV and AC algorithms, and for the same reasons.

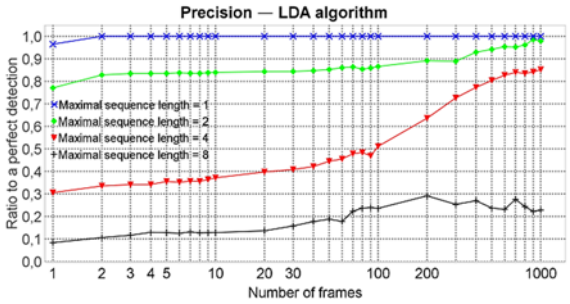


Fig. 26. Precision of the LDA algorithm relative to the trace size and parameterized by maximal sequence length.

There is also an approximately linear growth proportional to the size of the DLL trace, indicating that, unlike the AC algorithm, a learning-based technique keeps increasing its performance as its learning base becomes larger, making it more suitable for large volumes of data.

In the graph Fig. 27, we can see that the recall of the LDA algorithm is proportional to the maximal length of the sequences searched, similar to the VDV and AC algorithms, and for the same reasons.

We notice a phase of slight increase, then slight decrease of the recall, as the size of the DLL trace grows. This can be explained by the search for too few keywords, which causes most of the keywords to converge towards a limited number of sequences as the number of frames increases, resulting in a reduction in recall. However, when the maximal length of the sequences searched becomes long enough, this phenomenon seems to disappear.

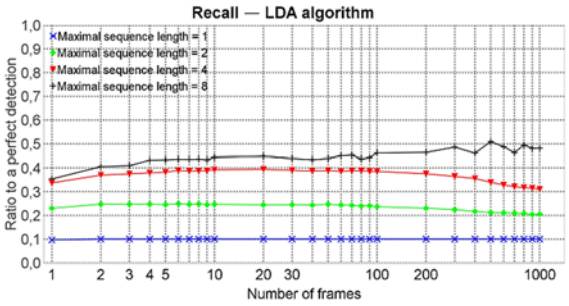


Fig. 27. Recall of the LDA algorithm relative to the trace size and parameterized by maximal sequence length.

From the graph in Fig. 28, we can conclude that for 10 keywords, a maximal perplexity gradient of 1, a maximal probability gradient of 0.1, an alpha of 1, and a beta of 0.0001, the maximal length of the sequences searched maximising the performance of the LDA algorithm is 4 bits.

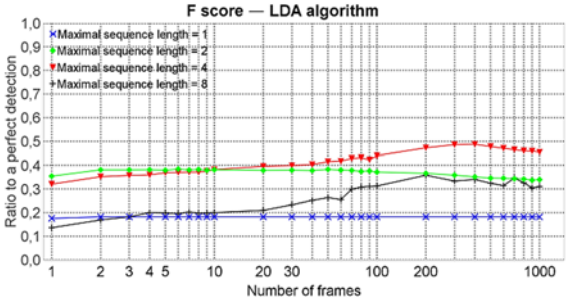


Fig. 28. F score of the LDA algorithm relative to the trace size and parameterized by maximal sequence length.

We can observe on the graph in Fig. 29 that the precision of the LDA algorithm is inversely proportional to the number of keywords assumed to be

present in the DLL trace. Indeed, since only the most probable sequences are selected for each keyword, if fewer keywords are assumed, then only the most probable sequences among those will be selected. These most probable sequences are generally remarkable sequences, which means more true positives, and hence higher precision.

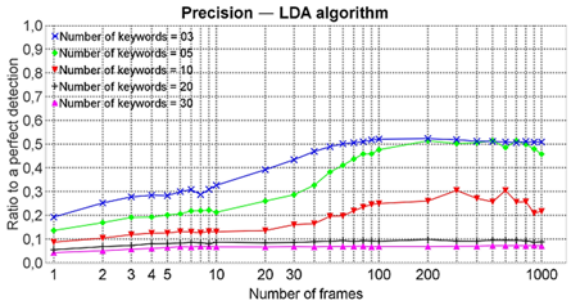


Fig. 29. Precision of the LDA algorithm relative to the trace size and parameterized by the number of keywords.

In addition, there are generally two phases: a linear growth with respect to the size of the DLL trace, for the same reason as for the previous LDA curves in Fig. 26-Fig. 28; and a constant phase corresponding to the maximal performance achievable with the value given to the other parameters.

In the graph Fig. 30, we can see that the recall of the LDA algorithm is proportional to the number of keywords assumed in the DLL trace. This seems logical, because the greater the number of keywords, the greater the number of sequences selected, and therefore the greater the recall.

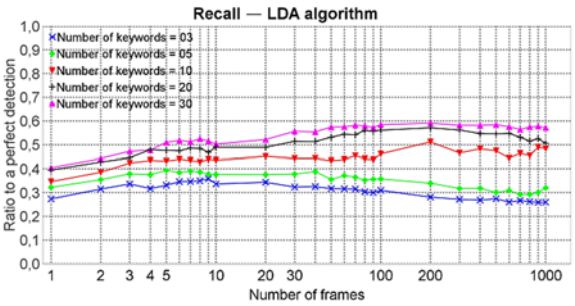


Fig. 30. Recall of the LDA algorithm relative to the trace size and parameterized by the number of keywords.

From the graph in Fig. 31, we can conclude that for a maximal length of 8 of the sequences searched, a maximal perplexity gradient of 1, a maximal probability gradient of 0.1, an alpha of 1, and a beta of 0.0001, the number of keywords maximising the performance of the LDA algorithm over the simulated interval is 3 or 5, but we will favour larger DLL trace sizes, so we will select 5.

We can see on the graph in Fig. 32 that the precision of the LDA algorithm is proportional to the value of the maximal perplexity gradient before the Gibbs sampler stops, between 0.3 and 3, and then stabilises. This means that iterating the sampler a large number of times is useless, and even counter-productive, which is very surprising.

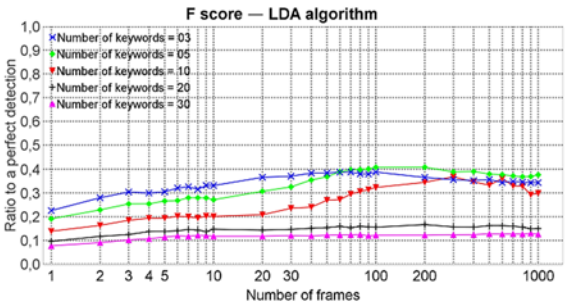


Fig. 31. F score of the LDA algorithm relative to the trace size and parameterized by the number of keywords.

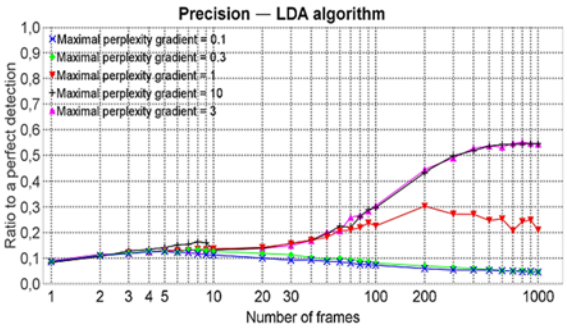


Fig. 32. Precision of the LDA algorithm relative to the trace size and parameterized by maximal perplexity gradient.

The graph in Fig. 33 shows that the recall of the LDA algorithm is inversely proportional to the maximal perplexity gradient. This means that iterating the Gibbs sampler a greater number of times favours a greater diversity of keywords, and therefore of sequences selected for these keywords.

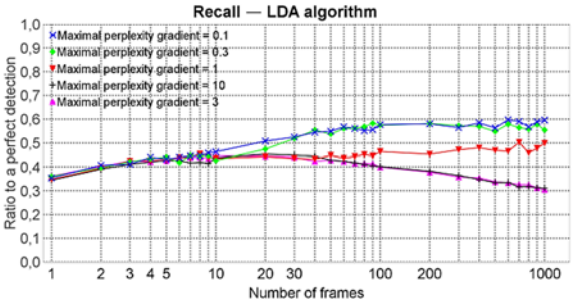


Fig. 33. Recall of the LDA algorithm relative to the trace size and parameterized by maximal perplexity gradient.

The behaviour of the recall relative to the size of the DLL trace is similar to the previous recall curves of the LDA algorithm in Fig 27 and Fig. 30, and for the same reason.

From the graph in Fig. 34, we can conclude that for 10 keywords, a maximal length of 8 of the sequences searched, a maximal probability gradient of 0.1, an alpha of 1, and a beta of 0.0001, the values of the maximal perplexity gradient maximising the performance of the LDA algorithm are those greater than or equal to 3.

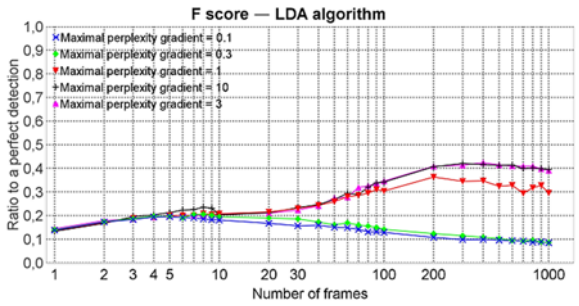


Fig. 34. F score of the LDA algorithm relative to the trace size and parameterized by maximal perplexity gradient.

We can see on the graph in Fig. 35 that the precision of the LDA algorithm is inversely proportional to the maximal probability gradient. This seems logical, as the lower this gradient is, the more the n-grams with the highest probability within each keyword will be favored, thus those most likely to be remarkable sequences of the protocol.

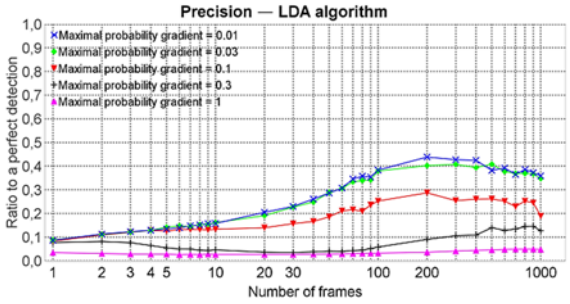


Fig. 35. Precision of the LDA algorithm relative to the trace size and parameterized by maximal probability gradient.

We also notice a behaviour generally resembling that of the precision parameterized by the number of keywords, with a growth phase, then a stabilization. The ceiling is probably due to the too high number of keywords and the too low perplexity gradient.

The graph in Fig. 36 shows that the recall of the LDA algorithm is proportional to the maximal probability gradient. This is due to the fact that as this gradient increases, the hardness of the n-gram selection decreases, therefore more n-grams are detected, resulting in an increase in diversity, and therefore recall.

The behaviour with respect to the number of frames is similar to that of all other recall curves of the LDA algorithm, and for the same reasons.

From the graph in Fig. 37, we can conclude that for 10 keywords, a maximal length of the sequences searched of 8, a maximal perplexity gradient of 1, an alpha of 1, and a beta of 0.0001, the value of the maximal probability gradient maximising the performance of the LDA algorithm is 0.03 or less.

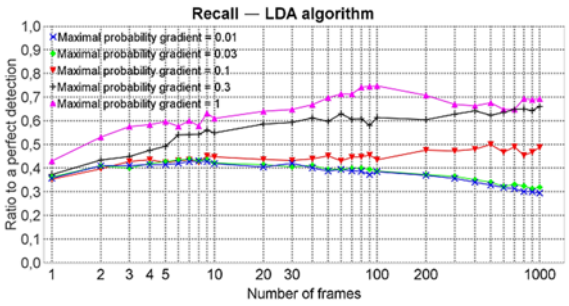


Fig. 36. Recall of the LDA algorithm relative to the trace size and parameterized by maximal probability gradient.

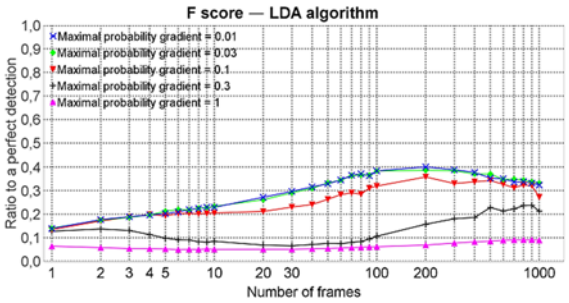


Fig. 37. F score of the LDA algorithm relative to the trace size and parameterized by maximal probability gradient.

On the graph in Fig. 38, we can see that the precision of the LDA algorithm is proportional to Alpha. By studying the characteristics of Dirichlet law, we can say that this means that the algorithm offers its best precision when we consider that the frames are composed of a relatively homogeneous mixture of all the keywords of the protocol, rather than just a few. This seems to correspond relatively well to reality, so these results are not surprising.

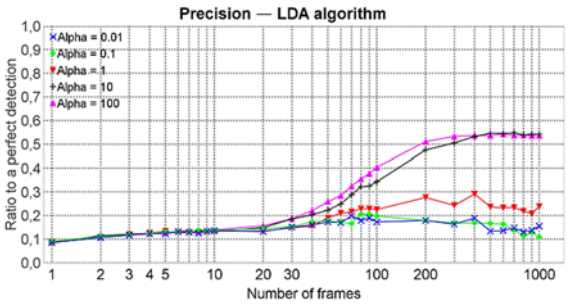


Fig. 38. Precision of the LDA algorithm relative to the trace size and parameterized by alpha.

The graph in Fig. 39 shows that the recall is inversely proportional to Alpha. Although we do not

have a precise explanation for this behaviour, it is consistent with the principle verified in all the previous curves in Fig. 10-Fig. 38, namely that precision and recall always show opposite trends when varying a parameter. Furthermore, the improvement in accuracy as Alpha increases is greater than the deterioration of the recall, resulting in an overall improvement in performance, which is in line with our hypothesis when observing the graph in Fig. 38.

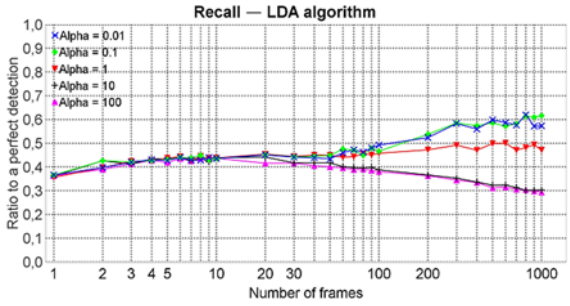


Fig. 39. Recall of the LDA algorithm relative to the trace size and parameterized by alpha.

From the graph Fig. 40, we can conclude that for 10 keywords, a maximal length of 8 of the sequences searched, a maximal perplexity gradient of 1, a maximal probability gradient of 0.1, and a beta of 0.0001, the values of Alpha maximising the performance of the LDA algorithm are those greater than or equal to 10.

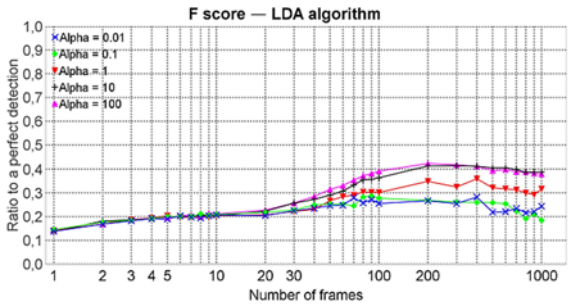


Fig. 40. F score of the LDA algorithm relative to the trace size and parameterized by alpha.

3.6. Comparison of the Best Parameter Sets

For each graph, we selected the best performing curve in the previous subsection and retrieved its corresponding algorithm parameters. Then, out of all these best performing curves, we select the best one for each algorithm, and address them in this subsection. We obtain the following parameter sets:

- For all algorithms, a maximal length of the sequences searched of 4.
- For the VDV algorithm, a number of flows randomly generated of 10 and a filtration threshold coefficient of 1.
- For the AC algorithm, a fusion threshold and a filtration threshold of 1.
- For the LDA algorithm, a number of keywords of 10, a maximal perplexity gradient of 1, a maximal probability gradient of 0.1, an alpha of 1, and a beta of 0.0001.

The graph in Fig. 41 shows that, for small DLL traces (less than 400 frames), the AC algorithm offers the best precision, followed by the LDA algorithm, and finally the VDV algorithm. For traces of more than 400 frames, the LDA algorithm becomes more precise than the AC algorithm, and the improvement in precision seems to continue beyond 1000 frames.

Maximum performance is around 85 % precision for LDA, and just under 80 % for AC, while VDV peaks at just over 20 %.

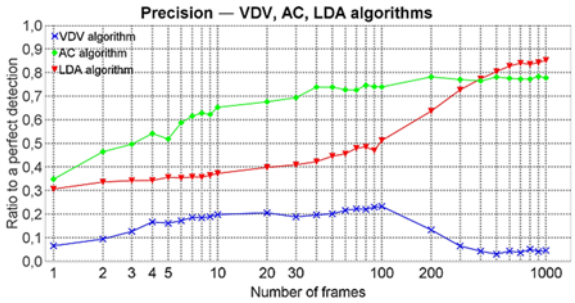


Fig. 41. Best precision of the algorithms VDV, AC and LDA.

On the graph in Fig. 42, we can see that the LDA algorithm has the best recall regardless of the size of the trace. Between 10 and 100 frames, the VDV algorithm has the second best recall, and the AC algorithm has the worst one, but outside these limits, AC is second, and VDV last.

Maximum performance is about 40 % recall for LDA, 35 % for VDV, and 30 % for AC.

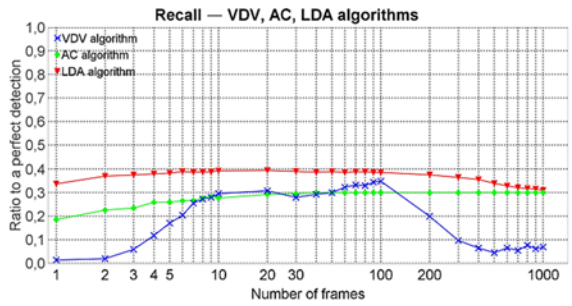


Fig. 42. Best recall of the algorithms VDV, AC and LDA.

The graph in Fig. 43 shows that the AC and LDA algorithms have approximately identical performance, with the LDA looking slightly better. The VDV algorithm offers significantly poorer results, which even determining the filtration threshold based on the size of the DLL trace could not fully compensate for;

at most, results at 1000 frames would be on the same order as of those obtained for 100.

Maximum performance is slightly under 50 % F score for LDA, slightly under 45 % for AC, and slightly under 30 % for VDV.

Let us now switch from the quality of the sequence detection to its usefulness for field detection.

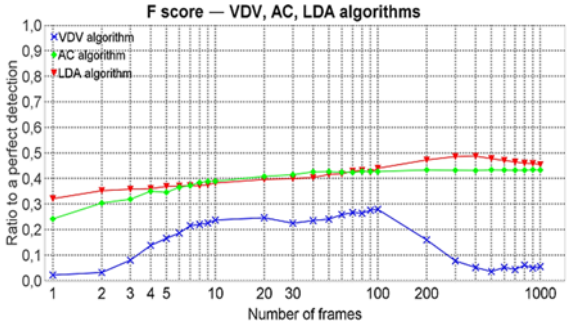


Fig. 43. Best F score of the algorithms VDV, AC and LDA.

The graph in Fig. 44 shows that the average fields lengths detection ratio of the LDA algorithm is the best, followed by that of AC, and finally VDV.

Maximum performance is slightly less than 60 % for LDA, a bit under 50 % for AC, and slightly under 30 % for VDV.

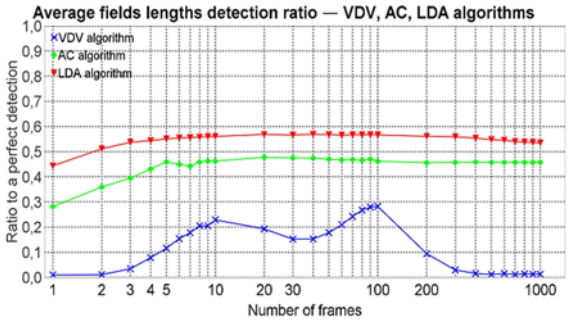


Fig. 44. Best average fields lengths detection ratio of the algorithms VDV, AC and LDA.

The same behaviour can be seen on the graphs in Fig. 45 and Fig. 46, concerning the average fields values and fields positions detection ratios. Their maximum performances are, respectively, 75 % for the LDA algorithm, slightly above 40 % for AC, slightly below 40 % for the VDV algorithm, and slightly below 60 % for the LDA algorithm, slightly below 50 % for AC, and slightly below 10 % for the VDV algorithm.

We summarize the relative performance of the three algorithms in Table 7. The number of stars stands, by decreasing order, for the best, average, and worst.

We clearly see that the LDA algorithm offers the best performance, followed by the AC algorithm, and lastly the VDV algorithm.

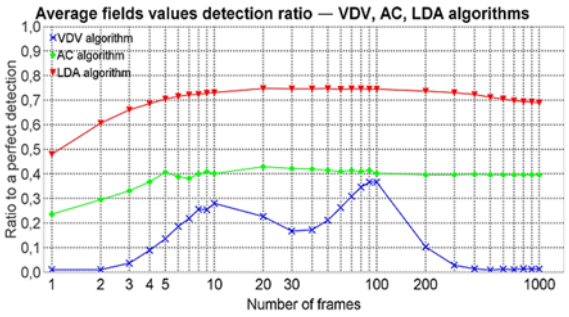


Fig. 45. Best average fields values detection ratio of the algorithms VDV, AC, and LDA.

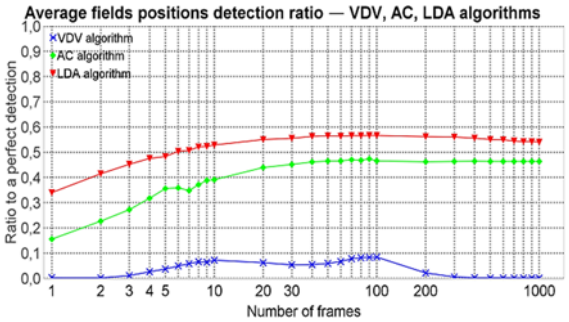


Fig. 46. Best average fields positions detection ratio of the algorithms VDV, AC and LDA.

Table 7. Relative performance summary of the algorithms VDV, AC and LDA.

Algorithms	VDV	AC	LDA
Precision	*	***	***
Recall	*	**	***
F score	*	***	***
Average fields lengths detection ratio	*	**	***
Average fields values detection ratio	*	**	***
Average fields positions detection ratio	*	**	***
Total	6	14	18

4. Conclusion

We wanted a communicating object to be able to communicate in an unknown environment, so we needed it to learn the protocols in that environment. We chose to study and evaluate the performance of three possible sequence identification techniques which could be used in that learning procedure: VDV, AC, and LDA. To that end, we simulated them applied to the analysis of ZigBee DLL traces, and compared them.

For the purpose of comparison, in addition to the classic metric F score, we defined our own, the fields detection ratio.

From the simulations results, we can clearly state that in this context the LDA technique offers the best

results, followed by the AC technique, and eventually the VDV technique.

A more powerful hardware could have allowed us to see if we could push further the performance of the LDA algorithm by analysing larger traces, and a more formal parameter optimization would have given us more precise maximal performance of the algorithms.

Nonetheless, with the results of the comparative simulation, we can now state that a technique based on Bayesian networks performs better than ones simply based on statistics or occurrences counting. This encourages us to further engage in Bayesian theory in the future, and design our own Bayesian network model.

References

[1]. O. Esoul, N. Walkinshaw, Finding clustering configurations to accurately infer packet structures from network data, in *ArXiv*, October 2016.

[2]. A. Trifilò, S. Burschka, E. Biersack, Traffic to protocol reverse engineering, in *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, July 2009, pp. 1-8.

[3]. A. V. Aho, M. J. Corasick, Efficient string matching: An aid to bibliographic search, *Commun. ACM*, Vol. 18, Issue 6, June 1975, pp. 333-340.

[4]. D. M. Blei, A. Y. Ng, M. I. Jordan, Latent Dirichlet allocation, *Journal of Machine Learning Research*, Vol. 3, May 2003, pp. 993-1022.

[5]. A. Li, C. Dong, S. Tang, F. Wu, C. Tian, B. Tao, H. Wang, Demodulation-free protocol identification in heterogeneous wireless networks, *Computer Communications*, Vol. 55, September 2014, pp. 102-111.

[6]. S. Kleber, L. Maile, F. Kargl, Survey of protocol reverse engineering algorithms: Decomposition of tools for static traffic analysis, *IEEE Communications Surveys and Tutorials*, Vol. 21, Issue 1, August 2018, pp. 526-561.

[7]. B. Sija, Y.-H. Goo, K.-S. Shim, H. Hasanova, M.-S. Kim, A survey of automatic protocol reverse engineering approaches, methods, and tools on the inputs and outputs view, *Security and Communication Networks*, Vol. 2018, February 2018, pp. 1-17.

[8]. P.-S. Gréau-Hamard, M. Djoko-Kouam, Y. Louet, A comparative Study of Sequence Identification Algorithms in IoT Context, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI' 2020)*, November 2020, pp. 137-143.

[9]. Y. Wang, N. Zhang, Y.-M. Wu, B.-B. Su, Y.-J. Liao, Protocol formats reverse engineering based on association rules in wireless environment, in *Proceedings of the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, July 2013, pp. 134-141.

[10]. Y. Wang, X. Yun, M. Shafiq, L. Wang, A. Liu, Z. Zhang, D. Yao, Y. Zhang, L. Guo, A semantics aware approach to automated reverse engineering unknown protocols, in *Proceedings of the 20th IEEE International Conference on Network Protocols (ICNP)*, October 2012, pp. 1-10.

[11]. E. I. George, G. Casella, Explaining the Gibbs sampler, *The American Statistician*, Vol. 46, Issue 3, August 1992, pp. 167-174.

[12]. G. Heinrich, Parameter estimation for text analysis, *Technical Note*, University of Leipzig, Germany, 2008.

[13]. C. A. Haydar, Trust-based recommender systems, Theses, Université de Lorraine, September 2014.

[14]. ZigBee Specification, ZigBee Standards Organization Std.

[15]. IEEE Std. 802.15.4-2003, IEEE Std.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 93-101

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

The Novel Computer Aided Diagnostic System on Medical Images for Parameter Calculation

Jong-Ha LEE
Keimyung University, Daegu, South Korea, 40210
Tel.: (82)53-258-3736
E-mail: segeberg@kmu.ac.kr

Received: 12 December 2020 /Accepted: 15 January 2021 /Published: 28 February 2021

Abstract: Recently, the number of Developmental Dysplasia of the Hip (DDH) that occurs during infant and child growth has been increasing a lot. DDH should be detected and treated as early as possible because it hinders infant growth and causes many other side effects. In this study, two modelling techniques were used for multiple training techniques. Based on the results after the first transformation, the training was designed to be possible even with a small amount of data. The vertical flip, rotation, width and height shift functions were used to improve the efficiency of the model. Adam optimization was applied for parameter learning with the learning parameter initially set at $2.0 \times 10e^{-4}$. Training was stopped when the validation loss was at the minimum. No significant difference in angle measurements was found between the model and doctor. The differences in the maximum, minimum, and mean base angles were 6.93, 0 and 1.81 degrees, respectively.

Keywords: non-rigid registration; image-guided surgery, emphysema, laser scanner.

1. Introduction

1.1. Research Background

The incidence of developmental dysplasia of the hip (DDH), which occurs during the developmental period of young children, has been increasing. DDH does not cause external deformities in young children. About 1~2 % of young children on average have DDH, and DDH can be passively detected by medical imaging in 15 % of DDH cases. DDH can be easily treatment if detected within the first year after birth but can incur serious problems if left untreated. Although infants are regularly screened for DDH in the United States, there is not yet a standardized and clear method of DDH screening. South Korea has included DDH in the National Health Screening Program for Infants since 2006. Research has also been started on DDH in South Korea but without clear research standards as is the case in the United States [1-2]. DDH is manually detected by a physician using ultrasonography (US) as a standard practice. However, factors such as the examiner and the system used to perform US can affect the interpretation of the US results. In US-based DDT screening, a medical team measures the acetabulum-femoral head angle on ultrasound scans. A physician then diagnoses DDT based on the angle measurement [3-9]. However, such measurements are subject to errors, which can affect the diagnosis results. US resolution can also have a significant impact on DDT screening results [10-13]. Since DDT is manually detected with the eyes, the resolution and accuracy of ultrasound images can significantly affect screening results [14]. Since DDT is manually screened by a human without a standard guideline, poor resolution can cause large errors. Additionally, a diagnosis is not just made based on an ultrasound image but also on length and angle measurements in

different sites [15-19]. Currently, a medical team manually obtains and provides a physician with these measurements. This manual process takes over 10 minutes per image, challenging health professionals in their endeavors to efficiently provide medical services [20-26].

Computational speed has been rapidly increasing due to recent technological advances. The improved performance of graphics processing units (GPU), used in image and video processing, has led to the increasing use of machine learning algorithms used for image processing in industries as well as in the fields of science and medicine. Visual interpretation of medical images is important in the field of medicine. Machine learning algorithms such as convolutional neural networks (CNNs) and improved Resnet 50 are increasingly used in ultrasound image analysis. Related literatures have demonstrated that artificial intelligence (AI)-based image analysis has significantly improved objectivity and productivity in medical settings and that these machine learning algorithms may be used to develop a computer-aided DDH diagnostic system using ultrasound images.

CNNs are a machine learning algorithm used in image analysis and are divided into machine learning and deep learning depending on the level of human intervention. The lower the level of human intervention, the closer an algorithm is to deep learning and the greater the amount of data required. However, it is often the case that there is not enough data to even just train a mode l. If more data are required for deep learning, then human intervention may be increased to resolve the data shortage problem. Thus, we propose a multi-training technique using a user filter.

In this study, we propose a MASK R-CNN-based DDH diagnostic system that can be trained on a small amount of ultrasound image data. The system uses a deep leaning algorithm to automatically calculate the acetabulum-femoral head angle and classify the hip bones of children as normal or abnormal.

1.2. Theoretical Background

1.2.1. Developmental Dysplasia of the Hip

The hip joint consists of an articulation between the acetabulum of the pelvis and the head of the femur. It supports the weight on the pelvis and enables the overall movement of the lower limbs to allow us to stand and run. DDH reduces the contact area between the acetabulum and the femoral head, resulting in increased pressure on the hip joint and thereby promoting degenerative changes of the joint cartilage. Such deformation impedes with smooth joint movements and induces a mismatch between articular surfaces, consequently causing pain or discomfort in daily life and inducing degenerative arthritis by applying an excessive force on the surrounding tissues and causing joint damage. DDH can be effectively treated if detected early in life. It can be easily treated if the treatment is started before an infant is on a walker. An assistive harness, double or triple diapers, Von Rosen splint, or Pavlik harness can be used to treat DDH in infants who are not yet on a walker. An assistive harness is recommended for infants who are less than six-months-old. For infants who are on walkers, a non-surgical method is available in which the hip is properly positioned and fixed in place using plaster bandages. However, infants often do not adapt to the treatment, and surgical procedures such as open reduction in which the fractured bone is put into place through a skin incision and bone correction surgery are more frequently performed.

A natural recovery from DDH without treatment is unlikely for infants. If DDH is not treated during the neonatal period, an infant may have short legs and muscle weakness resulting in limping. A false acetabulum may also form that promotes early degenerative changes. Secondary symptoms may also occur, the most common of which are scoliosis and lumbar pain. Infants with acetabular dysplasia or subluxation may have slightly short lower legs, limp slightly, or show no symptoms at all. However, infants with DDH still require active treatment as they are prone to acetabular cartilage or acetabular labral tears and degenerative changes, and approximately half of them have degenerative arthritis of the hip joint.

1.2.2. AI-based Medical Image Analysis

AI is a branch of computer science that plays a leading role in solving cognitive problems associated with human intelligence such as learning, problem solving, and pattern recognition. AI is currently considered the final stage of computer science. Recent technological advances have improved computational speed and efficiency and led to the emergence of several branches of AI. The advances in network computing have also led to the development of deep learning. There are several branches of AI including machine learning, deep learning, and reinforcement learning, which vary in the level of human intervention.

Machine learning algorithms are used for pattern recognition. Machine learning is a collection of algorithms that are trained on recorded data to make predictions and detect hidden characteristics in variables to concisely represent the data. Unlike previous techniques in which an algorithm designed by a software developer outputs results based on the given input, machine learning produces statistical code through a comparative analysis using a filter to derive an answer based on a previously produced model. It is important to ensure a high quantity and quality of data to increase the accuracy of a machine learning model.

Deep learning is a branch of AI that requires less human intervention compared to machine learning and is a more advanced form of machine learning. Whereas machine learning detects characteristics in answers during the training phase and infers results for

other examples, deep learning analyzes a large amount of data without human intervention and infers results on its own. Instead of forming related sets as is done by machine learning algorithms, deep learning uses nonlinear algorithms to produce a model that detects associations between inputs. Deep learning can make predictions about data whose complexity is beyond human cognitive ability or the analytical ability of existing software-based techniques or machine learning algorithms to be analyzed. Since deep learning requires more data than machine learning, it is appropriate for use in fields where a large amount of data are available.

1.3. Previous Research

1.3.1. Standardized Method of Measurement for DDH

Two methods are available to detect DDH: physical examination and US. In a physical examination, DDH may be diagnosed based on unequal skin folds in the femoral region between the left and right thighs. The Galeazzi test may also be used to diagnose DDH, which checks for a discrepancy in the height of the knees when an infant is lying on a flat surface with their knees up. In the Ortolani test, a click sound that can be heard when the dislocated femoral head is placed back in the pelvis is checked. An infant is first laid on the floor, and the knee and hip joints are bent. The femoral head is then pushed upward while the thighs are spread apart to hear a clicking sound and feel the femoral head fitting into the acetabulum. In the Barlow's test, dislocation is induced by pushing the femoral head while the thighs are bringing the thighs together with the hip joint adducted. These are the physical screening methods for DDH.

There are two methods of radiological examination of DDH: X-rays and US. Before an X-ray examination, it is necessary to check that the femoral head is in contact with the pelvic bone. Since the femoral head consists of cartilage in the neonatal period, it cannot be observed on X-rays. Although the femoral head is observed on X-rays starting 6-7 months after birth, US is more appropriate for early diagnosis. In US, which is the main topic of this study, ultrasound scans of the hip joint where the triradiate cartilage can be observed are obtained in the coronal plane while the ilium is placed horizontally. The ilium must be five degrees below the horizontal axis. A and β angles are measured to make a diagnosis.

1.3.2. Edge Detection Filter

There are many unnecessary areas on ultrasound scans. Unlike CT scans, ultrasound scans lack clear feature points and contain many areas that look similar and hinder feature detection. Unnecessary parts must be removed to extract feature points. In this study, edge detection was used as a filtering method. Since ultrasound scans are grey-scale images, edge detection can be performed without processing the ultrasound scans. Edge detection is the process of detecting edges on images. Edges are a sequence of points on an image where the intensity of a pixel drastically changes from the threshold value. Such changes in pixel intensity can be detected using derivatives. The first-order derivative captures the magnitude of the slope, and the second-order derivative captures the sign of the slope. Masks are used in edge detection, which are filters that have the same effect as a differential operator. The Sobel, Prewitt, and Roberts operators can be used in edge detection using first-order derivatives. The Sobel and Prewitt operators are less sensitive to noise and have a slow computational speed. The Roberts operator has a fast computational speed and can detect edges with high accuracy but is sensitive to noise. Edge detection using first-order derivatives can detect horizontal and vertical edges. Second order derive masks include the Laplacian mask and Canny mask. The Laplacian mask reduces noises by Gaussian smoothing and then performs the Laplacian of Gaussian operation. The Canny mask reduces noise by smoothing using a Gaussian filter, detects edges by performing a mask operation, and then applies non-maximum suppression to remove non-edges.

1.4. Objective

Fig. 1 shows diagram of the proposed system. As explained in previous studies, DDH must be diagnosed early in infants, and the most effective method of DDH detection is US. However, a relatively small amount of data can be obtained from ultrasound scans of infants compared to the amount of time and effort put into the data acquirement due to the low age of infants and the nature of the ultrasound scans. Even then, only a small portion of the obtained data are of acceptable quality. Ultrasound scans must be clear enough for examination with the naked eyes and the angle between the ilium of the hip joint and the baseline must measure 5 degrees or less to be used in an analysis. Currently, a medical team manually examines each image to determine its acceptability. The team measures and records lengths and angles on ultrasound scans using a computer to determine whether an infant has DDH. This is a highly time-consuming process. Owing to the active research on AI, AI may detect DDH in ultrasound scans more effectively than do existing programs. Ultrasound scans have many similar areas; therefore, it is difficult to detect features from ultrasound scans based on a naked-eye examination alone. CNNs can extract features from ultrasound scans. A large amount of data are required to perform machine learning or deep learning. Thus, it is difficult to effectively use AI algorithms without sufficient data. In most cases, data are lacking or not processed. A method to train a machine learning or deep learning model with a small amount of data and obtain a large amount of processed

data is needed. We proposed a MASK-R-CNN-based ultrasound DDH detection method that requires increased human intervention and reduced data requirements. The method produces a seed model that can process a large amount of data from a small initial dataset at a research laboratory or corporate hospital to develop an initial AI environment.

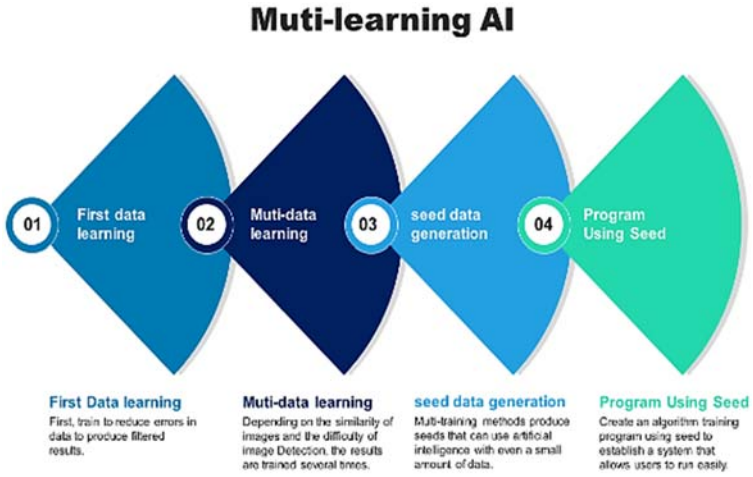


Fig. 1. The goal of the system.

2. Materials and Methods

2.1. Method

This study is aimed at developing a system that uses a multi-training method to process unprocessed data to increase the user accessibility. Previously, DDH was diagnosed by detecting the areas containing the ilium and acetabular roof, measuring the angles of the ilium and acetabular roof, and using a machine learning algorithm to make a prediction based on these angle measurements. Instead of measuring the angles from the ilium and acetabular roof following area detection, the areas containing the α and β angles of the ilium and acetabular roof are detected to measure the angles.

The Mask R-CNN method is used to first segment the shapes on ultrasound images formed by the ilium and acetabular head. Two points on the ilium and three points for calculating the α and β angles (four points in total with one overlapping point) are used to determine the acceptability of an image. The points are connected to obtain meaningful values for DDH diagnosis.

Based on a previous study in which segmentation and key points were used to extract a large amount of information from a given amount of data and achieved higher machine learning performance, the four points on the ilium and acetabular roof and segmentation were used to produce a deep learning model.

2.1.1. Segmentation Environment

Mask R-CNN was used for segmentation. Mask R-CNN is an extension of Fast R-CNN with the addition of a Mask-head. The Mask head is first used to detect the areas corresponding to the ilium and acetabular roof. A ResNet algorithm consisting of 50 layers modified using the background algorithm of Mask R-CNN. The ROIAlign and Feature Pyramid Network (FPN) were used for feature extraction.

2.1.2. File Conversion

The US data obtained from the participants were in the DICOM format. The DICOM files included information about ultrasound scans (date, type of equipment, imaged area, and patient) and image data. For personal information protection and memory efficiency reasons, this information was removed, and the ultrasound images were converted to 256-bit greyscale images.

2.1.3. Layer Extraction

A model was trained on ultrasound scans for segmentation. An area to be masked for the first training phase must be marked. The area is marked using the CVAT program developed by the MIT.

2.1.4. First Training

Polypoints were assigned to the areas on an image that contain the ilium and acetabular roof using the point selection tool. Points were assigned across a sequence of images. An extraction tool can be used to obtain XML or JSON files containing point records. The obtained XML and image files were used to train the first model.

2.1.5. Second Training

A model was trained using the output from the previously trained model. Using CVAT, the start and end of the ilium and the top and bottom of the acetabular roof were marked with dots. The start of the ilium was estimated based on a naked-eye examination, and the end of the ilium was designated to a point near the intersection between the three lines. The top of the acetabular roof was designated above the line extending to the acetabular roof. The bottom of the acetabular roof was designated below the line extending to the acetabular roof. Refer to the figure below for these points.

2.1.6. Final Results

Once the second training was complete, the first and second models were used to predict the angles and lengths of the ilium and acetabular roof. Hence, an AI system was developed that could assist physicians in diagnosing DDH based on these predicted values.

3. Results

A total of 2,692 ultrasound scans collected from 2016 and 2019 at Dongsan Medical Center were obtained. Of these, 1,242 were acceptable for use in the analysis. 321 of these scans were randomly selected and then arbitrarily divided into a training dataset containing 288 scans and a test dataset containing 33 scans.

To produce a machine learning model, the multi-training algorithm proposed in this study. The training parameter was set to $2.0e^{-3}$, and the model was trained until the validation loss was at the minimum. The trained model was then tested using 920 scans.

The 322 ultrasound scans obtained from Dongsan Medical Center were arbitrarily divided into a training dataset containing 288 scans and a validation set containing 33 scans. Of the total 920 scans, 389 were deemed acceptable (could be clearly interpreted by an orthopedic doctor at Dongsan Medical Center). The system correctly detected DDH in 369 out of the 389 scans, achieving a detection accuracy of 94.86 %.

3.1. Acceptable Data

Less than 20 % of all unprocessed data collected by a medical team is deemed acceptable due to the movements of the patient and the US equipment. Collecting acceptable data from infants is even more difficult. In this study, 1,242 of 2,692 raw scans were selected for use in the experiment (First screening). Next, from the test dataset containing 920 scans, 389 scans deemed acceptable for interpretation were selected to measure the performance of the system (Second screening).

Acceptable scan (Left) Unacceptable scan in the second screening (Middle) Unacceptable scan in the first screening (Right).

3.2. Method of US Scan Acquirement

Ultrasound scans were obtained from infants aged less than two years using HD7-XE from Philips. At least 30 scans were obtained per infant.

3.3. Performance Classification

To measure the performance of the system, the model's output was classified as appropriate detection, fail detection, different result, false image, inappropriate detection, or insufficient image. 'OK' indicated that the system produced normal outputs for normal data. 'Error' indicated that although an output was produced, an image was deemed inappropriate since the base angle value was not within the normal range. 'N/A' indicated that an output could not be produced due to non-available values.

The detection results of the total 920 scans were as follows: 'Appropriate detection' for 413 scans, 'Fail detection' for 98 scans, 'Different result' for 30 scans, 'False image' for 336 scans, 'Inappropriate detection' for 9 scans, and 'Insufficient image' for 34 scans. The figure below shows a bar graph the detection results starting with 'Appropriate Detection' from the far left followed by 'False Image', 'Different Result', 'Insufficient Image', 'Fail Detection', and 'Inappropriate Detection'. Fig. 2 shows the detection results.

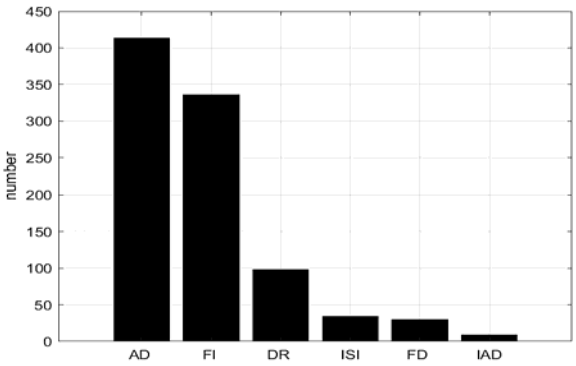


Fig. 2. Detection results.

3.4. Measuring System Performance

Performance parameters were used to assess the systems' performance. Since the answer must be divided into true or false, the classifier was also divided into OK (True) and error. A 2×2 matrix could then be created (Table 1). Of the total 510 scans excluding the ones classified as 'N/A', 320 were true positives (TPs), 48 were false positives (FPs), 49 were false negatives (FNs), and 93 were true negatives (TNs).

Table 1. 2×2 matrix based on true/false classification.

	AI OK	AI Error
Doctor OK	320 scans (TP)	48 scans (FP)
Doctor Error	49 scans (FN)	93 scans (TN)

3.5. System Performance Parameters

Based on the above classification, performance parameters could be calculated (Table 2). The equations below were used to calculate precision, recall, accuracy, F1 score, and fall-out. Precision was calculated at 0.87, recall at 0.87, accuracy at 0.81, F1 score at 0.87, and fall-out at 0.34.

Table 2. The values of the performance parameters (%).

	Value
Precision	$8.796e^{-3}$
Recall	$8.672e^{-3}$
Accuracy	$8.098e^{-3}$
F1 score	$8.684e^{-3}$
Fall-out	$8.404e^{-3}$

The vertical flip, rotation, width and height shift functions were used to improve the efficiency of the model. Adam optimization was applied for parameter learning with the learning parameter initially set at $2.0 \times 10e^{-4}$. Training was stopped when the validation loss was at the minimum. Fig. 3 shows the results.

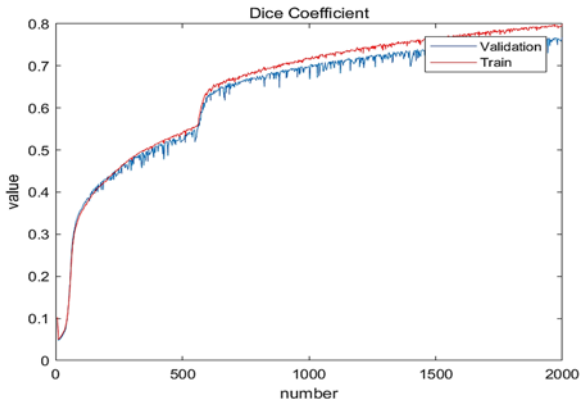


Fig. 3. Dice coefficients for the validation and training datasets.

Dice coefficients share a similar concept as intersection over union (IOU). They are calculated by dividing the area of overlap by the total number of pixels. They are commonly used to assess the accuracy of edge prediction and essentially represent the accuracy of a predicted bounding box. Dice coefficients close to 1 indicate good performance whereas those close to 0 indicate poor performance. Fig. 4 shows the results.

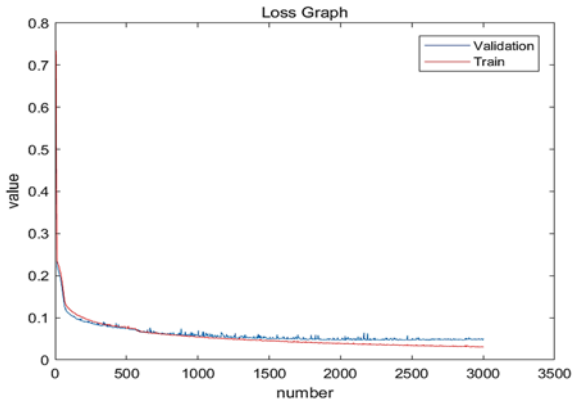


Fig. 4. Loss values for the validation and training dataset

The loss value graph shows loss values. Loss values are closer to 0 when a model makes correct predictions.

3.6. Measurement Comparison

No significant difference in angle measurements was found between the model and doctor. The differences in the maximum, minimum, and mean base angles were 6.93, 0, and 1.81 degrees, respectively (Fig. 5).

The differences in the maximum, minimum, and mean α angle were 19.78, 0.01, and 3.56 degrees, respectively (Fig. 6).

The differences in the maximum, minimum, and mean β angle were 25.92, 0.01, and 4.51 degrees, respectively (Fig. 7).

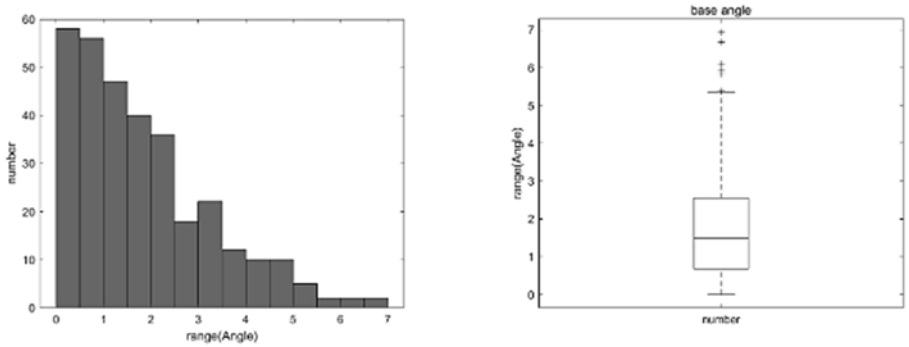


Fig. 5. Base angle range.

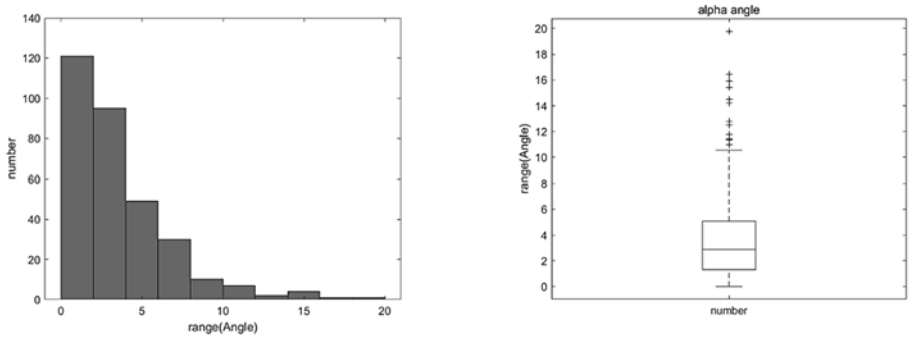


Fig. 6. α angle range.

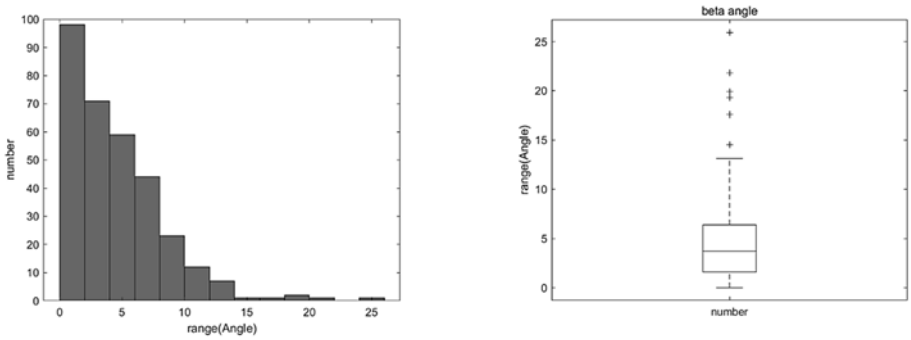


Fig. 7. β angle range.

4. Discussion

In this paper, the training was conducted with a smaller amount of data than the existing training method, and it was confirmed that the performance was better than the existing method when the multi-training technique presented in this paper was used with ultrasonic photography that is more complex than other image images. This technique is expected to be used in small laboratories or development teams that need to build AI with a small amount of data and can be used as a seed that training on a large of data based on this technique.

The techniques used in the paper are divided into machine learning and deep learning depending on the degree of user intervention in artificial intelligence. In the original image, train the part used for judgment first. It removes all areas other than the detected areas to enhance the filtering of noise. Re-training the part of the angle corresponding to the angle. Re-training the points corresponding to the angle. It is designed to detect parts that are needed even with less data than conventional training methods.

2,692 sheets of data were used in the experiment, but due to the nature of the ultrasound, all data that were highly swayed, noisy, or could not be verified by the human eye were excluded. As a result, the total number of data used for training was 1,242. Of these, 321 were brought in as data for modeling and construction. The data to be tested through the generated modelling was classified into 920 chapters. First of all, training data for modeling formation and test data for learning stabilization are required for modeling construction. We trained with 288 training data and 33 test data respectively. The second training was trained with the converted file of the data used for the first training. The most difficult thing about collecting effective data was that the subject was infants and that ultrasound had bigger artefacts than other medical images such as CT and MRI. Although medical staff with more than five years of experience filmed, the normal data acquisition rate was less than 50 percent. Because of the addition of medical elements, the photos were classified into six categories by medical measurement methods. The six categories factors actually require complex criteria. For quantitative measurement, the answer sheet for each image of doctors and AI was fixed with OK, Error, and NA. OK stands for normal range values for normal data. Error is normal data, but the base angle is not the normal range, which means an abnormal image. NA means when a value cannot be derived as non-available. Performance measurement methods were distinguished by general model evaluation classification indicators. 510 data were detected except for NA options that doctors could not detect. OK and Error of choice of doctor and artificial intelligence were calculated as a total of four based on two correct answers. Performance evaluation of AI used a performance evaluation index, Dice coefficient, and loss graphic, and the result was very good. The medical performance compared the differences in alpha beta angles. Comparing to the average, there was a difference of less than 4 degrees. This is a good result because it is the range of differences that occur on average when measuring the same picture between doctors.

Conclusions

Recently, the number of hip dysplasia (DDH) that occurs during infant and child growth has been increasing. DDH should be detected and treated as early as possible because it hinders infant growth and causes many other side effects. DDH's standard screening method is to utilize Ultrasonography (US). However, the Ultrasonography (US) may differ in the reading results depending on the proficiency of the physician. It also takes a lot of skill to define DDH levels depending on the quality of the Ultrasonography (US). In order to improve the doctor's proficiency, many patient cases should be encountered and large-capacity Ultrasonography (US) should be obtained. However, due to the nature of infants and toddlers, only 2 to 10 percent of the photos taken can be used as meaningful photos. Currently, significant photographs are used by doctors to determine DDH in ultrasonic images, such as the angle of the Acetabulum-Femoral head for ultrasonic image reading. However, this method causes a lot of errors because the doctor works manually. Recently, machine learning techniques using Convolutional Neural Networks (CNN) and improved Resnet50 have been widely used for ultrasound image analysis. Research shows that computer-aided image analysis greatly improves objectivity and productivity at medical sites. This shows that the DDH computer-aided diagnostic algorithm through ultrasound images can solve difficult challenges in orthopedics.

In this paper, we proposed a computer-aided diagnostic algorithm that can automatically measure and diagnose DDH using CNN. In this study, the algorithm was designed to automatically calculate the angle of the acetyaboulum-femoral head for normal/nonnormal reading of the infant hip joint and to perform diagnosis automatically by deep learning of existing images. Experiments have shown that the speed and accuracy of diagnosis are improved compared to doctors.

Acknowledgements

This research is supported by "The Foundation Assist Project of Future Advanced User Convenience Service" through the Ministry of Trade, Industry and Energy (MOTIE) (R0004840, 2018) and Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (NRF-2017R1D1A1B04031182) the Ministry of Trade, Industry and Energy(MOTIE) and Korea Health Technology R&D Project through the Korea Health Industry Development Institute (KHIDI), funded by the Ministry of Health & Welfare, Republic of Korea (grant number : HI17C2594).

References

[1]. Webster J. G., Encyclopedia of Medical Devices and Instrumentation, *John Wiley & Sons,* New York, 1988.

[2]. Geddes L. A., Baker L. E., Principles of Applied Biomedical Instrumentation, *John Wiley & Sons,* New York, 1989.

[3]. Cameron J. R., Skofronick J. G., Physics of the Body, *Medical Physics Publishing Corporation*, 1999.

[4]. Perloff D., Grim C., Flack J., Frohlich E. D., Hill M., M. McDonald, *et al*., Human blood pressure determination by sphygmomanometry, *Journal of Amreican Heart Association*, Vol. 88, Issue 5, 1993, pp. 2460-2470.

[5]. O'Grady N. P., Alexander M., Dellinger E. P., Gerberding G. L., Heard S. O., Maki D. G., *et al*., Guidelines for the prevention of intravascular catheter-related infection, *Clinical Infectious Diseases*, Vol. 52, Issue 9, 2011, pp. e162-e193.

[6]. Drzewiecki G. M., Melbin J., Noordergraaf A., The Korotkoff sound, *Ann. Biomedical Eng.*, Vol. 17, 1989, pp. 325-359.

[7]. Moraes J. C. T. B., Cerulli M., Ng P. S., Development of a New Oscillometric Blood Pressure measurement System, in *Proceedings of the IEEE Computers in Cardiology Conf.*, Vol. 26, 1999, pp. 467-470.

[8]. Wax D. B., Lin H. M., Leibowitz A. B., Incasive and concomitant noninvasive intraoperative blood pressure monitoring, *Anesthesiology*, Vol. 115, 2011, pp. 973-978.

[9]. Verkruysse W., Svaasand L. O., Nelson J. S., Remote plethysmographic imaging using ambient light, *Optics Express*, Vol. 16, Issue 26, 2008, pp. 21434-21445.

[10]. Poh M. Z., McDuff D. J., Picard R. W., Non-contact, automated cardiac pulse measurements using video imaging and blind source separation, Optics Express, Vol. 18, Issue 10, 2010, pp. 10762-10774.

[11]. Poh M. Z., McDuff D. J., Picard R. W., Advancements in noncontact, multiparameter physiological measurements using a webcam, *IEEE Transactions on Biomedical Engineering,* Vol. 58, Issue 1, 2011, pp. 7–11.

[12]. McDuff D., Gontarek S., Picard R. W., Improvements in remote cardiopulmonary measurement using a five band digital camera, *IEEE Transactions on Biomedical Engineering*, Vol. 61, Issue 10, 2014, pp. 2593–601.

[13]. Gesche H., Grosskurth D., Küchler G., Patzak A., Continuous blood pressure measurement by using the pulse transit time comparison to a cuff-based method, *European Journal of Applied Physiology*, Vol. 112, Issue 1, 2012, pp. 309-315.

[14]. Turk M., Pentland A., Eigenfaces for recognition, *Journal of Cognitive Neuroscience*, Vol. 3, Issue 1, 1991, pp. 71–86.

[15]. Zhao W., Chellappa R., Phillips P. J., Rosenfeld A., Face recognition: A literature survey, *ACM Computing Surveys,* Vol. 35, Issue 4, 2003, pp. 399–458.

[16]. Kamarainen J. K., Kyrki V., Kälviäinen H., Invariance Properties of Gabor filter-based features-overview and applications, *IEEE Transactions on Image Processing*, Vol. 15, Issue 5, 2006, pp. 1088–1099.

[17]. Wiskott L., Fellous J. M., Kuiger N., Malsburg C., Face Recognition by Elastic Bunch Graph Matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, Issue 7, 1997, pp. 775–779.

[18]. Brown L. G., A Survey of Image Registration Techniques, *ACM Computing Surveys*, Vol. 24, Issue 4, 1992, pp. 325-376.

[19]. Zitova B., Flusser J., Image Registration Methods: A Survey, *Image and Vision Computing*, Vol. 21, Issue 11, 2003, pp. 977-1000.

[20]. Rangarajan A., Chui H., Bookstein F., The Softassign Procrustes Matching Algorithm, *Medical Image Analysis*, Vol. 4, 1999, pp. 1-17.
[21]. Besl P. J., Mckay N. D., A Method for Registration of 3-D Shapes, *IEEE Transaction Pattern Analysis and Machine Intelligence*, Vol. 14, Issue 2, 1992, pp. 239-56.
[22]. Chui H., Rangarajan A., A New Point Matching Algorithm for Non-Rigid Registration, *Computer Vision and Image Understanding*, Vol. 89, Issue 2-3, 2003, pp. 114-141.
[23]. Rangarajan A., Gold S., Mjolsness E., A Novel Optimizing Network Architecture with Applications, *Neural Computing*, Vol. 8, Issue 5, 1996, pp. 1041-1060.
[24]. Myronenko A., Song X., Carreira-Perpinan M. A., Non-rigid Point Registration: Coherent Point Drift, *Advances in Neural Information Processing Systems*, Vol. 19, 2007, 1009-1016.
[25]. Belongie S., Malik J., Puzicha J., Shape Matching and Object Recognition Using Shape Contexts, *IEEE Transaction Pattern Analysis and Machine Intelligence*, Vol. 24, Issue 4, 2002, pp. 509-522.
[26]. Zheng Y., Doermann D., Robust Point Matching for Nonrigid Shapes by Preserving Local Neighborhood Structures, *IEEE Transaction Pattern Analysis and Machine Intelligence*, Vol. 28, Issue 4, 2006, pp. 643-649.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 102-109

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

The Automated Diagnosis Architecture and Deep Learning Algorithm for Cervical Cancer Cell Images

[1, *] Jong-Ha Lee and [2] Sangwoo Cho

[1] Dept. of the Biomedical Engineering, Keimyung University, Daegu, South Korea
[2] The Center for Advanced Technical Usability and Technologies, Keimyung University, Daegu, South Korea
[*] E-mail: Segeberg@gmail.com

Received: 22 January 2021 /Accepted: 22 February 2021 /Published: 28 February 2021

Abstract: Cervical cancer is the second most common cancer in women worldwide, with approximately 500,000 cases each year. Approximately 80 % of the reported deaths are observed in countries with poor medical conditions, such as developing countries; 230,000 people die each year from cervical cancer. When detected early, it is possible to prevent progression to invasive cancer by adequate treatment. Therefore, it is very important to detect human papillomavirus (HPV), which is known as a major cause of cervical cancer, and cervical cancer that has already progressed. The most well-known test to diagnose cervical cancer is the Pap test. Although the Pap test consumes more time to diagnose cervical cancer, this test has saved the lives of many patients through early detection of cells progressing into cervical cancer for treatment and has contributed significantly to the prevention of cervical cancer and the reduction of mortality due to cervical cancer. However, since the Pap test requires emanating the cell slide of each patient, which takes a lot of time, the pathologists performing Pap tests tend to become very tired, and it is very difficult to examine many patients. Therefore, the need for diagnostic aids to help pathologists make quick decisions is on the rise. This study aimed to address this problem by developing an automatic diagnostic aid tool for cervical cancer using Yolo V3, a deep learning algorithm. First, the RGB cell image is converted into a gray-scale image by pre-processing, and the noise in the image is removed using a 2-dimensional Gaussian smoothing filter. Next, each cell image for training was labeled by a pathologist. Finally, using the trained algorithm, the cells in the image from the Pap test are identified and marked by bounding boxes to aid the pathologist with rapid diagnosis. For training of the model, 5,631 pre-processed Pap test images were used, and the model was then tested using 563 images. In this process, the performance indicator of PASCAL Visual Object Classes was used, which was set by raising the threshold from 0 to 1.

The precision and recall values for all test images were obtained to calculate the average precision value corresponding to the recall value and to use it as an indicator to determine the performance of the algorithm. The average precision in this study was 73.34 %, and the model may be used as an auxiliary tool for pathologists performing Pap tests by improving accuracy using additional data in the future.

Keywords: Cervical, Cancer, Cell, Deep learning, AI.

1. Introduction

1.1. Background

Cervical cancer is the second most prevalent cancer among women worldwide. Approximately 500,000 women are diagnosed with cervical cancer each year, and approximately 230,000 women die each year due to cervical cancer. In particular, a majority of women who are diagnosed with cervical cancer belong to developing countries, where the incidence is 7-8 times higher than that of developed countries. In

South Korea, according to the data from Statistics Korea in 2017, a total of 3,469 cervical cancer cases were reported, and it is the fourth most common cancer among Korean women, next to breast cancer, gastric cancer, and thyroid cancer.

Compared to a previous report published by Statistics Korea, the proportion of Korean women with cervical cancer seems to have decreased, which is due to improved early diagnosis and treatment, with increased interest in cervical cancer. However, cervical cancer remains a common cancer in women, and it is fatal if left unattended. When detected early, it is possible to prevent cancer progression to an invasive tumor. In other words, early detection of human papillomavirus (HPV), the major cause of cervical cancer, and early diagnosis of cervical cancer are crucial.

1.2. Theoretical Background

1.2.1. Cervical Cancer

The uterus is divided into two parts: the body part, which occupies approximately three quarters of the uterus, and the cervical part connected to the vagina. Cervical cancer occurs in the cervix, which is the neck of the uterus. Cervical cancer occurs when normal epithelial cells in the epidermis of the cervix undergo microscopic morphological changes and form cervical intraepithelial dysplasia (or cervical intraepithelial neoplasia, CIN), a state between normal tissues and cancerous tissues; CIN then progresses to cervical epithelial cancer, with the presence of cancer cells only in the epithelium of the cervix, which is stage 0 cervical cancer. If cervical cancer is not detected at stage 0 and is not treated, it then progresses to invasive cervical cancer.

Normal cells gradually develop into cancerous cells, and such progression can take years to decades. Normal cells develop into ASCUS, L-SIL, ASCUS-H, H-SIL, and, eventually, to cervical cancer. Epithelial cells make up the outermost part of a tissue. The epithelium is located on the outermost part of the skin or the entire organ of the body, and the connective tissue is located below it. The region that separates the epithelium and the connective tissue is the basement membrane. When the cancer penetrates through this basement membrane, it is judged to have progressed to being invasive cancer. Squamous cell carcinoma (SCC), which is one of the two main types of cervical cancer, accounts for approximately 80 % of all cervical cancers. Adenocarcinoma, which is the other type, accounts for 10–20 % of all cervical cancers. Cases involving both adenocarcinoma and SCC are called mixed carcinoma, which account for 2–5 % of all cancers. SCC is then classified as 65 % of large cell non-keratinized subtype, 25 % of large cell keratinized subtype, and 5 % of small cell non-keratinized subtype. The large cell subtype shows a better prognosis than the small cell subtype.

In terms of the risk factors for cervical cancer, the incidence rate is relatively higher in women who have experienced sexual intercourse at an early age, women who have a sexual partner with HPV, and women who have multiple sexual partners, indicating that it is closely associated with sexual intercourse. It also tends to occur relatively more in smokers or in those who are less socially and economically privileged. In addition to these risk factors, it is known that sexual intercourse-induced HPV infection has a major effect on the onset of cervical cancer. Approximately 50–80 % of people are infected with HPV at least once in their lifetime. With regard to cervical cancer, HPV is divided into high-risk HPV and low-risk HPV, and there are approximately 150 kinds of viruses. The U.S. Centers for Disease Control and Prevention states that the most important cause of cervical cancer is persistent infection by HPV.

People who are engaged in sexual intercourse are constantly exposed to the risk of infection by HPV. Most of them are infected during sexual intercourse, but there is a low probability that they are infected through indirect sexual contact or by means other than sexual activity. Most HPV infections heal naturally within 6 months to 2 years. In addition, 56 % cases of cervical cancers occur in women who are not regularly screened. Therefore, early and continued participation is the most effective method to prevent cervical cancer. If abnormal cells are found in the cervix during regular screening, then appropriate treatment should be initiated.

One of the most common symptoms of cervical cancer is abnormal vaginal bleeding. Abnormal vaginal bleeding refers to vaginal bleeding after menopause or irregular bleeding before menopause but not during menstruation. Such bleeding occurs when blood vessels are distributed to newly formed cancer cells for nourishment. This bleeding is common after vaginal lavage, heavy exercise, sexual intercourse, or defecation. When cancer occurs due to secondary transmission or when cancer cells are necrotized, the secretions have strong odor, resulting in an increase in overall vaginal discharge. Cervical cancer progression and metastasis to other organs may result in kidney swelling and back pain.

Currently, regular screening is the best approach for detecting and preventing the formation of abnormal cells before developing cervical cancer. Regarding the screening method, the Pap test [1], known as the cervical cancer test, is performed, wherein cervical cells are collected using a small brush, and the pathologist directly detects abnormal cells under a microscope. Women who have had sexual intercourse or are 20 years of age or older are screened for cervical cancer, and it is recommended that they undergo the Pap test once a year. Moreover, an additional test for HPV is recommended for a more accurate examination.

When abnormal signs are detected during the examination, colposcopy and, in some cases, biopsy are performed, wherein part of the cervical tissue is removed and examined. Treatment of cervical cancer

depends on the disease progression and the patient's condition. Dysplasia and intraepithelial cancer can be cured by more than 95 % by treating the affected area with cryotherapy, laser therapy, conization, or loop electrosurgical excision procedure (LEEP). Pregnancy is also possible after treatment.

1.2.2. Cervical Cancer Screening

Cervical cancer screening is largely performed in two ways.

The Pap test, which is most frequently performed for the diagnosis of cervical cancer, came into use in clinical practice in 1943 after George N. Papanicolaou accidentally observed malignant cells in cervical decidual cell smear in 1928, suggesting that this test could be used for the diagnosis of cervical cancer. The Pap test has been recognized as the standard early screening method for cervical cancer for the past 50 years. Using this method, abnormal findings of cells can be observed under a microscope. First, a colposcope is inserted into the vagina, then the sample is collected with a brush and applied onto a glass slide, and smearing and staining are performed on the glass slide. It is a relatively safe method, is inexpensive with low procedural and testing cost; the sample collection procedures is simple, causing less pain, and has relatively high accuracy, i.e., 75–85 %. However, the greatest limitation of the Pap test is that it may yield a false-negative result, in other words, there are cases in which a lesion is not accurately detected [2]. Despite this false-negative rate, the Pap test is the most efficient and the only screening method with proven effects of screening, and it has contributed significantly to the prevention of cervical cancer and the reduction of mortality because cervical cancer is the most essential testing method.

Liquid-based cytology (LBC) is another widely used approach after the Pap test. LBC is different from the Pap test in terms of sample preparation for testing. As in the conventional Pap test, cells are collected directly from the cervix with a small brush, but the collected cells are stored in a bottle containing a small preservative instead of directly smearing them on a microscope slide and testing them immediately.

A slide is prepared through a separate treatment when used for observation under a microscope. This method shows high stability in terms of cell damage, which is a limitation in the Pap test and demonstrates better ability to detect H-SIL. However, there is a disadvantage that the cost of the test is relatively higher than that of the Pap test.

1.3. Previous Studies

1.3.1. Machine Learning

Machine learning, defined by the renowned computer scientist Arthur Samuel as "a field of research that allows computers to learn without being explicitly programmed by humans". Machine learning algorithms are often classified into supervised learning algorithms or unsupervised learning algorithms. First, supervised learning provides correct answers to the algorithms to let them draw conclusions by learning the data. In other words, the conclusion is drawn by learning the data to find a pattern to produce a more accurate result based on the example provided and then by undergoing the feedback process through an algorithm without external intervention to correct the result. Representative supervised learning algorithms include KNN and XGBoost. Unsupervised learning refers to an algorithm that presents the correct answer by itself without giving the correct answer data to the algorithm. Representative unsupervised learning algorithms include K-means [3] and PCA [4].

Deep learning is attracting attention under the concept of machine learning. Deep learning consisting of neural networks, as shown in Fig. 1, has the accuracy that existing algorithms have not been able to achieve in various areas. This deep learning is classified as a type of supervised learning in that the correct answer data are given, but it is also classified into a new category called machine learning, with the use of neural networks due to the advent of various deep learning model structures. In particular, in this study, deep learning was used to aid in diagnosis by identifying cells from cervical cell images that have unstructured data and by distinguishing abnormal cells from normal cells.

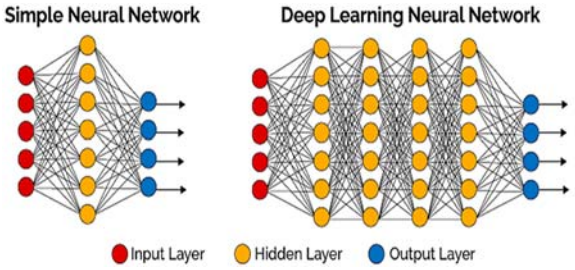


Fig. 1. Examples of differences between neural networks and deep learning networks.

1.4. Purpose

The Pap test, which is the most frequently used test to diagnose cervical cancer, detects lesions of cervical cancer, and its accuracy largely depends on the pathologist's skill, with a false-negative rate of 6–55 %. The pathologist also takes a lot of efforts to process a lot of data when examining a lesion sample. To address these problems, studies on automatized diagnosis of cervical cancer are being actively conducted. To automatically detect cervical cancer, it is necessary to divide the progression of cervical cancer by stage. There are three main ways to divide the stages: the method of dividing into normal and abnormal cells; the Bethesda System that divides progression into four stages, namely, normal cell, L-SIL, H-SIL and CIS; the World Health Organization

classification into seven stages, namely, superficial, intermediate, columea, mild dysplasia, moderate dysplasia, severe dysplasia, and carcinoma. In this study, data are classified according to "The Pap Test and Bethesda" to assist pathologists in making quick decisions. Data were collected from 987 patients from 2017 to 2018, and Yolo V3, a powerful algorithm for object detection, was used to find cells in the cervical cancer images, marking them by bounding boxes, and to distinguish abnormal cells from normal cells. This study aimed to reduce the time required for detecting lesions and to improve the efficiency of cervical cancer screening by helping the pathologist make judgments [5].

2. Materials and Methods

2.1. Materials

2.1.1. Experimental Environment

This study was conducted using 64 GB RAM, Intel i5-9600KF, and NVIDIA GeForce RTX 2080TI. The model was configured using Tensorflow version 1.14, the Python-based machine learning library of Google, and it was produced using Python packages such as Numpy and Open-cv.

2.1.2. Data and Labeling

The data of 987 patients were collected during 2017-2018. In 5,631 pieces of data, 2,827 normal cells and 2,804 abnormal cells were classified by a pathologist to construct ground truth. Fig. 2 shows the sample of images.

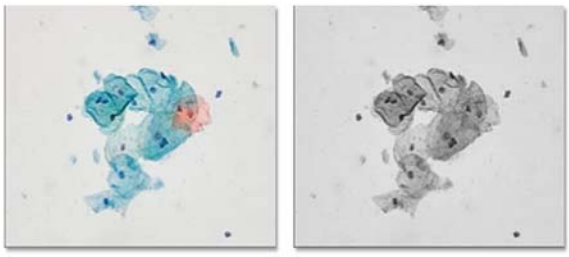


Fig. 2. Raw image (left) and pre-processed image (right).

2.2. Methods

2.2.1. Data Pre-processing Process

The cell images are displayed in color, as Papanicolaou staining has been applied to all the cells. The cytoplasm contains alkaline substances and has a strong affinity to dyes; such characteristics allow Papanicolaou staining to stain cells in various colors. After staining, the structure of the cell appears more clearly. Since the cytoplasm appears bright, providing a clear view of the nuclear chromatin, it is easy to distinguish abnormal cells from the normal ones. Papanicolaou staining is applied only to facilitate the identification of morphological features of cervical cells. The color difference caused by staining does not act as a factor that distinguishes negative cells from the positive cells when analyzing the cervical cancer images; therefore, the RGB image is converted to gray-scale image and used for training to prevent the model from recognizing the colors as a factor for training as shown in Fig. 3. In addition, 2-DGSF (2-Dimensional Gaussian Smoothing Filter) was used to remove noise in the cell image [6].

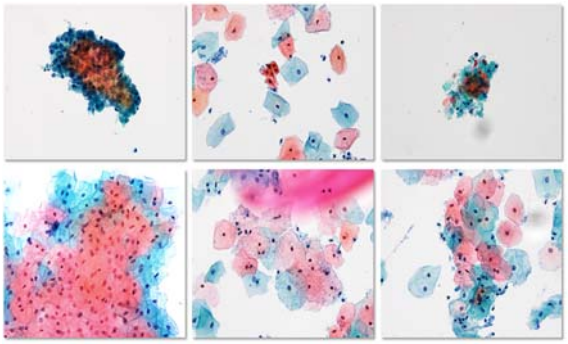


Fig. 3. Data used for training.

2.2.2. Design of Deep Learning-based Object Recognition Algorithm Yolo V3 Model

Yolo V3 [7], an object recognition algorithm based on deep learning, has proven performance in various fields. In particular, the algorithm is light enough to run smoothly on low-performance devices while maintaining a high level of accuracy. This study therefore used Yolo V3 to build a model that distinguishes abnormal cells from the normal ones among the cells in the cervical cancer images by marking the cells using bounding boxes. First, Yolo V3 predicts the position of the bounding boxes by generating anchor boxes, which are also bounding boxes with predefined shapes, by K-means clustering. While the previous Yolo 9000 algorithm predicted the center point of the grid, the Yolo V2 algorithm calculates the distance from the upper left offset and predicts the ratio of the width and height of the anchor box to be adjusted by involution.

In addition, the backbone network of Yolo V3 helped the model become much lighter. As the VGG model, the backbone network of the previous Yolo 9000, was very complex, the Darknet-19 architecture [8], which uses a relatively small number of parameters and has good performance, was applied.

A layer consisting of 3×3 Convolution and 1×1 Convolution was continuously configured, and the stride is set to 2 to lower the resolution of the feature map when performing convolution instead of MaxPooling. Then, the residual value is obtained by skip connection, and after passing through the average

pooling and fully connected layer in the last layer, the result is obtained through softmax. In Yolo V3, Darknet-53 with much more layers by applying the skip connection proposed by ResNet [9] to Darknet-19 as shown in Fig. 4.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Fig. 4. Darknet 53 architecture, the backbone network of Yolo V3.

These characteristics enable the configuration of a lightweight model that can be mounted on a single board computer with low computing capacity, which is suitable for application as a diagnostic tool in areas with relatively poor medical infrastructure. Therefore, this study aimed to use Yolo V3 to distinguish abnormal cells from normal cells in cervical cancer cell images. Fig. 5 shows the proposed system.

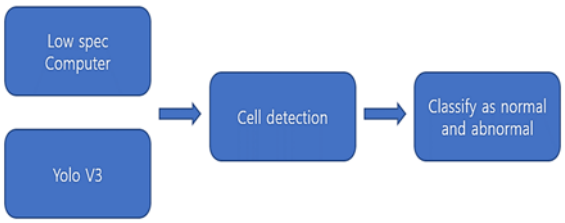


Fig. 5. Proposed assistance system for cervical cancer diagnosis.

2.2.3. Training

In the training process, among 5,631 pieces of data, 2,827 normal cells and 2,804 abnormal cells were identified and labeled by a pathologist.

The training parameters consisted of 600 epochs in a batch size of four each, and the weights learned from the COCO data set, which is the public data set of Microsoft from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), were applied by fine-tuning. The COCO data set is composed of 80 categories, each of which is labeled. It consists of approximately 300,000 images and 1.5 million objects, and information of each image can be obtained as a JSON file. Fine-tuning is a type of transfer-leaning among weight initialization techniques that takes the weight of an existing trained model for retraining the data to be trained. Fine-tuning greatly reduces the epochs required for training and improves the performance of the model.

For the learning rate applied to learning, the first stage was set up to 200 epochs, and the value 1e-4 was applied for training. The second stage was set up to 600 epochs, and the value 1e-6 was applied for loss convergence. As shown in Fig. 6, training was conducted using the Tensor board of the deep learning framework Tensorflow to check the progress.

3. Results

3.1. Study Results

3.1.1. Performance Indicator

To evaluate the previously trained model, the performance indicator of the Pascal VOC competition is commonly used in object detection algorithms [10]. In this performance verification method, the average precision using recall and precision is expressed in each class, that is, the mean average precision (mAP) is measured by averaging the values for normal cells and abnormal cells. The value of recall is obtained by dividing the entire true proposition by the true items detected by the model, and this indicates how accurately the target object is identified. The value of precision is obtained by dividing all the items identified by the model in the correct answer label, which indicates how accurate the detected result is, that is, how much of the actual correct answer label is included among the detected results. The threshold is raised from 0 to 1 based on the IoU value, an indicator showing how much the correct answer label coincides with the detected bounding box, to find the average precision. Items for evaluating most object detection algorithms use this method to verify the performance of the model, and it was used in this study to assess the performance of the model. In addition, the mean accuracy was obtained by calculating the accuracy of the two classes of detected cells.

3.1.2. Model Performance Evaluation

The data used in the model evaluation were classified the 563 cervical cancer cell images; when testing cervical cancer cell images with a threshold of 0.45, it was judged by outputting the image as shown in Fig. 7.

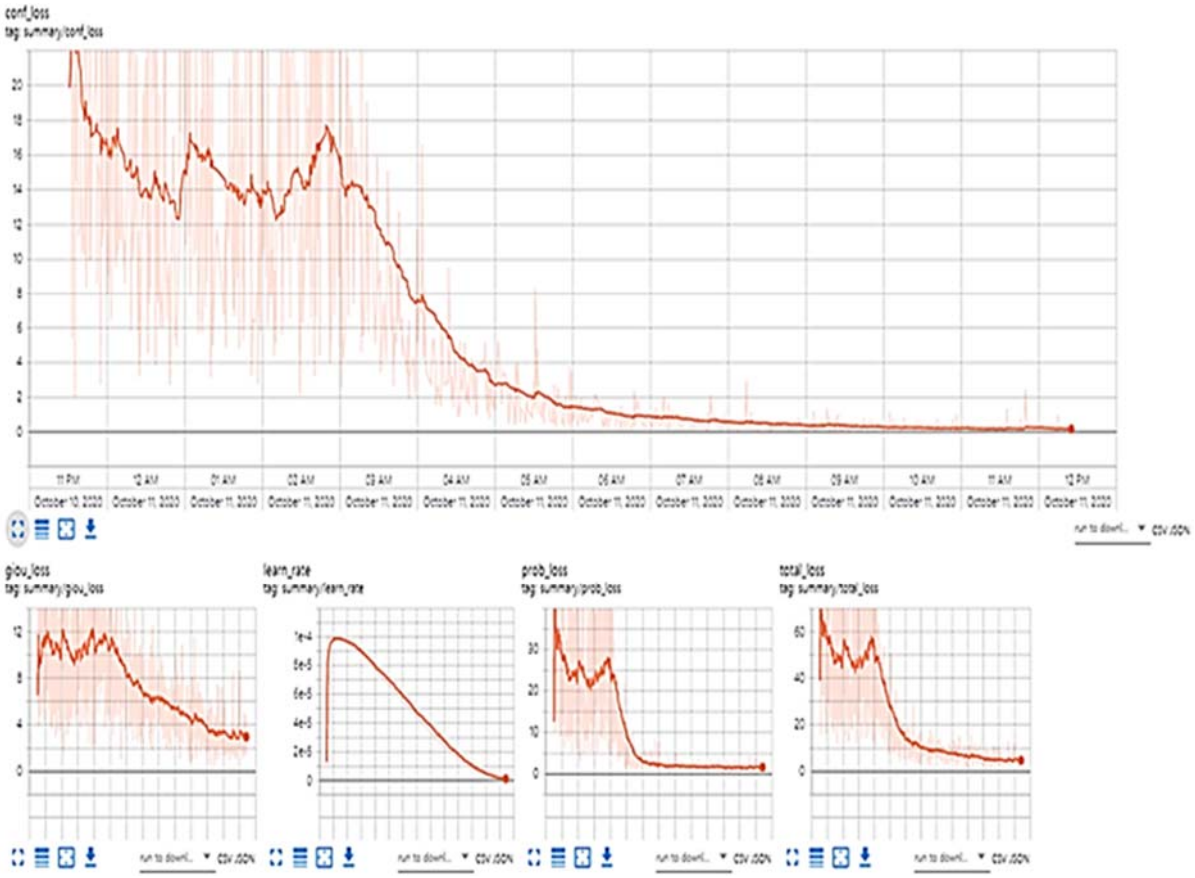


Fig. 6. Loss value of the Tensor board.

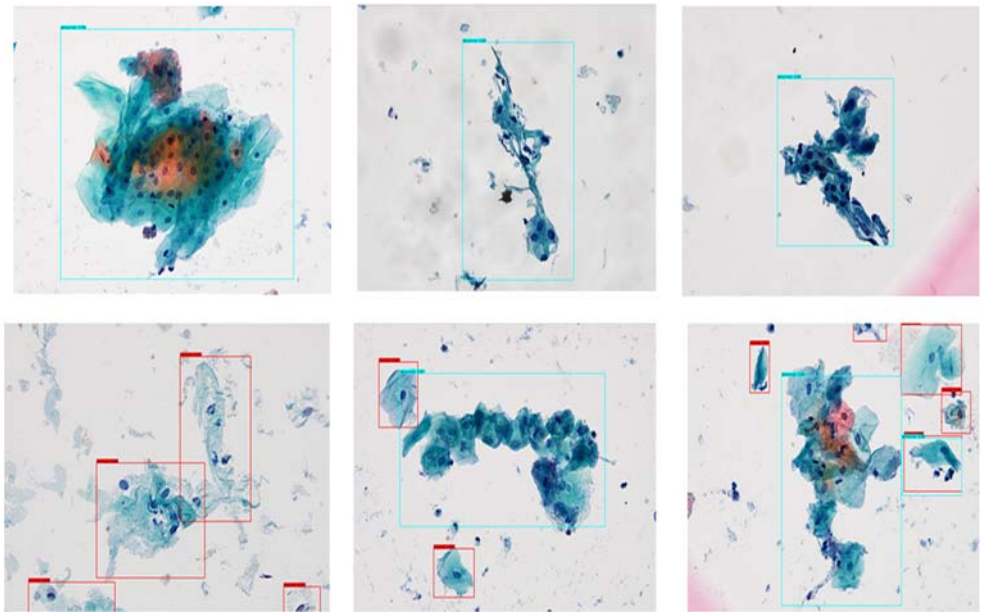


Fig. 7. Cervical cancer images classified using the model.

Fig. 8 shows our Mean average precision and accuracy result. Mean average precision was 73.34 %, which was quite high, and mean accuracy was 86.45 %. Considering the nature of the deep learning model, the larger the size of the ground truth, the higher the accuracy tends to be. It can be used as an auxiliary tool for pathologists who conduct the Pap test, by improving the accuracy using additional data later on.

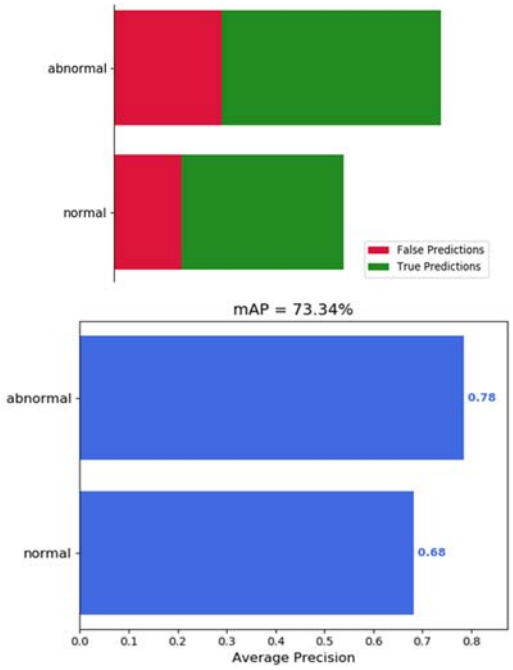


Fig. 8. Mean average precision and mean accuracy of the model.

4. Discussion

This study proposed a method to distinguish abnormal cells from normal cells in cervical cancer cell images, even with low computing power using the deep learning model Yolo V3. Despite relatively fewer data, the performance was higher than the mean average precision of other object detection algorithms using the COCO dataset which was used for transfer training. However, its current performance is not enough to assist pathologists in cervical cancer diagnosis. Nevertheless, there is ample room for improving the accuracy, as the nature of deep learning algorithms allows the accuracy to be improved by obtaining additional data. Even though various deep learning techniques such as data argumentation are not applied, its performance is competitive with that of other object detection algorithms using Microsoft COCO data set, suggesting that Yolo V3 works without problems even in pathological images.

The nature of deep learning demonstrates strong performance in unstructured data such as images; therefore, the use of a deep learning algorithm such as Yolo V3, which shows high performance even on a single board computer, as an auxiliary tool for automatic diagnosis would be useful for developing countries that have a relatively high incidence of cervical cancer. This approach will be very useful to pathologists who spend more time and therefore experience high levels of fatigue due to Pap-testing when screening for cervical cancer, especially when the number of pathologists continues to decrease.

The total number of data used for this study was 5,631, which were classified into 2,827 normal cells and 2,804 abnormal cells. Since the size of the data is significantly small, there remains room for significant improvement in the accuracy of the model by adding data.

Moreover, when the cells identified as abnormal cells are classified into each stage according to the progression of cervical cancer, that is, when the number of classes of abnormal cells is increased, the number becomes very low, resulting in data imbalance. To this end, additional data sets should be obtained to enable the classification of cervical cancer by stage. Then, the techniques used in various deep learning should be applied to improve the accuracy and to reinforce the training against noise data. As Yolo V3 has demonstrated satisfactory performance with pathological images, it can be applied to a wide variety of diseases other than cervical cancer by constructing a model through training with other pathological images.

5. Conclusion

The Pap test, which is the most frequently performed test for the diagnosis of cervical cancer, is a method for observing abnormal findings of cells under a microscope. It is performed by inserting a colposcope into the vagina, collecting a sample with a brush, applying the sample to a glass slide, and smearing and staining the slide. It has advantages such as being inexpensive in terms of procedural and test costs, having a simple sample collection process causing little pain, and having relatively high accuracy. However, the Pap test involves a significant amount of time for diagnosis. Contrary to the recent surge in demand for testing, the number of pathologists is decreasing. Moreover, in developing countries, where early diagnosis of cervical cancer is difficult due to poor medical infrastructure, the mortality rate is 7-8 times higher than that in developed countries.

To compensate for these problems, this study proposed a diagnostic aid tool using Yolo V3, a deep learning-based object detection algorithm that can be applied to a single board computer, which is relatively easy to distribute due to low cost.

A total of 5,631 cell images were obtained to train the model, of which 2,827 normal cells and 2,804 abnormal cells were labeled by a pathologist. Moreover, as a data pre-processing process, the RGB scale was converted to gray-scale, and a 2-Dimensional Gaussian Filter was used to remove noise.

To apply fine tuning, a method of transfer learning, the weight from the Yolo V3 model trained with the COCO data set was used as the initial weight.

The results of the training showed that the mean average precision was 73.34 %, which was high, and the mean accuracy was 86.45 %.

In this study, despite the small number of data, Yolo V3 showed satisfactory performance. As Yolo V3 was proven to be applied to a single board

computer with its lightweight algorithm, it seemed possible to develop an auxiliary tool for automatic diagnosis of cervical cancer at low cost to compensate for the high fatigue experienced by pathologists and to allow utilization in developing countries with relatively poor medical infrastructure. In addition, Yolo V3 has demonstrated smooth learning of pathological images, suggesting the possibility for application in the diagnosis of various other diseases.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A1B04031182/ NRF2020R1I1A1A01072101).

References

[1]. J. H. Park, The Usefulness of Cervicovaginal Cytology as a Primary Screening Test, *The Korean Journal of Cytopathology*, Vol. 19, Issue 2, 2008, pp.107-110.

[2]. Y. T. Kim, Causes and Diagnoses of Cervical Cancer, *Korean Med Assoc*, Vol. 50, Issue 9, 2007, pp. 769-777.

[3]. T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, A. Y. Wu, An efficient k-means clustering algorithm, Analysis and implementation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, Issue 7, 2002, pp. 881-892.

[4]. W. Chen, M. J. Er, S. Wu, PCA and LDA in DCT domain, *Pattern Recognition Letters*, Vol. 26, Issue 15, 2005, pp. 2474-2482.

[5]. N. A Mat-Isa, An automated cervical pre-cancerous diagnostic system, *Artificial Intelligence in Medicine*, Vol. 42, Issue 1, 2008, pp.1-11.

[6]. B. Bosi, G. Bois, Y. Savaria, Reconfigurable pipelined 2-D convolvers for fast digital signal processing, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 7, Issue 3, pp. 299-308.

[7]. J. Redmon, A. Farhadi, Yolov3: An incremental improvement, *arXiv:1804.02767, Computer Vision and Pattern Recognition*, 2018.

[8]. J. Redmon, Darknet: Open source neural networks in c, http://pjreddie.com/darknet/, 2013–2016.

[9]. K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 2016, pp. 770–778.

[10]. M. Everingham, L.Van Gool, C. K. I Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes (VOC) Challenge, *IJCV*, Vol. 88, 2010, pp. 303-338.

Sensors & Transducers, Vol. 249, Issue 2, February 2021, pp. 110-118

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2021
http://www.sensorsportal.com

Experimental Comparison of Transformers and Reformers for Text Classification

*** Roghayeh Soleymani, Julien Beaulieu and Jérémie Farret**
Inmind Technologies Inc., Montreal, Canada
Tel.: + 1(514) 871-0470
* E-mail: esoleymani@inmindtechnologies.com

Received: 30 November 2020 /Accepted: 15 January 2021 /Published: 28 February 2021

Abstract: In this paper, we present experimental analysis of Transformers and Reformers for text classification applications in natural language processing. Transformers and Reformers yield the state-of-the-art performance and use attention scores for capturing the relationships between words in the sentences which can be computed in parallel on GPU clusters. Reformers improve Transformers to lower time and memory complexity. We will present our evaluation and analysis of applicable architectures for such improved performances. The experiments in this paper are done in Trax on Mind in a Box with four different datasets and under different hyperparameter tuning. We observe that Transformers achieve better performance than Reformers in terms of accuracy and training speed for text classification. However, Reformers allow the training of bigger models which would otherwise cause memory failures with Transformers.

Keywords: Natural Language Processing, Text Classification, Transformers, Reformers, Trax, Mind in a Box.

1. Introduction

Text classification is one of the important natural language processing (NLP) tasks that is applicable in real-world problems such as topic tagging, question answering, spam detection, sentiment analysis, news categorization, etc. Text classification or tagging can be used in the Fog layer computations to make text data deep [1] or to enrich data. This task includes an NLP step immediately after data acquisition, in Fog layer or in the cloud to organize and structure data. This strategy makes organizing, accessing, transmitting, and processing the data more efficient.

Designing text classification models with classical machine learning methods follows a two-step process: feature extraction and classification. Feature extraction consists of defining some hand-crafted feature vectors and designing high quality features. This step requires strong domain knowledge, tedious feature engineering and analysis, and may not be transferable between different applications.

Since 2012, deep learning models are being applied to different tasks in the world of machine learning and therefore in NLP. Neural network architectures used for text classification include recurrent neural networks (RNNs), convolutional neural networks (CNNs), Capsule Nets, Transformers, Reformers, and more (see the review by Minaee, *et al.* [2]). Most of these models learn feature representations automatically with less hand-engineering and allow for knowledge transfer between different applications of NLP.

In this paper, we focus on Transformers and Reformers that both rely on attention mechanisms for relating different positions of a single sequence in parallel and yield state-of-the-art results in many NLP tasks [2-4]. These models are useful to train language models on Open Data and perform transfer learning on

the application on hand (e.g. Deep Data and Data enrichment as a target application). Attention mechanism allows sequential processing of text to compute a representation of the sequence and capture meaningful relationships between words in a corpus. Transformers allow for much more parallelization than CNNs and RNNs and make it possible to efficiently train very big models on large amounts of data on GPU clusters [2]. Reformers improve the efficiency of Transformers to train larger models on larger corpuses [4].

The experiments presented in the original Reformer paper [4] are performed only on generative and language model training tasks including imagenet64, enwik8-64K, and WMT 2014 English-to-German translation task. In this paper, we extend our experiment on text classification problems including News Classification, and Sentiment Classification compared to the published conference version [5]. We carry out the experiments on four datasets and use different settings to compare the models in terms of performance over training time. The experiments are done using Trax library, an end-to-end library for deep learning by Google Brain team with JAX backend [6].

We use for this study the integrated solution Mind in a Box Catalyst™, an On Premise and Fog computing device that features up to 4 GeForce RTX 2080 Ti for accelerated AI training and inference in its legacy version, and now up to 4 Nvidia Tesla V100 Cards.

2. Background

Transformers originally proposed by Vaswani, *et al.* [3] are used for building language models to learn contextual text representations. These language models can then be used for text generation and classification. Each Transformer block consists of a masked multi-head attention module, followed by a normalization layer and a position-wise feed forward layer (see [3] and Fig. 1). The attention module estimates the level of correlation between each word/character with the other elements using an attention vector. Then, it takes the weighted sum of the elements in this vector as the prediction of the target.

Transformers are more efficient than RNNs for sequential processing of text in parallel, however they are still memory-inefficient to train on long sequences and large models. Attempts to achieve more efficient versions of the Transformer model's self-attention mechanism include augmenting self-attention with persistent memory, adaptive attention span, factorized sparse representation of attention, and product-key attention [2]. The latest state of the art attempt is reported as Reformer model [4].

Reformer model [4] was proposed to solve the memory problem with Transformer models training using approximate attention computation based on locality-sensitive hashing and reversible layers [4]. With Locality Sensitive Hashing (LSH), nearby word embedding vectors get the same hash with high probability and the vectors that have the same hash are assigned to the same LSH bucket. Then LSH buckets are converted to sorted chunks (batches) to allow parallel processing. However, LSH buckets may be uneven in size which makes batching difficult. To tackle this, the batch size is set such that they may contain nearby hashes. Then attention is applied only to these batches and neighbor hash vectors rather than the whole input vector as being done with Transformers (see Fig. 2 and [4] for further details). In addition to LSH, Reformers use reversible layers mechanism, meaning instead of storing all the activations in memory, it recomputes the input of each layer on demand during the back propagation process. Two sets of activations are stored for each layer, one captures changes to the first layer and the other captures the last layer and then they are subtracted (see [4] and [7] for further details). This model has been reported to be memory and time efficient to train on long sequences and large models.

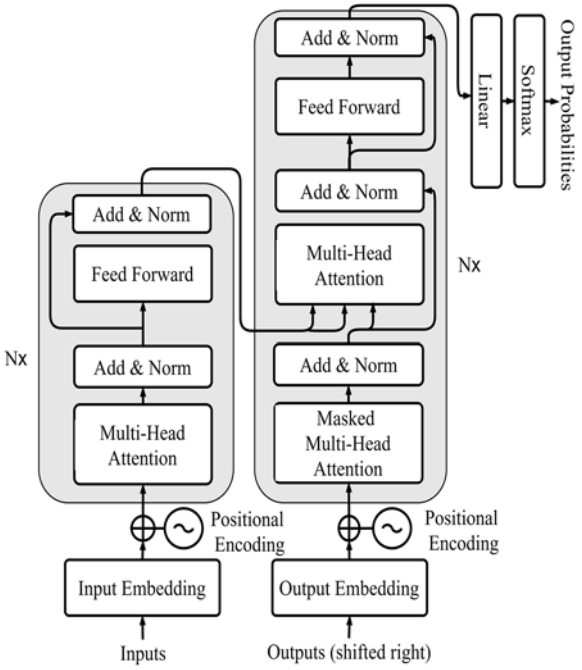


Fig. 1. The Transformer - model architecture [3].

3. Applicable Architectures

The experimental results in this paper rely on the integrated solution, Mind in a Box Catalyst™ with 4 GPU units, for balanced and accelerated training and inference capabilities in various text classification models (see Fig. 3). In this architecture, the objective is to integrate a pretrained language model on Open Data to learn the natural language structure. This model can read, understand, translate, summarize text with near human performance. Then this language model is used to do transfer learning for the target application of NLP on Big Data and generate Deep Data.

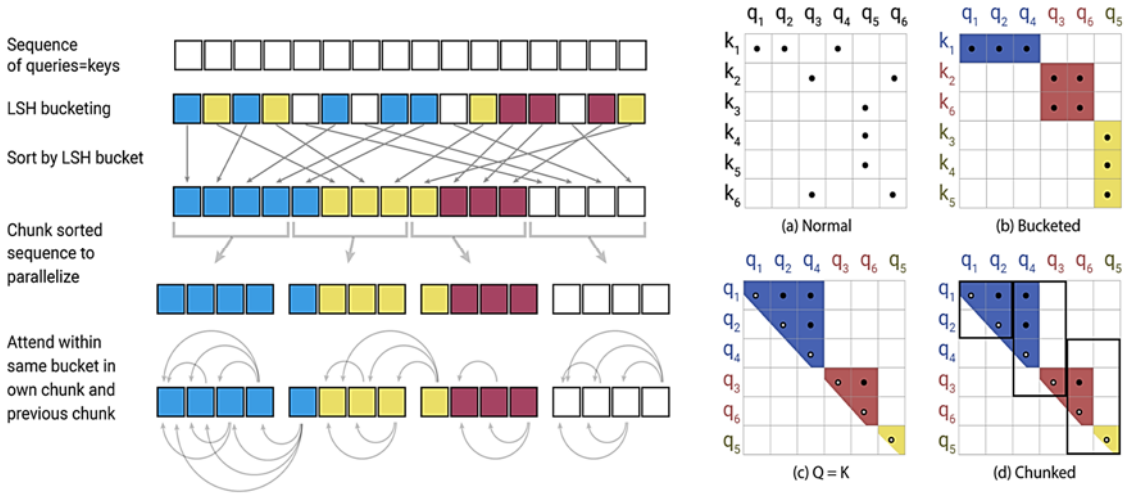


Fig. 2. Simplified depiction of LSH Attention showing the hash-bucketing, sorting, and chunking steps and the resulting causal attentions. (a-d) Attention matrices for these varieties of attention [4].

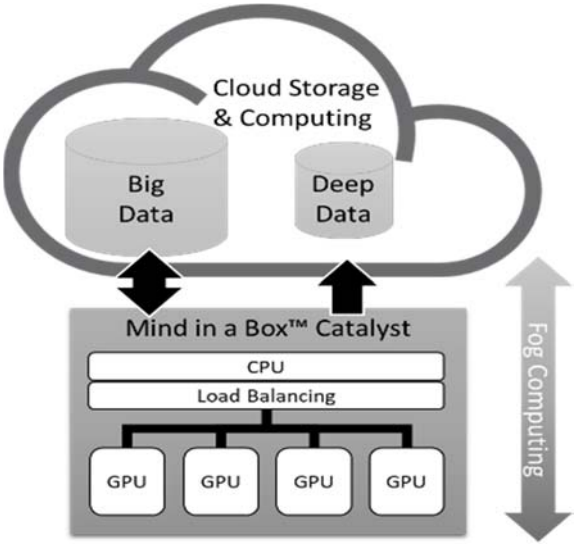


Fig. 3. On Premise/Fog AI functional architecture diagram.

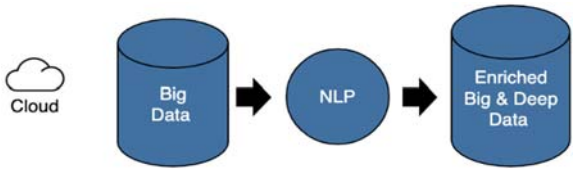


Fig. 4. Using NLP (text classification) in the cloud level to enrich big data.

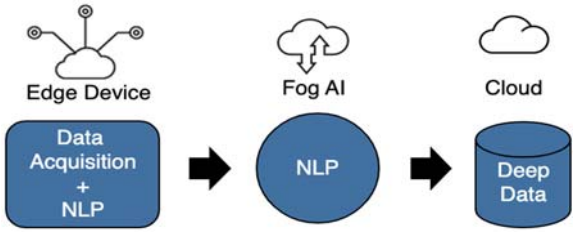


Fig. 5. Using NLP (text classification) on the edge level after data acquisition and before sending it to the fog layer. The data can be further analyzed with NLP (text classification) on the fog level before being sent to the cloud.

The objective sought through this experimental protocol and its underlying architecture is to assess the optimization and acceleration potential of the various models and libraries, to seek real time and / or high-performance computing capabilities. The technical common denominator here is an integration with the Python scripting language, which is currently extensively used to operate many Edge and Fog AI architectures, including the Fog AI appliance used for this experiment.

The applicable architectures for the performance considerations supported in our results are many. But the prime objective is to assess Hybrid Computing Deep Data generation strategies and in particular optimization of the choice of libraries and models depending on the data to be analyzed. Thus, relevant architecture would be Cloud, Edge and Fog AI architectures applying NLP techniques to generate Deep text Data and / or Enriched Big text Data, as illustrated in Fig. 4 and Fig. 5.

Fig. 4 shows a protocol to apply NLP algorithms to large amount of data (Big Data) to obtain Deep Data. This means that irrelevant information is removed, and data is stored in a structured and organized manner. In addition, NLP text classification algorithms can be applied to add labels or tags to the stored data to enrich it and improve its quality for the analysis applied in the next steps of processing the data.

If data is not yet stored as Big data in the cloud, a more efficient protocol can be used for its acquisition and storage. In Fig. 5, such protocol is presented where, data is processed with NLP in the edge device immediately after acquisition, and then sent to the fog layer. In Fog layer, further text processing and classification is performed on the data and when it is sent to the cloud for storage, it is structured and organized and only the relevant information is being

transmitted and stored. With this architecture, the computation is distributed over all three layers of the network and the computation load is reduced in cloud level which in turn reduces the transmission load and latency.

4. Experiments and Results

The experiments presented in the original Reformer paper [4] are performed only on generative tasks and no results of experiments have been presented for comparing Reformers with Transformers on text classification task. In this paper, four datasets are used as Open Data for the experiments on text classification tasks. They are as follows: IMDB and Yelp for binary sentiment classification, Allocine for large-scale sentiment analysis in French, and AGNews for four-class topic classification problem.

For the experiments, we build a Reformer and a Transformer encoder for binary and multiclass classification. A caveat to mention for this implementation is that the Reformer model does not use LSH Attention as referenced in the original Reformer paper. Rather, we use a memory efficient self-attention layer implemented by the Trax team. This efficient self-attention layer has two main differences when compared with the standard self-attention layer:

1. The query weights Q and the key weights K are shared in the dot-production attention which lowers memory consumption.

2. Attention masking is implemented by associating an integer – typically the sequence position - with each query and key vector and defining a function to compute attention masks from this information. The standard attention in comparison is less memory efficient because it instantiates $O(n^2)$-size attention masks.

Overall, the Reformer we use is still much more memory efficient than a Transformer encoder model because of the points mentioned above but also because it uses reversible layers. Reversible layers allow the model to save memory on GPUs and TPUs since we can re-compute inputs for backward propagation, instead of having to store them during the forward propagation. This alone saves a great deal of memory.

Different settings are used to tune the hyperparameters of Transformers and Reformers including optimizers (Adam and AdaFactor [8]), number of layers, number of attention heads, attention type in Reformers, the effect of vocabulary size, the number of features in the encoder inputs (model dimension), and maximum length of the sequences.

It is worth noting that, the length of attention key and value vectors that exists only for Reformer is set to 64.

In experiments of this paper, the performance is compared in terms of minimum testing cross entropy (Min CE), maximum test accuracy (per batch of text) and training time in seconds.

4.1. Adam and Adafactor Optimizers

We start the experiments by comparing the Adam and Adafactor optimizers. In Table 1, results are shown for comparing 2 layers of Transformer and Reformer encoder blocks and 32 attention heads ended with a dense layer and a softmax layer.

The results show that Adafactor performs better than Adam optimizer for both Transformer and Reformers. Therefore, for the rest of the experiments we use AdaFactor optimizer.

Table 1. Results of experiments with Transformers (TR) and Reformers (RF) (2 layers of TR/FR encoder blocks and 32 attention heads, a dense layer and a softmax layer) to compare Adam and AdaFactor optimization algorithms.

Dataset	Optimiser	Train Steps		Dropout rate		Train CE		Test CE		Test Accuracy		Train Time (sec)		Inference Time (sec)	
		TR	RF	TR	RF	TR	RF	TR	RF	TR	RF	TR	RF	TR	RF
IMDB	**Adam**	3000	5000	0.3	0.3	0.308	0.377	0.347	0.395	0.833	0.868	**1491.872**	1084.734	**2.42**	**3.84**
IMDB	**AdaFactor**	3000	5000	0.3	0.3	**0.246**	**0.321**	**0.324**	**0.362**	**0.90**	**0.885**	1540.54	**1068.767**	2.726	3.853
Yelp	**Adam**	5000	10000	0.2	0.2	0.323	**0.271**	0.308	0.276	0.871	0.886	558.18	2684.084	**2.329**	**3.711**
Yelp	**AdaFactor**	5000	10000	0.2	0.3	**0.212**	0.277	**0.202**	**0.219**	**0.920**	**0.910**	**533.50**	**2158.91**	2.405	3.792
AGNews	**Adam**	10000	10000	0.2	0.2	0.271	**0.245**	0.245	0.234	0.893	0.918	1201.45	**2192.94**	**2.454**	4.087
AGNews	**AdaFactor**	10000	10000	0.2	0.2	**0.209**	0.251	**0.219**	**0.203**	**0.910**	**0.935**	**1146.337**	2307.161	2.569	**3.912**

4.2. Attention Type in Reformers

Since we have the option of using standard self-attention layers as well as a more memory efficient implementation, we want to test if using a more efficient model will have an impact on our classification task's performance. We trained the Reformer model using two different approaches.

1. Alternating between memory efficient self-attention layers, as well as standard self-attention layers, for a total of six layers. We will referrer to this as a model having hybrid self-attention layers.

2. Using six memory-efficient-only self-attention layers. This approach will be referred to as efficient self-attention layers.

For these two approaches, we compared results using both 8k and 32k vocabulary sizes.

Results in Table 2 shows that:

- In almost all cases, it did not make a notable difference in accuracy to use either hybrid or efficient self-attention layers.
- In terms of training time, the impact was negligible, and on average efficient self-attention layers trained slightly faster.

In conclusion it is more beneficial to use the efficient version if memory is a concern since it will not impact performance or training time.

4.3. Number of Attention Head

In Table 3, we analyze the performance of the Transformers and Reformers when changing the number of attention heads. As shown, the number of attention heads tested are 6, 16 and 32.

These results show that increasing the number of attention heads improved the performance of these classifiers because it allows for better capturing the underlying context of the text.

Table 2. Results of experiments with Reformers (RF) to compare the effect of attention type on performance.

Dataset	Model	Vocab	D_Model	Encloder layers	Number of heads	Max length	Attention type	Training time	Max accuracy	Min CE
IMDB	Reformer	8k	512	6	6	512	hybrid	3168	87.27	0.33
	Reformer	8k	512	6	6	512	efficient	3024	**87.99**	**0.32**
	Reformer	32k	512	6	6	512	hybrid	3211	**88.44**	**0.30**
	Reformer	32k	512	6	6	512	efficient	3150	85.94	0.33
Yelp	Reformer	8k	512	6	6	512	hybrid	2997	91.02	0.21
	Reformer	8k	512	6	6	512	efficient	2917	**92.32**	0.17
	Reformer	32k	512	6	6	512	hybrid	3057	**93.83**	0.16
	Reformer	32k	512	6	6	512	efficient	3147	93.44	0.16
Allocine	Reformer	32k	512	6	6	512	hybrid	3048	93.3	0.18
	Reformer	32k	512	6	6	512	efficient	3090	**93.67**	0.18
AGNews	Reformer	8k	512	6	6	512	hybrid	2640	90.78	0.24
	Reformer	8k	512	6	6	512	efficient	2552	**90.94**	0.25
	Reformer	32k	512	6	6	512	hybrid	2593	**92.97**	0.22
	Reformer	32k	512	6	6	512	efficient	2657	92.73	0.21

Table 3. Results of experiments with Transformers (TR) and Reformers (RF) to compare the effect of the number of attention heads on performance.

Dataset	Model	Vocab	D_Model	Encloder layers	Number of heads	Max length	Attention type	Training time	Max accuracy	Min CE
IMDB	Reformer	8k	512	6	6	512	hybrid	3168	87.27	0.3323
	Reformer	8k	512	6	16	512	hybrid	5400	87.27	0.3232
	Reformer	8k	512	6	32	512	hybrid	6120	**88.44**	0.29
	Transformer	8k	512	6	6	512		2324.8	87.42	0.31
	Transformer	8k	512	6	16	512		2647	85.50	0.33
	Transformer	8k	512	6	32	512		2476	**89.06**	0.27
Yelp	Reformer	8k	512	6	6	512	hybrid	2997.6	91.02	0.21
	Reformer	8k	512	6	16	512	hybrid	4500	91.8	0.23
	Reformer	8k	512	6	32	512	hybrid	3385	**92.50**	0.19
	Transformer	8k	512	6	6	512		2425	**92.42**	0.19
	Transformer	8k	512	6	16	512		2627	90.0	0.23
	Transformer	8k	512	6	32	512		2388	92.27	0.20
Allocine	Reformer	8k	512	6	6	512	hybrid	3048	**93.3**	0.18
	Reformer	8k	512	6	16	512	hybrid	4440	92.11	0.20
	Reformer	8k	512	6	32	512	hybrid	3152	92.58	0.19
	Transformer	8k	512	6	6	512		2333	92.66	0.19
	Transformer	8k	512	6	16	512		2591	**95.08**	0.17
	Transformer	8k	512	6	32	512		1796	93.12	0.16
AGNews	Reformer	8k	512	6	6	512	hybrid	2640	90.78	0.24
	Reformer	8k	512	6	16	512	hybrid	3601	90.63	0.26
	Reformer	8k	512	6	32	512	hybrid	2322	**91.33**	0.28
	Transformer	8k	512	6	6	512		1860	90.94	0.25
	Transformer	8k	512	6	16	512		1911	90.94	0.26
	Transformer	8k	512	6	32	512		1469	**91.25**	0.24

4.4. Effect of Vocabulary Size

We are interested in analyzing the effect of pretrained vocabulary size on classification performance. All vocabularies used are subword implementations. 8k.subword and 32k.subword are both built using Tensor2Tensor's Subword Text Encoder class. Subword Text Encoder from Tensor2Tensor works similarly to a Byte Pair Encoding (BPE) tokenizer, but on top of splitting to subwords, it also does tokenization of spaces and punctuation. In other words, unlike BPE, it also

encodes the spaces – or absence of spaces – so that the raw sequence is fully reproducible. This is not the case with BPE tokenizer.

Results of experiments in Table 4 shows that:

- The 32k subword vocabulary does not consistently provide better results than the 8k subword vocabulary for all the datasets.
- Training time is usually slightly higher for 32k subword but not significantly.

Table 4. Results of experiments with Transformers and Reformers to compare the effect of vocab on performance.

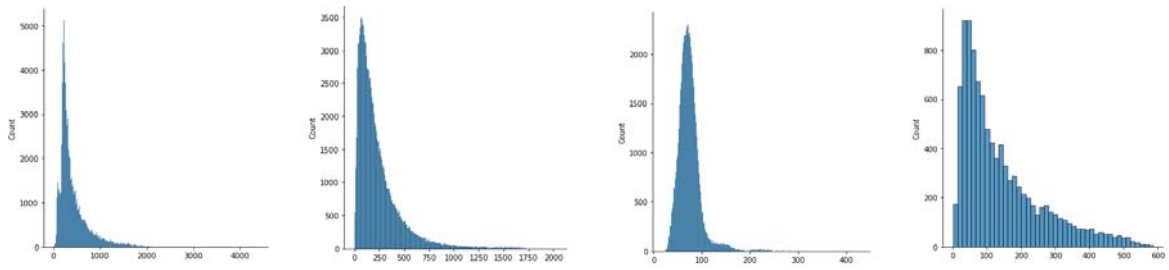
Dataset	Model	Vocab	D_Model	Encloder layers	Number of heads	Max length	Attention type	Training time	Max accuracy	Min CE
IMDB	Reformer	8k	512	6	6	512	hybrid	3168	**87.27**	0.33
	Reformer	32k	512	6	6	512	hybrid	6048	83.75	0.33
	Reformer	8k	512	6	6	512	hybrid	3211	88.44	0.30
	Reformer	32k	512	6	6	512	hybrid	10260	**89.14**	0.30
	Transformer	8k	512	6	6	512		2325	87.42	0.31
	Transformer	32k	512	6	6	512		2351	**87.94**	0.27
	Transformer	8k	512	6	6	512		2451	**89.69**	0.27
	Transformer	32k	512	6	6	512		3182	88.44	0.31
Yelp	Reformer	8k	512	6	6	512	hybrid	2998	**0.91**	0.21
	Reformer	32k	512	6	6	512	hybrid	5123	0.90	0.23
	Reformer	8k	512	6	6	512	hybrid	3057	**93.83**	0.16
	Reformer	32k	512	6	6	512	hybrid	10740	91.64	0.18
	Transformer	8k	512	6	6	512		2425	92.42	0.19
	Transformer	32k	512	6	6	512		2265	**92.66**	0.23
	Transformer	8k	512	6	6	512		2373	**93.98**	0.16
	Transformer	32k	512	6	6	512		2350	93.13	0.18
AGNews	Reformer	8k	512	6	6	512	hybrid	2640	**90.78**	0.24
	Reformer	32k	512	6	6	512	hybrid	8700	89.50	0.30
	Reformer	8k	512	6	6	512	hybrid	2657	**92.73**	0.21
	Reformer	32k	512	6	6	512	hybrid	9180	91.56	0.24
	Transformer	8k	512	6	6	512		1860	90.94	0.25
	Transformer	32k	512	6	6	512		1921	**91.72**	0.26
	Transformer	8k	512	6	6	512		1981	92.5	0.23
	Transformer	32k	512	6	6	512		1920	**92.97**	0.23

4.5. Effect of Maximum Length of the Sequences

Since the Reformer model is more memory efficient, we wanted to test whether increasing the maximum length of the sequences that will be read has an impact on performance.

We experimented with both shorter and longer sequence lengths (256, 512, 1024) and analyzed results comparing the Reformer encoder model with the Transformer encoder.

We also compared the maximum length hyperparameter with the average length of the sequences for each dataset to see if there is a correlation between the two. Table 5 shows the results and Fig. 6 shows the histogram of maximum length of the sequences in all datasets used in this paper. The average sequence length for IMDB dataset is 416 and the results show that a maximum length of 512 clearly performed better than lengths 256 and 1024 for both the Reformer and the Transformer model. The average sequence length for Yelp is 230. For Yelp dataset there was no clear pattern since sequence length of 1024 performed better for the Reformer model, and sequence length 512 performed the best for the Transformer model. The average sequence length for Allocine is 141. In this case, 256 also performed better on average than lengths 512 or 1024. The average sequence length for AGnews is 78. In this case, a maximum length of 256 performed better than 512 or 1024.



(a) IMDB (b) Yelp (c) Allocine (d) AGNews

Fig. 6. Sequence length distribution of the text samples in datasets.

Table 5. Results of experiments with Transformers and Reformers to compare the effect of max length on performance.

Dataset	Model	Vocab	D_Model	Encloder layers	Number of heads	Max length	Attention type	Training time	Max accuracy	Min CE
IMDB	Reformer	8k	512	6	6	256	hybrid	1633	83.20	0.44
	Reformer	8k	512	6	6	512	hybrid	3168	**87.27**	0.33
	Reformer	8k	512	6	6	1024	hybrid	2520	84.34	0.38
	Transformer	8k	512	6	6	256		1139	83.83	0.47
	Transformer	8k	512	6	6	512		2325	**87.42**	0.31
	Transformer	8k	512	6	6	1024		1918	79.69	0.46
Yelp	Reformer	8k	512	6	6	256	hybrid	1769	91.09	0.18
	Reformer	8k	512	6	6	512	hybrid	2998	91.02	0.21
	Reformer	8k	512	6	6	1024	hybrid	2777	**92.42**	0.23
	Transformer	8k	512	6	6	256		1297	91.95	0.22
	Transformer	8k	512	6	6	512		2425	**92.42**	0.19
	Transformer	8k	512	6	6	1024		2134	91.33	0.22
Allocine	Reformer	32k	512	6	6	256	hybrid	1888	**93.23**	0.16
	Reformer	32k	512	6	6	512	hybrid	3048	93.2	0.18
	Reformer	32k	512	6	6	1024	hybrid	2929	92.63	0.19
	Transformer	32k	512	6	6	256		1557	**93.05**	0.18
	Transformer	32k	512	6	6	512		2333	92.66	0.19
	Transformer	32k	512	6	6	1024		2312	91.51	0.20
AGNews	Reformer	8k	512	6	6	256	hybrid	2474	**91.71**	0.25
	Reformer	8k	512	6	6	512	hybrid	2640	90.78	0.24
	Reformer	8k	512	6	6	1024	hybrid	2592	91.41	0.24
	Transformer	8k	512	6	6	256		1921	**92.11**	0.26
	Transformer	8k	512	6	6	512		1860	90.94	0.25
	Transformer	8k	512	6	6	1024		1887	90.78	0.27

4.6. Effect of the Model Dimension

We experiment with 256, 512 and 1024 as the number of features so that each representation of subwords of our vocabulary contains more information. Based on the results in Table 6, depending on the dataset, and the model used, a dimension of 512 or 1024 will yield a better performance and the dimension of size 256 perform worse on average.

Table 6. Results of experiments with Transformers and Reformers to compare the effect of model dimension on performance.

Dataset	Model	Vocab	D_Model	Encloder layers	Number of heads	Max length	Attention type	Training time	Max accuracy	Min CE
IMDB	Reformer	8k	512	6	6	512	hybrid	3168	87.27	0.33
	Reformer	8k	1024	6	6	512	hybrid	6048	83.75	0.33
	Reformer	32k	512	6	6	512	hybrid	3211	88.44	0.30
	Reformer	32k	1024	6	6	512	hybrid	10260	89.14	0.30
	Transformer	8k	512	6	6	512		2325	87.42	0.31
	Transformer	8k	1024	6	6	512		2351	87.94	0.27
	Transformer	32k	512	6	6	512		2451	89.69	0.27
	Transformer	32k	1024	6	6	512		3182	88.44	0.31
Yelp	Reformer	8k	512	6	6	512	hybrid	2998	0.91	0.21
	Reformer	8k	1024	6	6	512	hybrid	5123	90.23	0.23
	Reformer	32k	512	6	6	512	hybrid	3057	93.83	0.16
	Reformer	32k	1024	6	6	512	hybrid	10740	91.64	0.18
	Transformer	8k	512	6	6	512		2425	92.42	0.19
	Transformer	8k	1024	6	6	512		2266	92.66	0.20
	Transformer	32k	512	6	6	512		2373	93.98	0.16
	Transformer	32k	1024	6	6	512		2350	93.13	0.18
Allocine	Reformer	8k	512	6	6	512	hybrid	3048	93.3	0.18
	Reformer	8k	1024	6	6	512	hybrid	7740	93.2	0.18
	Transformer	32k	512	6	6	512		2333	92.66	0.19
	Transformer	32k	1024	6	6	512		3097	92.42	0.177
AGNews	Reformer	8k	512	6	6	512	hybrid	2640	90.78	0.24
	Reformer	8k	1024	6	6	512	hybrid	8700	89.50	0.30
	Reformer	32k	512	6	6	512	hybrid	2657	92.73	0.21
	Reformer	32k	1024	6	6	512	hybrid	9180	91.56	0.24
	Transformer	8k	512	6	6	512		1860	90.94	0.25
	Transformer	8k	1024	6	6	512		1921	91.72	0.26
	Transformer	32k	512	6	6	512		1981	92.5	0.23
	Transformer	32k	1024	6	6	512		1920	92.97	0.23

4.7. Best Performance of Transformers and Reformers

In Table 7, we present the top 3 performance of Transformers and Reformers among all the experiments that were done under different hyper-parameter settings. In the case of IMDB, Yelp, and Allocine datasets, Transformers show the best performance in terms of accuracy. For AGNews however, the best performance is achieved by both Transformers and Reformers in terms of accuracy and in this case, the Reformer takes less time to train.

Table 7. Results of experiments with Transformers and Reformers to compare their top three performance.

Dataset	Model	Vocab	D_Model	Encloder layers	Number of heads	Max length	Attention type	Training time	Max accuracy	Min CE
IMDB	Reformer	8k	512	6	32	512	hybrid	6120	88.44	0.29
	Reformer	32k	512	6	6	512	hybrid	3211	88.44	0.30
	Reformer	8k	512	6	12	512	hybrid	10260	89.14	0.30
	Transformer	32k	1024	6	12	512		3182	88.44	0.31
	Transformer	8k	512	6	6	512		2476	89.06	0.27
	Transformer	32k	512	6	6	512		2451	**89.69**	0.27
Yelp	Reformer	32k	512	6	6	512	hybrid	2999	93.2	0.17
	Reformer	32k	512	6	6	512	efficient	3147	93.44	0.16
	Reformer	32k	512	6	6	512	hybrid	3057	93.83	0.16
	Transformer	32k	1024	6	12	512		2365	93.13	0.19
	Transformer	32k	512	6	6	512		2373	93.98	0.16
	Transformer	8k	512	8	6	512		3172	**95.16**	0.12
Allocine	Reformer	32k	512	6	6	256	hybrid	1888	93.23	0.16
	Reformer	32k	512	6	6	512	hybrid	3048	93.3	0.18
	Reformer	32k	512	6	6	256	efficient	3090	93.67	0.18
	Transformer	32k	512	6	6	256		1557	93.05	0.18
	Transformer	8k	512	6	32	512		1796	93.12	0.16
	Transformer	8k	512	6	16	512		2591	**95.08**	0.17
AGNews	Reformer	8k	512	6	6	256	hybrid	2474	91.71	0.25
	Reformer	32k	512	6	6	512	hybrid	2657	92.73	0.21
	Reformer	32k	512	6	6	512	efficient	2593	**92.97**	0.22
	Transformer	32k	512	6	6	512		1981	92.5	0.23
	Transformer	32k	512	6	6	512		2450	92.89	0.21
	Transformer	32k	512	8	6	1024		**1920**	**92.97**	0.23

5. Conclusions

In this paper, we compared Transformers and Reformers for text classification application. The results showed that increasing the number of attention heads improved the performance of both Transformers and Reformers because it allows for better capturing the underlying context of the text. Also, we observed that higher number of features results in better performance and setting the maximum length of sequences closer to the average length of sequence in the dataset results in better performance. In literature, Reformers have shown to outperform Transformers in terms of memory and time efficiency at the similar level of accuracy on language model training and text generative tasks. However, the experiments in this paper showed that Transformers might be preferred to Reformers for Text classification application with short text sequences and smaller models. The benefit of Reformers versus Transformers was observed to be the possibility of training larger models. Therefore, Reformers may still be a better option for training our language model on big corpus Open Data and do transfer-learning for the applications in hand. Benchmarking text classification using the language models based on Reformers and Transformers for the text classification is the future work related to this article.

Acknowledgements

This work was supported by the National Research Council through their Industrial Research Assistance Program (IRAP) and abide by their experimental research requirements.

References

[1]. Deep text analytics. (https://www.poolparty.biz/what-is-deep-text-analytics/)

[2]. S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep learning based text classification: A comprehensive review, *arXiv*:2004.03705, 2020.

[3]. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in *Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS' 2017),* 2017, pp. 5998–6008.

[4]. N. Kitaev, L. Kaiser, A. Levskaya, Reformer: The efficient transformer, in *Proceedings of the*

International Conference on Learning Representations, 2020.

[5]. R. Soleymani, J. Farret, Text Classification with Transformers and Reformers for Deep Text Data, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI' 2020)*, Berlin, Germany, November 2020, pp. 239-243.

[6]. Trax, an end-to-end library for deep learning by Google Brain team. (https://github.com/google/trax).

[7]. A. N. Gomez, M. Ren, R. Urtasun, R. B. Grosse, The reversible residual network: Backpropagation without storing activations, in *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA*, 2017, pp. 2214-2224.

[8]. N. Shazeer, M. Stern, Adafactor: Adaptive learning rates with sublinear memory cost, *arXiv*:1804.04235, 2018.

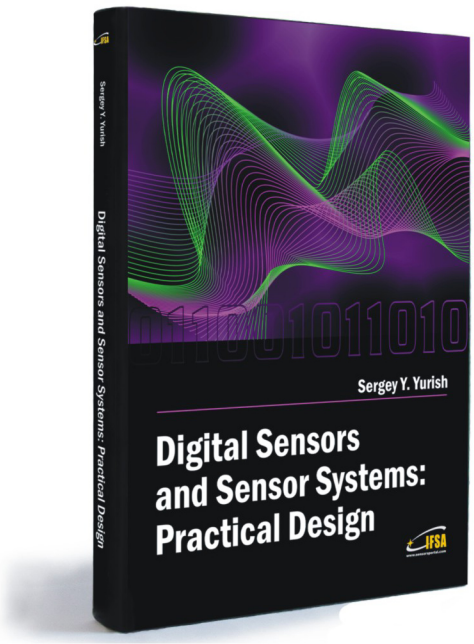
Theory:

Digital Sensors and Sensor Systems: Practical Design

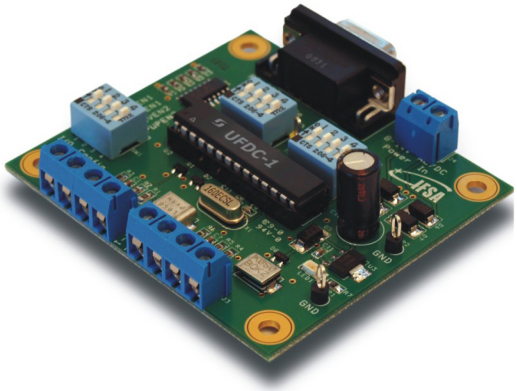
and

Practice:

Development Board EVAL UFDC-1/UFDC-1M-16



+



Buy book and Evaluation board together. **Save 30.00 EUR**

Development Board EVAL UFDC-1 / UFDC-1M-16

Full-featured development kit for the Universal Frequency-to-Digital Converters UFDC-1 and UFDC-1M-16. 2 channel, 16 measuring modes, high metrological performance, RS232/USB interface, master and slave communication modes. On-board frequency reference (quartz crystal oscillator). Operation from 8 to 14 V AC/DC. Development board software is included.

All existing frequency, period, duty-cycle, time interval, pulse-width modulated, pulse number and phase-shift output sensors and transducers can be directly connected to this 2-channel DAQ system. The user can connect TTL-compatible sensors' outputs to the Development Board, measure any output frequency-time parameters, and test out the sensor systems functions.

Applications:

- Digital sensors and sensor systems
- Smart sensors systems
- Data Acquisition for frequency-time parameters of electric signals
- Frequency counters
- Tachometers and tachometric systems
- Virtual instruments
- Educational process in sensors and measurements
- Remote laboratories and distance education

Order online:
http://www.sensorsportal.com/HTML/BOOKSTORE/Digital_Sensors_and_Board.htm

IFSA

International Frequency Sensor Association Publishing

www.sensorsportal.com

ISSN 1726- 5479