

Image Description Generator using Residual Neural Network and Long Short-Term Memory

Mahesh Kumar Morampudi, Nagamani Gonthina,
Nuthanakanti Bhaskar, V. Dinesh Reddy

Abstract

Human beings can describe scenarios and objects in a picture through vision easily whereas performing the same task with a computer is a complicated one. Generating captions for the objects of an image helps everyone to understand the scenario of the image in a better way. Instinctively describing the content of an image requires the apprehension of computer vision as well as natural language processing. This task has gained huge popularity in the field of technology and there is a lot of research work being carried out. Recent works have been successful in identifying objects in the image but are facing many challenges in generating captions to the given image accurately by understanding the scenario. To address this challenge, we propose a model to generate the caption for an image. Residual Neural Network (ResNet) is used to extract the features from an image. These features are converted into a vector of size 2048. The caption generation for the image is obtained with Long Short-Term Memory (LSTM). The proposed model is experimented on the Flickr8K dataset and obtained an accuracy of 88.4%. The experimental results indicate that our model produces appropriate captions compared to the state of art models.

Keywords: Image, description generator, ResNet, LSTM.

MSC 2020: 68T45, 68T50, 68U15.

ACM CCS Codes: Image Processing and Natural Language Processing → Scene Recognition and Text generation.

1 Introduction

Human beings possess a basic talent of describing an image clearly and elaborately by just watching it for a few seconds. Steady research is being conducted in the areas of machine learning and artificial intelligence in building a machine to mimic the capabilities of humans. In the past, extensive research progress was conducted in the areas such as the diagnosis of objects in a prescribed image, feature categorization, image grading, and categorization of activities by humans [1]–[4]. Detection of the image as well as producing the caption with natural language processing (known as an image caption generator system) by the computer is a tedious job, image caption generator system can be used as a solution to various problems such as self-driving cars, aid to the blind, etc. The captions or descriptions for an image are generated from an inverse dictionary that is formed during the training of the model. Automatic image description generation is useful in various fields like picture cataloging, blind persons, social media, and various natural language processing applications.

Generating the description for a picture involves multiple jobs like analyzing the importance in usage of semantics, and framing the meanings in a phrase through which humans can understand. To analyze the usage of semantics, the machine must grasp the relations amongst the things within an image. Generally, presentation in humans happens using natural language, hence building a computer system that generates captions that are acceptable to humans is an exciting task. We have multiple ways to build descriptions, like recognizing visual depiction of objects, setting up relations between the objects, and creating descriptions that are both grammatically and meaningfully perfect. In recent times, there is an immense growth in the availability of digital information on the Internet. One such application is Flickr, a means utilized for exporting, assembling, and distributing computerized information like pictures, audio, and video which can host more than 7 billion images, and this number is going to increase exponentially in the coming years. This application helps to find the images for training and testing a model with ease and helps the research community to describe an image. The image description is a simple process of

allocating words or phrases, which forms meaning to an image and describes the image. In earlier times, picture caption generation techniques agglomerated image knowledge with static object class libraries in the picture and were designed with the help of statistical language models. Few unintended techniques are even proposed to deal with picture caption descriptor issues, like the query evolution process suggested by Yagcioglu et al. [5], through fetching related pictures within a huge dataset and employing the allocation represented with a correlation of the fetched images. A huge amount of research done was prone to the issue of ranking descriptions for a given picture [3], [4], [6]. These methodologies depend on the possibility of co-embedding pictures and text data into a related vector space. Neural networks are utilized to co-embed pictures and text together, picture selections, and sub-sentences [7], [8], yet didn't strive to create narrative captions. Detections and segments aggregated with an end caption using phrases holding distinguished objects and relationships are used to deduce a triad of scene components that are translated to text with the help of templates [9], [10]. A further complicated graph of recognition behind triads is shown by Kulkarni et al. [10]. Deep learning architecture was used for image caption generation in [11]. In [12], [13], the authors used Convolutional Neural Network (CNN) in combination with Long Short-Term Memory (LSTM) to produce image descriptions but fail to achieve promising accuracy.

The proposed work aims at creating relevant, fluent captions like humans do for the given images without depending on any object identifiers, classifiers, transcribed regulations, or heuristics. We used Flickr8K dataset to find the efficiency of the proposed model.

1.1 Motivation

The aim and motivation of the proposed work are the latest advancements in machine conversion, where the job is converting a sentence "Z" taken in an original language, to "X" of the destination language, by enhancing the conditional probability $P(Z|X)$. Machine translation was even attained through a set of different jobs like converting words separately, joining words, rearranging, and so on from the past few

years. The latest research shows that conversion can be achieved efficiently with Residual Neural Network (ResNet) and can achieve better performance when compared to the state of art approaches [14].

1.2 Contributions

- We propose a model to generate the captions for an image using ResNet and LSTM [15].
- A generative approach is presented in this article that issues textual data, instead of using computer vision techniques (cv). Rather than using object identifiers, the data in a fresh image is estimated depending on the image's similarity to accessible images in the dataset, and the description of the image is output.
- Our approach creates relevant, fluent captions, like humans do for the given images not depending on any object identifiers, classifiers, transcribed regulations, or heuristics.
- This generative approach has experimented on the publicly available Flickr8K dataset. Better results are achieved compared to the conventional model by investigating and intelligently extracting the semantic knowledge encoded in the image descriptions.

2 Related Work

The task of creating captions similar to natural language, derived from visual data has been studied in computer vision mostly in video applications [16], [17]. It produces complicated machines containing visual primitive recognizers mixed with a structured formal language, e.g., And-Or Graphs or logic systems, that were additionally changed to natural language through rule-based frameworks. These things were mostly handcrafted, comparatively sensitive, and are shown uniquely in restricted spaces, e.g., sports or traffic scenes.

A vast amount of work was prone to the issue of ranking descriptions for a given picture [3], [4], [6]. Such methodologies depend on the possibility of co-embedding images and text in a similar vector space.

For a picture query, captions that fall close to the image in the embedding space are retrieved. Mostly, neural networks are utilized to co-embed images and sentences together [8], or even combine cropped pictures and sub-sentences [7] together. They didn't strive to create narrative captions. The previously mentioned methods could not represent earlier hidden compositions of objects, even though the individual objects might have been seen in the training data. Besides this, they avoid conveying the issue of assessing how well the created caption is.

Most recently the issue of picture caption generation using natural text has acquired attention. Utilizing the improvements in object recognition, their features, and positions, permits us to take forward natural language generation systems, even though they are restricted in their phrasing. Farhadi et al. [8] proposed a method to deduce a triad of scene components that are translated to text using the templates. Li et al. [9] introduced a method to provide an end caption with phrases consisting of distinguished objects and relationships. An even more complicated graph of recognition behind triads is proposed by Kulkarni et al. [10] using template-based text generation. In Vinyals et al. [12], the generative model is trained in such a way that, given the training image the likelihood of the final description of sentence is maximized. The methods discussed so far are capable of narrating images "in the wild", but they are mostly handcrafted and fixed in the case of text generation. Lebrecht et al. [18] proposed a simple language model based on caption syntax statistics to produce appropriate captions for an identified test image with the phrases deduced. N. K. Kumar et al. [11] proposed Regional Object Detector (RODe) to detect, recognize, and generate descriptions that aim at deep learning to still enhance on top of the prevailing image description generator systems.

Kinghorn et al. [19] proposed a region-based deep learning architecture in image caption generation by using a regional object detector, recurrent neural network (RNN)-based attribute prediction, and an encoder-decoder language generator embedded with two RNNs to generate processed and thorough captions for an identified image. Z. Zhou et al. [20] proposed a deep hierarchical framework to recognize images and a syntactic tree-based model to generate the natural language respectively. To support on-line image search, these two models

are described to evenly draw out the features of human beings and various object classes to generate well-proportioned sentences narrating the exact actions in the image. Bo Dai et al. [21] framework is dependent on Conditional Generative Adversarial Networks (CGAN), which mutually analyzes a generator to generate captions constrained on images and an evaluator to examine the fitness of caption in the visual content. Y. H. Tan et al. [13] proposed phi-LSTM, which decodes image captions from phrase to sentence. It contains a phrase decoder that decodes noun phrases of irregular size, and an abbreviated sentence decoder to decode the abbreviated form of the image description. An absolute image description is generated by combining the generated phrases with sentences in the course of the conclusion stage.

The literature reveals that the techniques designed to generate the captions for an image uses convolutional neural networks; the other deep learning models fail to generate the appropriate description of the scenario in the given image. Therefore, we proposed a model using ResNet and LSTM to provide a description of the scenario in the given image efficiently and accurately.

3 Proposed System

We proposed an approach to create relevant, fluent captions like humans do for the given images, without depending on any object identifiers, classifiers, transcribed regulations, or heuristics. The proposed system consists of two phases: the training and testing phase. Pre-processing, data preparation, and creation of a model are the different operations involved in the training phase of the proposed system. During the testing phase, the image vector is generated from the image, and a description of the image is displayed as an output using the generated model. Figure 1 depicts the generic architecture of the proposed model.

3.1 Preprocessing

This phase consists of two tasks, 1) Preprocessing images 2) Preprocessing captions.

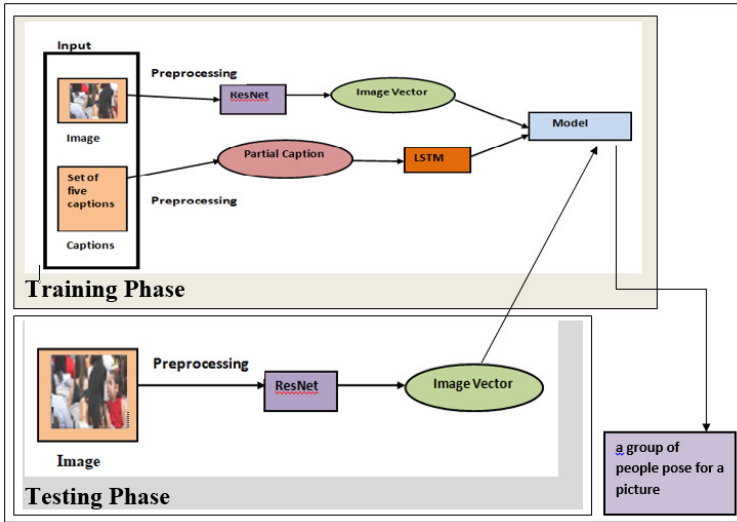


Figure 1. Block diagram of the proposed architecture

3.1.1 Preprocessing images

Image preprocessing is the process considered to set up images before training the model and inference. Images are given as input to the model in the form of an Image vector. Every image needs to be converted to a constant length vector that can be given as input to the neural network. To achieve the aforesaid, ResNet (Convolutional Neural Network) is used [14]. We created a dictionary named “image features”, where the key will be the name of the image and the value will be output from the ResNet model. CV2 reads the image in BGR format, so the image is converted into RGB and resized to 224×224 . It can be transferred into ResNet and reshaped to 2048 vector values. Retrieving only the name of the image from the whole path of the image is achieved using the slice function.

3.1.2 Preprocessing captions

During this step, each word is assigned an index. As there are 1652 unique words present in the database, each word is assigned a number from 1 to 1652. We also compute the maximum length of the caption.

The model will predict the caption for an image at the end. Given an image, it is difficult to predict the entire caption at once. So, we will predict Word by Word. In this regard, each word is encoded into a fixed-sized vector.

3.2 Data Preparation

Machine Learning or Deep Learning models cannot directly take the text and fit it into a model. Initially, the text is to be cleaned by dividing it into words, handling punctuation and case sensitivity problems. As English words can't be understood by the computers directly, they should be represented with numbers. Every word of the vocabulary should be matched to a unique index value, and each word is to be encoded into a fixed-length vector. There after every word is represented as a number. The text represented in binary numbers only can be readable by the machine and captions for the image can be generated. When the captions are run, the output will be images and the number of captions for a single image along with it. To achieve this, a dictionary is created with key as the image and all the other 5 captions of that image as the value. For the 1500 images available in the dataset, we need to check whether all the 1500 images append their captions. If the image name is available in the image feature and not in the captions dictionary, then we set the image name as a key and append those captions. Now we add *startofseq* and *endofseq* to the tokens. The prediction of the next word for a given image and partial description for text and numbers is shown in Figure 2 and Figure 3.

Create a dictionary Vocabulary that contains all the words in the captions is created with count=1 and checked word by word using the split function. If there is no count in *count_words*, we set that word as a key and count as an integer value, and the value of the count is incremented by 1. $\text{len}(\text{count_words})$ is obtained as 40461. The words are converted into integer values since the neural network can only work with integer values. Previously there was: key – image name and values – captions; but now: key – integer and value – integer. Three variables, where the first one holds the image features, the second variable holds previous words, and the last variable holds

X_i			Y_i
j	Image Feature Vector	Partial Description	Target word
1	Image_1	startofseq	a
2	Image_1	Startofseq a	group
3	Image_1	Startofseq a group	of
4	Image_1	Startofseq a group of	people
5	Image_1	Startofseq a group of people	pose
6	Image_1	Startofseq a group of people pose	for
7	Image_1	Startofseq a group of people pose for	a
8	Image_1	Startofseq a group of people pose for a	picture
9	Image_1	Startofseq a group of people pose for a picture	endofseq
10	Image_2	startofseq	cyclists
11	Image_2	Startofseq cyclists	in
12	Image_2	Startofseq cyclists in	a
13	Image_2	Startofseq cyclists in a	race
14	Image_2	Startofseq cyclists in a race	of
15	Image_2	Startofseq cyclists in a race of	bike
16	Image_2	Startofseq cyclists in a race of bike	is
17	Image_2	Startofseq cyclists in a race of bike is	riding
18	Image_2	Startofseq cyclists in a race of bike is riding	the
19	Image_2	Startofseq cyclists in a race of bike is riding the	busy
20	Image_2	Startofseq cyclists in a race of bike is riding the busy	street
21	Image_2	Startofseq cyclists in a race of bike is riding the busy street	endofseq

Figure 2. Data matrix for both images and captions

the next word to be predicted, are considered. Now we import the packages *to_categorical* and *pad_sequences* from *keras*. The value of *MAX_LEN* is 36. The *VOCAB_SIZE* will be the length of the count of words. We append image features to the *x* variable, *y_in* for input, and *y_out* for predicting the next word. *Pad_sequence* is used to convert the variable length to *MAX_LEN*. Figure 4 shows how the zeros are appended to each sequence to make them same length of 36. The *to_categorical* converts the out sequence into vocab size. It appends 0's and 1's, where 0 means the least probability, and 1 means maximum probability. When we check the length of all these variables, it will be 96528. So, to achieve faster execution, all the variables are converted to NumPy arrays.

X_i			Y_i
j	Image Feature Vector	Partial Description	Target word
1	Image_1	[9]	10
2	Image_1	[9,10]	1
3	Image_1	[9,10,1]	2
4	Image_1	[9,10,1,2]	8
5	Image_1	[9,10,1,2,8]	6
6	Image_1	[9,10,1,2,8,6]	4
7	Image_1	[9,10,1,2,8,6,4]	7
8	Image_1	[9,10,1,2,8,6,4,7]	5
9	Image_1	[9,10,1,2,8,6,4,7,5]	3
10	Image_2	[9]	4
11	Image_2	[9,4]	7
12	Image_2	[9,4,7]	10
13	Image_2	[9,4,7,10]	5
14	Image_2	[9,4,7,10,5]	2
15	Image_2	[9,4,7,10,5,2]	6
16	Image_2	[9,4,7,10,5,2,6]	1
17	Image_2	[9,4,7,10,5,2,6,1]	11
18	Image_2	[9,4,7,10,5,2,6,1,11]	8
19	Image_2	[9,4,7,10,5,2,6,1,11,8]	13
20	Image_2	[9,4,7,10,5,2,6,1,11,8,13]	12
21	Image_2	[9,4,7,10,5,2,6,1,11,8,13,12]	3

Figure 3. Data matrix after replacing the words with their indexes

3.3 Creating a model

The architecture of the model is shown in Figure 5. The left layer represents captions, the right layer represents images as input, the center represents the LSTM model which is a concatenation of both values, and the bottom layer is the dense layer. Figure 5 consists of three parts: 1) Feature extractor, 2) Sequence Processor, and 3) Decoder. Feature extractor helps to extract the features from the image. The feature vector generated from this phase is of size 1×2048 . The second part generates the image caption from the extracted features by using LSTM model. LSTM helps to carry out the relevant information and to discard non-relevant information. The last part decodes the output by concatenating the above two layers. It has 4074 probabilities for each vocabulary.

j	X_i		Y_i
	Image Feature Vector	Partial Description	Target word
1	Image_1	[9,0,0,....,0]	10
2	Image_1	[9,10,0,0,....,0]	1
3	Image_1	[9,10,1,0,0,....,0]	2
4	Image_1	[9,10,1,2,0,0,....,0]	8
5	Image_1	[9,10,1,2,8,0,0,....,0]	6
6	Image_1	[9,10,1,2,8,6,0,0,....,0]	4
7	Image_1	[9,10,1,2,8,6,4,0,0,....,0]	7
8	Image_1	[9,10,1,2,8,6,4,7,0,0,....,0]	5
9	Image_1	[9,10,1,2,8,6,4,7,5,0,0,....,0]	3
10	Image_2	[9,0,0,....,0]	4
11	Image_2	[9,4,0,0,....,0]	7
12	Image_2	[9,4,7,0,0,....,0]	10
13	Image_2	[9,4,7,10,0,0,....,0]	5
14	Image_2	[9,4,7,10,5,0,0,....,0]	2
15	Image_2	[9,4,7,10,5,2,0,0,....,0]	6
16	Image_2	[9,4,7,10,5,2,6,0,0,....,0]	1
17	Image_2	[9,4,7,10,5,2,6,1,0,0,....,0]	11
18	Image_2	[9,4,7,10,5,2,6,1,11,0,0,....,0]	8
19	Image_2	[9,4,7,10,5,2,6,1,11,8,0,0,....,0]	13
20	Image_2	[9,4,7,10,5,2,6,1,11,8,13,0,0,....,0]	12
21	Image_2	[9,4,7,10,5,2,6,1,11,8,13,12,0,0,....,0]	3

Figure 4. Appending zeros to each sequence to make them all of same length 36

Fitting the model Model fitting is a measure of the extent to which a machine learning model generates data similar to the one it was trained on. A model which is well-fitted generates the most appropriate results. We initialized the *batch_size* = 512 and *epochs* = 90, and trained the model until we got the maximum accuracy, keeping in mind the problem of overfitting. The value of the epochs should be in such a way that the model should not get under-fitted or overfitted.

4 Experimental Results

Dataset: Flickr8k database [3] is used to validate the efficiency of the proposed model. The database consists of 8000 images and 5 English captions for each image which is taken from the online photo-sharing

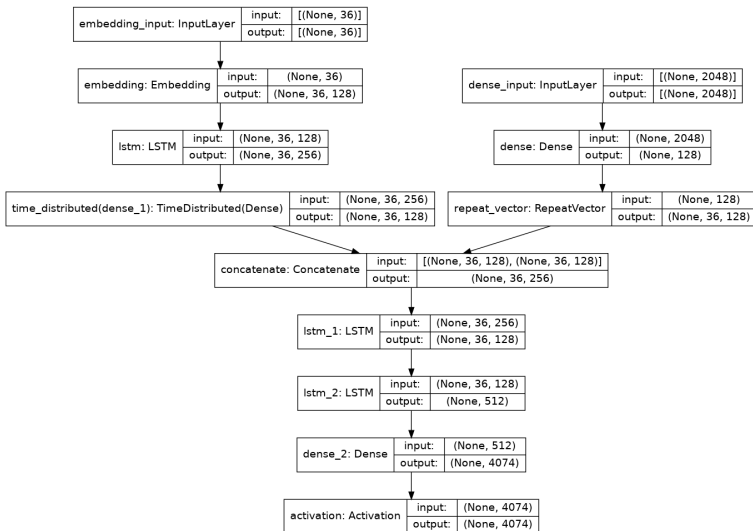


Figure 5. Architecture of the model

application Flickr.com. Out of which 6000 images are used for training and 1000 images are used for validation and testing. Annotators were asked to write sentences that describe the depicted scenes, situations, events, and entities (people, animals, other objects). Spoken captions for Flickr8k were collected by [22] having Amazon Mechanical Turk workers pronounce the originally written captions.

Training The first step in the process of generating comments to the image is to create a fixed-length vector that effectively summarizes the content of an image. We use CNN, in particular the ResNet50 architecture. This network is preliminarily trained for 1.2 million images of the ImageNet dataset. Therefore, ResNet50 has a reliable initialization for object recognition and allows reducing training time. For any image from the training set, we get the output vector representation of size 2048 from the last convolutional layer. This vector is fed to the LSTM input. During implementation, we considered 6000 images out of 8000 images for training.

Results: Figure 6 shows an example of how caption is generated using the proposed model. The first two images (Figure 6a and Figure 6b) and their captions are considered to train the model and the third image (Figure 6c) is used to test our model. It can be inferred from Figure 6 that the model builds the vocabulary using the captions of the train images. The model generates the caption for the test image using the vocabulary created during the training phase. The captions on some of the other images from the test dataset are shown in Figure 7a, Figure 7b, and Figure 7c. The accuracy achieved was 88.4% and the predictions made were almost correct.



(a)



(b)



(c)

Figure 6. An example: a) (Train image1) Caption: The black cat sat on the grass b) (Train image2) Caption: The white cat is walking on road c) (Test image) Caption: The black cat is walking on grass

The accuracy indicates that the proposed model is not an 100% best model and it also gives the captions wrongly in some scenarios. Figure 8 shows the images for which the model generates the captions wrongly. The color of the shirt got mixed with the color in the background in figure 8a. So, it generates the caption wrongly. The model classifies the famous tennis player Rafael Nadal as a woman in Figure 8b. This is due to his long hair. The caption generated in Figure 8c is grammatically incorrect.

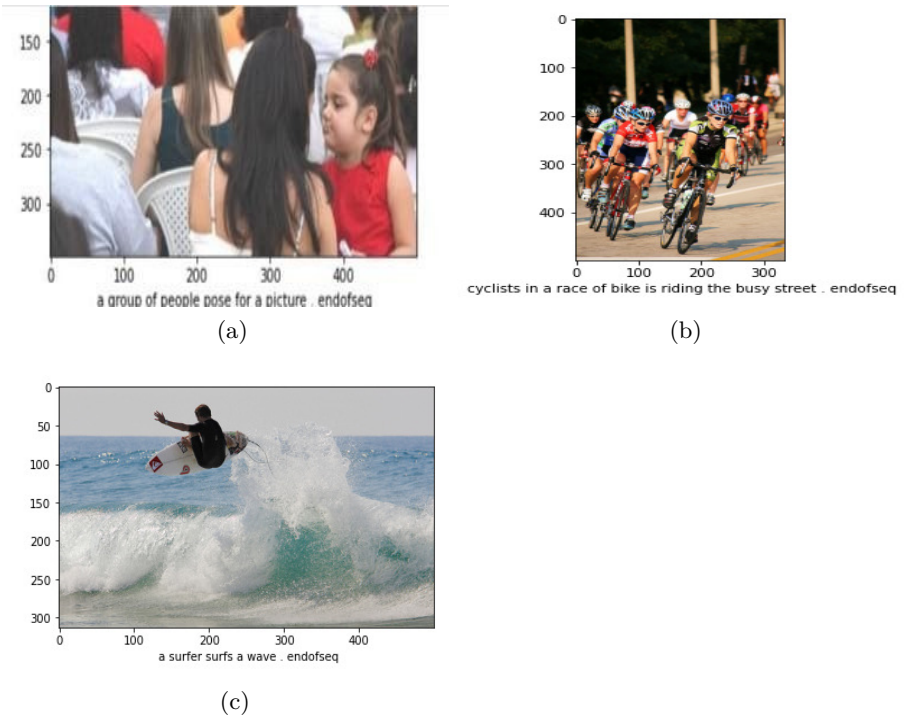


Figure 7. Some of the test images with their captions



(a)



(b)



(c)

Figure 8. The images for which the model generates the captions wrongly. a) man in black shirt is stakeboarding down ramp. b) a woman in tennis racket on the court. c) a boy is walking on the beach with ocean.

5 Conclusion and Future works

We introduced a system for creating relevant, fluent captions like humans do for the given images independent on any object identifiers, classifiers, transcribed regulations, or heuristics. Our model uses the ResNet to extract the features of an images and LSTM to provide the caption for an image. The proposed model is experimented on the publicly available database Flickr8K. The experimental results indicate that our model produces appropriate captions compared to the state-of-the-art methods.

Despite the fact that we have numerous enhancements in the area of image description generators, there is always a scope for development. Taking advantage of larger unsupervised data or weakly supervised methods is a challenge to explore in this area. Another major challenge could be generating summary or description for short videos. This work can also be extended to other sets of natural languages apart from English.

References

- [1] E. Kim, S. Helal, and D. Cook, “Human activity recognition and pattern discovery,” *IEEE pervasive computing*, vol. 9, no. 1, pp. 48–53, 2009.
- [2] R. Bernardi, R. Cakici, D. Elliott, A. Erdem, E. Erdem, N. Ikizler-Cinbis, F. Keller, A. Muscat, and B. Plank, “Automatic description generation from images: A survey of models, datasets, and evaluation measures,” *Journal of Artificial Intelligence Research*, vol. 55, pp. 409–442, 2016.
- [3] M. Hodosh, P. Young, and J. Hockenmaier, “Framing image description as a ranking task: Data, models and evaluation metrics,” *Journal of Artificial Intelligence Research*, vol. 47, pp. 853–899, 2013.
- [4] V. Ordonez, G. Kulkarni, and T. Berg, “Im2text: Describing images using 1 million captioned photographs,” *Advances in neural*

- information processing systems*, vol. 24, pp. 1143–1151, 2011.
- [5] S. Yagcioglu, E. Erdem, A. Erdem, and R. Cakıcı, “A distributed representation based query expansion approach for image captioning,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2015, pp. 106–111.
 - [6] A. Karpathy, A. Joulin, and L. Fei-Fei, “Deep fragment embeddings for bidirectional image sentence mapping,” 2014.
 - [7] R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng, “Grounded compositional semantics for finding and describing images with sentences,” *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 207–218, 2014.
 - [8] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth, “Every picture tells a story: Generating sentences from images,” in *European conference on computer vision*. Springer, 2010, pp. 15–29.
 - [9] S. Li, G. Kulkarni, T. Berg, A. Berg, and Y. Choi, “Composing simple image descriptions using web-scale n-grams,” in *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, 2011, pp. 220–228.
 - [10] G. Kulkarni, V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, “Babytalk: Understanding and generating simple image descriptions,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2891–2903, 2013.
 - [11] N. K. Kumar, D. Vigneswari, A. Mohan, K. Laxman, and J. Yuvaraj, “Detection and recognition of objects in image caption generator system: A deep learning approach,” in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*. IEEE, 2019, pp. 107–109.
 - [12] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell:

- A neural image caption generator,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.
- [13] Y. H. Tan and C. S. Chan, “Phrase-based image caption generator with hierarchical lstm network,” *Neurocomputing*, vol. 333, pp. 86–100, 2019.
- [14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [16] R. Gerber and N.-H. Nagel, “Knowledge representation for the generation of quantified natural language descriptions of vehicle traffic in image sequences,” in *Proceedings of 3rd IEEE international conference on image processing*, vol. 2. IEEE, 1996, pp. 805–808.
- [17] B. Z. Yao, X. Yang, L. Lin, M. W. Lee, and S.-C. Zhu, “I2t: Image parsing to text description,” *Proceedings of the IEEE*, vol. 98, no. 8, pp. 1485–1508, 2010.
- [18] R. Lebrecht, P. O. Pinheiro, and R. Collobert, “Simple image description generator via a linear phrase-based approach,” *arXiv preprint arXiv:1412.8419*, 2014.
- [19] P. Kinghorn, L. Zhang, and L. Shao, “A region-based image caption generator with refined descriptions,” *Neurocomputing*, vol. 272, pp. 416–424, 2018.
- [20] Z. Zhou, K. Li, and L. Bai, “A general description generator for human activity images based on deep understanding framework,” *Neural Computing and Applications*, vol. 28, no. 8, pp. 2147–2163, 2017.
- [21] B. Dai, S. Fidler, R. Urtasun, and D. Lin, “Towards diverse and natural image descriptions via a conditional gan,” in *Proceedings*

of the *IEEE International Conference on Computer Vision*, 2017, pp. 2970–2979.

- [22] D. Harwath and J. Glass, “Deep multimodal semantic embeddings for speech and images,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 237–244.

Mahesh Kumar Morampudi, Nagamani Gonthina,
Nuthanakanti Bhaskar, V. Dinesh Reddy

Received December 17, 2021

Revised 1 – May 17, 2022

Revised 2 – August 15, 2022

Accepted September 10, 2022

Mahesh Kumar Morampudi

ORCID: <https://orcid.org/0000-0002-6888-4637>

Department of Computer Science and Engineering

SRM University AP

Amaravathi

Andhra Pradesh

India

E-mail: morampudimahesh@gmail.com

Nagamani Gonthina

ORCID: <https://orcid.org/0000-0001-9559-8030>

Department of Computer Science and Engineering

Institute of Aeronautical Engineering

Hyderabad

Telangana

India

E-mail: gnvsk1986@gmail.com

Nuthanakanti Bhaskar

ORCID: <https://orcid.org/0000-0001-9852-1004>

Department of Computer Science and Engineering

CMR Technical Campus

Hyderabad

V. Dinesh Reddy

ORCID: <https://orcid.org/0000-0003-3945-6171>

Department of Computer Science and Engineering

SRM University AP

Amaravathi

Andhra Pradesh

India

E-mail: dineshvemula@gmail.com

Prostate Cancer Classifier based on Three-Dimensional Magnetic Resonance Imaging and Convolutional Neural Networks

Ana-Maria Minda (Perea)

Adriana Albu

Abstract

The main reason for this research is the worldwide existence of a large number of prostate cancers. This article underlines how necessary medical imaging is, in association with artificial intelligence, in early detection of this medical condition. The diagnosis of a patient with prostate cancer is conventionally made based on multiple biopsies, histopathologic tests and other procedures that are time consuming and directly dependent on the experience level of the radiologist. The deep learning algorithms reduce the investigation time and could help medical staff. This work proposes a binary classification algorithm which uses convolutional neural networks to predict whether a 3D MRI scan contains a malignant lesion or not. The provided result can be a starting point in the diagnosis phase. The investigation, however, should be finalized by a human expert.

Keywords: prostate cancer, classification, decision-making, diagnosis, CNN, MRI.

MSC 2020: 68T07, 68T20, 68U10.

1 Introduction

Prostate cancer is the second most diagnosed type of cancer in men, after lung cancer [1], with 1 of 5 cancer cases in most of the countries [2], [3]. The diagnosis is based on biopsy [4], histopathology [1] or measurement of PSA (Prostate Specific Antigen) and DRE (Digital

Rectal Examination) in the incipient stages of the disease [3]. Nevertheless, the physicians encounter difficulties because of too many biopsies [4], with low accuracy [5] and specificity [3] in diagnosis. Another problem is the limited medical staff [1].

An MRI (Magnetic Resonance Imaging) scan creates detailed images of organs and tissue based on a magnetic field and radio waves [6]. The use of prostate MRI scans reduces the number of biopsies. Further, convolutional neural networks (CNN) can be trained for automatic processing and interpretation of MRI scans, improving the diagnosis of prostate cancer.

These images contain anatomic and functional parameters (multi-parameters MRI, or simple mp-MRI). The use of mp-MRI together with the structured reporting scheme PI-RADS (Prostate Imaging Reporting and Data System) is not quite simple, being strongly dependent on the experience of the radiologist. Therefore, the Computer-aided diagnosis (CAD) systems are a necessity in order to minimize the human effort in the diagnosis procedures [4]. Using deep learning algorithms, the radiologists might obtain an estimation of the severity of this disease, in a shorter time. This would give them the opportunity to focus on treatment and patient support rather than the diagnosis process.

The proposed solution is an algorithm for prostate cancer classification using 3D multi-parameter MRI. Generally, the examination of such an image is made in 30-45 minutes by a human expert. However, using a classification algorithm, the examination is done automatically. The algorithm will classify the prostate lesions as being either benign or malignant.

This work uses PyCharm, Anaconda Development Environment and Jupyter Notebook platform. The algorithm is implemented in Python language, using PyTorch library and MONAI open-source framework for deep learning and healthcare imaging. The classification is made based on DICOM (Digital Imaging and Communications in Medicine) files, which are used to train and test multiple architectures of neural networks. The most performant architecture is included further in the application. The pre-trained neural models were not considered.

2 The context of this research

From a clinical point of view, prostate cancer can be classified as being significant or not significant, according to the aggressivity level of the lesion [2]. The Gleason grade system is generally used to grade prostate cancer lesions. Each investigated tissue receives a Gleason score, which represents the diagnosis of tumor malignancy. This can be low, moderate, or high [1].

Medical images and artificial intelligence mechanisms are frequently used in diagnosis, with undeniable success. This naturally led to the idea of using them in prostate cancer detection [7]. Therefore, two challenges called PROSTATEx have been initiated to speed up the research in this medical field (PROSTATEx Challenge and PROSTATEx-2 Challenge) [8], [9]. A database of 3D MRI scans for prostate cancer has become available, and different algorithms for classification or grading of prostate cancer lesions have been developed. The ground truth set has been defined by physicians with over 20 years of experience, who annotated the images [3].

2.1 Related work

There are multiple studies that explored deep learning algorithms used for prostate cancer detection. For instance, [2], [3], and [7] propose different deep learning algorithms that classify prostate lesions. Another study [5] developed an architecture of deep CNN trained on 3D mp-MRI to diagnose prostate cancer. This architecture was inspired by VGG (Visual Geometry Group) networks [5], which use small convolutions (3x3) in order to have a better recognition of the models [2]. The aim of this study was to demonstrate the applicability of deep learning methods in medical imaging for cancer. It was observed that deep learning algorithms perform better than conventional models based on feature engineering [5]. There are also studies that proposed automatic methods for grading the lesions from mp-MRI, using Gleason system (GGG – Gleason Grade Group) [3].

All these studies started based on PROSTATEx challenges (PROSTATEx Challenge and PROSTATEx-2 Challenge) provided by American Association of Physicists in Medicine (AAPM) together with Society of

Photo-Optical Instrumentation Engineers (SPIE) and National Cancer Institute (NCI) [8], [9].

The PROSTATEx-2 challenge aimed to classify prostate cancer using the available image set mp-MRI. One of the solutions used the Ordinal Classification (or Ordinal Regression) technique, which performs a multi-class classification [10]. This solution sorted the lesions according to their aggressivity level $G1 < G2 < G3 < G4 < G5$.

The algorithms developed proved that CNN (VGG-16, for instance) are more efficient when they are used together with other methods, such as Ordinal Classifier. Nevertheless, the solutions have not been tested on other data sets in order to validate their efficiency [2].

Regarding the deep learning algorithms, the easiest way of evaluating them is to analyze the multitude of similar research articles and to run the provided prototypes in the field of interest. Therefore, two public repositories that analyze prostate lesions using the data sets provided by PROSTATEx challenges have been considered.

2.1.1 Lesion classification algorithm using TensorFlow library

Piotr Sobecki proposed a deep learning algorithm for prostate cancer classification based on the MRI data set provided by the first edition of PROSTATEx [11]. The algorithm analyzes the images and classifies the lesions according to their malignity level (the lesions are clinically significant or not). The training data set contains 330 lesions (76 malignant and 254 benign).

The classification algorithm is based on VGG architecture. The repository provides three distinct architectures to analyze the lesions:

- **CNN_VGG_SIMPLE** – performs a binary classification based on clinical signification of the lesion, using three sub-networks;
- **CNN_VGG_MODALITIES** – uses a mix of expert architectures, where each modality is able to make predictions;
- **CNN_VGG_PIRADS** – uses PI-RADS system, together with other expert architectures.

The architectures developed by Piotr Sobecki have the advantage of diversity for each one. On the other hand, there are necessary excessive computations, which lead to a training process that needs too many resources (software, hardware, separated drivers, long time for an epoch, etc.). These architectures have been locally run and some Key Performance Indicators (KPI) have been analyzed. The AUC (Area Under the Curve) values for all three architectures are in the range (0.3, 0.75), with the best results obtained for the architecture CNN_VGG_MODALITIES. These values demonstrate that the algorithm training is slow. In order to improve it, it is necessary to run more than 200 epochs per fold (it is not visible a significant improvement of the KPIs at 100 epochs). Another disadvantage is the old and limited technology (the first version of TensorFlow library was used, which is no longer supported).

2.1.2 Lesion detection and segmentation algorithm using PyTorch framework

This architecture proposes a completely automated system, which takes the mp-MRI prostate scans of a patient suspected of prostate cancer and localizes the lesions using the detection model Retina U-Net. It then performs a segmentation of these lesions and provides the most appropriate Gleason grade (GGG) [12]. This architecture has been developed using the data set provided by the second edition of PROSTATEx challenge. The training set contains 142 lesions (54 of them with GGG 1, 45 with GGG 2, 25 with GGG 3, 10 with GGG 4 and 8 with GGG 5).

The advantages of this architecture are the AUC value of 0.87 [12] and the technologies that were used. On the other hand, it has a limitation because of the Medical Detection Toolkit [13], which is dedicated to Linux operating system only.

2.2 Limitations of the prostate cancer classification

Firstly, the limitation in prostate cancer diagnosis is introduced by MRI scans themselves. The image evaluation is subjectively performed based on PI-RADS criteria or on Gleason system [2]. PI-RADS is a

structured reporting scheme for prostate mp-MRI used to evaluate a patient who is suspected of cancer [14]. But the results are strongly dependent on the human expert who makes the evaluation. This also happens with the Gleason system [2]. Therefore, the deep learning algorithms did not succeed in adequately classify the lesions. In most cases, the tumors were considered low malignant (grade 1) or high malignant (grade 5) [3].

Computer-aided diagnosis (CAD) requires large data sets for training, validation, and testing. The data sets available for prostate cancer are limited because the annotation of MRI scans is tiring and time consuming for human experts. Also, even if multiple patches can be extracted from a single scan, in many cases these are similar, even if the cancer tissues are variate.

Another factor that restricts the use of deep learning in prostate tumors classification is the data imbalance – the observations that belong to individual classes are disproportionate. These observations could not be enough for successfully learning the features of that class [1].

3 Materials and methods

3.1 Available datasets

3.1.1 PROSTATEx Challenge (“SPIE-AAPM-NCI Prostate MR Classification Challenge”)

This challenge took place from 21.11.2016 to 15.01.2017 and aimed to determine methods for image analysis in order to diagnose prostate cancer and to classify it from clinical point of view [8], [9], [15], [16]. The data set contains mp-MRI scans of 330 prostate lesions, as well as spatial coordinates, atomic coordinates, and the clinical significance. The testing data set contains 208 prostate lesions.

3.1.2 PROSTATEx-2 Challenge (“SPIE-AAPM-NCI Prostate MR Gleason Grade Group Challenge”)

The second challenge took place from 15.05.2017 to 23.06.2017 and focused on some biomarkers for mp-MRI scans, which are useful when

Gleason grade of prostate cancer is determined [8], [9], [15], [16]. Gleason Grade Group (GGG) is a standard for the measurement of the aggressiveness of prostate cancer, which makes possible the prediction of the pathological state and of the oncological result. A GGG grade is assigned to a possible lesion using histopathological analysis of biopsies. The GGG grades are natural numbers between 1 and 5, where 1 is a lesion that doesn't require treatment and 5 is the most severe prostate cancer lesion [1], [3].

The training set contains 112 prostate lesions, the spatial and atomic coordinates, as well as the GGG grade of each lesion. The testing set contains 70 lesions.

3.1.3 DiagSet

Besides the data sets provided by PROSTATEx challenges, there are also other data sets available that can be used to train deep learning algorithms. DiagSet is a histopathologic data set, which contains regions of prostate tissue, annotated in WSI (Whole Slide Image) scans. These regions have been marked with GGG grade, in order to specify the severity level of the disease. The annotation was made by human experts in histopathology [1].

This data set was used to train several CNN architectures such as AlexNet, VGG16, VGG19, ResNet50, and InceptionV3 in different configurations or different dimensions of the images. The best configurations have been used to create an ensemble of classifiers. The obtained results have been validated by physicians [1].

3.2 Image types

The solution presented in this work used the data set provided by PROSTATEx challenge (“SPIE-AAPM-NCI Prostate MR Classification Challenge”) [8], [9], [15], [16]. The main reason for this is that at least five 3D MRI sequence types are provided for each patient. The ground truth is available, and it is used in both the training and evaluation processes.

There are three anatomic planes that compound a 3D MRI: transverse plane (is perpendicular to the spine and divides the body into

superior and inferior parts), sagittal plane (divides the body into left and right parts), and coronal plane (divides the body in vertical plane into dorsal and ventral parts) [17]. Fig. 1 [8], [9] presents the scans for the three anatomic planes of a patient from training data set PROSTATEx.

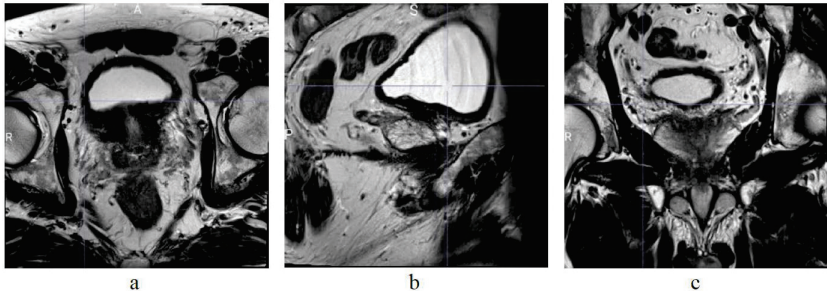


Figure 1. (a) transverse plane, (b) sagittal plane, and (c) coronal plane

The five sequence types [7] that are present in 3D MRI of all patients are:

- Transverse sequences T2W (T2-wieghted);
- Sagittal sequences T2W;
- Transverse sequences ADC (Apparent Diffusion Coefficient);
- Transverse sequences DWI (Diffusion-Weighted Imaging);
- Transverse sequences Ktrans (a measure for capillary permeability).

The four types of transverse images (T2W, ADC, DWI, and Ktrans) are presented in Fig. 2 [8], [9] (they were also selected from training data set PROSTATEx).

All sequences have been compressed in DICOM files, and then provided as a public data set in PROSTATEx challenge. DICOM (Digital Imaging and Communications in Medicine) is an international standard that defines medical image formats, being one of the most frequently

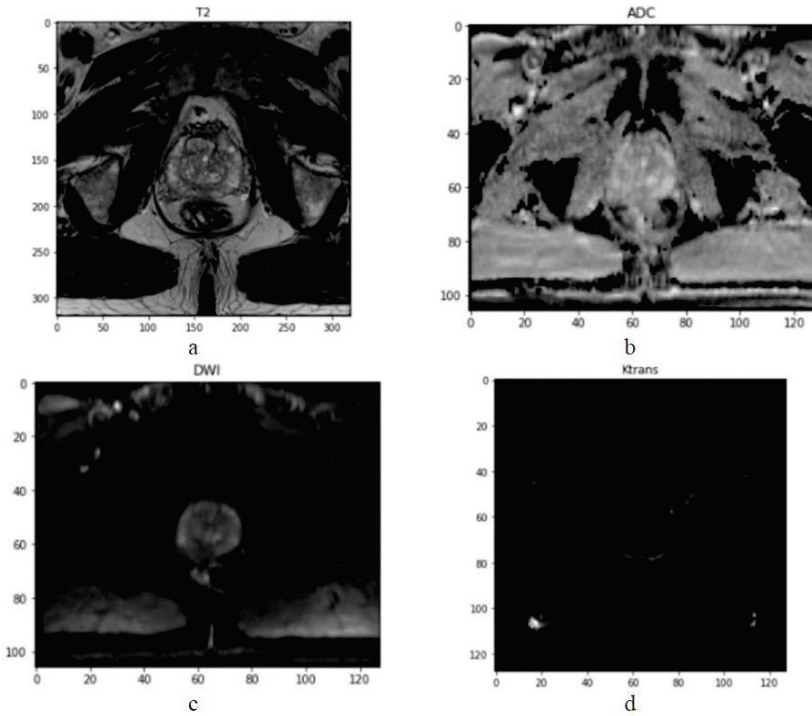


Figure 2. (a) a T2W slice, (b) an ADC slice, (c) a DWI slice, and (d) a Ktrans slice

used standards for medical information transfer. DICOM is implemented in almost all equipment from radiology, radiotherapy, cardiologic imaging, and other medical domains such as ophthalmology or stomatology [18]. DICOM standard is recognized by the International Organization for Standardization as standard ISO 12052 [18].

The current project also uses NIfTI (Neuroimaging Informatics Technology Initiative) files. This format is dedicated to neural imaging, having two versions: NIfTI-1 and NIfTI-2 (which is an update of NIfTI-1 that allows storing a larger quantity of data) [19].

3.3 Development environments

The following technologies were used throughout the development process:

- Anaconda.org - service for packages management;
- Pip and Conda - systems for packages management;
- PyCharm - development environment;
- Jupyter Notebook - application used for a visual perspective of the data and the proposed solution.

The programming language used for the entire project is Python, in addition to a couple of frameworks for training, validation, and testing the model (PyTorch and MONAI) and several libraries for data reading and visualization (Pandas, Matplotlib, Nibabel).

3.4 System's architecture

The proposed solution has a simple architecture (Fig. 3). The MRI scans (DICOM format) of the data set provided by PROSTATEx challenge are locally processed. The operating system of the workstation is Windows. Once the images are transformed into NIfTI format, they are processed using a DenseNet-121 classification algorithm. The output data of the training phase are the parameters of the model (the best results of the training). Then, the model can be evaluated (testing phase).

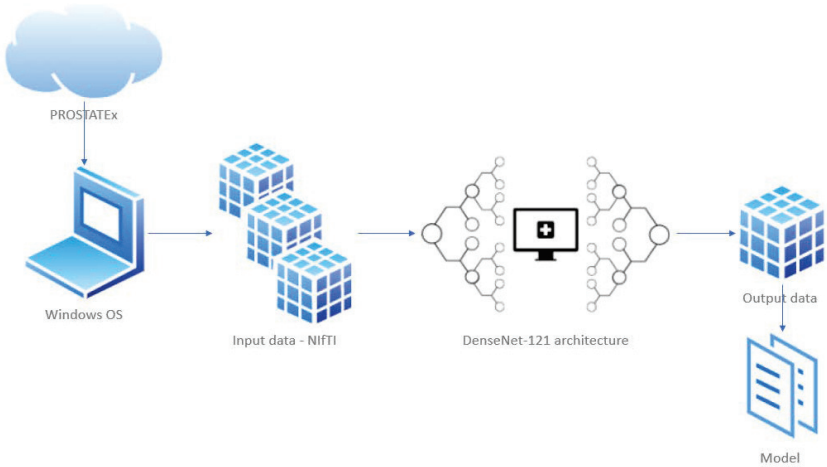


Figure 3. The architecture of the proposed solution

The main steps that were followed in the development of this solution are:

- Analysis of the existing repositories;
- Virtual environment setup for the Python packages and libraries;
- Available data (images and labels) visualization and analysis;
- Data processing;
- Training of the classification algorithm;
- Evaluation of the classification algorithm.

The classification algorithm uses a DenseNet-121 network, with the following features: one convolution 7×7 , 58 convolutions 3×3 , 61 convolutions 1×1 , 4 average pooling layers and one fully connected layer [20]. In a DenseNet network, each layer is directly connected to the next layer (Densely Connected Convolutional Network). Therefore, for n layers, there will be $n(n + 1)/2$ direct connections [20]. According to the MONAI documentation, because the data set images are three-dimensional, the DenseNet-121 is not pre-trained.

3.5 Model training and evaluation

The model was trained, validated, and tested using the PyTorch framework. In order to perform this procedure (specific to deep learning algorithms), the data set provided by PROSTATEx challenge has been randomly divided into sub-sets for training, validation, and testing. The data set contains 204 patients. Training and validation phases are performed using 80% of the patients (80% of them for training and 20% for validation). The rest of the patients (20% of the initial set) remain for testing phase. Therefore, 163 patients are used for training and validation (80% of 204) and 41 for testing (20% of 204).

Each image of these sub-sets has a label that specifies whether a lesion is malignant (clinically significant) or not. The labels were set by experienced radiologists.

Once the dataset is split and labeled, Transforms objects are defined. These objects scale the intensity of the images, re-dimension them and rotate them by 90° . The data type will also be verified, in order to ensure that these are PyTorch tensors [21]. Each sub-set has a DataLoader which contains an object of type ImageDataset and hardware specifications to parallelize the training, validation, and testing processes [22]. The ImageDataset objects contain a single-dimensional matrix for the images of the sub-set, a 1D matrix for the labels and a Transforms object.

After the DataLoader objects are created, the model can be trained, validated, and tested. Several parameters such as the number of parallel processed images (batch_size), the number of parallel threads (num_workers), or number of epochs for training phase can be set in a Python configuration file.

The training phase uses a simple PyTorch loop, which evaluates in each iteration the internal performance of the model using the validation sub-set. The epochs' parameters that improve the metrics are saved, being used in the final model, which is the best model produced.

At the end, the model is evaluated comparing the predictions with the labels. This will determine key performance indicators (KPI) based on the elements of the confusion matrix:

- TP (true positive) – both algorithm and label indicate a malig-

nant lesion;

- TN (true negative) – both algorithm and label indicate a benign lesion;
- FP (false positive) – the algorithm predicts a malignant lesion, but the label indicates a benign lesion;
- FN (false negative) – the algorithm predicts a benign lesion, but the label indicates a malignant lesion.

The KPIs that are considered in model evaluation are [23]:

- Accuracy – the most frequently used metric when an algorithm is evaluated, indicating the number of correct predictions reported to the total predictions (Eq. (1)):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}; \quad (1)$$

- Precision – the report between correct positive predictions and all positive predictions (Eq. (2)):

$$Precision = \frac{TP}{TP + FP}; \quad (2)$$

- Recall (sensitivity) – measures how many positive cases have been correctly predicted reported to all positive cases of the data set (Eq. (3)):

$$Recall = \frac{TP}{TP + FN}; \quad (3)$$

- F1-Score – the harmonic average of precision and recall (Eq. (4)):

$$F1-Score = 2 * \frac{Precision * Recall}{Precision + Recall}; \quad (4)$$

4 Experimental results and discussions

Fig. 4 represents a DataFrame object which contains information and results from the training and testing phases of the classification algorithm. The “color”, “loss_function”, “optimizer”, “learning_rate”, and “epochs” columns define the color code of the graphical representation, the loss function type, the optimizer type, and the learning rate, as well as the number of the epochs assigned to the classification algorithm. The “best_accuracy” column represents the best value obtained in the training phase. The last 8 columns are the KPI values obtained in the testing phase.

i	color	loss_function	optimizer	learning_rate	epochs	best_accuracy	TP	TN	FP	FN	accuracy	recall	precision	f1
0	red	Binary Cross Entropy	Adam	1e-3	100	0.7214	0	174	0	81	0.682	0.000	-0.100	-0.100
1	lightgreen	Binary Cross Entropy	Adam	1e-4	50	0.6990	0	136	0	126	0.519	0.000	-0.100	-0.100
2	blue	Binary Cross Entropy	Adam	1e-4	50	0.6768	26	160	12	64	0.710	0.289	0.684	0.406
3	yellow	Binary Cross Entropy	Adam	1e-4	100	0.6318	22	153	22	59	0.684	0.272	0.500	0.352
4	lightgray	Binary Cross Entropy	Adam	1e-4	200	0.7114	6	164	1	80	0.677	0.070	0.857	0.129
5	pink	Binary Cross Entropy	Adam	1e-4	50	0.5149	2	177	4	86	0.665	0.023	0.333	0.043
6	orange	Binary Cross Entropy	Adam	1e-4	50	0.8146	5	146	4	105	0.581	0.045	0.556	0.084
7	brown	Binary Cross Entropy	Adam	1e-5	50	0.6745	21	126	25	82	0.579	0.204	0.457	0.282
8	purple	Binary Cross Entropy	Adam	1e-5	200	0.7608	9	154	8	84	0.639	0.097	0.529	0.164

Figure 4. Information obtained during the training and testing process

Figs. 5, 6, and 7 present a visual representation of the internal results during the training process. The values of the loss function are directly related to the type of the function, to the number of epochs, to the optimizer type, and to the learning rate value. For instance, the larger the number of epochs, the smaller the value of the loss function – in Fig. 5, the loss function is around 0.2, while in Figs. 6 and 7, the values are less than 0.1.

Another aspect that can be observed in Figs. 5, 6, and 7 is that the internal accuracy of the algorithm in the validation stage reaches the highest values in the first 50 training epochs. Therefore, the classification algorithm doesn’t need a larger number of iterations to determine the optimal values of the parameters.

The best values of the evaluated models are represented in Fig. 8. During the training process, the model with the best results is the orange one (i=6, with an accuracy of over 80%) with the following fea-

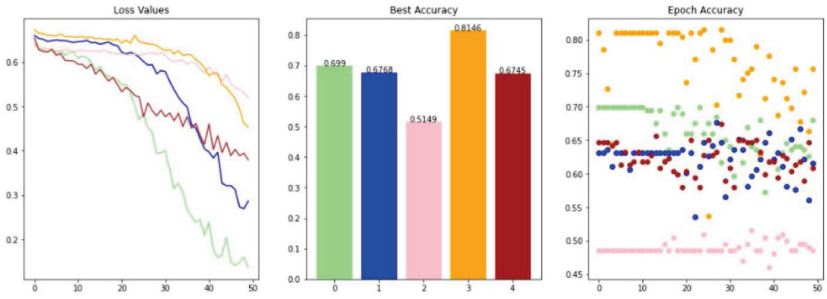


Figure 5. Loss function values (left), accuracy values (right), and the best accuracies of the models (middle) in the training phase – 50 epochs

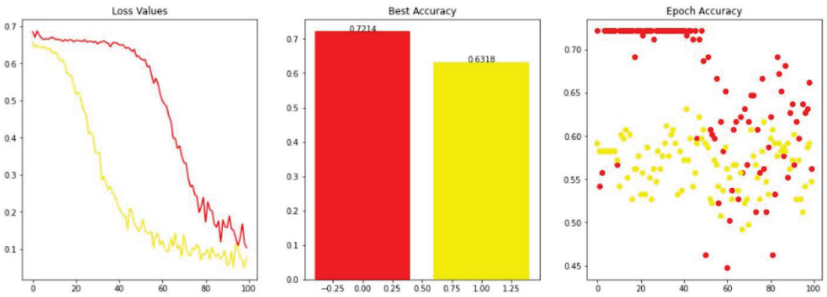


Figure 6. Loss function values (left), accuracy values (right), and the best accuracies of the models (middle) in the training phase – 100 epochs

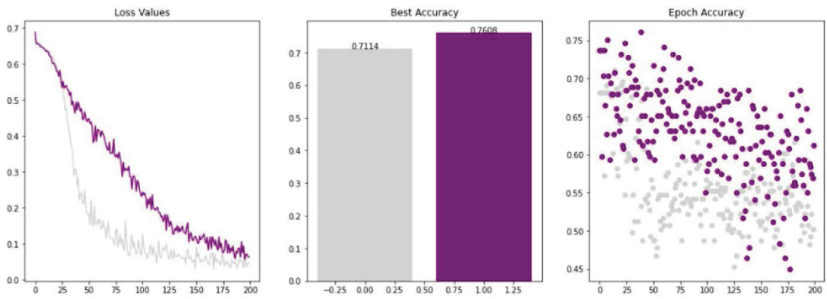


Figure 7. Loss function values (left), accuracy values (right), and the best accuracies of the models (middle) in the training phase – 200 epochs

tures: 50 epochs, Binary Cross Entropy loss function, Adam optimizer, and 1e-4 learning rate. Figs. 9 and 10 represent the KPI values ob-

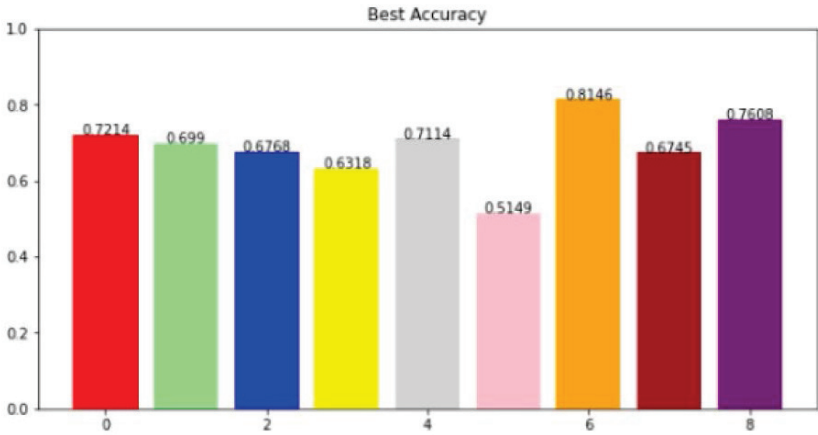


Figure 8. The best accuracy results in the training phase

tained in the testing phase. According to these results, the models that successfully identified malignant lesions are the blue one ($i=2$) and the yellow one ($i=3$). Nevertheless, the yellow model is in top of the models with the most FP values. Therefore, the blue model can be considered the most suitable for prostate lesions classification, according to the KPI values.

Looking at the representation of the metrics, the best accuracy is provided by the blue model ($i=2$), with over 70% correct predictions. Other models that correctly classified over 65% of the lesions are red ($i=0$), yellow ($i=3$), light grey ($i=4$), and pink ($i=5$). Regarding the recall and F1-score, the models blue and yellow have the best results. The precision has the highest values for the models light grey and blue. Nevertheless, the recall and F1-score values of the light grey mode are much smaller than those of the blue model. Therefore, the blue model ($i=2$) is the best model once the testing phase of the classification algorithm is completed.

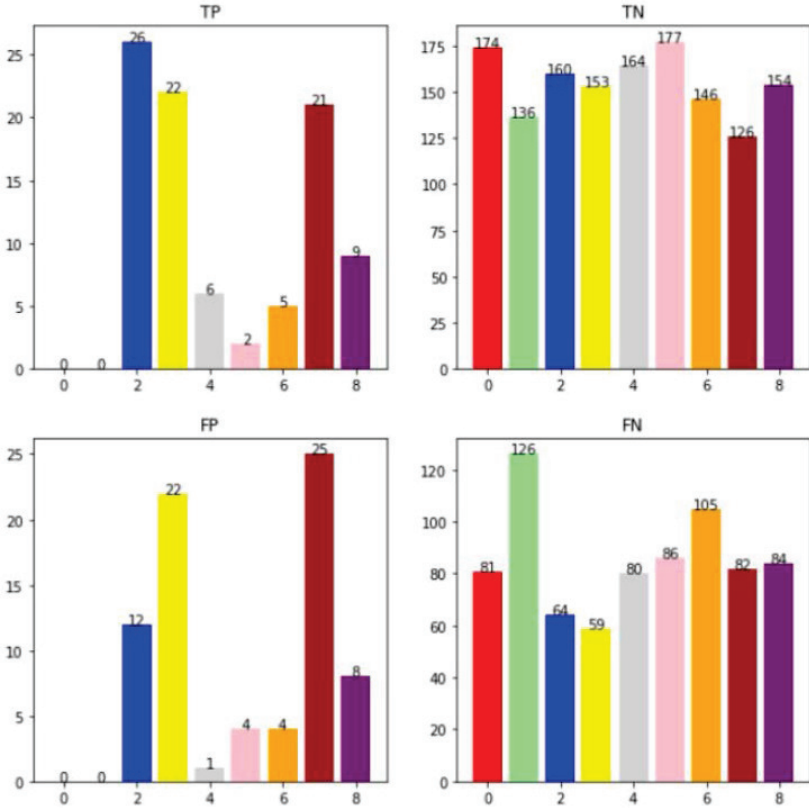


Figure 9. The elements of the confusion matrix in the testing phase

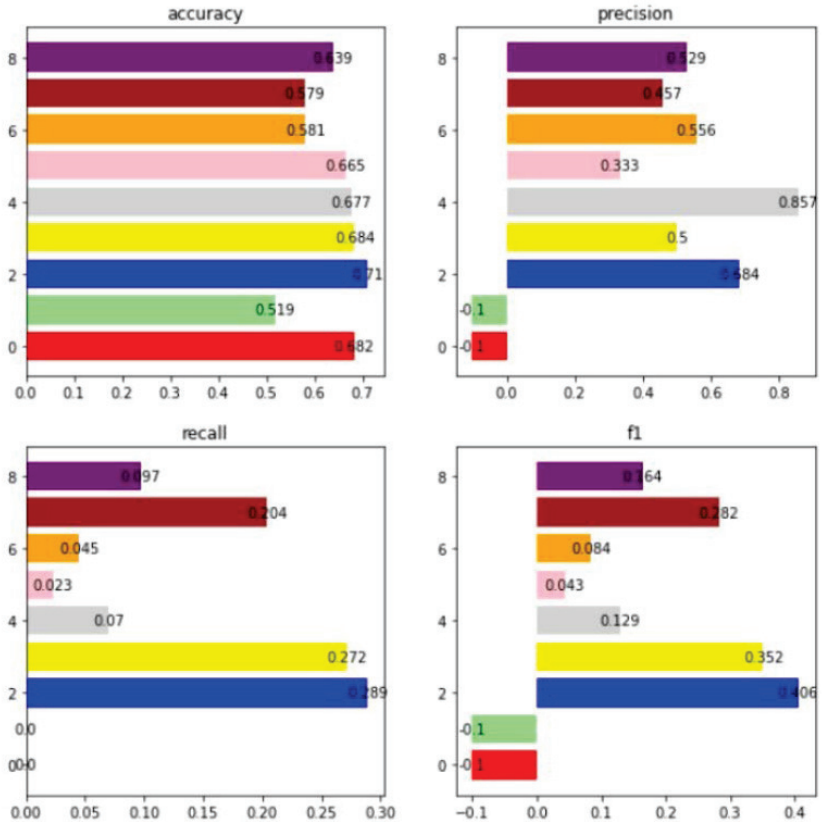


Figure 10. The performance metrics in the testing phase

4.1 Special aspects observed

During the evaluation of the classification algorithm, different values and trends of the KPI and of the loss function have been compared. The aim of the analysis of these values is to set the correct values and to determine the best parameters of the model.

The oscillations in the loss function indicate that learning rate is either too small or too large, even if it decreases exponentially during the epochs. The solution for such a problem is to train the classification algorithm with different optimizers, architectures, loss functions,

and learning rates. The process of finding the best parameters is an experimental one. Through the metrics analysis in the testing phase, the following can be observed:

- Precision – missing positive predictions ($TP+FP=0$). The precision in the red and light green models resulted in negative values. These negative values have been assigned to avoid and to mark a division by zero (during the testing process, no lesion was classified as malignant).
- Recall – missing correct positive predictions ($TP=0$). The recall was zero in the red and light green models. This means that no malignant lesion was correctly classified during the testing process.
- F1-score – irrelevant. This is directly dependent on precision and recall; as a result, for the previous two cases, the F1-score becomes irrelevant.

Therefore, all these three metrics (precision, recall and F1-score) are directly dependent on the number of positive predictions in the testing phase. The number of malignant lesions of the prostate in the data set is small. The solution in this direction is to increase the number of malignant lesions in the training sub-set by an oversampling process (a technique for the adjustment of a class distribution in a data set [24]).

5 Conclusion

This work proposes a binary classification algorithm for prostate lesions. It predicts whether an MRI scan contains a malignant lesion or not. The images and labels provided by PROSTATEx challenge have been analyzed. A new architecture, which uses current technologies (such as PyTorch and MONAI frameworks) has been created. Several models have been evaluated and the most performant architecture has been selected to present the performances of the classification algorithm.

The advantages of the proposed solution are given by the current technologies (which have constant technical support) and by the optimizations in the training and evaluation phases. The PyTorch frame-

work does not require the installation of additional drivers in order to use GPU. The analysis and the processing of the MRI sequences is faster using GPU. Unlike TensorFlow, this framework does not over-charge the hardware resources. Thus, models can be obtained with minimum degradation of the physical resources that are used.

Another advantage is the compatibility of the MONAI framework with multiple operating systems. The solution is easily configurable and accessible to the developers that lead research projects in this area. The source code is available on GitHub [25].

The solution provided by this research is the first step in development of a deep learning application for prostate lesions classification. The next steps involve the integration of this model into an application that provides a graphical user interface. The physicians might upload a 3D MRI into the application, obtaining a classification into “malignant lesion” or “benign lesion”. More than this, classification algorithms are the simplest method of analyzing 3D medical images. The development of algorithms for detection and segmentation of the lesions would improve the application.

References

- [1] M. Koziarski, B. Cyganek, B. Olborski, Z. Antosz, M. Zydak, B. Kwolek, P. Wasowicz, A. Bukala, J. Swadzb, and P. Sitkowski, “DiagSet: a dataset for prostate cancer histopathological image classification,” *arXiv, Electrical Engineering and Systems Science, Image and Video Processing*, May 2021. [Online]. Available: <https://arxiv.org/pdf/2105.04014.pdf>.
- [2] B. Abraham and M.S. Nair, “Automated Grading of Prostate Cancer using Convolutional Neural Network and Ordinal Class Classifier,” *Informatics in Medicine Unlocked*, Elsevier, vol. 17, no. 1, Article no. 100256, October 2019.
- [3] C. de Vente, P. Vos, M. Hosseinzadeh, J. Pluim, and M. Veta, “Deep Learning Regression for Prostate Cancer Detection and Grading in Bi-Parametric MRI,” *IEEE Trans. Biomedical Engineering*, vol. 68, no. 2, pp. 374–383, Feb. 2021.

- [4] S.G. Armato, H. Huisman, K. Drukker, L. Hadjiiski, J.S. Kirby, N. Petrick, G. Redmond, M.L. Giger, K. Cha, A. Mamonov, J. Kalpathy-Cramer, and K. Farahanif, “PROSTATEx Challenges for computerized classification of prostate lesions from multiparametric magnetic resonance images,” *Journal of Medical Imaging*, vol. 5, no. 4, Article no. 044501, Bellingham, 2018.
- [5] S. Liu, H. Zheng, Y. Fengc, and W. Li, “Prostate Cancer Diagnosis using Deep Learning with 3D Multiparametric MRI,” in *Proceedings of SPIE Medical Imaging, vol. 10134*, (Orlando, Florida, United States), 2017. [Online]. Available: <https://doi.org/10.1117/12.2277121>.
- [6] P. Lam and J. Marcin, “What to know about MRI scans,” *Medical News Today*, 2018. [Online]. Available: <https://www.medicalnewstoday.com/articles/146309>. Accessed on: Jan. 2022.
- [7] Hugesene, “3D CNN Classification of Prostate Cancer on PROSTATEx-2,” *Towards Data Science*, 2019. [Online]. Available: <https://towardsdatascience.com/3d-cnn-classification-of-prostate-tumour-on-multi-parametric-mri-sequences-prostatex-2-cced525394bb>.
- [8] “ProstateX challenge,” *Grand Challenge*. [Online]. Available: <https://prostatex.grand-challenge.org/>. Accessed on: January 2022.
- [9] G. Litjens, O. Debats, J. Barentsz, N. Karssemeijer, and H. Huisman, “ProstateX Challenge data,” *The Cancer Imaging Archive*, 2017. [Online]. Available: <https://doi.org/10.7937/K9TCIA.2017.MURS5CL>. Accessed on: Nov. 2021.
- [10] L. Gaudette and N. Japkowicz, “Evaluation Methods for Ordinal Classification,” *Advances in Artificial Intelligence, Canadian AI 2009* (Lecture Notes in Computer Science, vol. 5549), Berlin, Heidelberg: Springer, 2009, pp 207–210.
- [11] P. Sobceki, “Prostate lesion classification using Deep Convolutional Neural Networks,” GitHub. [Online]. Available: <https://github.com/piotrsobceki/PCa-CNNs>, 2020. Accessed on: Nov. 2021.

- [12] O.J. Pellicer-Valero, J.L. Marenco Jiménez, V. Gonzalez-Perez, J.L.C. Ramón-Borja, I.M. García, M. Barrios Benito, P. Pelechano Gómez, J. Rubio-Briones, M. José Rupérez, and J.D. Martín-Guerrero, “Deep Learning for fully automatic detection, segmentation, and Gleason Grade estimation of prostate cancer in multiparametric Magnetic Resonance Images,” *Scientific Reports*, vol. 12, Article no. 2975, 2022.
- [13] P.F. Jaeger, S.A.A. Kohl, S. Bickelhaupt, F. Isensee, T. Anselm Kuder, H.-P. Schlemmer, and K.H. Maier-Hein, “Retina U-Net: Embarrassingly Simple Exploitation of Segmentation Supervision for Medical Object Detection,” *arXiv, Computer Science, Computer Vision and Pattern Recognition*, Nov. 2018. [Online]. Available: <https://arxiv.org/pdf/1811.08661.pdf>.
- [14] M. Czarniecki and Y. Weerakkody, “Prostate Imaging-Reporting and Data System (PI-RADS),” *Radiopaedia.org*, 2021. [Online]. Available: <https://doi.org/10.53347/rID-27968>. Accessed on: Jan. 2022.
- [15] G. Litjens, O. Debats, J. Barentsz, N. Karssemeijer, and H. Huisman, “Computer-aided detection of prostate cancer in MRI,” *IEEE Trans. Medical Imaging*, vol. 33, pp.1083–1092, 2014. Available: <https://doi.org/10.1109/TMI.2014.2303821>. Accessed on: Nov. 2021.
- [16] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle, L. Tarbox, and F. Prior, “The Cancer Imaging Archive (TCIA): Maintaining and Operating a Public Information Repository,” *Journal of Digital Imaging*, vol. 26, no. 6, pp. 1045–1057, Dec. 2013, Available: <https://doi.org/10.1007/s10278-013-9622-7>. Accessed on: Nov. 2021.
- [17] “1.4D: Body Planes and Sections,” in *Anatomy and Physiology (Boundless), 1: Introduction to Anatomy and Physiology*, LibreTexts, Medicine, 2020.
- [18] “About DICOM: Overview,” *DICOM – Digital Imaging and Communications in Medicine*. [Online]. Available: <https://www.dicom-standard.org/about-home>. Accessed on: Jul. 2022.

- [19] C. Moore and H. Knipe, “NIFTI (file format),” *Radiopaedia.org*, 2014. [Online]. Available: <https://doi.org/10.53347/rID-72562>. Accessed on: Jul. 2022.
- [20] A. Ahmed, “Architecture of DenseNet-121,” *OpenGenus IQ: Computing Expertise & Legacy, Machine Learning (ML) Tutorial*. [Online]. Available: <https://iq.opengenus.org/architecture-of-densenet121/>. Accessed on: Jul. 2022.
- [21] “Transforms,” *MONAI*. [Online]. Available: <https://docs.monai.io/en/stable/transforms.html>. Accessed on: Jul. 2022.
- [22] “Datasets & DataLoaders,” *PyTorch*. [Online]. Available: https://pytorch.org/tutorials/beginner/basics/data_tutorial.html. Accessed on: Jul. 2022.
- [23] T. Kanstrén, “A Look at Precision, Recall, and F1-Score,” *Towards Data Science*, 2020. [Online]. Available: <https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec>. Accessed on: Jul. 2022.
- [24] N.V. Chawla, “Data Mining for Imbalanced Datasets: An Overview,” in *Data Mining and Knowledge Discovery Handbook*, Springer, 2010, pp. 875–886. ISBN 978-0-387-09823-4. Available: https://doi.org/10.1007/978-0-387-09823-4_45.
- [25] A.-M. Minda (Perea), “Prostate Cancer Classifier,” GitHub, Jul. 2022. [Online]. Available: https://github.com/pereaanamaria/prostate_cancer_classifier.

Ana-Maria Minda (Perea), Adriana Albu

Received November 01, 2022

Revised February 20, 2023

Accepted February 21, 2023

Ana-Maria Minda (Perea)

ORCID: <https://orcid.org/0009-0008-1415-4509>

Faculty of Automation and Computers,

Politehnica University Timisoara

Timisoara, Romania

E-mail: anamariaperea10@gmail.com

Adriana ALBU

ORCID: <https://orcid.org/0000-0003-1579-6163>

Department of Automation and Applied Informatics,

Politehnica University Timisoara

Timisoara, Romania

E-mail: adriana.albu@upt.ro

Bandwidth Allocation Algorithm for Makespan Optimization in a Fog-Cloud Environment: Monitoring Application

Bentabet Dougani, Abdeslem Dennai

Abstract

Fog computing technology has emerged to handle a large amount of data generated by the Internet of Things (IoT) terminals and cope with latency-sensitive application requests by allocating computation and storage resources at the edge of the Internet. In many IoT applications, the data acquisition procedures must apply the Directed Acyclic Graph (DAG) to get real-time results. The principal goal of DAG scheduling is to reduce total completion time without breaking priority constraints by properly allocating tasks to processors and arranging task execution sequencing. In this paper, we propose a bandwidth-aware workflow allocation (BW-AWA) that schedules tasks by priority to the resource and optimizes the total execution time (Makespan) in the entire computing system. The task allocation process needs to consider the dependency between tasks. The proposed approach is tested with a monitoring application case study, and the results are compared to well-known approaches to demonstrate its effectiveness in optimizing the Makespan.

Keywords: Cloud Computing, Fog Computing, IoT, List scheduling, DAG, Makespan.

MSC 2020: 68, 68M18, 68M20.

1 Introduction

With the Internet of Things emergence, real-time data access, processing, and storage require high bandwidth and low latency [1]. The

traditional Cloud Computing limitations in terms of working principle and low bandwidth may cause additional communication costs and latency. Therefore, Fog Computing technology has emerged and developed rapidly. A large amount of local data does not need to be uploaded to the Cloud for processing. The basic principle of Fog Computing is to process and analyze data on the data-generating side at the network Edge and provide services to the requests, thus effectively reducing the time delay generated by network transmission [2].

Fog Computing has some challenges. Computing nodes have limited resources, and tasks need to be distributed and scheduled according to the type and scale of actual tasks to avoid excessive load on some computing nodes, which affects system performance. The scheduling strategy aims to optimize the use of resources and the overall performance of task processing in the Fog Computing environment [3]. The performance objective can be achieved by using the allocated resources efficiently. Real-time monitoring of events in an environment is achievable through intelligent monitoring applications composed of multiple tasks cooperating in workflow models. Since the computerization of certain tasks in many IoT applications requires the result data of previous tasks, the workflow can be abstracted and modeled using a Directed Acyclic Graph (DAG), where the graph nodes represent subtasks, and the lines between nodes represent the priority constraint relationships between subtasks. The scheduling problem becomes even more complicated when computing systems are heterogeneous.

The execution time is the most important objective in the scheduling policy for the user in the real-time environment because the submitted workflow task has an urgent need for completion time. Since Fog nodes have limited processing capacity, the scheduling technique should consider the selection of the optimal processors while allocating the tasks. This paper presents workflow services for IoT applications in a Fog-Cloud Computing environment. We will propose a new list scheduling algorithm based on heuristics such as the most efficient and best-accepted category. This proposal algorithm will optimize the total execution time. Here are some factors considered when scheduling tasks:

- The algorithm is based on a task scheduling strategy for the work-

flow application in a heterogeneous Fog-Cloud computing environment.

- Calculate the priority of each task and according to the ranks, determine the order of the tasks.
- Improve the resource allocation phase: During the resource allocation phase, the algorithm will scan all the nodes for each task in order to find the node which has the most optimal bandwidth rate.

To simulate our scheduling algorithm, we applied it to dependent tasks case. This case contains the data collection related to IoT devices and their transfer to Fog nodes for processing and storage in the Cloud.

To evaluate the performance of our proposed algorithm by optimizing the Makespan, we have implemented it and run a set of experiments by modifying the number of tasks and the number of Fog nodes. The results obtained are compared with those given by the algorithms: FCFS (First Come First Serve), Popularity, Heterogeneous Earliest Finish Time (HEFT), and Critical Path On a Processor (CPOP).

The rest of this paper is structured as follows: Section 2 presents a synthesis of related works; Section 3 describes our proposed approach and addresses the task scheduling problem in a Fog-Cloud environment with our scheduling algorithm; the case study and the results of the experiments are presented in Section 4, before concluding and describing future work for the continuation of this research in Section 5.

2 Related work

In the literature, the task scheduling performance in heterogeneous computing systems has been evaluated in many types of research. A relevant solution for task scheduling methods uses optimization algorithms and performance metrics in real Fog scenarios to solve practical problems such as industry and factory [4], [5], transportation [6]–[8], and video processing [9]–[11]. Of course, the modeling and optimization algorithms' goals are different for different applications in different fields.

The Heuristic-based scheduling algorithms can be divided into three categories: List scheduling [12], clustering scheduling [13], and duplica-

tion scheduling [14]. The authors in [15] investigated the QoS (Quality of Service) parameters in a Fog network using three scheduling methods: Concurrent, FCFS, and delay-priority. The arriving tasks in the Concurrent method are assigned independently of consumption capacity in a concurrent manner. The tasks in the FCFS technique execute in the order of entry, and if the data center's processing power is less than the task request, the task is placed in the scheduler queue. Tasks are scheduled based on reduced latency in the delay-priority technique. The concurrent technique has a longer latency than FCFS and delay-priority methods, according to the results. However, these three strategies were not efficient for many problem instances. The workload can be divided into independent and dependent tasks based on the correlation between the scheduled tasks. Surely, for these different workloads, different algorithms are applicable [16].

Zeng et al. [17] studied the problem of minimizing the execution time of independent tasks by considering the placement of images in conjunction with task scheduling by assigning tasks and balancing the load among user devices and Fog devices, but without considering specific QoS requirements for the applications to be deployed. Therefore, developing an approximate algorithm is a good choice compared to the exact method.

Bitam et al. [18] compared a novel optimization algorithm named "Bees Life Algorithm" (BLA) with Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) algorithms in IoT-Fog environments. The results of this algorithm show a good performance of execution time and allocated memory.

In [19], a study of task scheduling for the IoT-based bag-of-tasks application in Fog and Cloud environment, Nguyen and colleagues proposed a GA algorithm to reduce the time required to complete the work. The authors have obtained better results than the BLA algorithm and MPSO (Modified PSO) algorithm, but the latency factor needs to be considered.

Rahbari and Nickray [20] proposed a knapsack-based scheduling algorithm that is optimized by symbiotic organisms and includes CPU and network usages in the fitness function. The proposed scenario was tested using iFogSim simulator. Compared to FCFS and Ordinary

Knapsack techniques, it offers gains in energy consumption and total network usage.

Jamil et al. [21] introduced a new Fog computing scheduler that may support IoE (Internet of Experience) service provisioning and therefore reduce delay and network use. A use case was also conducted with the goal of optimizing the task scheduling from end devices on Fog nodes and successfully addressing tasks on available resources on each Fog node. Latency and energy consumption were used as performance indicators in their study. They used iFogSim simulator to evaluate the suggested scheduling method in comparison to existing well-known approaches. Their results showed that the proposed latency and network utilization are more improved compared to the FCFS technique.

In [22], authors investigated an immune scheduling network-based task scheduling strategy in a Fog-Cloud environment. The suggested method uses the capability of distributed schedulers to produce efficient strategies for dealing with overloaded computing nodes and minimizing task completion times. According to the results, the proposed method outperforms the other compared strategies. Recently, exploiting iFogSim simulator, Guerrero et al. in [24], focus on resource provisioning in Fog environments. They propose a lightweight decentralized service placement policy for performance optimization in Fog computing to optimize service placement and reduce latency and network usage.

However, Intelligent algorithms are suitable for complex problems with strong constraints and multiple objectives. They are highly scalable, but they are difficult to be applied to scenarios with high real-time requirements, such as distributed online problems. While the number of workflow tasks increases, the scheduling time using intelligent algorithms will also increase. Since the computing system has limited resources, such as computation, storage, bandwidth, and battery power, it is more realistic to give the scheduling optimization model with limited resources as constraints.

Pham and Huh [23] focused on task scheduling in Fog-Cloud systems in order to find a balance between task executing times and cloud resource budgets. In their investigation, a DAG is applied to define the entering workflow. They show the effectiveness of collaboration be-

tween Fog nodes and rented Cloud nodes to run large-scale offloading workloads for the end user.

Topcuoglu et al. [12] proposed HEFT and CPOP as list scheduling algorithms under precedence constraint and dependency between tasks. In the first phase, HEFT calculates the priority of the task and ranks them in ascending order. A task ranking depends on the upward rank (rank u). This value is calculated for each task based on the execution and communication cost task. Then the resource with the lowest completion time is chosen to execute the task. But this is without taking into consideration the resource computation costs when executing the allocated task. CPOP is in the same category as the previous one. It tries to reduce the execution time by targeting the critical path. The algorithm creates a list of tasks in priority order. In accordance with HEFT, CPOP follows the same while considering another variable, the downward rank of nodes. The sum of the upward rank (rank u) and downward rank (rank d) for each node determines the tasks that are part of the critical path. Thus, we will have two categories: critical tasks, and the other tasks. The critical tasks will be processed with priority before the other tasks. However, the scheduling length may further rise as a result of the fact that all critical tasks are assigned to the resource with high processing capacity, thereby causing load imbalance among the computing nodes.

Table 1 resumes the main related works with a focus on the strategy, the scenario used, and the limitations. Workflow scheduling is an NP-complete problem resource [25]. Therefore, our paper proposes a workflow scheduling algorithm in a heterogeneous Fog-Cloud infrastructure based on a list scheduling algorithm. It sorts the tasks by priority. Then it selects the highest to the lowest priority task from the tasks set to be scheduled. The determination of priority is related to task ranking but the resource allocation deals with the bandwidth rate (bandrate) of each task in all processing nodes with the aim to select the optimal computing node which provides the optimal band rate. BW-AWA can increase task scheduling effectiveness when it is used to schedule the lower bandrate task in the high Bandwidth computing resource in the Fog server. This algorithm can therefore enable to optimize the makespan in the Fog-Cloud computing system.

Table 1. Related Work

Authors	Algorithms	Applica- tion	Limitations
Bitten- court et al [15]	FCFS, Delay- Priority	Surveillance Application	<ul style="list-style-type: none"> • Unclear Task Rank- ing • High Execution Time
Zeng et al [17]	Mixed-Integer Non Linear	General Case	<ul style="list-style-type: none"> • Homogeneous Nodes • High Complexity • No Task Ranking
Bitam et al [18]	BLA	Healthcare System	<ul style="list-style-type: none"> • High Complexity • No Task Ranking
Nguyen et al [19]	Based Genetic Algorithm	General Dataset	<ul style="list-style-type: none"> • High Complexity • No Task Ranking
Jamil et al [21]	SJF	Healthcare System	<ul style="list-style-type: none"> • High Execution Time • No Task Ranking
Wang et al [22]	Immune Scheduling	E immune System	<ul style="list-style-type: none"> • No Task Ranking • High Execution Time
Rahbari et al [20]	Knapsac	Surveillance Application	<ul style="list-style-type: none"> • No Task Ranking • High Execution Time
Topcuoglu et al [12]	HEFT-CPOP	Task Graph	<ul style="list-style-type: none"> • The issue of this pa- per does not target Fog-Cloud.
Guerrero et al [24]	Popularity	Sock Shop Demo	<ul style="list-style-type: none"> • Pre-fixed Task Ranking • High Execution Time
Pham et al [23]	HEFT-based Algorithm	General Case	<ul style="list-style-type: none"> • No Delay-Sensitive
Our Proposed	Bw-AWA Algorithm	Monitoring Application	<ul style="list-style-type: none"> • Optimizing Makespan • Task Ranking

3 Proposed Approach

3.1 System Model

The Fog-Cloud environment model is composed of heterogeneous computational resources. We define the Processor Graph (\mathcal{PG}) as a Workflow graph that $\mathcal{PG} = \{Q, L\}$, where: $Q = \langle q_1, q_2, \dots, q_m \rangle$ represents all the computational nodes, including Cloud server, the set of Fog nodes, and IoT devices. L defines the interconnection between computational nodes (i.e., the interconnection between q_i and q_j , noted: $q_i \leftrightarrow q_j \in L$).

We define computational nodes: $\vec{Q}(q_j) = \{Q_C \cup Q_F \cup Q_D\}$, respectively the set of Cloud server, the set of Fog nodes, and the set of end devices. Cloud technology may provide high processing capabilities, but it suffers from high latency and real-time processing limitations. Fog Computing is a distributed computing model that collects data at the edge of the network and processes the data in real time. The benefits of Fog computing include reducing bandwidth usage, optimizing data performance, and reducing latency.

Each node ($q_i \in Q$) has the attributes below:

The computational capacity of network node in *MIPS* (Million Instructions Per Second).

The memory (Ram) capacity in *Megabyte*.

The communication links between resources have two bandwidth levels: $UpBw(Kb/s)$, and $DownBw(Kb/s)$, respectively, represent Upper and Down bandwidth for each node.

3.2 Application Model

The IoT application consists of dependents tasks as Workflow Graph: $\mathcal{TG} = (\mathcal{MP}, \mathcal{EP}, \vec{Vmp}, \vec{VEp})$, where \mathcal{MP} denotes the set of n dependents task nodes M_1, M_2, \dots, M_n , each node M_i representing a task in the Workflow graph. \mathcal{EP} denotes the set of k directed edges between task nodes $E_j = E_1, E_1, \dots, E_K$. Specifically, each task $M_i \in \mathcal{MP}$ must be assigned to one resource $q_j \in Q$ with respecting the dependency constraints. The set $Succ(M_i)$ denotes the successors of task (M_i). The set $Pred(M_i)$ denotes the predecessors of task (M_i). If $Pred(M_i) = \emptyset$, then the task is called an entry task, and the task without a successor is called an exit task. Each task $M_i \in \mathcal{MP}$ has a computation weight,

and Each edge $E_j \in \mathcal{EP}$ has a communication weight representing the amount of data between two tasks (M_i, M_k) .

3.3 Proposed Solution

The basic idea is to generate a task scheduling list by assigning a priority to the task in the workflow, sorting it according to the priority, and allocating each ranked task to the optimal resource which is determined based on the bandwidth factor. Below is the diagram (see Figure.1) of our proposed solution:

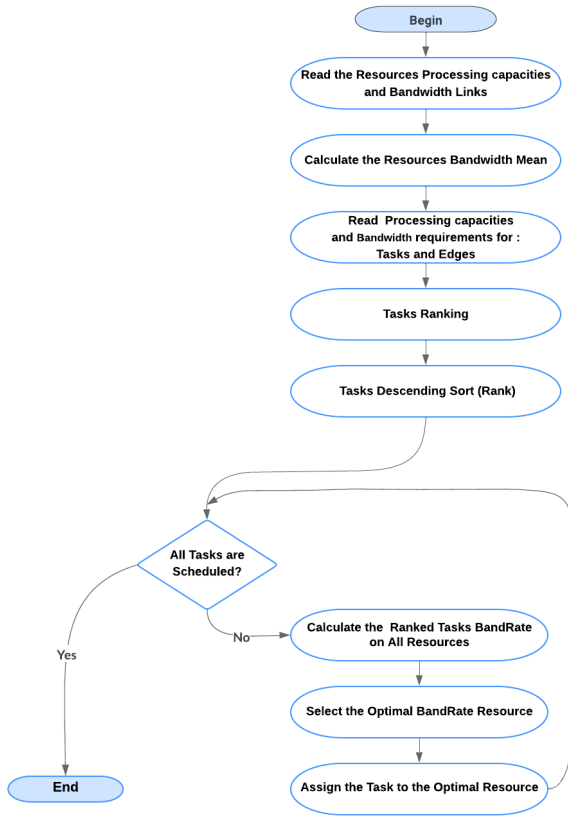


Figure 1. Our Proposed solution

Let $\overrightarrow{Vmp}(M_i)$ represent requirements of task node $M_i \in \mathcal{MP}$, and

$\overrightarrow{VEp}(E_j)$ is the characteristics for each edge in \mathcal{EP} .

For each task node $M_i \in \mathcal{MP}$, $\overrightarrow{Vmp}(M_i) = \langle Com_c(MIPS), Ram(Mb), \text{ and } mBW(Kb) \rangle$, respectively, representing the Computation cost, the memory required amount by the task, and the task node Bandwidth requirements. Tuples are the communication units between task nodes that represent the encapsulated $Data_{i,j}$ transferred between task nodes (i.e., between M_i and M_j). The IoT application receives the incoming tuples from the IoT Sensors which represent the Workflow graph source.

$$\overrightarrow{VEp}(E_j) = \langle T_{cc}(MIPS), BwE(Kb) \rangle,$$

where T_{cc} represent the Communication cost, and $BwE(M_i, M_j)$ is the bandwidth levels of E_j (M_i, M_j).

3.3.1 Task Ranking Phase

The Rank of each task is determined by three attributes computation: cost Com_c , encapsulated data amount in Tuple as communication cost T_{cc} (see Eq(1)), and predecessor task ranking $Rank(M_j)$ [26].

$$Rank(M_i) = \begin{cases} M_i, & \text{If } M_i = M_{exit} \\ Com_c(M_i) + \min_{M_j \in pred(M_i)} (T_{cc}(M_j) + Rank(M_j)), & \text{otherwise} \end{cases} \quad (1)$$

We define $Com_c(M_i)$ as the processing cost needed for the task M_i , and $T_{cc}(M_j)$ represents the data encapsulated amount in the Tuple (out) from the direct predecessor M_j to M_i , where the rank of the direct predecessor task is $Rank(M_j)$, and the rank value related to the exit task is $M_i = M_{exit}$.

3.3.2 Task Sorting Phase and Allocation Resources

In this phase, all priority of tasks is calculated and arranged in descending order from high to low. The task scheduling phase selects the task with respect to dependency according to the task rank list. The tasks will be unstacked one by one. The algorithm will scan the resources set for each task to find the resource that will optimize the bandwidth rate. The computing resources have a double link (full-duplex). Let

us note that the set of ($Mean_{Bw}$) represents the Average bandwidth of nodes in the Fog-Cloud architecture as shown in Eq(2):

$$Mean_{Bw}(q_k) = [upBw(q_k) + dwBw(q_k)]/2. \quad (2)$$

For each ranked task M_i , the algorithm uses the set mBW to calculate the bandwidth $BandRate$ on all the set of Average bandwidth $Mean_{Bw}(q_k)$, then it will scan all the resources to find the node q_k that minimizes $BandRate$ of the task M_i (see Eq(3)):

$$BandRate(M_i, q_k) = mBW(M_i)/Mean_{Bw}(q_k). \quad (3)$$

3.3.3 Algorithm

Algorithm 1 represents the Bw-AWA Algorithm, which allocates each ranked task respecting the priority order and chooses the best resource to execute the current task in terms of bandwidth metrics.

Algorithm 1.

Input: \mathcal{TG} (Tasks Graph), \mathcal{RG} (Resources Graph).

Output: A mapped tasks to optimal resources.

Initialisation: $Order_{list} = \emptyset, Mean(q_k) = \emptyset, bandRate(M_i, q_k) = \emptyset$.

Read $Com_c(M_i)$ and $T_{cc}(M_i)$

Calculate $rank(M_i)$ for all tasks (see Algorithm 2)

Order tasks: Place tasks in descending order in $Order_{list}$

While non-placed task in $Order_{list}$ {

Select the first task from the $Order_{list}$

$q = q_1$

Assign task M_i to the resource q

For Each $q_k \in Q - \{q_1\}$

{

$Mean(q_k) = [upBw(q_k) + dwBw(q_k)]/2$

$bandRate(M_i, q_k) = mBW(q_k)/Mean(q_k)$

IF $bandRate(M_i, q_k) < bandRate(M_i, q_{k-1})$ and q_k is Empty

{ $q = q_k$

Assign current task M_i to the resource q_k }

}

Remove task M_i from $Order_{list}$ }

The ranking phase is represented in the Algorithm 2, with the aim to order the tasks according to the rank calculated in descending order.

Algorithm 2.

Input: $Com_c(M_i)$, $T_{cc}(M_i)$
Output: Ranked tasks
Initialisation: $M_i = Task_{exit}$
Rank $(M_i) = Com_c(M_i) + \min_{M_j \in pred(M_i)} (T_{cc}(M_j) + Rank(M_j))$

4 Use Case and Experimental Results

4.1 Workflow System for Smart Monitoring

This scenario is based on a dispersed network of security, manufacturing, transportation, and healthcare surveillance cameras [27].

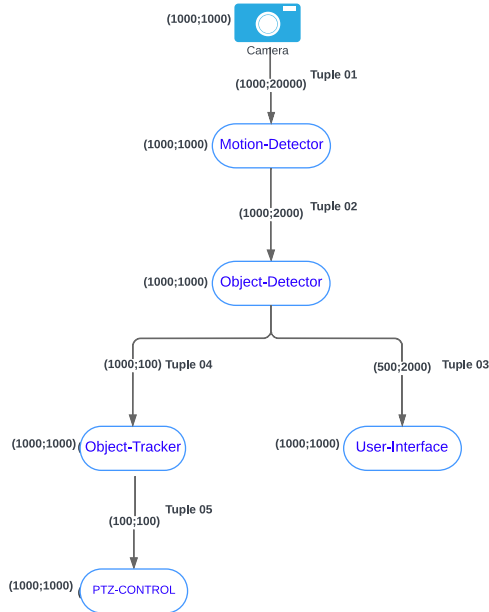


Figure 2. A Monitoring Workflow Graph

The tasks are dependent on the positioning of an IoT application in the Fog-Cloud environment. The application consists of five tasks, in which each Fog device will be in charge of a security camera number. Using the raw video stream (Tuple 01), the Motion-Detector locates any motion. The motion video stream (Tuple 02) is then sent on to the Object-Detector, which compares the images and finds the moving object. Following moving object recognition, the discovered object is delivered to the User-Interface (Tuple 03) or to an Object Tracker for location monitoring (Tuple 04). The Object-Tracker sends a location to the PTZ (Pan Tilt Zoom)-Control (Tuple 05), which serves as an actuator to turn the camera there. Tasks and Edges have two main attributes (processing requirements and bandwidth requirements) respectively (see Figure 2).

The system consists of IoT Devices, gateways (Fog nodes), and a Cloud platform. The system configuration parameters used for the monitoring case are presented in Table 2 [27]. The latency between

Table 2. Configuration Details in the System

Type	MIPS	RAM	Up BW	Link	Down Link BW
Cloud	44800	40000	100		10000
Proxy	2800	4000	10000		10000
Gateway	2800	4000	10000		10000
IoT devices	500	1000	10000		10000

network resources is 2, 4, and 100 (ms) respectively between [IoT Devices-Gateway], [Gateway-Proxy], and [Proxy-Cloud]. iFogSim simulator [27] is used to simulate the proposed scenario. Table 3 represents the task ranking phase (see Eq (1)) with the priority order.

The MIPS Rate ($MIPS_{Rate}$) values of the application tasks in different network resources are calculated using Eq (4) and presented in Table 4:

$$MIPS_{Rate} = MIPS_{Task} / MIPS_{resource}. \quad (4)$$

This parameter is used in HEFT and CPOP algorithms in the task allocation phase.

Table 3. Task ranking

Task	ranking	order
motion detector	7000	1
object detector	3000	2
object tracker	1000	3
user interface	1000	4

Table 4. MIPS Rate Of All Tasks In different network resources

Network Resources	$MIPS_{Rate}$
CLOUD Device	0.02336
Proxy	0.3571
Gateway	0.3571
CAMERA Devices	2.0

Table 5 contains two main parameters used in our proposed algorithm. The first one is the bandwidth Mean of different kinds of resources (see Eq 2). The bandwidth Mean result is used to calculate the second parameter which is the BandRate (see Eq 3).

Table 5. Bandwidth Rate Of All Tasks In different network resources

resource	$Mean_{Bw}$	$bandRate$
CLOUD Device	5050.0	0.198
Proxy	10000.0	0.1
Gateway	10000.0	0.1
Camera Devices	10000.0	0.1

4.2 Experimental Results

4.2.1 Variation in Number of Fog Nodes and Cameras

In our case study, the application has three different settings:

1. The simulation is based on four configurations: (1,4,7), (2,4,12), (3,4,17), and (4,4,22). Each configuration contains three parameters representing, respectively, Fog devices, camera devices, and the total resources including one proxy and one Cloud server.
2. The number of IoT devices (refer to the camera devices) related to each Fog node is augmented to 8,10,12,16.
3. The number of Fog nodes is augmented to 10,20,30,40 nodes.

4.2.2 Makespan

This is the value given by the total execution time. It is calculated based on the simulation start time and the simulation end time. The simulator's executive clock is used to calculate this value in milliseconds [28](see Eq 5):

$$Makespan = \sum (finish(Mi)) - \sum (start(Mi)), \quad (5)$$

where the function $finish(Mi)$ gives the completion time related to the last task in the workflow and $start(Mi)$ gives the start time of the first task scheduled. Our work is compared to HEFT, CPOP [12], Popularity [24], and FCFS [15] strategies. Firstly, we compare BW-AWA with HEFT and CPOP well-known priority scheduling algorithms that aim to reduce the Makespan for scheduled tasks. The approach HEFT is used as an algorithm in the Cloud-Fog-IoT infrastructure [29], it is a placement algorithm for heterogeneous systems. However, in Bw-AWA, the determination of priority is also related to task ranking but the resource allocation deals with the rated bandwidth of each task in all processing nodes to select the optimal processing node. In the algorithm [24], the tasks are placed along the shortest network path between the user and the cloud provider, and priority should be given when allocating an application's interconnected tasks. The topological arrangement of the tasks determined the placement order, putting the initial tasks with the pre-fixed priority nearer to the Fog devices. The figures below show the impact of changing the number of resources on Makespan (milliseconds). In the graphics, we used a logarithmic scale on the Y-axis to highlight the differences between the strategies. On the X-axis, we have taken four configurations. The time taken

by an algorithm for the execution of the tasks is crucial in choosing the algorithm. Figures 3, 4, and 5 demonstrate the simulation results graphs in varied configurations.

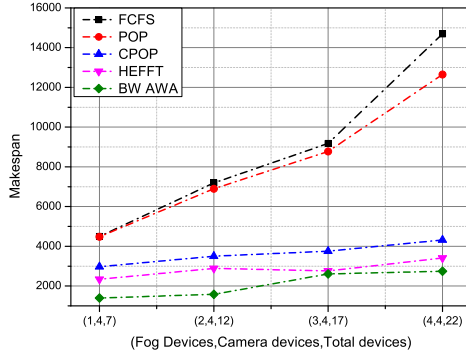


Figure 3. Makespan Results (First Configuration)

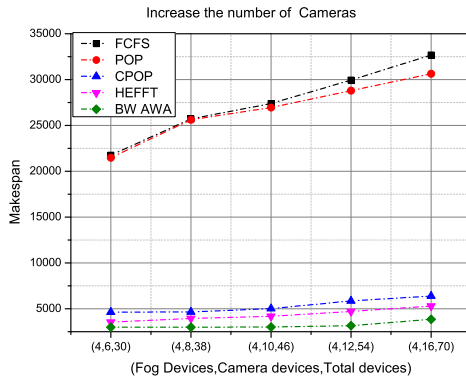


Figure 4. Increase IoT Devices Number

Differences among the five algorithms are shown in Tables 6 and 7. The BW-AWA algorithm results outperform other approaches for the following reasons:

Firstly, BW-AWA leverages Fog devices as processing nodes for allocating resources. When the devices number increases, the Makespan

Table 6. Makespan results

Figure	Config	FCFS	Popularity	HEFT	CPOP	BW-AWA
Figure 3	(1,4,7)	4494	4482	2340	2973	1396
	(2,4,12)	7193	6894	2886	3502	1580
	(3,4,17)	9179	8772	2758	3750	2607
	(4,4,22)	14694	12644	3408	4319	2746
Figure 4	(4,6,30)	21756	21483	3545	4629	2984
	(4,8,38)	25712	25624	3945	4661	2998
	(4,10,46)	27394	26945	4170	5024	3005
	(4,12,54)	29942	28792	4716	5855	3153
	(4,16,70)	32691	30633	5292	6382	3858
Figure 5	(10,4,52)	28558	20732	5022	8790	3195
	(20,4,102)	47912	41405	7566	9354	5645
	(30,4,152)	71519	66949	11219	11683	8003
	(40,4,202)	94456	85983	13239	14597	10714

Table 7. Improvement ratio of Makespan of the proposed solution

Figure	Config	FCFS	Popularity	HEFT	CPOP
Figure 3	(1,4,7)	68,94%	68,85%	40,34%	53,04%
	(2,4,12)	78,03%	77,08%	45,25%	54,88%
	(3,4,17)	71,60%	70,28%	5,47%	30,48%
	(4,4,22)	81,31 %	78,28%	19,42%	36,42%
Figure 4	(4,6,30)	86,28 %	86,11%	15,83 %	35,54%
	(4,8,38)	88,34 %	88,30 %	24,01 %	35,68%
	(4,10,46)	89,03 %	88,85 %	27,94 %	40,19%
	(4,12,54)	89,47 %	89,05 %	33,14 %	46,15%
	(4,16,70)	88,20 %	87,41 %	27,10 %	39,55%
Figure 5	(10,4,52)	88,81%	84,59 %	36,38 %	63,65 %
	(20,4,102)	88,22 %	86,37 %	25,39 %	39,65 %
	(30,4,152)	88,81 %	88,05 %	28,67 %	31,50%
	(40,4,202)	88,66 %	87,54%	19,07 %	26,60 %

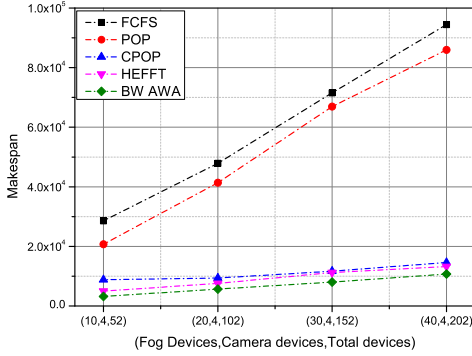


Figure 5. Increase Fog nodes Number

for all approaches increases, and it always remains reduced compared to the other four approaches. Secondly, according to Table 5, BW-AWA considers the task's band rate, which is resided in the gateway devices. As a result, extra communication time to the Cloud nodes is reduced. Because the resources with huge bandwidth could finish their work quickly, other resources with lower Bandwidth would still operate for a longer period of time when the disparity between resources is large. So, the proposed algorithm saves time more when there are more Fog devices.

HEFT and CPOP are almost similar in that they rely on processing for placement. These processing algorithms will tend to place on powerful nodes according to the MIPS Rate values in Table 4. In this case, we will have massive use of the Cloud, which has a high latency of 100 ms, but it is necessary to consider nearby resources in the allocation phase. Although CPOP and HEFT have the same processing complexity, CPOP processing time exceeds that of HEFT. Because it performs another pass to calculate rank downward tasks, it tries to reduce the critical path impact of the graph.

FCFS method processes tasks in the order they are entered, regardless of the length or size of the task. It is difficult to reduce the scheduling time. The results show that as the number of IoT devices increases, the Makespan increases. Popularity presents a priority rule

in which the most popular tasks are placed first in the devices closest to the end-users. Furthermore, Popularity assigns also to upper levels despite the Fog device’s capacities and the task’s requirements. Popular tasks had reduced latency, while less popular ones saw greater lag.

4.2.3 Throughput

Figures 6,7, and 8 show a comparison of throughput rates. The total number of tasks completed successfully in a certain time period is referred to as Throughput [30].

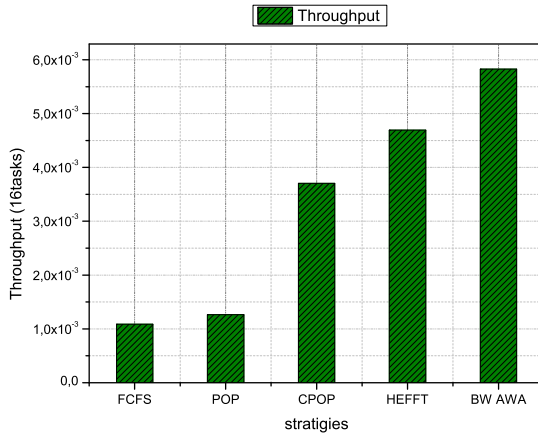


Figure 6. 16 Tasks

It is proportional to the number of tasks locally and remotely allocated based on the number of camera devices generating the task.

The effectiveness of scheduling approaches is demonstrated by using Eq 6:

$$Throughput = Tasks\ Number / Makespan. \tag{6}$$

The findings show that, when the task number rises, the throughput rate rises as well. When a rate is higher, it improves the execution of the tasks and it also reduces the makespan [30]. The throughput of our proposed approach is compared to that of the other chosen techniques. As a result, it provides better results. Three configurations, with 16,40,

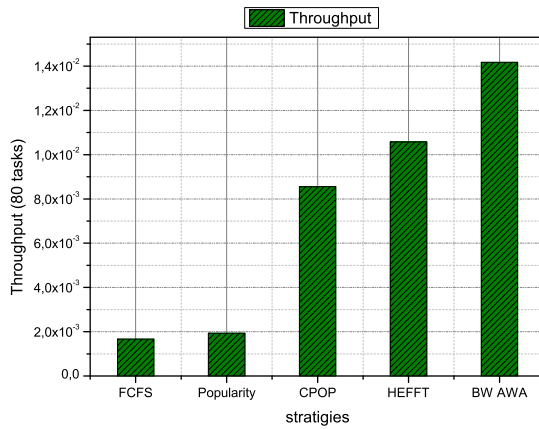


Figure 7. (80 Tasks)

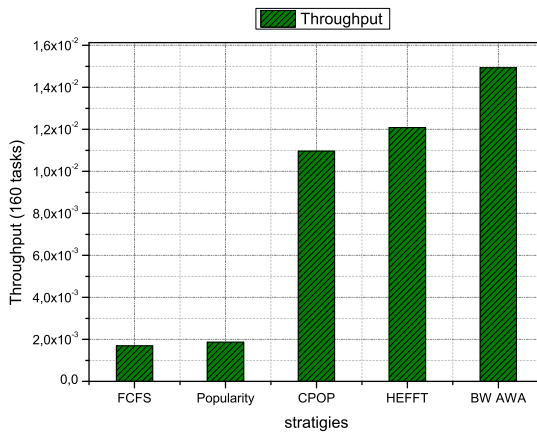


Figure 8. (160 Tasks)

and 160 tasks, respectively, were used to analyze throughput. As expected, the results show that the throughput increases as the number of processing nodes increases (202 total resources used in 160 tasks). Along with this, as the number of tasks increases, the throughput also increases.

We can conclude that the proposed algorithm BW-AWA can increase task scheduling effectiveness based on the experimental results mentioned above. This algorithm can therefore enable to reduce the makespan in the Fog-Cloud computing system.

5 Conclusion and Future Work

Fog Computing is an emerging computing paradigm that supports and unleashes the full reach of IoT and solves the limitations of the Cloud. In order to improve the processing node's efficiency and optimize performance indicators, the optimizing task scheduling problem in a Fog-Cloud environment has taken a lot of intentions by researchers. In this paper, we have dealt with the problem of workflow application task scheduling in a Fog-Cloud environment, focusing on the makespan optimization which is the workflow execution time. The proposed list scheduling algorithm (BW-AWA) takes into consideration task priority, communication cost between tasks, and processor choice in terms of bandwidth over the entire schedule. The comparative study between the results obtained from our approach and those of the selected algorithms clearly shows that our proposed algorithm provides efficient and effective solutions in terms of Makespan with the task priority constraint. In order to improve our approach, we intend in our future work to improve our algorithm to support other QoS (Quality of Service) metrics, such as budget constraints and power consumption through the processor Dynamic Voltage and Frequency Scaling (DVFS) technique, and subsequently formally move from a single-objective optimization problem to a multi-objective optimization problem.

References

- [1] V. G. Skobelev and V. V. Skobelev, "A general analytical model of queuing system for Internet of Things applications," *Computer*

- Science Journal of Moldova*, vol. 28, no 1(82), pp. 3–20, 2020.
- [2] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, “Zenith: Utility-Aware Resource Allocation for Edge Computing,” in *2017 IEEE International Conference on Edge Computing (EDGE)*. Honolulu, HI, USA: IEEE, 2017, pp. 47–54. DOI: 10.1109/IEEE.EDGE.2017.15.
 - [3] J. K. Zao et al., “Augmented Brain Computer Interaction Based on Fog Computing and Linked Data,” in *2014 International Conference on Intelligent Environments*. Shanghai, China: IEEE, 2014, pp. 374–377. DOI: 10.1109/IE.2014.54.
 - [4] N. Verba et al., “Modeling industry 4.0 based Fog computing environments for application analysis and deployment,” *Future Generation Computer Systems*, vol. 91, pp. 48–60, 2019.
 - [5] J. Wan, B. Chen, S. Wang, M. Xia, D. Li, and C. Liu, “Fog Computing for Energy-Aware Load Balancing and Scheduling in Smart Factory,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4548–4556, Oct. 2018. DOI: 10.1109/TII.2018.2818932.
 - [6] R. Yayla, E. Albayrak, U. Yüzgeç, “Vehicle Detection from Unmanned Aerial Images with Deep Mask R-CNN,” *Computer Science Journal of Moldova*, vol. 30, no 2(89), pp. 148–169, 2022.
 - [7] D. Ye, M. Wu, S. Tang, and R. Yu, “Scalable Fog Computing with Service Offloading in Bus Networks,” in *2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud)*. Beijing, China: IEEE, 2016, pp. 247–251. DOI: 10.1109/CSCloud.2016.34.
 - [8] X. Wang, Z. Ning, and L. Wang, “Offloading in Internet of Vehicles: A Fog-Enabled Real-Time Traffic Management System,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4568–4578, Oct. 2018. DOI: 10.1109/TII.2018.2816590.
 - [9] R. K. Lomotey, J. Pry, S. Sriramoju, “Wearable IoT data stream traceability in a distributed health information system,” *Pervasive and Mobile Computing*, vol. 40, pp. 692–707, 2017.
 - [10] S. E. Mahmoodi, R. N. Uma, and K. P. Subbalakshmi, “Optimal Joint Scheduling and Cloud Offloading for Mobile Applications,” *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 301–313, 2019. DOI: 10.1109/TCC.2016.2560808.

- [11] S. Sharma, H. Saini, “A novel four-tier architecture for delay aware scheduling and load balancing in Fog environment,” *Sustainable Computing: Informatics and Systems*, vol. 24, Article no. 100355, 2019.
- [12] H. Topcuoglu, S. Hariri, and Min-You Wu, “Performance-effective and low-complexity task scheduling for heterogeneous computing,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260–274, March 2002. DOI: 10.1109/71.993206.
- [13] A. Agarwal and P. Kumar, “Economical Duplication Based Task Scheduling for Heterogeneous and Homogeneous Computing Systems,” in *2009 IEEE International Advance Computing Conference*. Patiala, India: IEEE, 2009, pp. 87–93. DOI: 10.1109/IADCC.2009.4808986.
- [14] X. Tang, K. Li, G. Liao, R. Li, “List scheduling with duplication for heterogeneous computing systems,” *Journal of parallel and distributed computing*, vol. 70, no. 4, pp. 323–329, 2010.
- [15] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, “Mobility-Aware Application Scheduling in Fog Computing,” *IEEE Cloud Computing*, vol. 4, no. 2, pp. 26–35, 2017. DOI: 10.1109/MCC.2017.27.
- [16] P. Varshney and Y. Simmhan, “Characterizing application scheduling on edge, fog, and cloud computing resources,” *Software: Practice and Experience*, vol. 50, no. 5, pp. 558–595, 2020.
- [17] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, “Joint Optimization of Task Scheduling and Image Placement in Fog Computing Supported Software-Defined Embedded System,” *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3702–3712, 2016. DOI: 10.1109/TC.2016.2536019.
- [18] S. Bitam, S. Zeadally, and A. Mellouk, “Fog computing job scheduling optimization based on bees swarm,” *Enterprise Information Systems*, vol. 12, no. 4, pp. 373–397, 2018.
- [19] B. M. Nguyen, H. T. T. Binh, T. T. Anh, and D. B. Son, “Evolutionary algorithms to optimize task scheduling problem for the IoT based bag-of-tasks application in Cloud–Fog computing environment,” *Applied Sciences*, vol. 9, no. 9, Article no. 1730, 2019. <https://doi.org/10.3390/app9091730>.

- [20] D. Rahbari and M. Nickray, "Scheduling of fog networks with optimized knapsack by symbiotic organisms search," in *2017 21st Conference of Open Innovations Association (FRUCT)*. Helsinki, Finland: IEEE, 2017, pp. 278–283. DOI: 10.23919/FRUCT.2017.8250193.
- [21] B. Jamil, M. Shojafar, I. Ahmed, A. Ullah, K. Munir, H. Ijaz, "A job scheduling algorithm for delay and performance optimization in Fog computing," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 7, Article ID. e5581, 2020.
- [22] Y. Wang, C. Guo, and J. Yu, "Immune scheduling network based method for task scheduling in decentralized Fog computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID. 2734219, 2018. <https://doi.org/10.1155/2018/2734219>.
- [23] X.-Q. Pham and E.-N. Huh, "Towards task scheduling in a cloud-fog computing system," in *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. Kanazawa, Japan: IEEE, 2016, pp. 1–4. DOI: 10.1109/APNOMS.2016.7737240.
- [24] C. Guerrero, I. Lera, and C. Juiz, "A lightweight decentralized service placement policy for performance optimization in Fog computing," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 6, pp. 2435–2452, 2019.
- [25] A. Brogi, S. Forti, C. Guerrero, and I. Lera, "How to place your apps in the Fog: State of the art and open challenges," *Software: Practice and Experience*, vol. 50, no. 5, pp. 719–740, 2020.
- [26] M. Sulaiman, Z. Halim, M. Waqas, and Doğan Aydın, "A hybrid list-based task scheduling scheme for heterogeneous computing," *The Journal of Supercomputing*, vol. 77, pp. 10252–10288, 2021.
- [27] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments," *Software: Practice and Experience*, vol. 47, no 9, pp. 1275–1296, 2017.
- [28] L. Hamid, A. Jadoon, and H. Asghar, "Comparative analysis of task level heuristic scheduling algorithms in Cloud computing," *The Journal of Supercomputing*, vol. 78, pp. 12931–12949, 2022.

- [29] X.-Q. Pham, N. D. Man, N. D. T. Tri, N. Q. Thai, and E.-nam Huh, “A cost-and performance-effective approach for task scheduling based on collaboration between Cloud and Fog computing,” *International Journal of Distributed Sensor Networks*, vol. 13, no. 11, Article no. 1550147717742073, 2017.
- [30] M. Aljarah, M. Shurman, and S. Alnabelsi, “Cooperative hierarchical based Edge-computing approach for resources allocation of distributed mobile and IoT applications,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no 1, pp. 296–307, 2020.

Bentabet Dougani, Abdeslem Dennai,

Received December 24, 2022

Revised January 25, 2023

Accepted January 28, 2023

Bentabet Dougani

ORCID: <https://orcid.org/0000-0003-4295-3830>

CCAI Research Team, SGRE Laboratory,

University of BECHAR, Algeria

E-mail: dougani.bentabet@univ-bechar.dz

douganibentabetinfo@gmail.com

Abdeslem Dennai

ORCID: <https://orcid.org/0000-0003-2133-0765>

CCAI Research Team Leader, SGRE Laboratory,

University of BECHAR, Algeria

E-mail: dennai.abdeslam@univ-bechar.dz

A Neural Attention-Based Encoder-Decoder Approach for English to Bangla Translation

Abdullah Al Shiam, Sadi Md. Redwan, Md. Humaun Kabir,
Jungpil Shin

Abstract

Machine translation (MT) is the process of translating text from one language to another using bilingual data sets and grammatical rules. Recent works in the field of MT have popularized sequence-to-sequence models leveraging neural attention and deep learning. The success of neural attention models is yet to be construed into a robust framework for automated English-to-Bangla translation due to a lack of a comprehensive dataset that encompasses the diverse vocabulary of the Bangla language. In this study, we have proposed an English-to-Bangla MT system using an encoder-decoder attention model using the CCMatrix corpus. Our method shows that this model can outperform traditional SMT and RBMT models with a Bilingual Evaluation Understudy (BLEU) score of 15.68 despite being constrained by the limited vocabulary of the corpus. We hypothesize that this model can be used successfully for state-of-the-art machine translation with a more diverse and accurate dataset. This work can be extended further to incorporate several newer datasets using transfer learning techniques.

Keywords: Neural Machine Translation (NMT), Machine Translation (MT), Encoder-Decoder Model, Neural Attention.

ACM CCS 2020: Computing methodologies-Artificial intelligence-Machine learning.

MSC 2020: 68T50.

1 Introduction

Bangla is one of the most common languages spoken worldwide, with approximately 300 million native speakers and another 37 million as second language speakers. However, reliable machine translation (MT) systems for the language have not been implemented until very recently. Machine Translation (MT) is the translation of text from one natural language (source language) to another language (target language) using a computerized system with or without human interaction [1]. MT is an automated system associated with Natural Language Processing (NLP), which uses other language resources and bilingual datasets to build language and phrase models for text translation. Ideally, MT is a batch process that is applied to a given text to produce a perfect translated text [2]. The aim is to fill the communication gap between different societies with language diversity. Manual human translation is time-consuming for any language. But an efficient MT system can reduce both the time and cost involved in the translation. English being the language of choice internationally, is used in most documents, papers, journals, books, and records in today's world. Subsequently, MT systems like google translate have been popularized by most native Bangla speakers for English-to-Bangla translation. Since there is an abundance of comprehensive English articles online, translation of Bangla to English is much more accurate compared to translating English to Bangla. This is due to the fact that most Bangla translations available online are not refined enough for an NLP-based approach that can capture the nuance and subtlety of the language. The Bangla language has some differences from English and some other languages since Bangla has a large and diverse vocabulary, while the exact words can have different contextual meanings. It is multi-disciplinary research to facilitate machine translation systems to capture the contextual meaning of a sentence while translating to other languages. Hence it is essential for researchers to study the literature describing the differences between specific language pairs to explain the critical mistakes made by the systems and optimize them accordingly. The neural machine translation (NMT) approach for English-to-Bangla translation has been proposed in a recent study [3]. Using the SUPara and Glob-

alVoices corpus, it achieved a BLEU score of +0.30 and +0.69 over previously implemented MT systems. This study has shown NMT to be more efficient and accurate compared to PBMT (phrase-based MT) systems such as shu-torjoma [4] and other SMT (statistical MT) systems [5]. Taking these findings into account, we aim to demonstrate the potential adequacy of NMT systems using a larger dataset, namely the CCMatrix dataset [6],[7]. The proposed encoder-decoder model trained with the CCMatrix dataset has achieved a BLEU score of 15.68 and a NIST (National Institute of Standards and Technology) score of 5.12 in English-to-Bangla translation.

2 Related Works

Traditional MT systems include Direct-Based MT systems that translate individual words in a sentence at a time from one language to another using a phrasebook. Corpus-Based MT relies on the study of bilingual text corpora. Statistical MT (SMT) and Example-based MT fall under this category. SMT is good for catching exclusions to rules. The primary advantage of the SMT is that it does not require philological information in the translation process. Knowledge-Based MT, on the other hand, requires to be formed based on ontology and the semantic web. Lexical-Based MT systems translate individual words with lexical information [8]. The encoder-decoder model has emerged relatively recently and has been successful in many state-of-the-art translation frameworks [9]. With the growing popularity of machine learning models, NMT has been established as the new baseline for MT systems. Contemporary NMT systems have been modernized by Google [10] with phrase alignment [11] and attention mechanisms [12]. A comparison of NMT and SMT systems in recent bilingual studies is given in Table 1.

The first notable Bangla-to-English MT system is the phrase-based MT method proposed in 2010 [15]. Rule-based machine translation (RBMT) was first proposed in a 2012 study to include assertive-affirmative, negative, and interrogative sentences [16]. In the following year, a tense-based (TBMT) system has been proposed [17]. Recurrent neural networks have also seen some success in English-to-Bangla translation over the years [18]. [19] used the neural encoder-decoder

Table 1. Comparison of NMT and SMT Systems for English to other Languages [13],[14]

Language	System	BLEU
DE (German)	SMT, NMT	41.5, 61.2
EL (Greek)	SMT, NMT	47.0, 56.6
PT (Portuguese)	SMT, NMT	57.0, 59.9
PT (Portuguese)	SMT, NMT	41.9, 57.3

model for text normalization. On the basis of recent deep learning work, [20] proposed a bidirectional encoder-decoder model for addressing the problem of Arabic NER, in which the encoder and decoder are bidirectional LSTMs. Character-level embeddings are used in addition to word-level embeddings, and they are combined via an embedding-level attention mechanism. [21] proposes a novel Multimodal Encoder-Decoder Attention Networks (MEDAN). The MEDAN is composed of cascaded Multimodal Encoder-Decoder Attention (MEDA) layers that can capture rich and reasonable question features as well as image features by associating question keywords with important object regions in the image. In conclusion, NMT has become the cornerstone of most recent works on Bangla-English translation [3],[22]. Some of the relevant studies are discussed in Table 2.

3 Corpus Data

3.1 Dataset Description

We obtain the CCMatrix dataset from OPUS [13]. CCMatrix is built in a multilingual sentence space using a margin-based bitext mining technique, resulting in many parallel sentences and tokens in different languages. Training NMT systems for multiple language pairs was used to assess the quality of the mined bitexts. English-Bangla paral-

Table 2. Novel Bangla-English MT Studies

System	Year	Author
Phrase-Based MT	2010	Islam Z et al. [23]
RBMT	2012	Rhaman MK et al. [24]
TBMT	2013	Muntarina K et al. [25]
LSTM-RNN	2019	Islam MS et al. [16]
NMT	2019	Hasan MA et al. [26]
RNN	2020	Siddique S et al. [27]
Attention-based NMT	2021	Abujar S et al. [17]

lel datasets with more than 10M combined tokens available in OPUS and two other corpora relevant to this work are shown in Table 3 for comparison. The CCMatrix dataset is chosen in this work since the dataset is entirely built using the available sentence pairs on the internet, and it is not curated by human experts. The rationale behind choosing the dataset is to enable the MT system to access a wide range of vocabulary without being constrained by a specific corpus or human expertise. Another justification is that the CCMatrix dataset is also one of the most extensive datasets in terms of the number of tokens, as shown in Table 3.

Table 3. Corpus Data

Corpus	Bn Tokens	En Tokens
WikiMatrix v1	35.3M	1000M
wikimedia v2021040	10.3M	349.2M
CCMatrix v1	88.6M	98.7M
CCAligned v1	37.8M	38.9M
Tanzil v1	6.1M	5.6M
GlobalVoices	3.3M	4.9M
SUPara	244K	202K

3.2 Data Pre-processing

After downloading the dataset, the raw text data is preprocessed in several steps as shown in Figure 1. First of all, we perform Unicode normalization and split the punctuations. Then we add a ‘start’ and ‘end’ token to each sentence. Then each sentence is cleaned by removing special characters (if any). Then a word index is created and reversed so that each ‘token id’ points to a unique word.

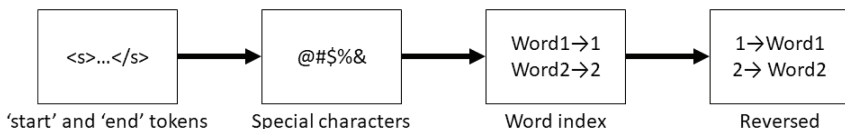


Figure 1. Preprocessing of the raw text

Traditional encoding schemes include sub-word regularization [28] and byte pair encoding (BPE) [29], which have been used in some NMT systems in the past. However, these encoding methods have been shown to be suboptimal for pretraining in some cases [30]. As an alternative, we vectorize the text data after the tokenization step in this work. Vectorization refers to transforming the strings into an array of token indices by using the Tensorflow [31] TextVectorization layer. This layer implements an ‘adapt’ method that reads the input epoch-by-epoch, much like model training. Importantly, the final dictionary of tokens and words is also used to decode the output of the model.

4 Proposed Framework

4.1 Encoder-Decoder Model

The encoder-decoder model operates such that the conditional probability of the target sentence given the source sentence is maximized. The encoder transforms the input phrase sequence into a dense vector form, and the decoder takes that representation and converts it into subsequent word sequences [23]. This leads to the model’s performance being constrained by the maximum sentence length in training data.

Any sentence longer than the maximum length may lead to inaccurate translation. The neural attention mechanism is used to optimize this model with more flexibility. Instead of directly encoding the input sentence into a fixed-length vector, this method converts it into a sequence of vectors. When translating the encoded text, a subset of these vectors is chosen by employing this mechanism [24],[25], as shown in Figure 2.

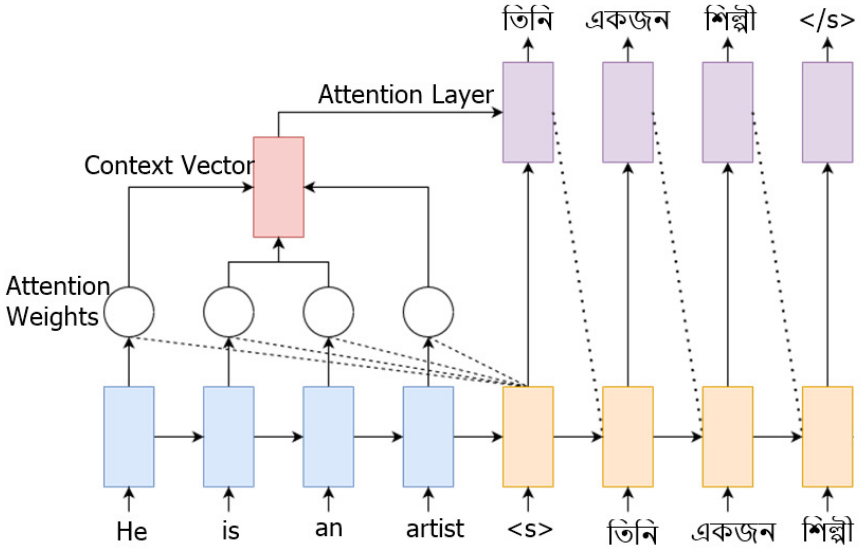


Figure 2. Neural Attention-Based Encoder-Decoder

The encoder part of the model is a Gated Recurrent Unit (GRU). This can also be implemented using bidirectional LSTM. The GRU is initialized with Glorot uniform and acts as the encoder combined with an embedding layer. The uniform distribution $U[-a,a]$ is defined as follows.

$$a = \sqrt{\frac{6}{n_{in} + n_{out}}}, \quad (1)$$

where n_{in} is the number of input neurons in the weight tensor, and n_{out} is the number of output neurons in the weight tensor.

4.2 Neural Attention

The attention or alignment mechanism for the encoder-decoder model retains all the input sentence's hidden states during the decoding phase. The attention model proposed by Bahdanau et al. used in this work produces hidden states for each of the elements in the input sequence from the encoder. The alignment score of each encoder output with respect to the decoder inputs and hidden state is then calculated at each step by multiplying the decoder's hidden state by all of the encoder's hidden states defined as follows.

$$e^t = [s_t^T h_1, \dots, s_t^T h_{T_x}]. \quad (2)$$

Then the probability distribution is calculated as follows.

$$\alpha^t = \text{softmax}(e^t). \quad (3)$$

This gives the context vector as follows

$$a_t = \sum_{i=1}^{T_x} \alpha_i^t h_i. \quad (4)$$

The decoder uses this context vector to generate new hidden states. The decoder produces a tensor, which is then passed through a text vectorization layer to produce the final output.

4.3 Experimental Setup

The encoder-decoder model is implemented using Keras [27] with an embedding dimension of 256 and 1024 units in each layer. The encoder uses the Embedding layer to produce an embedding vector for each token in an input array of tokens and then transforms the vectors using the GRU layer. The processed sequence is passed as the attention inputs, implemented by the Additive Attention layer. The decoder does the exact same thing and uses the output of the GRU layer as the 'query' to the attention layer. The model is trained using an RTX 2060 GPU with Nvidia CUDA 11.2. The loss function used for training is Sparse Categorical Cross-entropy, and the optimizer is the Adam

optimizer [32]. The total number of trainable parameters is 41,650,940. In this paper, the model is trained for 30 epochs and tested on the Tatoeba dataset [33].

5 Experimental Results and Discussion

The model training results are shown in Figure 3.

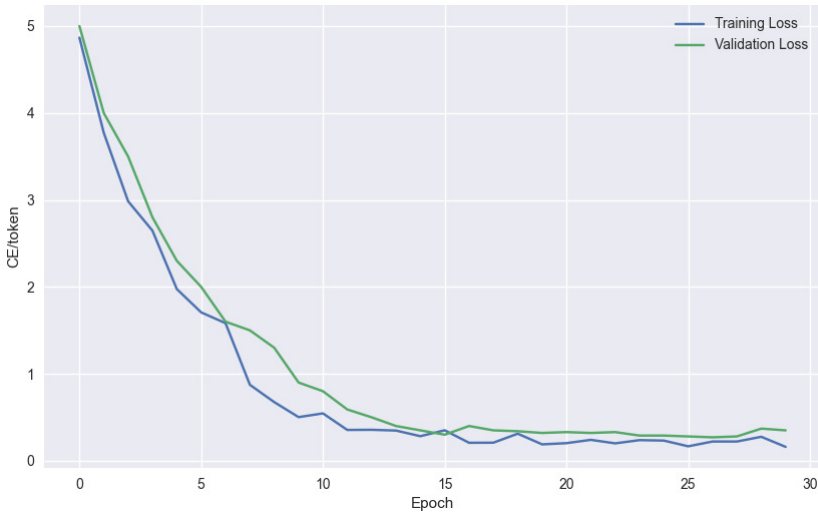


Figure 3. Training and validation loss curves during the model training. CE/token denotes cross-entropy loss per token in training and validation data

The BLEU score [34] is the primary metric used to quantify the model’s performance. NIST calculates the score by giving more weight to the rarer correct n-gram [26], whereas BLEU measures edit distance using n-grams up to length four. The geometric average of modified n-gram precisions is used to calculate BLEU. The brevity penalty (BP) is then calculated as

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}, \quad (5)$$

where c is the length of the candidate translation and r is the length of the effective reference corpus. The BLEU score is as follows:

$$\text{BLEU} = \text{BP} \cdot \exp \left(\sum_{n=1}^N w_n \log p_n \right). \quad (6)$$

A higher BLEU score indicates improvements in translation. Table 4 shows the BLEU score of the proposed model compared with the current state-of-the-art models [4]. The proposed model’s performance is on-par with the current best-performing models, as shown in Table 4. Interestingly, the BLEU score of the model is higher than the Phrase-based SMT, while the NIST score is marginally lower. The primary intuition is that the CCMatrix corpus has a large number of commonly used sentences, while the more complex and nuanced sentences are more frequent in the SUPara and GlobalVoices corpora. Further studies using other datasets built using text mining can lead to a better understanding of the model’s performance.

Table 4. Translation Accuracy

System	Dataset	BLEU	NIST
Phrase-based SMT + large LM	SUPara, GlobalVoices	15.27	5.13
Attention-based NMT	SUPara, SUPara	15.57	4.72
Attention-based NMT with BPE	SUPara, GlobalVoices	16.26	5.18
Proposed Model	CCMatrix	15.68	5.12

The attention plot for a sample translation (How are you?-কমেন আছেন আপনি?) shows that the majority of the weights are concentrated on the diagonal of the matrix. This denotes which parts of the input sentence have the model’s attention while translating, as shown in Figure 4.

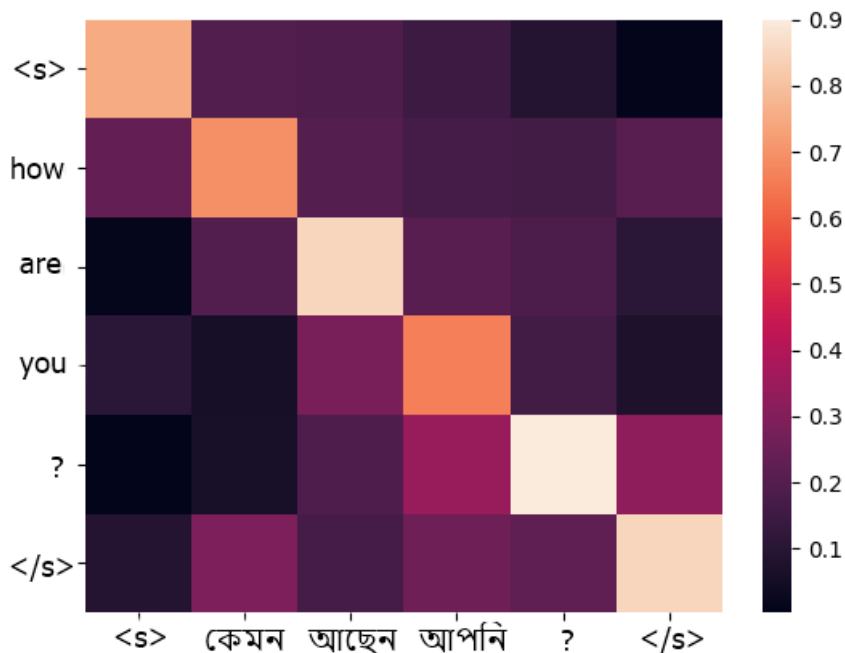


Figure 4. Attention plot of a sample translation

The model can be hypothetically improved upon by occasionally training with its own translations since the attention mechanism facilitates access to the previous output. Another aspect that demands further study is how well the model performs for a specific vocabulary and sentence length. Future works should also consider using transfer learning for a more generalized model that can adapt to multiple datasets. The model was not trained for Bangla-to-English translation, which should also be considered an important criterion for a generalized MT solution.

6 Conclusion

In this work, we have used a relatively newer approach for automated English-to-Bangla translation using the CCMatrix dataset. The system

can be extended further to deal with complex and compound sentences with a more diverse vocabulary for translation. The same approach can also be implemented for building a multi-language translation system. Furthermore, the inclusion of a more extensive database of bilingual dictionaries and adding more and more words to the lexicon can yield an even better BLEU score for uncommon words and sentence structures. Successful implementation can deliver a sound expert system for translating any text document from English to Bangla and vice versa.

7 Source Availability

The source code for this work is available at the following link <https://github.com/sadiredwan/cbmt-en-bn> under CC by 4.0. Please follow the ODC Attribution-Sharealike Community Norms and publish any derivative works under a similar open license.

References

- [1] M. A. Cheragui, “Theoretical overview of machine translation.” in *ICWIT*. Citeseer, 2012, pp. 160–169.
- [2] A. H. Homiedan, “Machine translation,” *African University, Adrar, Algeria*, 2012.
- [3] M. A. Al Mumin, M. H. Seddiqui, M. Z. Iqbal, and M. J. Islam, “Neural machine translation for low-resource english-bangla,” *Journal of Computer Science*, vol. 15, no. 11, pp. 1627–1637, 2019.
- [4] M. Mumin, M. H. Seddiqui, M. Z. Iqbal, and M. J. Islam, “shutorjoma: An english to bangla statistical machine translation system,” *Journal of Computer Science (Science Publications)*, 2019.
- [5] M. A. Hasan, F. Alam, S. A. Chowdhury, and N. Khan, “Neural machine translation for the bangla-english language pair,” in *2019 22nd International Conference on Computer and Information Technology (ICCIT)*. IEEE, 2019, pp. 1–6.

- [6] H. Schwenk, G. Wenzek, S. Edunov, E. Grave, and A. Joulin, “Ccmatrix: Mining billions of high-quality parallel sentences on the web,” *arXiv preprint arXiv:1911.04944*, 2019.
- [7] G. Wenzek, M.-A. Lachaux, A. Conneau, V. Chaudhary, F. Guzmán, A. Joulin, and E. Grave, “Ccnnet: Extracting high quality monolingual datasets from web crawl data,” *arXiv preprint arXiv:1911.00359*, 2019.
- [8] A. Godase and S. Govilkar, “Machine translation development for indian languages and its approaches,” *International Journal on Natural Language Computing*, vol. 4, no. 2, pp. 55–74, 2015.
- [9] J. Zhang, H. Luan, M. Sun, F. Zhai, J. Xu, and Y. Liu, “Neural machine translation with explicit phrase alignment,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1001–1010, 2021.
- [10] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.
- [11] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv preprint arXiv:1409.1259*, 2014.
- [12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation,” *arXiv preprint arXiv:1609.08144*, 2016.
- [13] S. Castilho, J. Moorkens, F. Gaspari, I. Calixto, J. Tinsley, and A. Way, “Is neural machine translation the new state of the art?” *The Prague Bulletin of Mathematical Linguistics*, no. 108, 2017.
- [14] L. Liu, M. Utiyama, A. Finch, and E. Sumita, “Neural machine translation with supervised attention,” *arXiv preprint arXiv:1609.04186*, 2016.

- [15] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.
- [16] M. S. Islam, S. S. S. Mousumi, S. Abujar, and S. A. Hossain, “Sequence-to-sequence bangla sentence generation with lstm recurrent neural networks,” *Procedia Computer Science*, vol. 152, pp. 51–58, 2019.
- [17] S. Abujar, A. K. M. Masum, A. Bhattacharya, S. Dutta, and S. A. Hossain, “English to bengali neural machine translation using global attention mechanism,” in *Emerging Technologies in Data Mining and Information Security*. Springer, 2021, pp. 359–369.
- [18] B. Zhang, D. Xiong, and J. Su, “Neural machine translation with deep attention,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 1, pp. 154–163, 2018.
- [19] J. Tiedemann, “The tatoeba translation challenge—realistic data sets for low resource and multilingual mt,” *arXiv preprint arXiv:2010.06354*, 2020.
- [20] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.
- [21] G. Doddington, “Automatic evaluation of machine translation quality using n-gram co-occurrence statistics,” in *Proceedings of the second international conference on Human Language Technology Research*, 2002, pp. 138–145.
- [22] J. Tiedemann and L. Nygaard, “The opus corpus-parallel and free: <http://logos.uio.no/opus>.” in *LREC*, 2004.
- [23] Z. Islam, J. Tiedemann, and A. Eisele, “English to bangla phrase-based machine translation,” in *Proceedings of the 14th Annual conference of the European Association for Machine Translation*, 2010.

- [24] M. K. Rhaman and N. Tarannum, “A rule based approach for implementation of bangla to english translation,” in *2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT)*. IEEE, 2012, pp. 13–18.
- [25] K. Muntarina, M. G. Moazzam, and M. A.-A. Bhuiyan, “Tense based english to bangla translation using mt system,” *International Journal of Engineering Science Invention*, vol. 2, no. 10, pp. 30–38, 2013.
- [26] M. A. Hasan, F. Alam, S. A. Chowdhury, and N. Khan, “Neural machine translation for the bangla-english language pair,” in *2019 22nd International Conference on Computer and Information Technology (ICCIT)*. IEEE, 2019, pp. 1–6.
- [27] S. Siddique, T. Ahmed, M. Talukder, R. Azam, M. Uddin *et al.*, “English to bangla machine translation using recurrent neural network,” *arXiv preprint arXiv:2106.07225*, 2021.
- [28] Y. Shibata, T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara, and S. Arikawa, “Byte pair encoding: A text compression scheme that accelerates pattern matching,” 1999.
- [29] K. Bostrom and G. Durrett, “Byte pair encoding is suboptimal for language model pretraining,” *arXiv preprint arXiv:2004.03720*, 2020.
- [30] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, “{TensorFlow}: a system for {Large-Scale} machine learning,” in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.
- [31] T. Kudo, “Subword regularization: Improving neural network translation models with multiple subword candidates,” *arXiv preprint arXiv:1804.10959*, 2018.
- [32] M. Lusetti, T. Ruzsics, A. Göhring, T. Samardzic, and E. Stark, “Encoder-decoder methods for text normalization,” in *Proceedings*

of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018), 2018, pp. 18–28.

- [33] M. N. Ali and G. Tan, “Bidirectional encoder–decoder model for arabic named entity recognition,” *Arabian Journal for Science and Engineering*, vol. 44, no. 11, pp. 9693–9701, 2019.
- [34] C. Chen, D. Han, and J. Wang, “Multimodal encoder-decoder attention networks for visual question answering,” *IEEE Access*, vol. 8, pp. 35 662–35 671, 2020.

Abdullah Al Shiam, Sadi Md. Redwan,
Md. Humaun Kabir, Jungpil Shin

Received November 30, 2022
Revised 1 – January 25, 2023
Revised 2 – February 22, 2023
Accepted February 28, 2023

Abdullah Al Shiam
ORCID: <https://orcid.org/0000-0002-8787-5584>
Department of Computer Science and Engineering,
Sheikh Hasina University,
Netrokona-2400, Bangladesh
E-mail: shiam.cse@shu.edu.bd

Sadi Md. Redwan
ORCID: <https://orcid.org/0000-0002-1859-1617>
Department of Computer Science and Engineering,
University of Rajshahi,
Rajshahi-6205, Bangladesh.
E-mail: sadi.redwan@ru.ac.bd

Md. Humaun Kabir
ORCID: <https://orcid.org/0000-0001-7225-0736>
Department of Computer Science and Engineering,
Bangamata Sheikh Fojilatunnesa Mujib Science and Technology University,
Jamalpur-2012, Bangladesh.
E-mail: humaun@bsfmstu.ac.bd

Jungpil Shin
ORCID: <https://orcid.org/0000-0002-7476-2468>
School of Computer Science and Engineering,
The University of Aizu Aizuwakamatsu,
Fukushima, 965-8580, Japan.
E-mail: jpshin@u-aizu.ac.jp

A Secret Image Sharing Based on Logistic-Chebyshev Chaotic Map and Chinese Remainder Theorem

Asmaa Hilmi, Soufiane Mezroui, Ahmed El Oualkadi

Abstract

Visual secret sharing (VSS) was introduced in order to solve information security issues. It is a modern cryptographic technique. It involves breaking up a secret image into n secured components known as shares. The secret image is recovered with utmost secrecy when all of these shares are lined up and piled together. A (3, 3)-secret image sharing scheme (SIS) is provided in this paper by fusing the Chinese Remainder Theorem (CRT) and the Logistic-Chebyshev map (LC). Sharing a confidential image created with CRT has various benefits, including lossless recovery, the lack of further encryption, and minimal recovery calculation overhead. Firstly, we build a chaotic sequence using an LC map. The secret value pixel for the secret image is permuted in order to fend off differential attackers. To encrypt the scrambled image, we apply our CRT technique to create three shares. Finally, the security analysis of our (3, 3)-SIS scheme is demonstrated and confirmed by some simulation results.

Keywords: Visual Cryptography, Logistic-Chebyshev map, Chinese Remainder Theorem, share.

MSC 2010: 68R10, 68Q25, 05C35, 05C05.

1 Introduction

Visual secret sharing (VSS) consists of decoding visually a secret image without involving any complex computations by superimposing the shares. Based on Naor and Shamir's method [1], a visual cryptographic scheme (VCS) is possible to encrypt an image into n shares that are secret. k or more shares should be used to rebuild the secret image.

No information can be extracted with less than k shares. The (k, n) -threshold scheme is one type of such strategy. The shares are printed on distinct transparency [2]. When they are transmitted after applying the encrypt and decrypt cryptographic techniques and then superimposed, the secret image should become visible. In addition to the definition of contrast proposed by Noar and Shamir, other researchers suggest various other definitions to improve contrast's quality [1], [3]–[6]. Ensuring a secure transmission of secret images is a challenging task in VSS. Developing an optimal encryption and decryption algorithm to tackle this issue is the main goal of several researches done on VSS, most of these studies try to use well-known algorithms such as RSA, BLOWFISH, AES, etc. For example, in [7], the authors have generated separate color matrices R_i , G_i , and B_i from the RGB colors. Then, the basic matrices, which represent the shares of the original image, have been created by dividing the values of each pixel in R_i , G_i , and B_i by 2. Once the shares are created, Advanced Encryption Standard (AES) algorithm is used to both encrypt and decrypt them separately. The proposed method of K. Shankar and P. Eswaran is used to generate multiple shares from the extracted pixel values of RGB. The use of ordinary colors is an easy way to define the colors available in the original image. After that, blocks are created from the image's shares [8], [9]. The Elliptical Curve Cryptography (ECC) method is used to encrypt the share blocks and to decrypt the encrypted blocks. However, these traditional encryption methods have enabled to meet the current image encryption requirements. Recently, the encryption algorithm of image has been considered. The two main approaches for encrypting images are diffusion and confusion, where diffusion modifies the value of pixels and confusion modifies their position. We will include a chaotic map into our secret image sharing because of its essential properties, such as sensitivity to beginning conditions and ergodicity, which have attracted the interest of cryptographers to create new system methods. In this paper, our secret image sharing scheme based on $(3, 3)$ -threshold cryptography is proposed. We generate three shares images using the Chinese Remainder Theorem (CRT); if only we have three shares, we can reconstruct the secret image; however, with fewer than three, we are unable to collect any useful information about the secret image.

The structure of this paper is as follows: Section 2 provides an analysis of the literature on image security; Section 3 contains a brief introduction of the related method used in this paper; in Section 4, the proposed method is presented, which includes the permutation phase, sharing phase, and reconstructing phase. The performance analysis and the experimental results are discussed in Section 5. Color image encryption is described in Section 6. Section 7 presents analysis and comparison. Section 8 concludes the paper.

2 Related work

Many authors have suggested different methods for image encryption and decryption using chaotic systems or CRT. In [10], the authors have proposed a threshold secret sharing scheme for digital images; the proposed method uses the Mignote [11] and Asmuth-bloom [12] solutions that were introduced in 1983. This scheme suggested in [10] is based on the CRT and divides a secret image into k shares so that a group with k shares can recover the secret image but a group with fewer shares cannot divulge any information about the hidden image. In [13], it is suggested to use a Secret Image Sharing (SIS) system that uses diminutive shadow images and is based on the CRT. Shadow images can be made to almost $1/k$ of the size of the original secret image. Auxiliary encryption is not required for this technique because random bits are added to binary representations of the CRT's random factors. In [14], a new technique is explained how to create a collection of n shared images from a set of n hidden images. The CRT and Boolean Exclusive-OR (XOR) operation are used in this method, which also introduces symmetric and modified masking coefficients. The goal is to overcome issues induced when an odd number of secret images is used. In [15], the authors used two chaotic systems: a cat map is used firstly for image pixels permutation, and the logistic map is used secondly for image pixels diffusion. M. L. Sahari et al. [16] presented a color image encryption, the secret key is connected with the plain image, and the 3D chaotic map creates the sequence in confusion and diffusion. In [17], the authors used the Chinese remainder theorem (CRT) with arithmetic compression coding to develop a secret image-sharing system. The piecewise linear map is utilized to create the compression

coding scheme, and the CRT is used to create the compression code sharing scheme. The overview of the related work revealed that most existing (k, n) schemes face the problem of if we have any k shares, we can reconstruct the secret image. Another problem raised is the difference in weight between the participants, because this can cause a problem of confidentiality. To overcome the above stated in terms of security, the work presented in this paper proposes a scheme named $(3, 3)$ -secret image sharing scheme (SIS); in other words, to rebuild a secret image, all n shares must be present; if even one share is missing, the hidden image cannot be restored. Our scheme is based on the Logistic-Chebyshev chaotic map and CRT theorem. It is a solution to improve the level of security by using an LC chaotic map to confuse the secret image and the CRT for encrypting the confused image. It is important to note that CRT is especially used to generate shares using a pure mathematical analysis which focuses on changing pixels' values. In our case, we use chaotic map to permute the position of pixel's image. Then we apply the CRT method to encrypt the scrambled image into three shares. Our results demonstrate that we obtain a high quality reconstructed secret image when all three decrypted shares are used. However, lower quality is produced when less than three shares are decrypted. That makes our technique robust and more secure.

3 Brief introduction to the related method

The Logistic-Chebyshev map is chaotic systems that can enough to design secure crypto-systems, it is a hybrid 1-D chaos map that combines two well-known chaotic systems in order to improve the discontinuous ranges of its original ones.

3.1 Logistic-Chebyshev chaotic map

The Logistic-Chebyshev map combines the two popular 1D chaotic systems, Chebyshev and the Logistic. It can be expressed as [18]

$$LC_{i+1} = (\alpha * LC_i(1 - LC_i) + \frac{(4 - \alpha)\cos(A * \arccos(LC_i))}{4}) \bmod 1, \quad (1)$$

where $\alpha \in (0, 4)$ is the control parameter and $LC_0 \in (0, 1)$ is the original value. The Chinese Remainder Theorem has been employed

vastly in cryptography. It is an old Chinese technique for resolving a collection of linear congruences.

3.2 Chinese Remainder Theorem

The simplest version of the **Chinese Remainder Theorem** is given by [10].

Theorem 1. *Let us consider n_1, \dots, n_k being pairwise relatively prime positive integers, that is $\gcd(n_i, n_j) = 1$, when $i \neq j$. Then for all integers, there is a unique integer modulo $n = \prod_{i=1}^k n_i$, such as:*

$$x \equiv a_1(\text{mod } n_1), \dots, x \equiv a_k(\text{mod } n_k).$$

Then there exists only one solution:

$$x = a_1(\text{mod } N_1 N_1^{-1}) + \dots + a_k(\text{mod } N_k N_k^{-1})(\text{mod } N), \quad (2)$$

where $N_i = \frac{N}{n_i}$, $N_i N_i^{-1} = 1(\text{mod } n_i)$.

Based on the CRT, in ref [12], The Asmuth-Bloom system was developed by Charles Asmuth and John Bloom, and it is defined as follows.

3.3 Asmuth-Bloom's (r,n)-threshold secret sharing scheme

Definition 1. [12] *Let a group of integers be $m_0, m_1, m_2, \dots, m_n$. These integers should satisfy the following conditions:*

- (i) $(m_i, m_j) = 1$ for $i \neq j$;
- (ii) $m_0 \prod_{i=0}^{r-2} m_{n-i} < \prod_{i=1}^r m_i$.

As before, n here stands for the number of shares. The scheme works as follows:

- *The secret S is chosen as a random integer.*
- *Let A be a random integer such that $(S + Am_0) < m_1, \dots, m_r$. Shares are determined by $y_i = (S + Am_0) \text{mod}(m_i)$ for all $1 \leq i \leq n$. During the decoding procedure, the following system of congruences is considered for any r shares $y_i, 1 \leq i \leq r$ out of n shares:*

$$x \equiv y_i \text{mod}(m_i). \quad (3)$$

x can easily be calculated by using the CRT since m_i values are pairwise co-prime.

4 Our proposed method

4.1 Permutation phase

As we already know, image information differs from text information in a number of ways, including significant levels of redundancy and a strong correlation between neighboring pixels. As stated in the previous section, we construct our permutation phase according to the steps below:

- 1- Let us consider a $m \times n$ gray-scale image.
- 2- We convert our $m \times n$ gray-scale image in one vector $[1, m \times n]$.
- 3- A chaotic vector $X = X_0, X_1, \dots, X_{m \times n - 1}$ is generated from Logistic-Chebyshev map taking $m \times n$ iterations from 0 to $(m \times n - 1)$
- 4- The sequence obtained is sorted in descending order based on the index of the sequence X to create a new index j .
- 5- Our image vector $[1, m \times n]$ is permuted according to the new index j .

Figure 1 describes the steps detailed above for an example of a (3,3) gray-scale image.

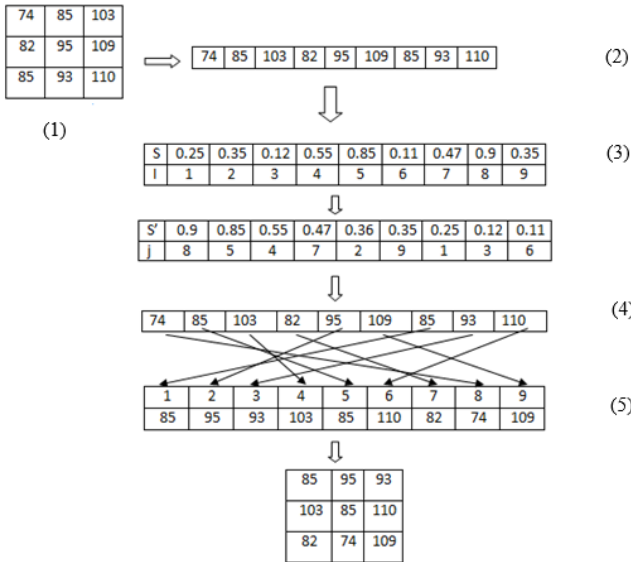


Figure 1. Permutation phase for gray-scale image

4.2 Sharing phase

In the following, we use the CRT theorem to encrypt the confused image; we consider a pixel X which is an integer value between 0 and 255. Since the decomposition of 255 in primary numbers is given by $255 = 17 \times 5 \times 3$, we obtain:

$$\begin{aligned} share_1 &\equiv X \pmod{17} \oplus K_m \\ share_2 &\equiv X \pmod{5} \oplus K_m \\ share_3 &\equiv X \pmod{3} \oplus K_m \end{aligned} \tag{4}$$

with K_m is a matrix random key whose size is the same as that of the secret image. Figure 2 shows an example of obtained share1, 2, and 3 from image of size (3,3) before applying the eXclusive-OR with the K_m .

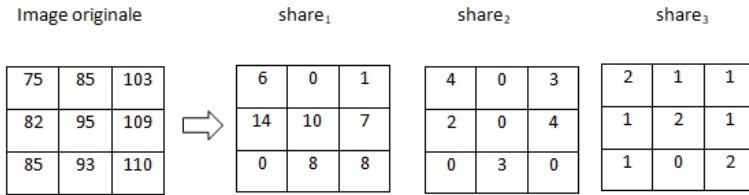


Figure 2. Example of creation shares with our proposed method

4.3 Reconstructing phase

The inverse of the sharing phase is the reconstruction phase. It is to note that we decrypt the encrypted images using the inverse of CRT Eq(2). Then, to obtain the original image, the reconfused image is permuted by using the inverse of the permutation phase as follows:

- 1- Let us consider the reconfused image after decrypting all shares.
- 2- We convert it in one vector.
- 3- We generate the same chaotic vector X from the Logistic-Chebyshev map obtained in permutation phase; cited by $(S; I)$.
- 4- The sequence obtained X is sorted in descending order based on their index I , to create a new index J ; cited by (S', J) .
- 5- We reconstruct the secret image as follows: the X pixel in J^{th} position of the reconfused image vector is placed in I^{th} position as shown

in Figure 3.

6- The reconstructed image (m, n).

Figure 3 describes the reconstructed steps of the reconfused image detailed above.

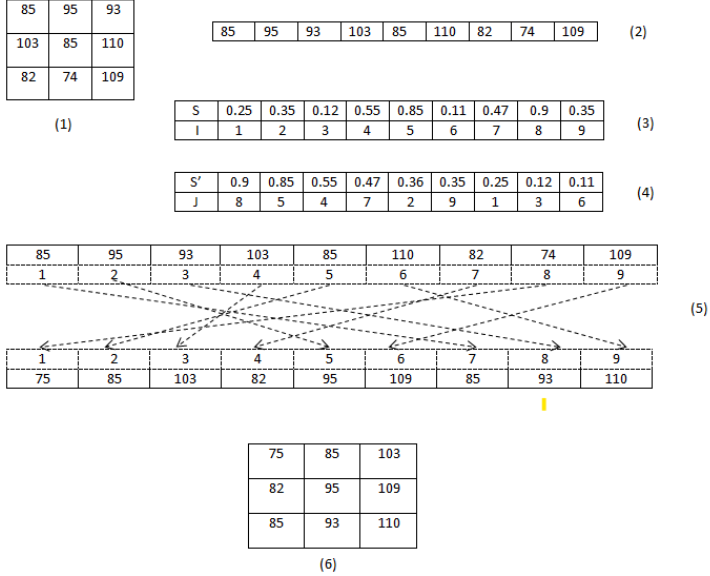


Figure 3. Reconstructed steps of reconfused image

5 Experimental results and analysis

In this paper, the feasibility and applicability of the suggested solution are presented, and the secret images of three sizes (a) 220×220 px, (b) 174×290 , and (c) 642×640 are encrypted. As shown in Figure 4, in our solution, all three shares can restore the original secret images without distortion. We demonstrate that with two parameters, $psnr$ and $ssim$.

5.1 The PSNR

PSNR is defined as the difference between the highest achievable power of the signal and the power of the corrupted noise [20] and given by



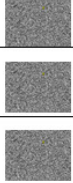



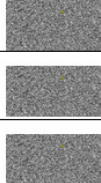



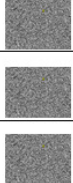

Secret image	Confused image	Encrypted shares	Decrypted image
 (a)			
 (b)			
 (c)			

Figure 4. Encryption and decryption of gray image

$$PSNR(dB) = 20 \log_{10} \left(\frac{max_i}{\sqrt{MSE}} \right), \quad (5)$$

where the image's maximum potential pixel value is max_i , it is equal to 255. The MSE stands for Mean Squared Error and represents the average square of the error between the original and the reconstructed image. The pixels are represented using 8 bits per sample. Theoretically, PSNR can be infinite if MSE equals 0; in this instance, the original and the reconstructed images are identical, i.e., the corresponding pixels of both images have similar values. The PSNR can also be calculated for color image by

$$PSNR(dB) = \frac{1}{n} \times (PSNR(Red) + PSNR(Green) + PSNR(Blue)), \quad (6)$$

where n is the number of all RGB color components.

Table 1. The PSNR value

Images	PSNR(dB)
(a)	infinite
(b)	infinite
(c)	infinite

From Table 1, the image revealed by three share images represents a PSNR infinite which means they have similar values to the original image. We will demonstrate later that fewer than three share images cannot give any information about the original secret image, but all share images can be used to reconstruct the original secret image without distortion.

5.2 The SSIM

A quality evaluation index is the SSIM Index. The computation of three parameters: the luminance, the contrast, and the structural parameters forms the basis of this method. According to [23], the overall index is given by

$$SSIM(x, y) = [l(x, y)]^\alpha \cdot [c(x, y)]^\beta \cdot [s(x, y)]^\gamma \quad (7)$$

and

$$\begin{cases} l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \\ c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \\ s(x, y) = \frac{\sigma_{xy} + C_3}{\mu_x^2 + \mu_y^2 + C_3} \end{cases}, \quad (8)$$

where, x and y represent two input images; μ_x, μ_y are the local means; σ_x, σ_y , are the standard deviations; and s is a cross-covariance for images x, y . α, β , and γ are factors used to modify the relative importance of the luminance, contrast, and structure; C_1, C_2 , and C_3 are constants used to avoid instability [23]. In the case when $\alpha = \beta = \gamma = 1$ and

$C_3 = \frac{C_2}{2}$, Eq.(7) can be expressed by

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}. \quad (9)$$

Table 2 shows the values of SSIM. The obtained SSIM index is a decimal value between -1 and 1. An integer between -1 and 1 represents the obtained SSIM index. Only in the situation of two identical images x and y , the SSIM is a value equal to 1, signifying perfect structural similarity. It is necessary to note that, in our case, x is the original image and y is the reconstructed image.

Table 2. The SSIM value

Images	SSIM
(a)	1
(b)	1
(c)	1

By analyzing the results presented in Tables 1 and 2, we conclude that the proposed method is lossless. Furthermore, to rebuild the hidden image using this approach, all shares must be present and be the same size as the original image. The comparison of the proposed schemes with other references is shown in Table 3.

6 Security analysis

We examine the scheme's security in this section. We shall examine its resistance to the most significant attacks in particular. The security of the (3, 3)-SIS scheme relates to that we encrypt our image in three shares, and all three shares should be presented to restore the secret image. We demonstrate that with fewer than three shares, we rebuild the original image, but the quality is quite poor.

6.1 Statistical analysis

The statistical test seeks to ascertain the proposed scheme's confusion and diffusion features. A correlation test of the nearby pixels in the

Table 3. The comparison of the schemes presented in [17], [22], [25], [26] and the scheme proposed in this paper

	Scheme in [17]	Scheme in [22]	Scheme in [25]	Scheme in [26]	Our proposed
Encryption and decryption	CRT	CRT	CRT	CRT	CRT
The same size	Different size	the same size	the same size	Different size	the same size
Additional Information	Compression code	-	Confusion using chaotic map	-	Confusion using chaotic map
Is lossless	Yes	Yes	Yes	Yes	Yes
Threshold	(n, n)	(k, n)	(k, n)	(k, n)	(3, 3)

original image and their share, the entropy of Shannon, and histogram analysis are presented in this section.

6.1.1 Correlation coefficient factor

Correlation coefficient factor is a crucial element for evaluating an encryption algorithm's quality (CC). This analysis helps to explain how closely the original image and encrypted images resemble one another. According to Eqs(10)(11)(12)(13), the CC factor of the adjacent pixel [21] can be stated.

$$r_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}} \quad (10)$$

$$cov(x, y) = \frac{1}{N} \sum_{i=1}^N (x_i - E(x))(y_i - E(y)) \quad (11)$$

$$D(x) = \frac{1}{N} \sum_{i=1}^N (x_i - E(x))^2 \quad (12)$$

$$D(x) = \frac{1}{N} \sum_{i=1}^N x_i \quad (13)$$

where x and y represent the values of two neighboring pixels in the original or encrypted images, $E(x)$ is the mean value of x , $D(x)$ is the variance with respect to mean, $cov(x, y)$ represents the estimation of the covariance between neighboring pixels x and y , and r_{xy} represents the correlation coefficient between x and y . For better encrypted image, correlation coefficient should be close to zero or very low. Table 4 lists the computed correlation coefficients of original and share images. It is clear from Table 4 that the two neighboring pixels in the original images are strongly correlated to each other, while the correlation coefficients for the encrypted images are so close to zero.

Table 4. The correlation coefficient of images (a), (b), and (c) and their corresponding Share images

Images	Horizontal	Vertical	Diagonal
(a)	0.9207	0.9582	0.8909
Share1	-0.0083	-0.0031	0.0026
Share2	-0.0085	-0.0038	0.0043
Share3	-0.0081	-0.0043	0.0043
(b)	0.9759	0.9710	0.9573
Share1	0.0074	0.0017	0.0011
Share2	-0.0079	-0.0020	-0.0017
Share3	-0.0079	0.0015	0.0016
(c)	0.9308	0.9131	0.8812
Share1	-0.0010	-0.0011	-0.0005
Share2	-0.0009	-0.0008	0.0009
Share3	-0.0009	-0.0008	0.0006

6.1.2 Entropy of Shannon

The Shannon entropy over the ciphertext is a standard metric for evaluating the effectiveness of image encryption. It was first proposed by Claude Shannon in 1948, [24] and defined by Eq (14).

$$H(X) = - \sum_{i=1}^n Pr(x_i) \log_2 Pr(x_i). \quad (14)$$

X denotes the test random variable, x_i denotes the i_{th} possible value of X , and $Pr(x_i)$ is the probability of $X = x_i$. An ideally encrypted image is completely random if an encrypted image is very random-like. It is believed that information entropy is very close to theoretical maximum; therefore, the pixel values are uniformly distributed within a random encrypted image. Table 5 shows the entropy of encrypted images, which is very close to 8, the theoretical upper bound of entropy for an 8-bit image. Since the image entropy is a quantitative measurement, it is an equivalent test to the histogram analysis, which plots the distribution of Pr and is commonly used for security analysis in the image encryption literature.

Table 5. The entropy value

Images	secret image	im-share 1	share 2	share 3
(a)	7.4721	7.9964	7.9956	7.9947
(b)	7.6451	7.9959	7.9459	7.9958
(c)	7.3405	7.9994	7.9990	7.9986

6.1.3 Histogram of analysis

The statistical properties of images are shown by the histograms. The histograms of the secret and reconstructed images are displayed in Fig. 5, where it can be seen that for images (a), (b), and (c), the histogram following the decryption process is identical to that of the secret image; whereas, before decryption, the encrypted images' histograms are uniformly distributed and significantly different from those of the original images.

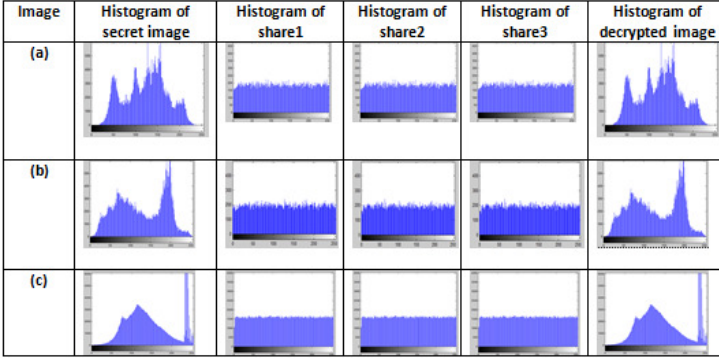


Figure 5. The histogram analysis of secret, encrypted, and decrypted images

6.2 Security of the sharing scheme

The logistic-Chebyshev map and CRT serve as the foundation for the sharing scheme's security. The threshold in this method is three, and any shadow images with two or less cannot obtain enough data to reconstruct the hidden image. From Section 4, in our case $r = n$, then $\min(r) = \prod_{i=1}^r m_i$ and $\max(r - 1) = \prod_{i=1}^{n-r+2} m_i$ we choose an arbitrary secret value $S = 127$, which is a positive integer and $\max(r - 1) < S < \min(r)$. Let a_i denote the remainder of S modulo m_i . In our scheme, we have three prime integers $m_1 = 17, m_2 = 5$, and $m_3 = 3$, which $3 < 5 < 17$, and $a_1 = 8, a_2 = 2$, and $a_3 = 1$. The S can be restored by all three sharing values. We compute $y = CRT[(a_1, a_2, a_3), (m_1, m_2, m_3)] = 127$. If we choose $w = CRT[(a_1, a_2), (m_1, m_2)] = 25 \neq 127$ or $w = CRT[(a_2, a_3), (m_2, m_3)] = 187 \neq 127$ or $w = CRT[(a_1, a_3), (m_1, m_3)] = 170 \neq 127$.

In Figure 6, (a), (b), and (c) are the secret images; however, (a-1), (b-1), and (c-1) are the images revealed by 2 shares. Moreover, (a-2), (b-2), and (c-2) are the images revealed by 1 share. According to the experimental results, two or one share images cannot divulge any information about the original secret image, but the three share images can be used to accurately reconstruct the original secret image.

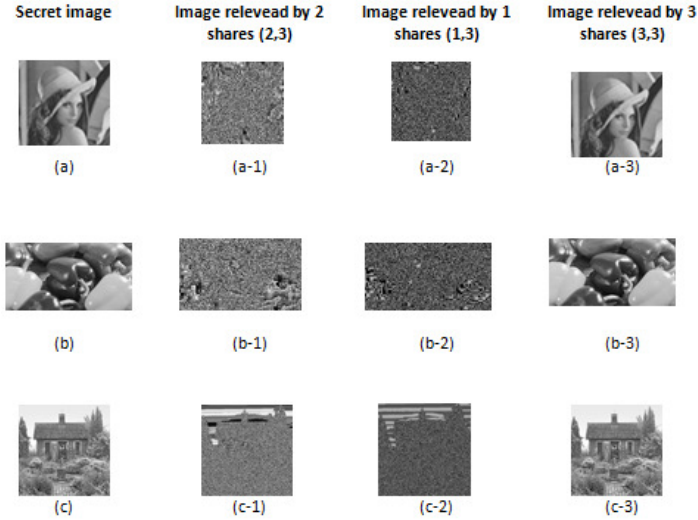


Figure 6. The experimental results of (1,3), (2,3), and (3,3) secret image sharing based on Chinese remainder theorem

To better clarify the experimental results, the PSNR values of images (a-1), (a-2), (a-3), (b-1), (b-2), (b-3), (c-1), (c-2), and (c-3) are presented in Table 6. According to Table 6, the PSNR values of the images revealed by one or two share images are very low ($< 10dB$).

Table 6. The PSNR value

Images	PSNR(dB)
(a-1)	9.314
(a-2)	9.181
(b-1)	9.257
(b-2)	9.187
(c-1)	8.278
(c-2)	9.102

7 Color image

Now, the scheme is extended to apply to the encryption of color image. After testing, the effect is good. The scheme can also be used for color image.

7.1 Simulation results

The simulation is realized for the "Lena" color image. We first extracted the original image in three components, R, G, and B (Red, Green, and Blue); then we used the Logistic-Chebyshev map to scramble each component; after that we have (3, 3)-scheme based on CRT to encrypt each component in three shares as shown in Figures 7 and 8.



Figure 7. The color secret image and the confused images for each component image

7.2 Security analysis

This section shows directly the PSNR value of the reconstructed images when less than n shares are decrypted. For information, please refer to Section 5.1.

As shown in Figures 9, 10, and 11, the images reconstructed by less than nine shares (three shares for each components) present bad-quality of images. This is demonstrated by the PSNR values of the SIS schemes (8,9), (7,9), and (6,9) which do not exceed 11.99 dB.

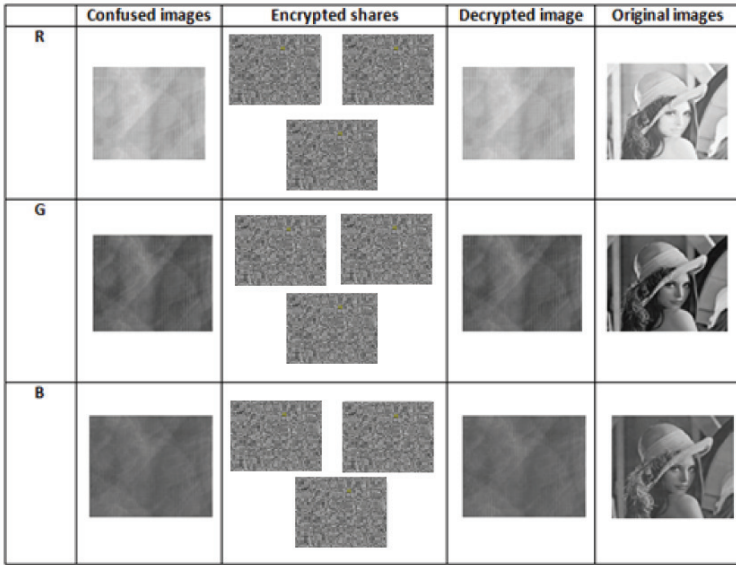


Figure 8. The confused images, encrypted shares, decrypted shares, and reconstructed images

7.2.1 Correlation coefficient factor

For details of correlation, see Section 6.1.1. This section directly shows the correlation coefficients of the R, G, and B components of the "Lena" color image. As shown in Table 7, our scheme can resist statistical analysis.

8 Analysis and comparison

This section analyzes the experimental findings of the (3, 3) secret sharing scheme based on the Chinese remainder theorem and compares it to the secret image sharing schemes provided in Reference [25].

8.1 Analysis

The (3, 3)-SIS scheme based on the Chinese remainder theorem is examined in this section in view of the experimental findings in Section


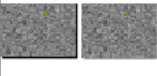


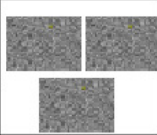

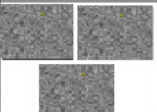

	Input image	Encrypted shares	Decrypted images	Output image
	R			<i>PSNR = 11.99 dB</i> 
	G			
	B			

Figure 9. The reconstructed image using (8, 9)-SIS schemes with PSNR value


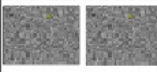

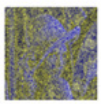
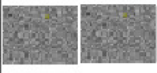

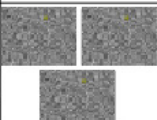

	Input image	encrypted shares	Decrypted images	Output image
	R			<i>PSNR = 9.88 dB</i> 
	G			
	B			

Figure 10. The reconstructed image using (7, 9)-SIS schemes with PSNR value

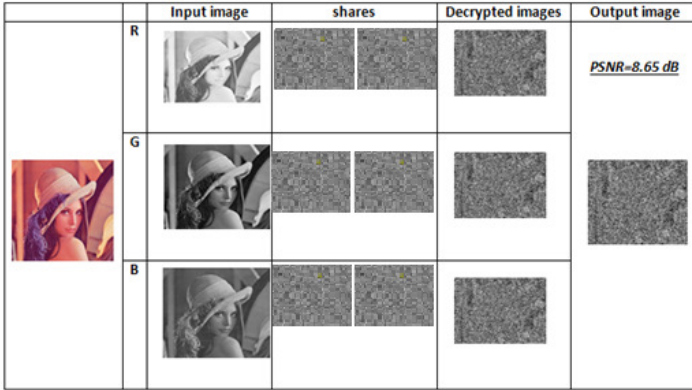


Figure 11. The reconstructed image using (6, 9)-SIS schemes with PSNR value

Table 7. The correlation coefficient of Lena and its corresponding share images

Images	Horizontal	Vertical	Diagonal
R	0.9433	0.9716	0.9150
Share1	0.0114	0.0128	0.0138
Share2	0.0041	-0.0037	-0.0114
Share3	0.0043	0.0058	-0.0006
G	0.9257	0.9618	0.8980
Share1	-0.0044	0.00938	0.0060
Share2	0.0061	0.0074	0.0018
Share3	0.0026	-0.0097	-0.0030
B	0.9100	0.9435	0.8783
Share1	-0.0040	-0.0066	0.0052
Share2	0.0030	0.0022	-0.0031
Share3	0.0017	-0.0068	-0.0017

5, and the following conclusions are drawn:

- a) Each and every one of the shared images is just random noise, revealing nothing about the original secret image.
- b) A recovery process cannot represent any information from the original secret image with fewer than three sharing images.
- c) To recreate the original secret image without distortion, three sharing images are used.
- d) Since there is no pixel expansion in the suggested approach, each share image is the same size as the original image.
- e) The approach can get rid of pixel correlations from the original secret image.
- f) Our proposed scheme is very suitable for color images.

8.2 Comparison

In the reference [25], a secret digital image is split into n pieces and then disseminated among the n participants, and only by using the cooperation of arbitrary r parties can the hidden digital image be rebuilt. But in our scheme, we choose three prime numbers that satisfy the CRT, and all these three parties are totally indispensable and should be present, just with $r \leq 2$ we can not rebuild the hidden image. The second point is about the permutation phase. In [25], the transformation is done via an equation. In our case, the permutation is totally different. The permutation phase is done according to the new index j as detailed in Section 4.1.

9 Conclusion

This paper proposes a lossless secret image sharing for grayscale and color image based on (3, 3) threshold cryptography, i.e., (3, 3)-SIS scheme. In order to strengthen our solution and make it more resistant against differential attack, we use a chaotic map to scramble the secret image before constructing three share images using our CRT technique. For the (3, 3)-SIS scheme, all three participants should be existed to restore the secret value. From the experiment, we can see that our (3, 3)-SIS scheme using CRT is reliable and effective; more-

over, we demonstrate that when the number of participants is less than three, the relevant information is very bad, which increases the level of security.

References

- [1] M. Naor and A. Shamir, “Visual cryptography,” in *EUROCRYPT: Workshop on the Theory and Application of Cryptographic Techniques*, (Italy), Springer, 1994.
- [2] W. Q. Yan, D. Jin, and M. S. Kankanhalli, “Visual cryptography for print and scan applications,” in *IEEE International Symposium on Circuits and Systems*, (Vancouver), 2004.
- [3] F. Liu, C. Wu, and X. Lin, “Step Construction of Visual Cryptography Schemes,” *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 27–38, 2009.
- [4] E. Verheul and H. V. Tilborg, “Constructions and properties of k out of n visual secret sharing schemes,” *Des. Codes Cryptogr.*, vol.11, pp. 179–196, 1997.
- [5] P. A. Eisen and D. R. Stinson, “Threshold visual cryptography schemes with specified whiteness levels of reconstructed pixels,” *Des. Codes Cryptogr.*, vol. 25, pp. 15–61, 2002.
- [6] F.Liu, C.Wu, and X.Lin, “A new definition of the contrast of visual cryptography scheme,” *Information Processing Letters*, vol. 110, no. 7, pp. 241–246, 2009.
- [7] K. Shankar and P. Eswaran, “Sharing a Secret Image with Encapsulated Shares in Visual Cryptography,” *Procedia Computer Science*, vol. 70, pp. 462–468, 2015.
- [8] K. Shankar and P. Eswaran, “RGB-Based Secure Share Creation in Visual Cryptography Using Optimal Elliptic Curve Cryptography Technique,” *Journal of Circuits, Systems, and Computers*, vol. 25, pp. 1–23, 2016.

- [9] K. Shankar and P. Eswaran, “RGB Based multiple Share Creation in Visual Cryptography with Aid of Elliptic Curve Cryptography,” *China Communications*, vol. 14, pp. 118–130, 2017.
- [10] S. J. Shyu and Y. -R. Chen, “Threshold Secret Image Sharing by Chinese Remainder Theorem,” in *2008 IEEE Asia-Pacific Services Computing Conference*, (Yilan, Taiwan), 2008, pp. 1332–1337. DOI: 10.1109/APSCC.2008.223.
- [11] M. Mignote, “How to share a secret,” in *Cryptography. EURO-CRYPT 1982* (Lecture Notes in Computer Science, vol 149), T. Beth, Ed. Berlin, Heidelberg: Springer, 1983, pp. 371–375. DOI: https://doi.org/10.1007/3-540-39466-4_27.
- [12] C. Asmuth and J. Bloom, “A Modular Approach to Key Safeguarding,” *IEEE Transactions on information theory*, vol. 29, pp. 208–210, 1983.
- [13] Jinrui Chen, Kesheng Liu, Xuehu Yan, Lintao Liu, Xuan Zhou, and Longdan Tan, “Chinese Remainder Theorem-Based Secret Image Sharing with Small-Sized Shadow Images,” *Symmetry*, vol. 10, no. 8, Article No. 340, 2018. DOI: <https://doi.org/10.3390/sym10080340>.
- [14] P. Heri and M.G. Jing, “A Note on Multiple Secret Sharing Using Chinese Remainder Theorem and Exclusive-OR,” *IEEE Access*, vol. 7, pp. 37473–37497, 2019.
- [15] Guanrong Chen, Yaobin Mao, and Charles K. Chui, “A symmetric image encryption scheme based on 3D chaotic cat maps,” *Chaos, Solitons and Fractals*, vol. 21, no. 3, pp. 749–761, 2004. DOI: <https://doi.org/10.1016/j.chaos.2003.12.022>.
- [16] M. L. Sahari and I. Boukemara, “A pseudo-random numbers generator based on a novel 3D chaotic map with an application to color image encryption,” *Nonlinear Dynamics*, vol. 94, pp. 723–744, 2018. DOI: <https://doi.org/10.1007/s11071-018-4390-z>.

- [17] Wei Hua and Xiaofeng Liao, “A secret image sharing scheme based on piecewise linear chaotic map and Chinese remainder theorem,” *Multimedia Tools and Applications*, vol. 76, no. 5, pp. 1–17, 2016. DOI: 10.1007/s11042-016-3364-8.
- [18] A. Alanezi, B. Abd-El-Atty, H. Kolivand, A. A. Abd El-Latif, B. Abd El-Rahiem, S. Sankar, and H. S. Khalifa, “Securing Digital Images through Simple Permutation-Substitution Mechanism in Cloud-Based Smart City Environment,” *Security and Communication Networks*, Volume 2021, Article ID 6615512, 2021. DOI: <https://doi.org/10.1155/2021/6615512>.
- [19] S. Iftene and I.C. Boureanu, “Weighted threshold secret sharing based on the Chinese remainder theorem,” *Scientific Annals of Cuza University*, vol. 15, pp. 161–172, 2005.
- [20] A. T. Nasrabadi, M. A. Shirsavar, A. Ebrahimi, and M. Ghanbari, “Investigating the PSNR calculation methods for video sequences with source and channel distortions,” in *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, (Beijing, China), 2014, pp. 1–4. DOI: 10.1109/BMSB.2014.6873482.
- [21] Sura F. Yousif, “Grayscale Image Confusion and Diffusion Based on Multiple Chaotic Maps,” in *1st International Scientific Conference of Engineering Sciences - 3rd Scientific Conference of Engineering Science (ISCES)*, pp. 114–119, 2018.
- [22] S. Zhang and F. Miao, “Secret Image Sharing Based on Chinese Remainder Theorem over a Polynomial Ring,” in *Machine Learning for Cyber Security. ML4CS 2020* (Lecture Notes in Computer Science, vol 12486), X. Chen, H. Yan, Q. Yan, X. Zhang, Eds. Springer, Cham., 2020, pp. 634–643.
- [23] W. Zhou, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image Quality Assessment: From Error Visibility to Structural Similarity,” *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 2004.

- [24] C.E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, 1948.
- [25] C. Hu, X. Liao, and D. Xiao, "Secret Image Sharing Based on Chaotic Map and Chinese Remainder Theorem," *International Journal of Wavelets, Multiresolution and Information Processing*, vol. 10, no. 3, 2012.
- [26] S. J. Shyu and Y. R. Chen, "Threshold secret image sharing by Chinese Remainder theorem," in *Asia-Pacific Services Computing Conference, 2008. APSCC'08. IEEE*, pp. 1332–1337, 2008.

Asmaa Hilmi, Soufiane Mezroui,
Ahmed El Oualkadi

Received October 16, 2021
Revised 1 – October 24, 2022
Revised 2 – February 09, 2023
Accepted February 16, 2023

Asmaa Hilmi

ORCID: <https://orcid.org/0000-0002-4665-7177>

Abdelmalek Essaadi University National School
of applied sciences of Tangier (ENSAT)

Laboratory of Information and Communication Technologies (LabTIC)

ENSA Tanger, Route Ziaten, BP 1818, Tanger principale, Morocco.

E-mail: asmaa00hilmi@gmail.com

Soufiane Mezroui

ORCID: <https://orcid.org/0000-0003-2705-9806>

Abdelmalek Essaadi University National School
of applied sciences of Tangier (ENSAT)

Mathematics and Intelligent Systems Team (MASI) ENSA Tanger,

Route Ziaten, BP 1818, Tanger principale, Morocco.

E-mail: mezroui.soufiane@yahoo.fr

Ahmed El Oualkadi

ORCID: <https://orcid.org/0000-0002-4953-1000>

Abdelmalek Essaadi University National School
of applied sciences of Tetuan (ENSATe)

Laboratory of Information and Communication Technologies (LabTIC)

ENSA Tetouan, Avenue Palestine B.P 2222, M'hannech II-Tetouan, Morocco.

E-mail: eloualkadi@gmail.com

A new type of digital signature algorithms with a hidden group

Dmitriy N. Moldovyan

Abstract

The known designs of digital signature schemes with a hidden group, which use finite non-commutative algebras as algebraic support, are based on the computational complexity of the so-called hidden discrete logarithm problem. A similar design, used to develop a signature algorithm based on the difficulty of solving a system of many quadratic equations in many variables, is introduced. The significant advantage of the proposed method compared with multivariate-cryptography signature algorithms is that the said system of equations, which occurs as the result of performing the exponentiation operations in the hidden group, has a random look and is specified in a finite field of a higher order. This provides the ability to develop post-quantum signature schemes with significantly smaller public-key sizes at a given level of security.

Keywords: finite associative algebra, non-commutative algebra, commutative finite group, discrete logarithm problem, hidden logarithm problem, public key, digital signature, multivariate cryptography, post-quantum cryptosystem.

MSC 2010: 68P25, 68Q12, 68R99, 94A60, 16Z05, 14G50.

1 Introduction

Multivariate public key cryptosystems (MPKCs) represent a class of post-quantum cryptographic algorithms since they are based on the computational complexity of solving a system of non-linear (usually quadratic) equations in many variables, which is set over a finite field [1], [2]. Hypothetic quantum computers are not efficient at solving

the said computational problem. One of three finalists of the NIST competition on developing post-quantum public-key algorithms [3] in the nomination of digital signatures, namely the algorithm Rainbow, is a prominent example of the said class of cryptosystems. The problem of solving a system of quadratic equations is in general an NP-hard problem; however, while designing an MPKC, one should use a set of quadratic polynomials defining a map in a finite field, which has a trapdoor. The design of the MPKCs consists mainly of the development of secure multivariate trapdoors. Inversion of a map with a trapdoor is connected with solving a system belonging to a subclass of systems of quadratic equations. Therefore, the security of the MPKCs is not guaranteed by the NP-hardness of the random or generic systems of quadratic equations and requires expert evaluation.

Currently, the development of the public-key digital signature algorithms, the security of which is based on the so-called hidden discrete logarithm problem (HDLP) defined in a finite non-commutative algebra (FNAA), is also considered an attractive approach to designing practical post-quantum public-key cryptoschemes [4], [5], [7]. A specific feature of the HDLP-based signature schemes is the use of a hidden (secret) group in which exponentiation operations are executed when generating a public key and signature [6]–[8]. Usually, the signature is calculated in the form of two integers (e, s) , where e is a randomization element and s is a fitting one. Several HDLP-based signature schemes use an additional fitting element of the signature, which represents an algebra element S [9], [10] used as one of the multipliers of the signature verification equation. The latter potentially is a source of forging signature algorithms. To prevent such attacks, the technique of doubling the verification equation is proposed [9]. A general problem of the HDLP-based approach is a justification of the post-quantum security since, in any case, we have a problem of finding a discrete logarithm value, which can be solved on a quantum computer in polynomial time [11]–[13].

The present article introduces a new design of signature algorithms with a hidden group, in which the single fitting element of the signature represents an algebra element S , and the attacks of the mentioned type are prevented due to using a verification equation with three entries of

the signature element S . The introduced signature scheme uses the exponentiation operations executed in a hidden commutative group; however, its security is based on the computational difficulty of solving a system of many quadratic equations in many variables (unknowns), which is set over the ground finite field $GF(p)$. The mentioned type of the underlying computational problem defines post-quantum security of the developed signature scheme, a significant merit of which relates to the fact that the said system is not directly connected with the used trapdoor, since the latter is set at the level of FNAA, when selecting random masking (secret) m -dimensional vectors.

2 The FNAAs used as algebraic support

Suppose a finite m -dimensional vector space is defined over a finite field, for example, the ground field $GF(p)$. If, in addition to the operation of addition and scalar multiplication, a vector multiplication operation is defined so that it is distributive at the right and at the left relative to the addition operation, then the full set of the m -dimensional vector composes an m -dimensional finite algebra. Some algebra element A can be denoted in the following two forms: $A = (a_0, a_1, \dots, a_{m-1})$ and $A = \sum_{i=0}^{m-1} a_i \mathbf{e}_i$, where $a_0, a_1, \dots, a_{m-1} \in GF(p)$ are called coordinates; $\mathbf{e}_0, \mathbf{e}_1, \dots, \mathbf{e}_{m-1}$ are basis vectors.

The vector multiplication operation of two m -dimensional vectors A and B is set as $AB = \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} a_i b_j (\mathbf{e}_i \mathbf{e}_j)$, where every of the products $\mathbf{e}_i \mathbf{e}_j$ is to be substituted by a single-component vector $\lambda \mathbf{e}_k$, where $\lambda \in GF(p)$, indicated in the cell at the intersection of the i th row and j th column of the so-called basis vector multiplication table (BVMT). To define associative vector multiplication operation, the BVMT should define associative multiplication of all possible triples of the basis vectors $(\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k)$: $(\mathbf{e}_i \circ \mathbf{e}_j) \circ \mathbf{e}_k = \mathbf{e}_i \circ (\mathbf{e}_j \circ \mathbf{e}_k)$.

In the developed digital signature scheme, it is proposed to use i) the 4-dimensional FNAA with the multiplication operation set by the BVMT shown in Table 1 or ii) the 6-dimensional FNAA defined by the BVMT shown in Table 2. In the first (second) case, the algebra is defined over the field $GF(p)$ of prime order $p = 2q + 1$, where q is a 256-bit (128-bit) prime.

Table 1. The sparse BVMT [14] setting the used 4-dimensional FNAA ($\lambda \neq 0$)

\cdot	\mathbf{e}_0	\mathbf{e}_1	\mathbf{e}_2	\mathbf{e}_3
\mathbf{e}_0	\mathbf{e}_0	0	0	\mathbf{e}_3
\mathbf{e}_1	0	\mathbf{e}_1	\mathbf{e}_2	0
\mathbf{e}_2	\mathbf{e}_2	0	0	$\lambda \mathbf{e}_1$
\mathbf{e}_3	0	\mathbf{e}_3	$\lambda \mathbf{e}_0$	0

The structure of the used 4-dimensional FNAA had been studied in detail in [14], where it had been proven that this algebra contains $p^2 + p + 1$ different commutative subalgebras of order p^2 , including i) $\frac{p(p+1)}{2}$ subalgebras with multiplicative groups of order $(p-1)^2$, every one of the latter is generated by a minimum generator system containing two vectors of the same order (such groups are called groups having 2-dimensional cyclicity); ii) $\frac{p(p-1)}{2}$ subalgebras with cyclic multiplicative groups of order $p^2 - 1$, and iii) $p + 1$ subalgebras with cyclic multiplicative groups of order $p(p-1)$. This algebra contains the global two-sided unit $E = (1, 1, 0, 0)$. The scalar vectors have the form $(j, j, 0, 0)$, where $j = 1, 2, \dots, p-1$. A vector $G = (g_0, g_1, g_2, g_3)$ is invertible, if its coordinates satisfy the following invertibility condition [14]:

$$g_0 g_1 \neq \lambda g_2 g_3. \tag{1}$$

The used 6-dimensional FNAA had been earlier considered as a particular case generated applying a unified method for defining associative algebras of arbitrary even dimensions [15]. The latter algebra also contains a sufficiently large number of commutative groups of orders $(p-1)^2$, $p^2 - 1$, and $p(p-1)$; however, its decomposition into the set of commutative algebras had not been studied in detail. The global two-sided unit of the algebra is the vector $E = (1, 0, 0, 0, 0, 0)$. The scalar vectors have the form $(j, 0, 0, 0, 0, 0)$, where $j = 1, 2, \dots, p-1$. A vector $G = (g_0, g_1, g_2, g_3, g_4, g_5)$ is invertible, if its coordinates satisfy

Table 2. The BVMT [15] setting the used 6-dimensional FNAA ($\lambda \neq 0$)

\cdot	\mathbf{e}_0	\mathbf{e}_1	\mathbf{e}_2	\mathbf{e}_3	\mathbf{e}_4	\mathbf{e}_5
\mathbf{e}_0	\mathbf{e}_0	\mathbf{e}_1	\mathbf{e}_2	\mathbf{e}_3	\mathbf{e}_4	\mathbf{e}_5
\mathbf{e}_1	\mathbf{e}_1	$\lambda\mathbf{e}_0$	\mathbf{e}_5	$\lambda\mathbf{e}_4$	\mathbf{e}_3	$\lambda\mathbf{e}_2$
\mathbf{e}_2	\mathbf{e}_2	\mathbf{e}_3	\mathbf{e}_4	\mathbf{e}_5	\mathbf{e}_0	\mathbf{e}_1
\mathbf{e}_3	\mathbf{e}_3	$\lambda\mathbf{e}_2$	\mathbf{e}_1	$\lambda\mathbf{e}_0$	\mathbf{e}_5	$\lambda\mathbf{e}_4$
\mathbf{e}_4	\mathbf{e}_4	\mathbf{e}_5	\mathbf{e}_0	\mathbf{e}_1	\mathbf{e}_2	\mathbf{e}_3
\mathbf{e}_5	\mathbf{e}_5	$\lambda\mathbf{e}_4$	\mathbf{e}_3	$\lambda\mathbf{e}_2$	\mathbf{e}_1	$\lambda\mathbf{e}_0$

the following invertibility condition [15]:

$$\begin{aligned} & \frac{1}{4}((g_0 + g_2 + g_4)^2 - \lambda(g_1 + g_3 + g_5)^2) \times \\ & \times ((g_0 - g_2)^2 + (g_0 - g_4)^2 + (g_2 - g_4)^2 - \\ & - \lambda(g_1 - g_3)^2 - \lambda(g_1 - g_5)^2 - \lambda(g_3 - g_5)^2)^2 \neq 0. \end{aligned} \tag{2}$$

3 The developed post-quantum signature algorithm

3.1 Generation of the public key

The first step of the procedure for generating a public key consists in selecting a random minimum generator system $\langle G, H \rangle$ of a commutative hidden group, which contains two vectors of the same order equal to q . For the case of the 4-dimensional FNAA, a suitable algorithm is described in [14]. For each of the cases of 4-dimensional and 6-dimensional FNAA's, one can apply the following algorithm for selecting a random pair of vectors G and H setting a hidden commutative group of order q^2 possessing 2-dimensional cyclicity:

1. Generate a random invertible vector R .
2. Calculate the vector $G' = R^{p(p+1)}$.
3. If the vector G' is a scalar vector or has order $\omega_{G'} < p - 1$, then go to step1.

4. Generate at random an integer k and a primitive element α modulo p and compute the scalar vector $L = \alpha E$ and the vector $H' = G'^k L$.

5. Calculate the vectors $G = G'^2$ and $H = H'^2$.

6. Output the pair of vectors $\langle G, H \rangle$, each of which possesses order q , as a minimum generator system of a hidden group possessing 2-dimensional cyclicity and having order q^2 .

The public key represents the set of six vectors Y_1, Z_1, Y_2, Z_2, T , and U and is calculated as follows:

1. Generate at random a minimum generator system $\langle G, H \rangle$ of a commutative primary group $\Gamma_{\langle G, H \rangle}$ possessing the 2-dimensional cyclicity.

2. Using the invertibility conditions (1) and (2), generate random invertible vectors A, B, D , and F satisfying the following inequalities $AB \neq BA, AD \neq DA, AF \neq FA, BD \neq DB, BF \neq FB, DF \neq FD, AG \neq GA, BG \neq GB, GD \neq DG$, and $GF \neq FG$. Then calculate the vectors A^{-1}, B^{-1}, D^{-1} , and F^{-1} .

3. Generate random integers $y_1, y_2, z_1, z_2 \in GF(p)$ and calculate the vectors Y_1, Z_1, Y_2, Z_2, T , and U :

$$\begin{aligned} Y_1 &= AG^{y_1} B, & Z_1 &= FH^{z_1} A^{-1}; \\ Y_2 &= DH^{y_2} B, & Z_2 &= FG^{z_2} D^{-1}; \\ T &= AHB, & \text{and } U &= FGD^{-1}. \end{aligned} \tag{3}$$

The size of public key is equal to ≈ 6144 bits (768 bytes) in the case of the 4-dimensional FNAA used as algebraic support and to ≈ 4608 bits (576 bytes) in the case of the 6-dimensional FNAA. The integers y_1, y_2, z_1, z_2 and vectors G, H, A, B^{-1}, D^{-1} , and F^{-1} represent a private key having the size equal to ≈ 7168 bits (896 bytes) in the case $m = 4$ and to ≈ 5120 bits (640 bytes) in the case $m = 6$.

3.2 The signature generation procedure

Suppose the owner of the public key $(Y_1, Z_1, Y_2, Z_2, T, U)$ is to compute a signature to the electronic document M , using some specified h -bit hash-function f (where $h = 512$ bits and $h = 256$ bits for the cases of 4-dimensional and 6-dimensional FNAA's, correspondingly). Then the

digital signature is computed using the secret values $y_1, y_2, z_1, z_2, G, H, A, B^{-1}, D^{-1}$, and F^{-1} and the following algorithm:

1. Generate at random integers $k < q$ and $t < q$ and compute the vector

$$R = AG^k H^t D^{-1}.$$

2. Compute the hash-function value $e = e_1 || e_2$ (the first signature element) from the document M to which the vector R is concatenated: $e = e_1 || e_2 = f(M, R)$, where e_1 and e_2 are $(h/2)$ -bit non-negative integers.

3. Calculate the integers n and u :

$$n = \frac{k - y_1 e_1 - z_2 e_2 - 1}{e_1 + e_2 + 1} \bmod q;$$

$$u = \frac{t - z_1 e_1 - y_2 e_2 - 1}{e_1 + e_2 + 1} \bmod q.$$

4. Calculate the second signature element S :

$$S = B^{-1} G^n H^u F^{-1}.$$

The size of the output signature (e, S) is equal to ≈ 1280 bits (160 bytes) in the case $m = 4$ and to ≈ 896 bits (112 bytes) in the case $m = 6$. Computational difficulty w of the signature computation procedure is roughly equal to four exponentiation operations in the FNAA used as algebraic support of the signature scheme. For the case of 4-dimensional FNAA, we have $w \approx 12,288$ multiplications modulo a 257-bit prime or $w \approx 49,152$ multiplications modulo a 129-bit prime. For the case of 6-dimensional FNAA, we have $w \approx 27,648$ multiplications modulo a 129-bit prime or $w \approx 6,912$ multiplications modulo a 257-bit prime.

3.3 Signature verification procedure

The verification of the signature (e, S) to the document M is performed using the public key $(Y_1, Z_1, Y_2, Z_2, T, U)$ as follows:

1. Calculate the vector R' :

$$R' = (Y_1 S Z_1)^{e_1} (T S U) (Y_2 S Z_2)^{e_2}.$$

2. Compute the hash-function value e' from the document M to which the vector R' is concatenated: $e' = f(M, R')$.

3. If $e' = e$, then the signature is genuine. Otherwise reject the signature.

The computational difficulty w' of the signature generation procedure is roughly equal to two exponentiation operations in the m -dimensional FNAA used as algebraic support of the signature scheme. For the case $m = 4$, we have $w' \approx 6,144$ multiplications modulo a 257-bit prime or $w' \approx 24,576$ multiplications modulo a 129-bit prime. For the case $m = 6$, we have $w' \approx 13,824$ multiplications modulo a 129-bit prime or $w' \approx 3,456$ multiplications modulo a 257-bit prime.

Correctness proof of the signature algorithm consists in proving that the correctly computed signature (e, S) will pass the verification procedure as genuine signature. Indeed, taking into account that the order of the mutually permutable vectors G and H is equal to the prime q , we have the following:

$$\begin{aligned}
 R'_1 &= (Y_1SZ_1)^{e_1} (TSU) (Y_2SZ_2)^{e_2} = \\
 &= (AG^{y_1}BB^{-1}G^nH^uF^{-1}FH^{z_1}A^{-1})^{e_1} (AHBB^{-1}G^nH^uF^{-1}FGD^{-1}) \times \\
 &\quad \times (DHY_2BB^{-1}G^nH^uF^{-1}FG^{z_2}D^{-1})^{e_2} = \\
 &= (AG^{y_1+n}H^{u+z_1}A^{-1})^{e_1} (AHG^nH^uGD^{-1}) (DHY_2+uG^{m+z_2}D^{-1})^{e_2} = \\
 &= AG^{e_1(y_1+n)}H^{e_1(u+z_1)}A^{-1}AG^{m+1}H^{u+1}D^{-1}DH^{e_2(y_2+u)}G^{e_2(n+z_2)}D^{-1} = \\
 &= AG^{e_1(y_1+n)+n+1+e_2(n+z_2)}H^{e_1(u+z_1)+u+1+e_2(y_2+u)}D^{-1} = \\
 &= AG^{m(e_1+e_2+1)+e_1y_1+e_2z_2+1}H^{u(e_1+e_2+1)+e_1z_1+e_2y_2+1}D^{-1} = \\
 &= AG^{\frac{k-y_1e_1-z_2e_2-1}{e_1+e_2+1}(e_1+e_2+1)+e_1y_1+e_2z_2+1} \times \\
 &\quad \times H^{\frac{t-z_1e_1-y_2e_2-1}{e_1+e_2+1}(e_1+e_2+1)+e_1z_1+e_2y_2+1}D^{-1} = \\
 &= AG^kH^tD^{-1} = R \Rightarrow f(M, R') = f(M.R) \Rightarrow e' = e.
 \end{aligned}$$

4 Discussion

Like the known HDLP-based signature schemes, the proposed algorithm uses exponentiation operations in a hidden group. However, in the HDLP-based algorithms, exponentiations define the HDLP; whereas, in the introduced algorithm, the exponentiation operations

are used to set the randomization vector R in a form for which one can find a solution of the vector verification equation with three entries of the variable S (possibility to find a solution is also connected with the representation of the public-key elements in the form of formulas (3)). The random integers $y_1, z_1, y_2,$ and z_2 that are degrees of respective exponentiation operations are used only in the framework of a technique for selecting operations are used only in the framework of a technique for selecting random vectors $G' = G^{y_1}, H' = H^{z_1}, H'' = H^{y_2},$ and $G'' = G^{z_2}$ from the hidden group. This technique reduces the number of exponentiations when generating a signature. Actually, the developed signature scheme can be modified so that the randomization vector R is calculated as $R = G^{k_1} H^{t_1} G'^{k_2} H'^{t_2} G''^{k_3} H''^{t_3}$ (where integers $k_1, k_2, k_3, t_1, t_2,$ and t_3 are generated at random), and for generating a signature, the values $y_1, z_1, y_2,$ and z_2 are not needed. This illustrates that the HDLP is not an underlying computationally hard problem in the proposed signature scheme.

To forge a signature, a forger can propose an attack connected with the computation of the private key (or alternative private key) using the following system of 11 quadratic vector equations (defined by formulas (3)) with 10 unknowns $A, B^{-1}, D, F, G, G', G'', H, H', H''$:

$$\begin{cases} Y_1 B^{-1} = AG'; & Y_2 B^{-1} = DH''; \\ Z_1 A = FH'; & Z_2 D = FG''; \\ TB^{-1} = AH; & UD = FG; \\ GG' = G'G; & GG'' = G''G; \\ GH = HG; & GH' = H'G; \\ GH'' = H''G, \end{cases} \quad (4)$$

where the last five equations define the pairwise permutability of the vectors G, G', G'', H, H', H'' . The system (4) reduces to i) the system of 44 quadratic equations (over the field $GF(p)$) with 40 unknowns for the case of the 4-dimensional FNAA used as algebraic support and ii) the system of 66 quadratic equations (over the field $GF(p)$) with 60 unknowns for the case of the 6-dimensional FNAA. Evidently, the system (4) has (on construction) at least one solution.

For the case $m = 4$, in the system of 44 quadratic equations in the field $GF(p)$ every one of the lasts contains 4 members. A comparatively small number of members is defined by the use of the algebra defined by a sparse BVMT. Therefore, for this case, we propose to define the 4-dimensional FNAA over the finite field $GF(p)$ with 257-bit characteristic p ; whereas, for the case of 6-dimensional FNAA used as algebraic support, we specify using 129-bit characteristic p . For the case $m = 6$, in the system of 66 quadratic equations in the field $GF(p)$, every one of the lasts contains 12 members.

Taking into account that for the signature algorithm Rainbow (one the finalists of the NIST competition), it is used hardness of solving a system of 96 quadratic equations in 188 variables, which is set in a finite field of order 2^8 ; one can suppose that in the $m = 4$ ($m = 6$) version of the proposed signature algorithm, it is sufficient to use a 128-bit (64-bit) characteristic p . However, a detailed security analysis of the proposed signature scheme should be performed as independent research work.

The computational difficulty of the systems of quadratic equations set over a finite field is used in multivariate public-key cryptosystems [1], [2] that are attractive as post-quantum ones. However, the estimation of the computational difficulty of solving the system (4) represents a topic of individual study.

A rough comparison of the proposed signature scheme with some known candidates for post-quantum signature schemes is presented in Table 3, where a procedure execution time* is estimated in multiplications in $GF(p)$ with 129-bit characteristic. One can see that the developed signature algorithm has advantages in the size of parameters and performance (lower execution time) against algorithms Falcon [17], Dilithium [18], and Rainbow [3] which are finalists of the NIST's competition on the development of the post-quantum signature standard [19]. The HDLP-based signature algorithms introduced in [14], [16] look more practical. However, one can expect that the proposed method for development of the post-quantum signature algorithms has an internal potential to optimize the design and to get more practical signature schemes. For example, if further research will show that finding a solution to the system (4) is an infeasible compu-

Table 3. Comparison of the proposed and known signature schemes

Signature scheme	signature size, bytes	public-key size, bytes	signature generation time*	signature verification time*
HDLP-based [14]	96	384	$\approx 49,200$	$\approx 36,800$
HDLP-based [16]	96	384	$\approx 12,400$	$\approx 24,800$
Falcon [17]	1280	1793	$\approx 80,000$	$\approx 160,000$
Dilithium [18]	2701	1472	$\approx 200,000$	$\approx 350,000$
Rainbow [3]	1,632	1,683 kB	–	–
Proposed ($m = 4$)	160	768	$\approx 49,152$	$\approx 24,576$
Proposed ($m = 6$)	112	576	$\approx 27,648$	$\approx 13,824$

tational problem in the case of 64-bit (128-bit) prime p for the case of 6-dimensional (4-dimensional) FNAA used as algebraic support.

5 Conclusion

The proposed versions of the signature algorithm on FNAAAs can be attributed to the cryptoschemes with a hidden group and to the MP-KCs, but not to the HDLP-based signature algorithms. First shown as the development of digital signature algorithms on FNAAAs leads to the creation of a MPKC. The introduced digital signature scheme has principal differences from both the known MPKCs and the known HDLP-based signature algorithms and can serve as a promising starting point that can be used for preparing a new application for participating in the NIST competition on developing a standard on a post-quantum signature algorithm (NIST is going to propose such possibility at the fourth round of its competition [19]).

References

- [1] Q. Shuaiting, H. Wenbao, Li Yifa, and J. Luyao, “Construction of Extended Multivariate Public Key Cryptosystems,” *International*

- Journal of Network Security*, vol. 18, no. 1, pp. 60–67, 2016.
- [2] D. Jintai and S. Dieter, “Multivariable Public Key Cryptosystems,” 2004. [Online]. Available: <https://eprint.iacr.org/2004/350.pdf>. Accessed on: December 14, 2021.
 - [3] Rainbow Signature. One of three NIST Post-quantum Signature Finalists, 2021. [Online]. Available: <https://www.pqcraibow.org/>. Accessed December 14, 2021.
 - [4] A. S. Kuzmin, V. T. Markov, A. A. Mikhalev, A. V. Mikhalev, and A. A. Nechaev, “Cryptographic Algorithms on Groups and Algebras,” *Journal of Mathematical Sciences*, vol. 223, no. 5, pp. 629–641, 2017.
 - [5] D. N. Moldovyan, “Post-quantum public key-agreement scheme based on a new form of the hidden logarithm problem,” *Computer Science Journal of Moldova*, vol. 27, no. 1(79), pp. 56–72, 2019.
 - [6] A. A. Moldovyan and N. A. Moldovyan, “Post-quantum signature algorithms based on the hidden discrete logarithm problem,” *Computer Science Journal of Moldova*, vol. 26, no. 3(78), pp. 301–313, 2018.
 - [7] N. A. Moldovyan and A. A. Moldovyan, “Finite Non-commutative Associative Algebras as carriers of Hidden Discrete Logarithm Problem,” *Bulletin of the South Ural State University. Ser. Mathematical Modelling, Programming & Computer Software*, vol. 12, no. 1, pp. 66–81, 2019. DOI: 10.14529/mmp190106.
 - [8] D. N. Moldovyan, A. A. Moldovyan, and N. A. Moldovyan, “A new design of the signature schemes based on the hidden discrete logarithm problem,” *Quasigroups and Related Systems*, vol. 29, no. 1, pp. 97–106, 2021.
 - [9] D. N. Moldovyan, A. A. Moldovyan, and N. A. Moldovyan, “Digital signature scheme with doubled verification equation,” *Computer Science Journal of Moldova*, vol. 28, no. 1(82), pp. 80–103, 2020.

- [10] N. A. Moldovyan and A. A. Moldovyan, “Candidate for practical post-quantum signature scheme,” *Vestnik of Saint Petersburg University. Applied Mathematics. Computer Science. Control Processes*, vol. 16, no. 4, pp. 455–461, 2020. <https://doi.org/10.21638/11701/spbu10.2020.410>.
- [11] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on quantum computer,” *SIAM Journal of Computing*, vol. 26, pp. 1484–1509, 1997.
- [12] A. Ekert and R. Jozsa, “Quantum computation and Shor’s factoring algorithm,” *Rev. Mod. Phys.*, vol. 68, p. 733, 1996.
- [13] R. Jozsa, “Quantum algorithms and the fourier transform,” *Proc. Roy. Soc. London Ser A*, vol. 454, pp. 323–337, 1998.
- [14] D. N. Moldovyan, “A practical digital signature scheme based on the hidden logarithm problem,” *Computer Science Journal of Moldova*, vol. 29, no. 2(86), pp. 206–226, 2021.
- [15] N. A. Moldovyan, “Unified Method for Defining Finite Associative Algebras of Arbitrary Even Dimensions,” *Quasigroups and Related Systems*, vol. 26, no. 2, pp. 263–270, 2020.
- [16] N. A. Moldovyan and A. A. Moldovyan, “Digital signature scheme on the 2×2 matrix algebra,” *Vestnik of Saint Petersburg University. Applied Mathematics. Computer Science. Control Processes*, vol. 17, no. 3, pp. 254–261, 2021. <https://doi.org/10.21638/11701/spbu10.2021.303>.
- [17] “Fast-Fourier lattice-based compact signatures over NTRU,” [Online]. Available: <https://falcon-sign.info/>. Accessed on: December 14, 2021.
- [18] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, “CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme.” [Online]. Available: <https://eprint.iacr.org/2017/633.pdf>. Accessed on: December 14, 2021. (see also <https://pq-crystals.org/dilithium/index.shtml>).

- [19] D. Moody, “NIST Status Update on the 3rd Round.” [Online]. Available: 2021. <https://csrc.nist.gov/CSRC/media/Presentations/status-update-on-the-3rd-round/images-media/session-1-moody-nist-round-3-update.pdf>. Accessed on: December 14, 2021.

Dmitriy N. Moldovyan

Received December 14, 2021

Accepted June 07, 2022

ORCID: <https://orcid.org/0000-0001-5039-7198>

Department of Information Systems,
Saint Petersburg Electrotechnical University "LETI",
Prof. Popov, 5, St. Petersburg, 197022,
Russia
E-mail: mdn.spectr@mail.ru

Performance comparison of CPU and GPGPU calculations using three simple case studies

Branislav Lipovský, Slavomír Šimoňák

Abstract

In this work, we have prepared and analyzed three case studies comparing CPU and GPGPU calculations. After briefly introducing the topic of parallel programming by means of contemporary CPU and GPGPU technologies, we provide an overview of selected existing works closely related to the topic of the paper. For each of the case studies, a set of programs has been implemented using the following technologies: pure CPU, CPU SIMD, CPU multi-threaded, CPU multi-threaded with SIMD instructions, and GPU - Cuda. We also illustrate the basic idea of the operation of selected algorithms using code snippets. Subsequently, the particular implementations are compared, and obtained results are evaluated and discussed.

Keywords: CUDA, Multi-threading, SIMD, Matrix multiplication, Sobel operator, Template matching.

MSC 2020: 68W10, 68U10, 94A08.

ACM CCS 2020: Computing methodologies – Parallel computing methodologies – Parallel algorithms – Vector / streaming algorithms, Computing methodologies – Computer graphics – Image manipulation.

1 Introduction

In today's world, we strive to integrate modern computer systems into our lives. This is done simply because of the general desire for modernization, to increase comfort, to speed up various processes, or for automation. All these processes need ever-increasing computing power.

When faced with such new tasks, people may not know which technologies to use, which ones will be the easiest to use, which ones will be the most appropriate, and which ones they should avoid [1].

Moore's Law tells us that the number of transistors in an integrated circuit doubles every 2 years.

Huang's law, as observed by Jensen Huang, CEO of Nvidia, says that graphics card performance more than doubles every 2 years. With the ever-increasing performance of graphics cards, people often ask about CPU usage cases, SIMD CPU cases, or multi-core calculations versus GPU calculations.

A common practice, we can observe presently quite often, is using the GPU implementations for increasing the effectiveness of solutions to a wide range of computational problems [2]–[4]. However, contemporary CPUs also offer solid options for efficient execution of parallel programs [5], [6]. In this work, we would like to explore several practical problems in order to find if parallel CPU implementation could be more efficient than the GPU implementation.

2 Parallel programming

Parallel computing is a type of calculation in which many calculations or processes are performed at the same time. Large problems can often be divided into smaller ones, which can then be solved simultaneously [7].

The main point of parallel programming is the use of concurrency. Concurrency exists in a computational problem if the problem can be decomposed into subproblems that can be safely run at the same time. In order to take advantage of concurrency, it must be possible to structure the code itself so that the problems run at the same time. Most of the major computational problems involve usable concurrency. Parallel programming presents unique challenges:

- Often, concurrent subproblems can contain various dependencies that will need to be identified and managed properly.
- The order of execution of subproblems can change the results of calculations in non-deterministic ways.

- It is necessary that the cost of concurrency control does not exceed the cost of running the program itself.
- Balancing the work between computing units may not be easy.
- Parallel algorithm that is very efficient on one platform may not be as effective on other platforms [8].

Parallel programming on the CPU can leverage the following methods.

2.1 SIMD Parallelism

Single instruction multiple data (SIMD) allows a single instruction to operate on multiple data – perform multiple calculations at once [6]. The most widely known form of SIMD parallelism is vector instructions¹.

2.2 Simultaneous Multithreading

Simultaneous Multithreading (SMT) allows an operating system to see a single processor core as a set of logical processors. The operating system can then schedule threads on these logical processors which will every cycle compete for the functional units of a processor core [6].

2.3 Multicore Processors

Multicore processors integrate multiple execution units into a single processor chip. The operating system can see multiple logical processors, all of which have their own execution units [6].

3 General purpose computing on graphics processing units

General purpose computing on graphics processing units (GPGPU) is a method for performing large-scale calculations using a graphics

¹https://cvw.cac.cornell.edu/vector/overview_simd

processor. Nowadays, we can also use GPGPU to perform, in addition to generic calculations, training of various machine learning models, image / video manipulation, cryptography, and even emulation [9]. For a program to be suitable for GPGPU it needs to meet the following criteria:

- parallelism – the ability to process multiple data at once
- throughput – the ability to process large amounts of data [10].

In order to utilize GPGPU, we need to use a GPGPU programming framework. The most popular GPGPU programming framework is Cuda.

Cuda is a platform developed by NVIDIA for utilizing graphics cards for non-graphic computations. Cuda C++ is a variant of C++ language that allows users to create and execute code on the GPU²³.

4 Related work

In this section, we summarize the results of some of the already existing comparisons. These comparison works also had an impact on the selection of problems we decided to use in our work.

4.1 Performance comparison of FPGA, GPU and CPU in image processing

In 2009, S. Asano, T. Maruyama, and Y. Yamaguchi compared the performance of FPGA, GPU, and CPU in various image processing tasks, specifically two-dimensional filters, stereo-vision, and k-means clustering. In their work, they found that GPU can match the performance of FPGA, however only when naive computation methods are used, otherwise the GPU may even be slower than CPU. In their tests, the CPU was reaching 1/12 – 1/7 the performance of FPGA [11].

²<https://developer.nvidia.com/cuda-zone>

³<https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>

4.2 Comparative performance of GPU, SIMD and OpenMP systems for raw template matching in computer vision

In 2011, J. Méndez, J. Lorenzo-Navarro, and M. Castrillón Santana compared the performance of CPU and GPU in raw template matching. In their research, they found the CPU to achieve better performance compared to the GPU. They, however, also note that newer GPU architectures could outperform the CPU computations given bigger mask sizes [12].

4.3 Comparing CPU and GPU Implementations of a Simple Matrix Multiplication Algorithm

In 2017, T. Dobravec and P. Bulić compared the performance between CPU and GPU in matrix multiplication. In their testing, the GPU outperformed CPU in all cases. It needs to be noted, however, that their testing hardware utilized integrated GPU with shared memory, and, as a result of that data transfer between the GPU and CPU, had minimal to no impact on performance [13].

4.4 Implementation of Sobel filter using CUDA

In 2021, A. Akasapu, V. Sailaja, and G. Prasad implemented the sobel filter using Cuda GPGPU programming framework. They found that, for smaller images, the CPU gives faster results, and, for larger scale and high-resolution data sets, it is better to use the GPU [4].

5 Testing environment

The testing system used was a gaming laptop Lenovo Legion 5 (2021), with the following configuration:

- Processor: Ryzen 7 5800H
- RAM: 64GB 3200Mhz CL20 Dual-Channel Dual Rank
- GPU: NVIDIA GeForce RTX 3070 Laptop 8GB VRAM

The testing system was using the following software:

- Operating system: Windows 10 Education
- Development environment: Visual studio 2019
- GPGPU programming framework: CUDA Toolkit 11.6

6 Selected problems for comparisons

6.1 Matrix multiplication

Matrix multiplication is one of the most important matrix operations. It is often used in network theory, solving linear equation systems, coordinate system transformations, and population modeling. When multiplying matrices, the number of rows in the first matrix must be the same as the number of columns in the second matrix. When calculating a certain number of the resulting matrix, we calculate the scalar product from the row of the first matrix and the columns of the second matrix in which the given number is located [14].

6.2 Raw template matching

Template matching is a method of finding and locating a template image in a larger image. Raw template matching is the simplest form of template matching in which we try to find an exact match of template in the larger image.

6.3 Edge detection

Edge detection is a common step in multiple image processing algorithms. In this work, we will be using the sobel operator⁴ for edge detection.

⁴<https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm>

7 Implementation

For each of algorithms chosen for comparison, a set of programs has been implemented:

- basic example on CPU
- CPU + SIMD
- multiple threads CPU
- multiple threads CPU + SIMD
- GPU - cuda

7.1 Matrix multiplication

For matrix multiplication, we have chosen a simplified algorithm that has hardcoded matrix sizes and matrix width and height of the same size.

In the actual implementation, we first swap columns of the second matrix with its rows and then we calculate dot product of the two matrixes. We verified the calculated results from our implementations against simple reference implementation.

```
void CPU_matrix_mult(float* h_a, float* h_b,
                    float* h_result) {
    for (int i = 0; i < MATRIX_SIZE; ++i)
    {
        for (int j = 0; j < MATRIX_SIZE; ++j)
        {
            float tmp = 0.0f;
            for (int h = 0; h < MATRIX_SIZE; ++h)
            {
                tmp += h_a[i * MATRIX_SIZE + h] *
                    h_b[h * MATRIX_SIZE + j];
            }
            h_result[i * MATRIX_SIZE + j] = tmp;
        }
    }
}
```

```
}  
}
```

At first, in our implementation, we used C++ `std::vector` to store the matrixes; however, because of performance and no native CUDA support, we changed to C-style arrays. Auto-vectorization worked with our compiler (MSVC) and used a similar approach to our initial SIMD implementation:

```
float dotProduct(float* arr1 , float* arr2) {  
    float ret ;  
    __m256 sum = __mm256_setzero_ps() ;  
    for (int i = 0; i < MATRIX_SIZE; i += 8) {  
        __m256 vec_arr1 = __mm256_loadu_ps(&arr1[i]);  
        __m256 vec_arr2 = __mm256_loadu_ps(&arr2[i]);  
        sum = __mm256_add_ps(sum ,  
            __mm256_mul_ps(vec_arr1 , vec_arr2)  
        );  
    }  
    float buffer [8];  
    __mm256_storeu_ps(buffer , sum);  
    ret = buffer [0] + ... + buffer [7];  
    return ret ;  
}
```

We further optimized this implementation by computing two sets of vectors instead of one.

In our GPU program, we used tiling for efficient matrix multiplication on GPU⁵.

As we can see in the graph (Fig. 1), with high matrix sizes – starting at 2048*2048 – our GPU program is faster. On the other hand, in lower matrix sizes, our CPU program is faster; in fact, at matrix sizes lower than 1024*1024, even the pure CPU program is faster than the GPU program.

⁵<https://penny-xu.github.io/blog/tiled-matrix-multiplication>

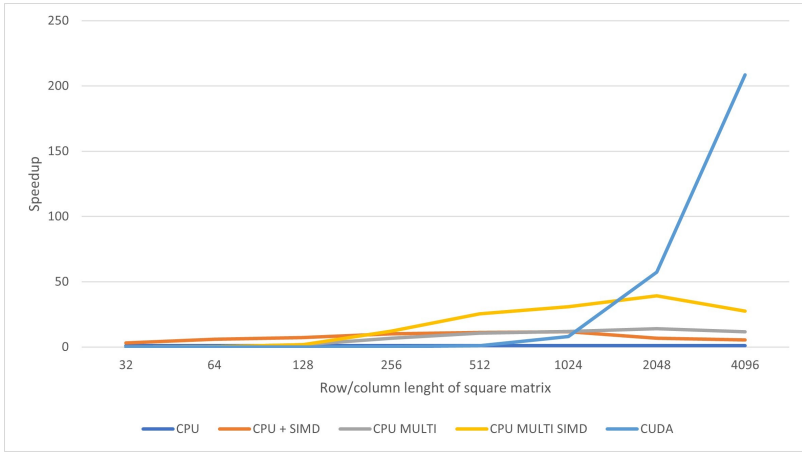


Figure 1. Graph of speedups achieved by our implementations of matrix multiplication.

7.2 Raw template matching

In our program, we load two images – the template and the image to search for the template. We loaded the images using bitmap library⁶ and we also use that library to convert input images to grayscale.

In the implementation, we first iterate through every group of pixels that the template can fit in and then we compare the pixels in the given group with pixels from the template:

```

int compareTemplate(Image img, Image templ,
                    int posx, int posy) {
    for (int j = 0; j < templ.height; j++) {
        for (int i = 0; i < templ.width; i++) {
            if (img.data[(posy+j)*img.width+posx+i] !=
                templ.data[j * templ.width + i]) {
                return 0;
            }
        }
    }
}

```

⁶<https://github.com/wernsey/bitmap>

```

}
return 1;
}

```

Our compiler (MSVC) could not auto-vectorize this algorithm. In our SIMD implementation, we created a function to compare two vectors, and we used this function to compare multiple pixels at once:

```

bool vec_equal(__m256i a, __m256i b) {
    __m256i pcmp = _mm256_cmpeq_epi32(a, b);
    unsigned bitmask = _mm256_movemask_epi8(pcmp);
    return (bitmask == 0xffffffffU);
}

```

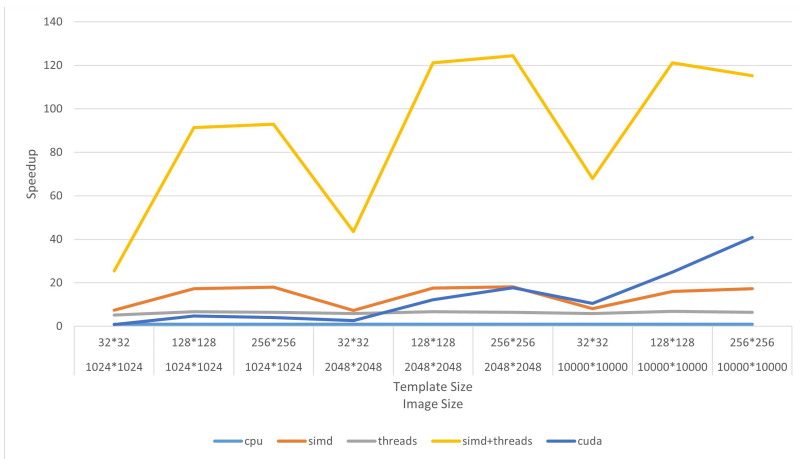


Figure 2. Graph of speedups achieved by our implementations of raw template matching.

From the graph in Fig. 2, we can make multiple observations:

- multi-threaded program is providing stable 6-7 times speedup compared to a single core;
- SIMD speedup is dependent on mask size: on 32*32 mask, there is 7-8 times speedup, on 128*128 and 256*256 – there is 16-18

times speedup;

- multithreaded program with SIMD is achieving up to 124 times speedup;
- GPU program, even though it is faster than CPU only, is reaching only slightly better performance and starts to be faster than SIMD only program at image size of 10000*10000.

7.3 Edge detection

Our program loads either one image in bmp format or a set of images in gif format using the bitmap library. The Sobel operator works by sliding two 3*3 convolution kernels over the input image and calculating gradients. Parts of the convolution kernels involve calculation which multiplies by zero, so we decided to unroll the calculations and skip the mentioned multiplies by zero:

```
sumX-=img [ frame ] . data [ x-1+(y-1)*img [ frame ] . width ] ;
sumX-=2*img [ frame ] . data [ x-1+(y) *img [ frame ] . width ] ;
sumX-=img [ frame ] . data [ x-1+(y+1)*img [ frame ] . width ] ;
sumX+=img [ frame ] . data [ x+1+(y-1)*img [ frame ] . width ] ;
sumX+=2*img [ frame ] . data [ x+1 +(y) *img [ frame ] . width ] ;
sumX+=img [ frame ] . data [ x+1 +(y+1)*img [ frame ] . width ] ;
```

Our compiler (MSVC) could not auto-vectorize this algorithm. In our SIMD implementation, we at first load 16 uint8_t numbers into __m128i vector, zero extend that vector to 256 bits, add/subtract the numbers, and then we used saturation to convert int16_t numbers to uint8_t. Example code of loading and adding/subtracting a vector of numbers:

```
vec_sumX=_mm256_sub_epi16(
    vec_sumX, _mm256_cvtepu8_epi16(
        _mm_loadu_epi8(
            &img [ frame ] . data [ x-1 +(y-1)*width ]
        )
    )
);
```

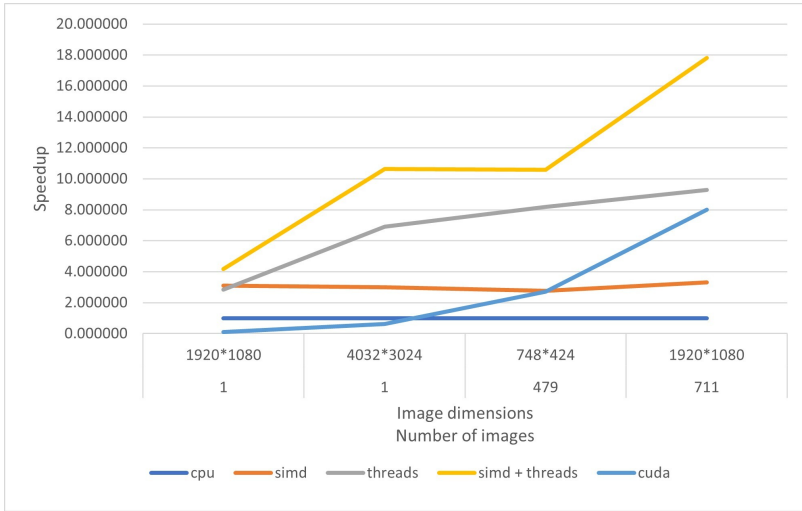


Figure 3. Graph of speedups achieved by our implementations of sobel algorithm.

In the graph in Fig. 3, we can see that our multithreaded program with SIMD instructions is always achieving greater speedup compared to our GPU program. Another result we can observe is that with single images the GPU program is slower than CPU program and only starts to achieve better performance when processing multiple images.

8 Evaluation

If we compare our matrix multiplication results with the results from T. Dobravec and P. Bulić [13], we can observe that in their case the calculation on the GPU was always faster. However, this comparison is not entirely appropriate, as they performed calculations on the NVIDIA GeForce 9400M graphics card that uses shared memory, so data transfer between CPU and GPU was virtually instantaneous. Furthermore, we can compare the results of work from K. Fatahalian, J. Sugerman, and P. Hanrahan [15], where they achieved results in which it was always

more appropriate to use a CPU when multiplying matrices. However, this work is relatively old (2004), and since then there have been many significant architectural changes in graphics cards.

We can observe that our template matching results agree with the conclusions of the work by J. Méndez, J. Lorenzo-Navarro, and M. Cas-trillón Santana [12] that the CPU achieves better results with small template dimensions, and only with large template/input dimensions will the GPU achieve better results. These results could be in part caused by the need to use synchronization in our GPU implementation, which causes performance penalty. Another factor could be using `uint8_t` data format for storing pixel values, which is not the preferred workload for GPUs.

Our edge detection results are consistent with the conclusions of work by A. Akasapu, V. Sailaja, and G. Prasad [4], where they claim that small images are achieving better performance on CPU, and for large images in large data sets it is better to use GPU. In this case, even though we did not use synchronization in our GPU implementation, we still used `uint8_t` data format for storing pixel values, which made this algorithm not perform as well on GPU as it did on CPU.

9 Conclusion

In this work, we briefly introduced the topic of parallel programming by means of contemporary CPU and GPGPU technologies. After this brief introduction we explored some of the existing works comparing CPU and GPU calculations. Based on our findings, we prepared three case studies and created programs for each type of parallelism. From the results obtained, we found that even in the cases that should be perfect for the GPU, like matrix multiplication, with small enough matrix sizes, calculations on the CPU could be more suitable. On the other hand, we found that for some image processing tasks, like edge detection and template matching, our GPU programs could not achieve better results than our CPU multithreaded programs with SIMD instructions despite both programs processing a relatively large number of pixels.

So we can conclude, that currently there certainly is a class of problems for which a parallel CPU implementation would be more efficient

than the GPU implementation. According to the results achieved in this work, we can formulate the following recommendations:

- Matrix multiplication: for matrices with dimensions up to 1024 x 1024, parallel CPU implementations were faster. However, for matrices of higher sizes, the GPU implementation is recommended, as it can achieve great speedup improvements over the CPU implementations.
- Raw template matching: SIMD multithreaded CPU implementation is recommended, as it achieved the best performance in all tested scenarios.
- Edge detection: SIMD multithreaded CPU implementation is recommended, as it achieved the best performance in all tested scenarios.

As we only performed our comparisons using three simple case studies, in the future it would be beneficial to include more diverse algorithms and create comparisons using bigger datasets. It would be also interesting to study the impact of the operating system and the compiler used on such comparisons. The additional research could help to better describe the class of problems for which CPU parallel implementation would be a better option. Another possibility to explore may be a combination of CPU and GPU parallel techniques in order to gain further efficiency improvements [16].

References

- [1] B. Lipovský, “Comparison of use cases of simd cpu and gpgpu calculations,” Master’s thesis, Technical University of Košice, 2022, (in Slovak).
- [2] J. R. Cheng and M. Gen, “Parallel Genetic Algorithms with GPU Computing,” in *Industry 4.0*, T. Bányai, A. Petrillo, and F. De Felice, Eds. Rijeka: IntechOpen, 2020, ch. 6. [Online]. Available: <https://doi.org/10.5772/intechopen.89152>

- [3] M. Arora, S. Nath, S. Mazumdar, S. B. Baden, and D. M. Tullsen, "Redefining the Role of the CPU in the Era of CPU-GPU Integration," *IEEE Micro*, vol. 32, no. 6, pp. 4–16, 2012.
- [4] A. Akasapu, V. Sailaja, and G. Prasad, "Implementation of sobel filter using CUDA," *IOP Conference Series: Materials Science and Engineering*, vol. 1045, no. 1, p. 012016, Feb. 2021. [Online]. Available: <https://doi.org/10.1088/1757-899x/1045/1/012016>
- [5] T. Singh, D. K. Srivastava, and A. Aggarwal, "A novel approach for CPU utilization on a multicore paradigm using parallel quick-sort," in *2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT)*, 2017, pp. 1–6.
- [6] T. Rauber and G. R unger, *Parallel programming*. Springer, 2013.
- [7] G. S. Almasi, *Highly Parallel Processing*. Reading, PA: Benjamin-Cummings Publishing Co., Subs. of Addison Wesley Longman, Nov. 1987.
- [8] T. G. Mattson, B. Sanders, and B. Massingill, *Patterns for parallel programming*. Pearson Education, 2004.
- [9] P. Jakub o and S.  imo ak, "Utilizing gpgpu in computer emulation," *Journal of Information and Organizational Sciences*, vol. 36, no. 1, pp. 39–53, 2012.
- [10] T. A. C. Center, "8 things you should know about gpgpu technology," 2017. [Online]. Available: <https://web.archive.org/web/20211126072222/https://www.tacc.utexas.edu/documents/13601/88790/8Things.pdf>
- [11] S. Asano, T. Maruyama, and Y. Yamaguchi, "Performance comparison of fpga, gpu and cpu in image processing," in *2009 International Conference on Field Programmable Logic and Applications*, 2009, pp. 126–131.
- [12] J. M endez, J. Lorenzo-Navarro, and M. Castrill n Santana, "Comparative performance of gpu, simd and openmp systems for raw template matching in computer vision," pp. 9–15, 01 2011.
- [13] T. Dobravec and P. Buli , "Comparing cpu and gpu implementations of a simple matrix multiplication algorithm," *Interna-*

tional Journal of Computer and Electrical Engineering, vol. 9, pp. 430–438, 01 2017.

- [14] B. D. Hahn and D. T. Valentine, “Chapter 6 - matrices and arrays,” in *Essential MATLAB for Engineers and Scientists (Seventh Edition)*, seventh edition ed., B. D. Hahn and D. T. Valentine, Eds. Academic Press, 2019, pp. 127–161. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9780081029978000129>
- [15] K. Fatahalian, J. Sugerman, and P. Hanrahan, “Understanding the efficiency of gpu algorithms for matrix-matrix multiplication,” in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, ser. HWWS '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 133–137. [Online]. Available: <https://doi.org/10.1145/1058129.1058148>
- [16] V. Skorych and M. Dosta, “Parallel CPU–GPU computing technique for discrete element method,” *Concurrency and Computation: Practice and Experience*, vol. 34, no. 11, p. e6839, 2022. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.6839>

Branislav Lipovský¹, Slavomír Šimoňák²

Received July 01, 2022

Revised September 20, 2022

Accepted September 23, 2022

^{1,2} Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Letná 9, 042 00 Košice, Slovak Republic

¹ ORCID: <https://orcid.org/0000-0001-7079-7519>
E-mail: branislawlipovsky98@gmail.com

² ORCID: <https://orcid.org/0000-0001-6505-3160>
E-mail: slavomir.simonak@tuke.sk