Princess Sumaya University for Technology
جامعة الأميرة سمية للتكنولوجيا

صندوق دعم البحث العلمي
Scientific Research Support Fund

Jordanian Journal of Computers and Information Technology

March 2022 VOLUME 08 NUMBER 01
ISSN 2415 - 1076 (Online)
ISSN 2413 - 9351 (Print)

JJCIT

www.jjcit.org jjcit@psut.edu.jo

An International Peer-Reviewed Scientific Journal
Financed by the Scientific Research Support Fund

Jordanian Journal of Computers and Information Technology (JJCIT)

The Jordanian Journal of Computers and Information Technology (JJCIT) is an international journal that publishes original, high-quality and cutting edge research papers on all aspects and technologies in ICT fields.

JJCIT is hosted and published by Princess Sumaya University for Technology (PSUT) and supported by the Scientific Research Support Fund in Jordan.

AIMS AND SCOPE

The JJCIT aims to publish the most current developments in the form of original articles as well as review articles in all areas of Telecommunications, Computer Engineering and Information Technology and make them available to researchers worldwide. The JJCIT focuses on topics including, but not limited to: Computer Engineering & Communication Networks, Computer Science & Information Systems and Information Technology and Applications.

INDEXING

JJCIT is indexed in:

EDITORIAL BOARD SUPPORT TEAM

JJCIT ADDRESS

WEBSITE: www.jjcit.org
EMAIL: jjcit@psut.edu.jo
ADDRESS: Princess Sumaya University for Technology, Khalil Saket Street, Al-Jubaiha
B.O. BOX: 1438 Amman 11941 Jordan
TELEPHONE: +962-6-5359949
FAX: +962-6-7295534

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

CLUSTERING VIETNAMESE CONVERSATIONS FROM FACEBOOK PAGE TO BUILD TRAINING DATASET FOR CHATBOT

Trieu Hai Nguyen, Thi-Kim-Ngoan Pham, Thi-Hong-Minh Bui and Thanh- Quynh-Chau Nguyen

(Received: 26-Sep.-2021, Revised: 10-Dec.-2021, Accepted: 28-Dec.-2021)

ABSTRACT

The biggest challenge of building chatbots is training data. The required data must be realistic and large enough to train chatbots. We create a tool to get actual training data from Facebook messenger of a Facebook page. After text preprocessing steps, the newly obtained dataset generates FVnC and Sample dataset. We use the Retraining of BERT for Vietnamese (PhoBERT) to extract features of our text data. K-Means and DBSCAN clustering algorithms are used for clustering tasks based on output embeddings from PhoBERT$_{base}$. We apply V-measure score and Silhouette score to evaluate the performance of clustering algorithms. We also demonstrate the efficiency of PhoBERT compared to other models in feature extraction on the Sample dataset and wiki dataset. A GridSearch algorithm that combines both clustering evaluations is also proposed to find optimal parameters. Thanks to clustering such a number of conversations, we save a lot of time and effort to build data and storylines for training chatbot.

KEYWORDS

BERT, Clustering, Language models, Feature extraction, Word embeddings.

1. INTRODUCTION

Chatbot is certainly not an unfamiliar name in the field of Natural Language Processing (NLP). Previously, traditional chatbot simply interacts with the user *via* predefined rules, which means that the user is only allowed to enter by these rules to get answers. However, NLP chatbot is not only a word recognition algorithm, but it can also understand what the user is saying. It is one of the pioneering applications using Artificial Intelligence (AI); namely, NLP, to help humans interact with machine like humans with humans via Virtual Assistant autoresponder. Currently, there are many parties developing NLP chatbot, which can be mentioned as Google's DialogFlow, Watson of IBM and Rasa.

In the process of building NLP chatbots, all chatbots require real datasets for training bot. The training datasets can be large or small depending on the size and intelligence level of the chatbots. Raw training data can be collected from past conversations through social media, archived user chats, previous questions, email chains or live telephone transcripts. But, these data are messy, not in any structure or order and come from various sources collected with huge amounts of raw data. Thus, the first priority when constructing a chatbot is to transform those raw data into useful data for the purpose of training bot.

In order to fit chatbot building orientation, that raw dataset needs to be divided into specific intents, which serves to build conversations to train a chatbot. There are many ways to process that raw dataset into specific intents (topics, conversations). The first method to be mentioned is using Supervised Learning task [1] to classify intents. In particular, this method requires labeling for the input examples and then it predicts labels for remaining data in the raw dataset. In this case, label prediction corresponds to classified raw data into the intents that have been labeled previously. However, building and labeling manual intents on large datasets lead to big challenges for chatbot developers. With a simple raw dataset of about 8000 conversations, analyzing how many intents are created is a conundrum.

Instead, we can approach the above problem by using the second method, which is clustering similar raw data together into corresponding intents. The advantage of this approach is that it uses

Trieu H. N. (ORCID: 0000-0002-6086-9598), T-K-Ngoan P., T-H-Minh B. and T-Q-Chau N. are with Faculty of Information Technology, Nha Trang University, Nha Trang City, Vietnam. Emails: {trieunh, ngoanptk, minhbth, chauntq}@ntu.edu.vn.

Unsupervised Learning technique [1], which is only based on the features of the data to perform specific tasks such as clustering. As expected, this method has a significant effect on the analysis of raw data. It saves us a lot of time and effort to make a training dataset for chatbot. There are many clustering algorithms, such as K-Means, DBSCAN, BIRCH and Spectral clustering [2]-[3]. In this article, we use K-Means [4] and DBSCAN [5] techniques to cluster our dataset and consider that clustering is a downstream NLP task. Each technique has its own advantages and disadvantages. K-Means algorithm is a simple and fast-implementation algorithm, but it requires knowing the number of clusters to perform clustering whereas DBSCAN does not. Nevertheless, DBSCAN technique is more difficult to implement and requires finding the optimal parameters [5], [6], [7], which leads to drastically increased costs, especially for large datasets. Thus, we can combine the advantages of both techniques to serve the purpose of efficient clustering.

The input of clustering algorithms in particular and downstream NLP tasks in general is document embeddings extracted from the dataset. There are many ways to extract information from text datasets; for example, we can use traditional machine learning algorithm like TF-IDF [8], proposed word embedding models in recent years such as Word2Vec, GloVe [9]-[10], FastText [11] or popular language models like GPT-2 [12], BERT model and its variations [13], [14], [15]. In this work, we use BERT (Bidirectional Encoder Representations from Transformers), which is state-of-the-art embeddings [13] to extract features of documents. Recently, a clustering approach with the BERT model has been proposed by O. Gencoglu [16]. As suggested in [17], clustering techniques using pre-trained transformer language models are applied to short text clustering. The combination of word embeddings using BERT models and clustering algorithms to obtain topics was presented in [18]. Distinctively, PhoBERT represents pre-trained language models for Vietnamese, being used to embed our Vietnamese dataset [15]. V-measure score [19] and Silhouette score [20] are used to evaluate the performance of clustering algorithms as well as the feature extraction efficiency of the language models.

The aim of the present paper is to study and apply PhoBERT model to our Facebook Vietnamese conversations dataset, thereby deriving document embeddings in order to serve the clustering task. The combination of both K-Means and DBSCAN clustering algorithms is proposed by us to achieve the best clustering results on the actual dataset. The finding of these data clusters allows us to simplify and accelerate the building of a training dataset for chatbot. In Section 2, we recall some theories of Transformer and BERT architecture proposed by Vaswani et al. in [21] and Devlin et al. in [13], respectively. In Section 3, we offer an approach to apply PhoBERT to the clustering task from the idea of classification task [15], [22]. Next, we also recall clustering algorithms in machine learning and evaluation metrics for unsupervised learning algorithms in Section 4. Some experiments on our Facebook Vietnamese conversations dataset (including FVnC and sample dataset) and wiki dataset, such as searching optimal parameters, clustering performance evaluations as well as clustering results, are considered in Section 5. In particular, we show that among the models that support Vietnamese, PhoBERT's feature extraction efficiency is the best based on V-measure score. Ultimately, we give some conclusions in Section 6. The code, datasets and pre-trained models are available at https://github.com/trieuntu/conversation_ clustering.

2. RELATED WORK

We provide some background knowledge about Transformer architecture, Pre-Trained Language Models, especially BERT. From these theoretical constructs, we apply them to solve our NLP tasks.

2.1 Transformer

Transformer architecture was first introduced in the paper "Attention Is All You Need" by [21]. At the time of launch, this architecture was considered a new breakthrough in the field of natural language processing and related tasks. Currently, when dealing with sequence-to-sequence models in NLP, the transformer is still one of the state-of-the-art (SOTA) types of model and completely replaces RNN/LSTM [23]. Transformer architecture overcomes the disadvantages of RNN and its variations. For instance, it doesn't take advantage of GPU parallelism, because it has to process input word-by-word sequentially into encoder/decoder and the information is easily lost during propagation through hidden layers for long input sentences.

Transformer architecture contains two parts; Encoder attention and Decoder attention. According to the original article of [21], the encoder part has 6 layers, each of which has two sublayers, which are multi-head self-attention and fully connected feed-forward. Decoder part is similar to the encoder part, but it adds a masked multi-head attention sublayer and the last layer of the encoder part will be passed to the multi-head attention sublayer in the decoder part. Note that the input of both parts is the sum of positional encoding vector and word vector embedding.

The attention mechanism is the most important component of transformer architecture. Self-attention sublayer is an attention mechanism, which contains the weight sets of the model W_q, W_k, W_v to be trained. The attention mechanism presents the relation of a word to all its related words in the sequence based on the adjustment of the above sets of weights. The product of the input embedding layer and W_q, W_k, W_v is matrices Query Q, Value V and Key K. In order to calculate Attention vector of word *i* to the rest of the words, Vaswani et al. [21] have given the formula:

$$Attention_i(Q_i, K_i, V_i) = softmax\left(\frac{Q_i K_i^T}{\sqrt{d_{K_i}}}\right).V_i$$

where d_K is the dimension of *K*. Each computed *Attention* obtains a head-attention. We can compute the *Attention* in parallel, which leads to the multi-head attention mechanism by concatenating head-attentions:

$$MultiHead(Q,K,V) = Concatenate(head_1, head_2, \ldots, head_n).W_O$$

where $head_1$ corresponds to $Attention_1$. Matrix W_O has the same number of columns as the input matrix.

2.2 Pre-trained Language Models: BERT

Training models from scratch on large datasets is impossible for most people. Thus, using pre-trained models is an inevitable trend in the development of Artificial Intelligence. Taking into account the advantage of the weights that can be learned from trained models, we just need to fine-tune them to suit specific purposes. Formerly, pre-trained models in NLP have been mentioned in many studies [9]-[10], [24][25][26]. One of the great advantages of the transformer's architecture is that it allows the creation of NLP models trained, which can be reused in downstream NLP tasks. Some of the pre-trained language models based on transformer's architecture have achieved state-of-the-art results, like BERT of [13] from Google, GPT of Radford and Narasimhan [27] from Open AI and their variations. These new models can do things that the old models can't, such as allowing transfer learning in NLP with both low- and high-level features. Transfer learning is a combination of reusing the architecture of pre-trained model and fine-tune parameters of the original layers to accommodate downstream tasks.

Specifically, BERT is an easily fine-tuned pre-train word embedding on a large unlabelled text corpus (unsupervised) which is trained based on Masked Language Model Task and Next Sentence Prediction Task. BERT's architecture is built only on the Encoder part of the Transformer. The input text before applying fine-tuning for Vietnamese in particular and other languages in general is a combination of Token Embeddings, Segment Embeddings and Position Embeddings. If the input text consists of two or more sentences (pair-sequence), we must add token [CLS] at the beginning of the sentence and token [SEP] to separate the sentences.

Masked Language Model task allows us to fine-tune word representations on any unsupervised text corpus. This task creates embeddings for the above Vietnamese dataset. The principle of operation of model training can be understood by predicting a missing word in the sequence instead of trying to predict the next word in the sequence itself. A missing word is equivalent to [MASK] token. We randomly mask 15% of the total tokens in the sequence and predict these [MASK] tokens. Note that a missing word can be replaced by [MASK] token 80% of the time, 10% of the time for a random token and 10% of the time for the unchanged token.

Next Sentence Prediction (NSP) is a binary classification task applied practically to the Question Answering (QA) task. NSP helps us understand the relationship between sentences. The input of the model is a pair-sequence, which has been added tokens [CLS], [SEP]. During model training, we select 50% of the time of the second sentence, which is the next sentence of the first one and labeled as IsNext, while the remaining 50% of the second sentence is randomly chosen from unrelated sentences in the dataset and labeled as NotNext.

There are many versions of BERT with different parameters on transformer architectures. The two most

basic models are $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$. In essence, both models are the same, but they are different in size. Specifically, according to Devlin et al. [13], these models have the following sizes:

$$\text{BERT}_{BASE}(L=12, H=768, A=12, Total\ Parameters=110M),$$
$$\text{BERT}_{LARGE}(L=24, H=1024, A=16, Total\ Parameters=340M)$$

where L is the number of layers in the Encoder part of transformer architecture, H is the hidden size and A is the number of heads in multi-head self-attention.

3. PHOBERT FOR TEXT CLUSTERING

PhoBERT represents pre-trained language models for Vietnamese proposed by Nguyen and Nguyen [15]. At the time of launch, pre-trained PhoBERT models established state-of-the-art results in most tasks related to Vietnamese NLP. Although BERT can be applied to many tasks, like Classification, Clustering, Dependency parsing, Sentiment analysis, Summarization text, Part-of-speech tagging, Question Answering, Named-entity recognition and Machine translation, in this work, we only focus on clustering task to analyze our Vietnamese conversations dataset.

$\text{PhoBERT}_{\text{base}}$ and $\text{PhoBERT}_{\text{large}}$ are two versions of PhoBERT, whose architectures are similar to the $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ above. PhoBERT uses RoBERTa, which is based on pytorch framework [28] to retrain the BERT models on new 20GB pre-training Vietnamese dataset. Since PhoBERT architecture is based on RoBERTa, it only trains BERT model with Masked Language Model task. Another difference between PhoBERT and RoBERTa is fastBPE used to tokenize input sentences. Currently there are many methods to tokenize, such as Word Level Tokenizer, Multi-Word-Level Tokenizer, Character Level Tokenizer, Subword Units Level (BPE algorithm) Tokenizer, but only BPE (Byte-Pair Encoding proposed by Sennrich et al. [29]) achieves SOTA and is applied to most modern NLP models.

BPE is a compression technique and is adapted for word segmentation tasks. Most words can be represented by subwords using the BPE method. It overcomes the disadvantages of Word and Character Tokenizers; for instance, words that do not appear in the dictionary can be represented in these subwords and the index length of sequence output is significantly shorter than Character Tokenizers. Code[1] of BPE algorithm to segment word into subword units was published by [29]. For example, assume that the given Vietnamese vocabulary is:

vocab = {'x i n h </w>': 10, 'đ ẹ p </w>': 20, 'x i n h _ đ ẹ p </w>': 10, 'x i n h _ x ắ n </w>': 15, 'x ắ n </w>': 8}

Notice that, unlike English, the Vietnamese language does not use white space to separate words, because Vietnamese words can have more than one syllable. For illustration take a simple Vietnamese sentence "*Cô ấy rất xinh đẹp*" (English version is "*She is very beautiful*"), which can be rewritten in the monosyllable form "*Cô_ấy* $_{She}$ *rất* $_{very}$ *xinh_đẹp* $_{beautiful}$ ". Therefore, we can apply a Multi-Word-Level Tokenizer on the pre-training Vietnamese dataset before going into BPE. There are many toolkits to support word segmentation based on Multi-Word-Level Tokenization, like RDRSegmenter from VnCoreNLP [30], pyvi [31] and underthesea [32]. In the example above, tokens </w> are appended to the end of the words to mark the end of a word in Vietnamese vocabulary. After merging the most frequent pair at the 9^{th} iteration, we obtain a new vocabulary as follows

vocab_{new}={'xinh': 10, '</w>': 10, 'đẹp</w>': 30, 'xinh_': 25, 'xắn</w>': 23}

It is clear that the word '*xinh_đẹp</w>*' can be represented by subwords '*xinh_*' and '*đẹp</w>*' from the above vocab_{new}. Especially, word '*xinh_xinh*' (English meaning is *pretty*) is out of vocabulary words, which can also be represented by the word pair '*xinh_*' and '*xinh*'.

In order to fine-tune PhoBERT for downstream tasks, we can use library packages, such as Transformers of Hugging Face [33] and FAIRSeq of Facebook [34], to implement. We use $\text{PhoBERT}_{\text{base}}$ with 12 block sub-layers of the Encode part to obtain the Embedding vectors as features of input sequences. More specifically, this Embedding vector is an output vector of the first token [CLS] from the final hidden state h (Figure 1). According to the idea of [22] [35], the vector of token [CLS] is the feature of the whole sentence for Classification task. To verify the idea just mentioned

[1] Scripts are available at https://github.com/rsennrich/subword-nmt

earlier, let's consider the following three sentences in Table 1.

Statement 1. *Assume that Embedding vectors* $\boldsymbol{E}^{i}_{[CLS]}$ *and* $\boldsymbol{E}^{j}_{[CLS]}$ *represent the whole sentences i and j, respectively. If sentences i and j are similar, then the Cosine Similarity between* $\boldsymbol{E}^{i}_{[CLS]}$ *and* $\boldsymbol{E}^{j}_{[CLS]}$ *must be sufficiently larger than a certain threshold and converges to 1 with identical sentences.*

Proof. The Cosine Similarity Formula is:

$$Cosine_{Similarity}\left(E^{i}_{[CLS]}, E^{j}_{[CLS]}\right) = \frac{E^{i}_{[CLS]} \cdot E^{j}_{[CLS]}}{\left\|E^{i}_{[CLS]}\right\| \left\|E^{j}_{[CLS]}\right\|} \quad (1)$$

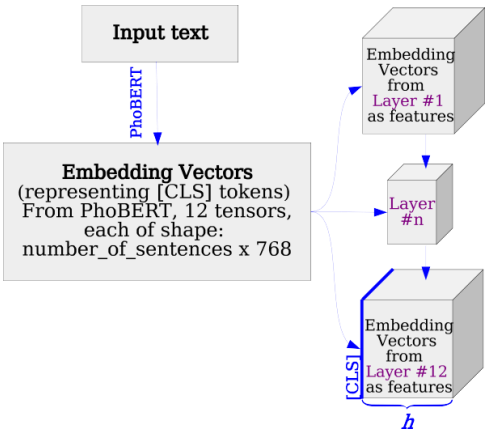


Figure 1. The first token [CLS] as feature of the input sentence.

Table 1. Three Vietnamese sentences are used as examples to extract Embedding vectors with PhoBERT model. Embedding vectors of sentences **A**, **B** and **C** are $E^{A}_{[CLS]}, E^{B}_{[CLS]}$ and $E^{C}_{[CLS]}$, respectively.

Sentence	Embedding Vector	Vietnamese	English
A	$E^{A}_{[CLS]}$	hà_nội là thủ_đô của việt_nam	Hanoi is the capital of Vietnam
B	$E^{B}_{[CLS]}$	thủ_đô của nước chxhcn việt_nam có tên gọi là hà_nội	The capital of the socialist republic of Vietnam is called Hanoi
C	$E^{C}_{[CLS]}$	hôm_nay trời sẽ có mưa dông, gió mạnh	Today there will be thunderstorms and strong winds

The statement above can be easily demonstrated through the example in Table 1. As observed, sentences **A** and **B** are almost similar and have higher similarity over sentence **C**. The computation of cosine similarity between Embedding vectors $E^{A}_{[CLS]}, E^{B}_{[CLS]}$ and $E^{C}_{[CLS]}$ is shown in Table 2. Since **A** and **B** are almost alike, their similarity metric will be high and converge to *1* and *vice versa* for **C**.

Table 2. Computing cosine similarity between embedding vectors for Table 1 with PhoBERT.

Cosine Similarity $(\boldsymbol{E}^{i}_{[CLS]}, \boldsymbol{E}^{j}_{[CLS]})$			
$\boldsymbol{E}^{A}_{[CLS]}, \boldsymbol{E}^{A}_{[CLS]}$	$E^{A}_{[CLS]}, E^{B}_{[CLS]}$	$E^{A}_{[CLS]}, E^{C}_{[CLS]}$	$E^{B}_{[CLS]}, E^{C}_{[CLS]}$
1.0	0.8519334	0.4461445	0.43029878

Using the Embedding vector of token [CLS] as a feature of the whole sentence, we adapt this idea to our Clustering task. The Clustering implementation process with PhoBERT_{base} model is shown in Figure 2. After obtaining the output embeddings of sentences with PhoBERT_{base} model, we use the algorithms K-mean and DBSCAN to cluster our text data. The output embeddings of sentences have the form as follows:

$$E^{i}_{[CLS]} = hW \quad (2)$$

where $W \in \Re^{d,H}$ and h are projection matrices at the linear projection layer and the final hidden state, respectively.

4. CLUSTERING ALGORITHM

K-Means is one of the most basic algorithms in unsupervised learning [3]. According to the K-Means algorithm, a set of N samples $E^i_{[CLS]}$ is divided into K disjoint clusters ($K < N$). Let Y is the set of all label vectors for N samples, i.e., each sample $E^i_{[CLS]}$ has a label vector $y_i = [y_{i1}, y_{i2}, \ldots, y_{iK}] \in Y$. If vector $E^i_{[CLS]}$ belongs to cluster k, then $y_{ik} = 1$ and $y_{ij} = 0, \forall i \neq k$. Each cluster is characterized by a cluster "centroids". A set of centroids is denoted $M = [m_1, m_2, \ldots, m_K]$. In K-means algorithm, the clustering problem will be reduced to the optimization for loss function $\mathcal{L}(Y, M)$.

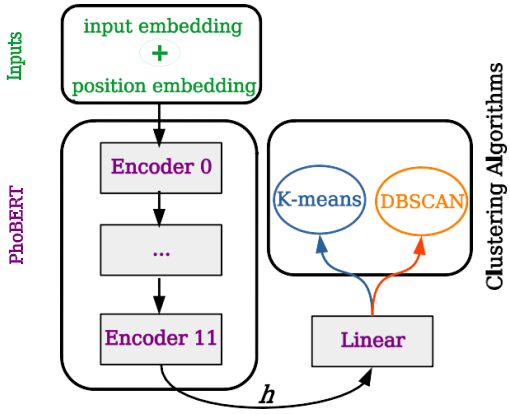


Figure 2. The overall flow for Clustering task with PhoBERT$_{base}$ model, starting from Vietnamese pre-training data, passing the layers Encoder$_{0\rightarrow11}$ to obtain embedding vectors from the final hidden state h through the linear projection layer and finally using unsupervised Learning algorithms K-mean and DBSCAN to cluster the text data.

$$\mathcal{L}(Y, M) = \sum_{i=1}^{N} \sum_{j=1}^{K} y_{ij} \left\| E^i_{[CLS]} - m_j \right\|^2 \quad (3)$$

Along with that, we also apply DBSCAN technique to cluster data points [5]. In real-life data, DBSCAN can work well for nonconvex clusters with arbitrary shapes and noises. DBSCAN algorithm focuses on radius *eps*–ε and the minimum number of neighbors required to create a cluster *minPts*. Radius *eps* defines a circle for each point to determine its neighbors. A point becomes *core point* if the circle surrounding this point with radius *eps* contains more than *MinPts* neighbors. In case that the number of neighbor points is less than *MinPts*, the *core point* is the *border points*. On the other hand, a point without any neighbors within radius *eps* is called *noise*. The relationship state of two points in DBSCAN can be *direct density reachable*, *density reachable* or *density connected*. A point is called *direct density reachable* for C_i point if and only if it lies within the circle centered *core point* C_i. If a *core point* is connected unidirectionally to any other *core point* through a chain of *core points*, there is a *density reachable* state between them. In case that there are two points, which are *density reachable* from the same point, they are *density connected* states. Pseudocode describing DBSCAN clustering algorithm [5], [36] is shown in Algorithm 1.

Unlike the evaluation metrics for supervised learning algorithms, the evaluation of clustering performance can be applied to datasets with known or unknown ground truth labels. If the ground truth class assignment of dataset is known, we use entropy-based measure, **V-measure** proposed in [19] is used to evaluate clustering performance for our sample dataset. The sample dataset will be described in detail in Section 5. Based on the conditional entropy analysis of two terms of *homogeneity* and *completeness*, V-measure is a harmonic mean function of those terms and can be calculated as follows:

$$v(\beta, h, c) = \frac{(1+\beta) \times h \times c}{(\beta \times h) + c} \quad (4)$$

where $h = \frac{1 - H(C|K)}{H(C)}$ and $c = \frac{1 - H(K|C)}{H(K)}$ are *homogeneity* and *completeness*, respectively. The conditional entropy $H(K|C)$ and entropy $H(K)$ are symmetric. In formula (4), β weight represents the contributions of homogeneity or completeness and the default value of β is equal to 1.

Unfortunately, in fact, we don't know anything about the ground truth classes for document clustering

task. Thus, we can evaluate clustering performance based on the partition obtained from clustering techniques and two types of proximities, which are similarity or dissimilarity between objects. Suggested in [20], **Silhouette** is a typical evaluation for this case. Besides providing a graphical overview of the partitional clustering (silhouette plot), Silhouette also allows evaluating clustering validity based on the average silhouette width. From those analyses, we can obtain a suitable number of clusters for the K-means algorithm. The Silhouette Coefficient $s(i)$ for object i has the form:

$$s(i) = \frac{b(i)-a(i)}{\max\{a(i),b(i)\}} \quad (5)$$

Algorithm 1: Pseudocode of original DBSCAN algorithm for our data

```
Data: E[CLS]
Input : ε, MinPts
Input : metric                          // calculating distance between data points
Input : cluster
foreach p in E[CLS] do                  // linear scan for all data points in E[CLS]
    if cluster(p) != unassigned then continue; // p is unassigned to any cluster or
     noise
    Neighbors N ← find_neighbors(E[CLS], metric, p, ε)
    if |N| < minPts then
        cluster(p) ← noise
        continue
    end
    c ← create a new cluster
    cluster(p) ← c
    S ← N \ {p}
    foreach q in S do
        if cluster(q) == noise then cluster(q) ← c;
        if cluster(q) != unassigned then continue;
        Neighbors N ← find_neighbors(E[CLS], metric, q, ε)
        cluster(q) ← c
        if |N| < minPts then continue;
        S ← N ∪ S
    end
end
```

Here, $a(i)$ is the average dissimilarity of object i to the remaining objects in the same cluster and $b(i)$ is the average dissimilarity of i to all objects of the next nearest cluster. The value of Silhouette Coefficient is in the range $[-1,+1]$, where near -1 indicates the object for incorrect clustering and *vice versa* for +1. The value around 0 represents overlapping clusters.

5. EXPERIMENT

We apply K-means and DBSCAN algorithms with PhoBERT$_{base}$ to cluster our Facebook Vietnamese conversations dataset (FVnC) and Sample dataset. In order to implement clustering task, we use PhoBERT$_{base}$ with the Transformers package of Hugging Face and Scikit-learn library [33], [37]. Furthermore, we search the optimal parameters for the clustering algorithms in this article. The clustering results will be used to build Intents for chatbot later.

5.1 Clustering Task Dataset

We evaluate our approach on Facebook Vietnamese conversations dataset. There are plenty of free tools or extensions to download conversations from a personal page, because it is quite simple. However, collecting conversations from public page is more difficult, so most tools or extensions to carry out this task are paid. In order to acquire this dataset, we created a tool named *NTUCrawler*[2] for scraping conversations from a Facebook messenger page of our University. This tool is written in Python language and based on Facebook's Graph API platform to get messages. It has two versions, one is

[2] Tool is available at https://archive.org/download/NTUCrawler

linux executable (run on Ubuntu distribution) and the other is Windows executable. The UI of the Windows version is shown in Figure 3. *NTUCrawler* requires users to provide four parameters, start time and finish time to get data, PageID and Token of page. Downloaded dataset contains 8000 conversations with more than 150 thousand raw text sentences of clients and admins of a Facebook page in the six-month period of the year 2020. The contents of the conversations are FAQ (frequently asked questions), which are related to information already published on the university website; for example, tuition fees, insurance, dormitory, English language test, course registration, …etc.

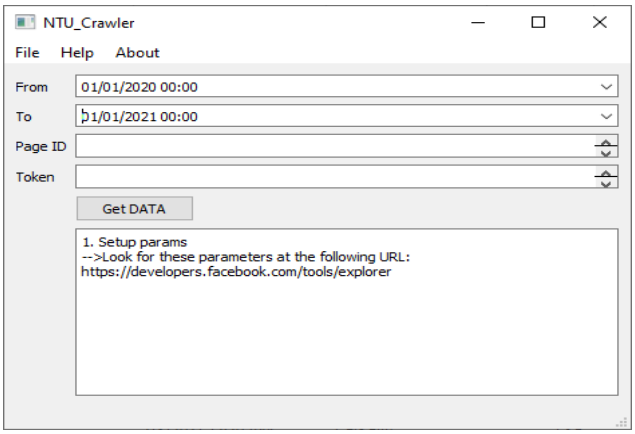


Figure 3. User interface of *NTUCrawler* on Windows.

5.2 Data Preprocessing

The raw downloaded dataset above must be preprocessed. Some punctuations "!", "", "?", "", ":" are removed from the dataset. Further, we eliminate stop words that have no value or no meaning to the NLP model, such as "dạ", "vâng", "chào ad", "vâng ạ", "alo", "ừ", "vậy", "ok", "nhé". We also filter out duplicate sentences or those that contain less than 4 words. After data preprocessing, 44846 text sentences are obtained and it is our desired **FVnC** dataset. From FVnC dataset, we randomly selected 95 sample text sentences (0.2% of FVnC dataset size) to form the **Sample** dataset. Besides that, we also use another sub-dataset which is called the **wiki** dataset. This dataset contains 396 text sentences of articles on 5 topics collected from Wikipedia. The Wikipedia library was applied to access and parse data from Wikipedia. The reason we use those Sample dataset and wiki sub-dataset is that we can evaluate the effectiveness of applying PhoBERT$_{base}$ for downstream task (clustering). Label assignment to sample text sentences and analyzing the number of clusters will cost less in our task. In addition, clustering on the small Sample dataset not only helps easily evaluate clustering performance, but also saves time compared to the original FVnC dataset. These sample text sentences were completed using manual labeling by us and divided into 3 classes, which describe questions between users (students) and admin (university) about information related to insurance, dormitory and English language test. Specifically, in order to specify the lables of classes, we rely on the experience of the specialists of the training department, who are responsible for answering students' questions directly or *via* social platforms. In their opinion, first- year students of our university are often concerned with insurance, dormitory and English language test. During data collection for each label, we carefully selected the sentences in the dataset that matched the recommendations of the specialists. Those 3 labels are one of the intents used to train the chatbot. Table 3 shows the details of the two datasets.

Table 3. Brief description of the datasets used for PhoBERT$_{base}$.

Dataset	Label	Description	Tasks	$E^i_{[CLS]}$ size
FVnC	unknown	clean downloaded dataset	Silhouette evaluation, clustering	44846 × 768
Sample	class 1	feature relating to insurance	clustering, clustering performance evaluation	31 × 768
	class 2	feature relating to English language test		38 × 768
	class 3	feature relating to dormitory		26 × 768

According to [15], input text must be already word-segmented before going through the BPE algorithm.

We use "pyvi" toolkit of Tran [31] to perform word segmentation in our datasets. After passing the fastBPE step, we have the index of tokens and attention masks for the text data. Taking them through PhoBERT's architecture leads to output embeddings of $E_{[CLS]}$. From this step, we use these embeddings as feature vectors to cluster text data.

5.3 Dealing with Varying Length

Because the length of the sentences in FVnC dataset is different, we have to use padding to make sure that the input texts have the same length. In particular, the maximum sequence length of PhoBERT is 256.

We truncate sentences with padding length less than 256 tokens. To choose the optimal padding length for all sentences, we can analyze the distribution of sentence lengths in Figure 4. Based on the above distribution, most sentences have lengths of less than 33 words. Thus, we decide that the padding length is equal to 33 in all datasets.

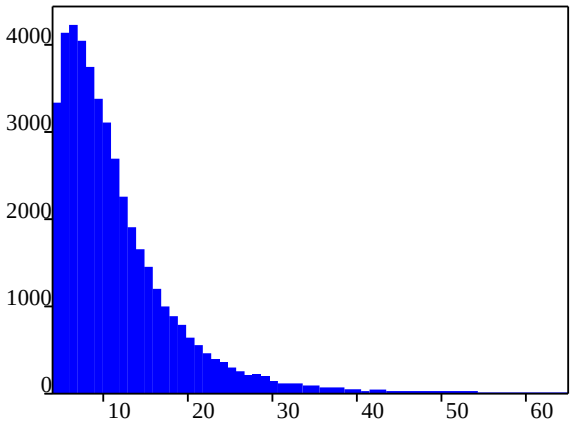


Figure 4. The distribution of the sentence lengths in FVnC dataset.

5.4 Parameter Optimization

When using the DBSCAN technique, we need to take care of two parameters; *MinPts* and ε. Choosing these two parameters is not easy. Their influence is very large on the clustering results. There is no way to accurately determine the parameter *MinPts*. However, there are several ways of choosing *MinPts* which have been proposed in [5]-[6]. Besides, the value of *MinPts* must also depend on domain knowledge and the data distribution observation. So, we derive that *MinPts* should be greater than the number of dimensionality of feature vector $\mathbf{E}_{[CLS]}$ (44846×768). As we can see, the number of dimensions of $\mathbf{E}_{[CLS]}$ is too large (*768*), which affects computation time and cost for large datasets. Therefore, we use the dimensionality reduction method to reduce $\mathbf{E}_{[CLS]}$ to lower-dimensional while retaining most of the original information. Principal Component Analysis (**PCA**) and t-distributed Stochastic Neighbour Embedding (**t-SNE**) are common techniques for data dimensionality reduction. **PCA** relies on eigenvalues and eigenvectors of $\mathbf{E}_{[CLS]}$ to reduce the original data to a specific number of dimensions (commonly known as *principal components*), but it still ensures a threshold of allowable variance. If we use **PCA,** then *MinPts* can be selected as follows:

$$MinPts \geq principal\ components + 1$$

Parameter ε can be found from a K-Distance graph, which is based on the average distance between objects and their MinPts nearest neighbors [7]. The K-Distance graph with MinPts=3 for FVnC and Sample dataset is shown in Figure 5.

The blue solid curve and red dashed curve correspond to the average distance of objects to MinPts nearest neighbors which are sorted in ascending order for FVnC and Sample dataset, respectively. Usually, a point at the position with the largest slope change in K-Distance graph or what we popularly call the "knee/elbow" of the graph is the optimal value of parameter ε [38]. Especially, the greatest slope change zones are highlighted in Figure 5(a) and Figure 5(b) for specific datasets. In order to take exactly the point mentioned above or the "knee point" of the graph, the kneedle algorithm is considered in our work [39]. The knee point obtained from the kneedle algorithm is determined by the intersection of the

specific data curve with the vertical straight line in Figure 6. The optimum values for parameter ε are 0.57 and 0.12 in the case of Figure 6 (a) and Figure 6 (b), respectively. However, optimizing parameter ε by choosing a fixed value of knee point in some cases does not lead to good clustering efficiency.

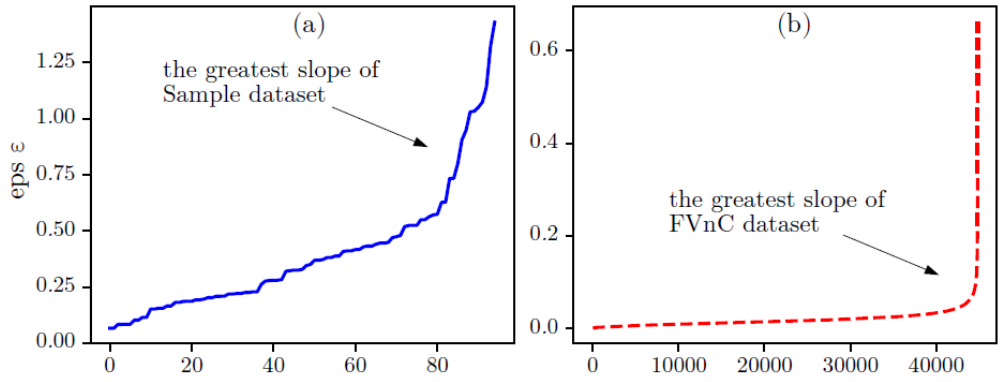


Figure 5. Data points are sorted ascending by the average distance to *MinPts* nearest neighbors. (a) Calculated on Sample dataset; (b) Calculated on FVnC dataset.

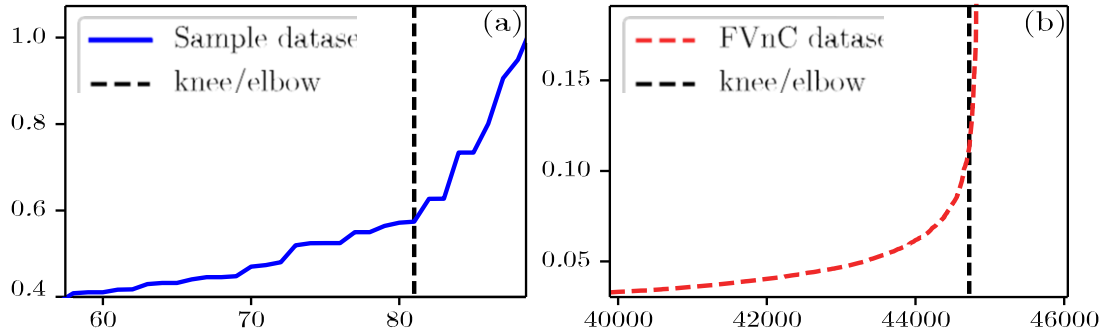


Figure 6. Determining knee points at the greatest slope change zones using the kneedle algorithm for Figure 5. (a) Solid blue curves – the greatest slope change zone of Sample dataset; (b) Dashed red curve – the greatest slope change zone of FVnC dataset.

Specifically in the test of the good separability between clusters in the sub-section below, Silhouette score is quite low. This could lead to objects being assigned to wrong clusters. With the expectation of improved clustering performance, in this work, we propose another technique based on the combination of K-Distance graph and clustering performance evaluations to find the right optimal value ε. Besides observing the greatest slope change zone of the line from a pair of points on the K-Distance graph, the maximum values of the Silhouette coefficient and V-measure score are also considered. The pseudocode for our technique is given in Algorithm 2. The sequence of MinPts is taken from principal components + 1 to 2 × principal components + 1 and incremented by a step of the minimum distance between data points. Gridsearch technique is applied in algorithm 2 for the sequence of MinPts and the greatest slope change zone in K-Distance graph. At the position where the Silhouette Coefficient is maximum, we obtain the optimal pair of values (ε, MinPts) for unlabeled data. For the labeled Sample dataset, the search of the optimal values (ε, MinPts) is based on the greatest mean of V-measure and Silhouette evaluation.

5.5 Result

In this part, our first task is to cluster text documents and evaluate clustering performance on the Sample dataset. As a consequence, analyzing the Sample dataset will be generalized to the general dataset FVnC, such as choosing the number of clusters using silhouette analysis for K-Means algorithm. As mentioned above, the Sample dataset has three labeled clusters (see Table 3). We use Silhouette evaluation to confirm that the Sample dataset has exactly three clusters and the way to choose the rightnumber of clusters when using it. Besides considering average silhouette scores, the silhouette plot is also an important factor in determining the number of clusters. Figure 7 represents the graphical overview

Algorithm 2: Pseudocode of the proposed technique to find appropriate ε and *MinPts*

Data : $\mathbf{E}_{[CLS]}$, label ; // label: points labels (unassigned or assigned)
Input : K-Distance
Input : *n_components*, step ; // step: minimum distance between points
Output: Index // the appropriate ε and $MinPts$ values for DBSCAN **Initial** : $\text{Max}_{Silhouette} \leftarrow -1$; $\text{Max}_{V\,measure} \leftarrow 0$; $\text{Max} \leftarrow -0.5$

```
slope ← CalculatingSlope(K-Distance)// the greatest slope change zone
nearest_neighbors ← arange(n_components+1, 2×n_components+1, step)
foreach ε in slope do
    foreach MinPts in nearest_neighbors do
        p ← PCA(E_[CLS], n_components)
        ClusterAssignment ← DBSCAN(p, MinPts, ε)
        SilCoeff ← SilhouetteScore(p, ClusterAssignment)
        if label = unassigned then
            if SilCoeff > Max_Silhouette then
                Max_Silhouette ← SilCoeff
                Index ← (ε, MinPts)
        else
            VScore ← VMeasureScore(label, ClusterAssignment)
            if (SilCoeff+VScore)/2 > Max then
                Max ← (SilCoeff + VScore)/2
                Index ← (ε, MinPts)
        end
    end
end
```

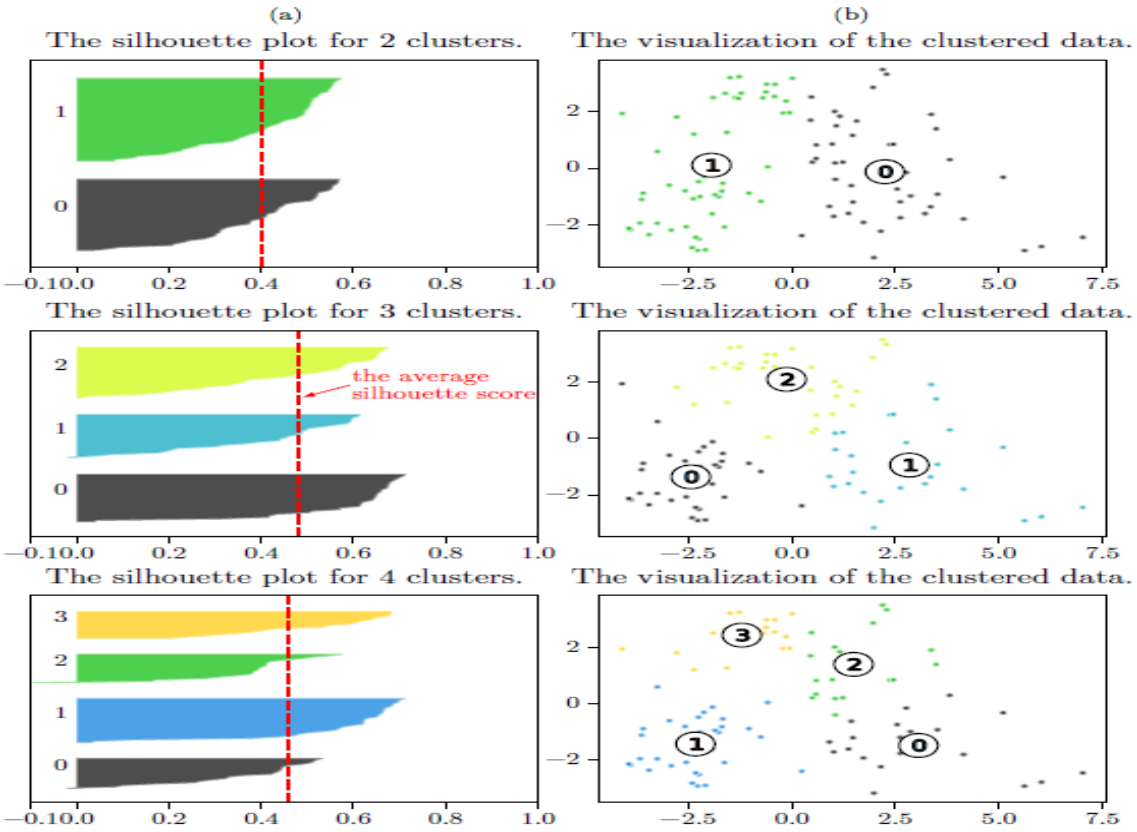


Figure 7. The graphical overview of the partitional clustering for 2, 3, 4 clusters using K-Means algorithm on the Sample dataset. (a) The silhouette plot; (b) The visualization of the clustered data for corresponding silhouette plots.

of the partitional clustering using K-Means algorithm on Sample dataset. X-axis of Figure 7 (a) corresponds to the silhouette coefficient values and the vertical dashed line is the average of the silhouette coefficients of data points. Clusters corresponding to silhouette plots in Figure 7(b) are visualized in two-

dimensional space using PCA. The maximum value of average silhouette scores is close to 0.481 for 3 clusters. On the other hand, the thickness of the silhouette plot for clusters is similar. The analysis outlined above fits the facts in Table 3 for the Sample dataset.

Moreover, in order to evaluate the efficiency of the PhoBERT$_{base}$ model in feature extraction for clustering tasks, we compared it with other models, such as BERT$_{base}$ Uncased, BERT$_{base}$ Multilingual Uncased [13], DistilBERT$_{base}$ Multilingual Cased [14], GPT-2 [12] based on V-measure score, among which GPT-2 and BERT$_{base}$ Uncased models do not support Vietnamese. We also use traditional approaches, like FastText, GloVe to compare with transformer models. Both models are applicable to Vietnamese language. However, both of these models are commonly used for learning word representations. Thus, to obtain sentence embedding, our method is averaging the word embeddings of all the words in the sentence. Especially, in order to make the comparison better, we also retrained the GloVe model on a new Vietnamese 20GB dataset based on the improvement proposed in [40]. Pre-trained word vectors **vncorpus.3B.100d** of new GloVe model correspond to the corpus 3B tokens, 1.3M vocab and 100d vectors and 1.17 GB download. Comparison results on the Sample dataset and wiki dataset using K-Means algorithm with 3 clusters and 5 clusters respectively are presented in Table 4. Through the obtained results, the models that support Vietnamese achieve significantly better V-measure scores than the rest of the models, especially the PhoBERT$_{base}$ model, which obtained the highest V-measure score on both the Sample dataset and wiki dataset (0.76 and 0.62, respectively). The comparisons obtained are in good agreement with the pointed theoretical and experimental works. Therefore, we use PhoBERT$_{base}$ model to extract features for the FVnC dataset, which leads to the output embeddings serving the clustering task.

Table 4. Evaluating feature extraction efficiency of models through V-measure scores for clustering task on the Sample dataset and wiki dataset.

Approach	Dataset	
	Sample	wiki
FastText (cc.vi.300)	0.64	0.27
GloVe (glove.6B.100d)	0.49	0.26
GloVe (vncorpus.3B.100d)	0.61	0.27
BERT$_{base}$ Uncased	0.11	0.19
BERT$_{base}$ Multilingual Uncased	0.36	0.43
DistilBERT$_{base}$ Multilingual Cased	0.37	0.12
GPT-2	0.04	0.04
PhoBERT$_{base}$	**0.76**	**0.62**

For DBSCAN clustering method on the Sample dataset, we need to find the optimal parameters ε, *MinPts* using Algorithm 2 and kneedle algorithm. Based on K-Distance graph of Sample dataset in Figure 5(a) and knee visualization in Figure 6(a), the greatest slope zone is selected in the range of [0.5,0.85] and the knee point value is 0.57. Due to "*principal components*" equal to 2 in PCA dimensionality reduction, the value of *MinPts* will be selected from 3 to 5. Some experimental results in finding the optimal parameters are shown in Table 5. As shown in the table, ε, *MinPts* and the average of V-measure and Silhouette score equal to 0.84, 4 and 0.42, respectively, are the best choice to cluster data points. Results in Table 5 also show that our approach – Algorithm 2- gives better clustering performance than kneedle algorithm on the Sample dataset. Using DBSCAN algorithm with the obtained parameters, 5 clusters are formed in Figure 8, in which cluster "-1" contains noisy objects in Figure 8 (a), noisy objects are removed in Figure 8 (b) and Silhouette plot of denoised Sample dataset is shown in Figure 8 (c). After removing noise, our clustering result has 4 clusters, while the Sample dataset has only 3 clusters. Based on actual observation in Figure 8 (b) and Silhouette plot in Figure 8 (c), cluster 0 and 1 must be merged into one cluster.

Remark 1. *DBSCAN is a density-based spatial clustering algorithm. The biggest disadvantage of DBSCAN is working in cases of varying-density clusters. From the Silhouette plot in Figure 8(c), there are two separate clusters of very different density comparing to the other two clusters. Two low-density clusters (cluster 0 and 1) are a matter of concern to us for predicting the number of clusters. Obviously, we hope that our clusters will be more equally distributed to choose the number of clusters more precisely.*

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

In order to solve this problem, we propose some solutions as follows:

Table 5. Experiment to find the optimal values of parameters ε, *MinPts* for DBSCAN algorithm on the Sample dataset.

Approach	eps (ε)	*MinPts*	V-measure score	Silhouette score	Average	N-Clusters
Algorithm 2	**0.84**	**4**	**0.56**	**0.28**	**0.42**	**5**
	0.78	4	0.52	0.27	0.4	5
	0.72	4	0.5	0.22	0.36	6
	0.76	3	0.47	0.22	0.34	4
	0.54	5	0.37	0.0	0.18	8
Knee point	0.57	3	0.42	0.15	0.29	9
	0.57	4	0.41	0.1	0.26	8
	0.57	5	0.41	0.05	0.23	7

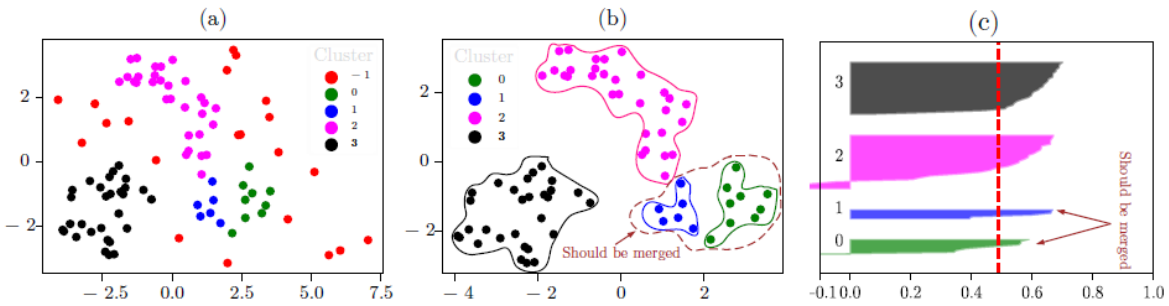


Figure 8. DBSCAN clustering on the sample dataset.

(i) Based on the thickness of the silhouette plot, the silhouette plots for clusters 0 and 1 are much smaller than for clusters 2 and 3. In particular, clusters 0 and 1 are close neighbors. Therefore, we have reasons to merge these small two clusters into a larger cluster. As a result, 3 clusters are the correct number of clusters when using DBSCAN clustering on the Sample dataset. After all, we should combine the selection of the optimal parameters and the width of Silhouette plots for clusters when using DBSCAN algorithm to obtain the most accurate number of clusters.

(ii) On the other hand, we can use a criterion for determining the minimum number of core points in a cluster. If at least two clusters are close to each other and have a smaller minimum number of core points than the specified criteria, we can merge them into a larger cluster.

Finally, we perform the main task, which is text clustering for the FVnC dataset. Note that the ground truth classes of the FVnC dataset are unknown. In order to estimate the number of specific clusters for K-Means clustering method, we take full advantage of DBSCAN algorithm. In the same way, we get the number of clusters from Algorithm 2. Specifically, we choose the number of clusters through Silhouette scores with large values corresponding to each pair of parameters ε and MinPts. For FVnC dataset, the optimal values of ε can be found in Figure 5(b). After applying DBSCAN algorithm, the number of clusters actually obtained must be subtracted by 1 for noisy objects (points labeled -1). Before using K-Means algorithm to perform clustering, these noisy objects are removed from FVnC dataset, which leads to the form of the corresponding denoised FVnC datasets. Lastly, we obtain clustering results of FVnC dataset without noise from the K-Means method based on the Silhouette evaluation. Silhouette scores test on original FVnC and denoised FVnC datasets for several different cluster numbers can be examined, as shown in Table 6. In case of using ε = 0.14 and MinPts = 25, the best obtained Silhouette scores for the original FVnC and denoised FVnC dataset are 0.3359 and 0.3460, respectively.

The Silhouette plots and clustering visualization of the best Silhouette scores can be seen in Figure 9. By comparing the Silhouette plots, we believe that the clustering result on the denoised FVnC dataset (see Figure 9 (b)) is better than on the original FVnC dataset (see Figure 9 (a)), because there are some noise points outside the clusters and the size of the silhouette plots of clusters 3 and 4 represents a wide fluctuation in the visualization of Figure 9(a). With the number of clusters received from the DBSCAN algorithm, the K-Means algorithm gives very good clustering results for denoised FVnC

datasets. After all, the clusters obtained from the clustering process are considered as big intents to help us build data for chatbot. For this reason, we can save time and effort and build chatbot faster.

Table 6. Silhouette scores using K-Means algorithm on original FVnC and denoised FVnC datasets for several different cluster numbers found from DBSCAN algorithm.

DBSCAN			K-Means Silhouette Scores	
eps ε	*MinPts*	N-Clusters	Original FVnC	Denoised FVnC
0.14	22	10	**0.3359**	0.3436
0.14	25	12	**0.3359**	**0.3460**
0.13	12	15	0.3310	0.3368
0.14	19	16	0.3294	0.3326
0.13	16	20	0.3267	0.3314
0.12	30	23	0.3252	0.3418
0.11	26	29	0.3225	0.3343
0.1	14	53	0.3196	0.3257
0.09	12	75	0.3196	0.3275
0.1	3	81	0.3217	0.3205
0.09	4	102	0.3220	0.3229
0.09	3	128	0.3207	0.3228
0.08	4	136	0.3212	0.3235

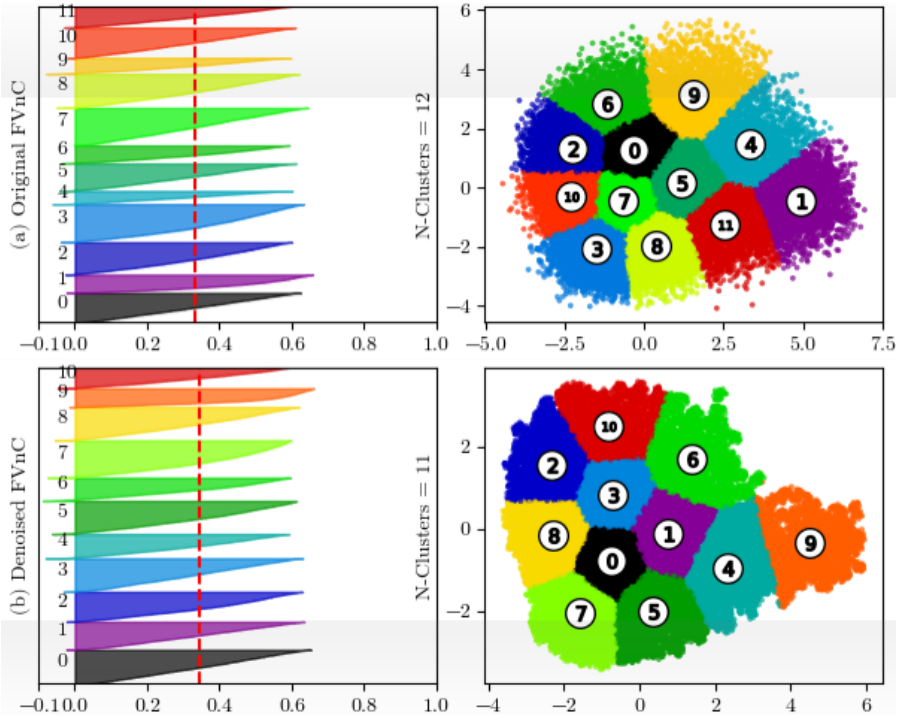


Figure 9. Silhouette plots and clustering visualization.

6. CONCLUSION

In this work, we research transformer architecture as well as pre-trained language models, such as BERT and PhoBERT. We also cover how to apply PhoBERT to our Facebook Vietnamese conversations dataset (FVnC). Furthermore, we have built a tool to crawl conversations from a Facebook Messenger Page. After extracting embedding vectors at the final hidden layer, we use the unsupervised learning algorithms K-means and DBSCAN to cluster text data. V-measure score and Silhouette score are used to evaluate the performance of clustering algorithms. A GridSearch algorithm that combines these two clustering evaluations is also proposed to find optimal parameters for the DBSCAN algorithm. The algorithm proposed by us obtained better clustering performance than kneedle algorithm through experimentations

based on V-measure scores and Silhouette score on the Sample and FVnC datasets. In addition, we compare the efficiency of the PhoBERT$_{base}$ model in feature extraction for clustering tasks with those of other models. PhoBERT$_{base}$ model achieves the best V-measure score on the Sample dataset and wiki dataset. We apply the K-Means clustering method with the number of clusters received from the DBSCAN algorithm to cluster the FVnC dataset. Topics obtained from clustering are similar to intents in building chatbot. From a pre-analysis data screening perspective, clustering results are valuable for building stories in our chatbot. Thanks to the implementation of this clustering, we save a lot of time and effort to build data and storylines for training chatbot.

ACKNOWLEDGMENTS

This work was partially supported by Nha Trang University (project TR2020-13-42). The authors thank Hien Thao Le for proofreading our manuscript and fruitful discussions. We also address special thanks to the reviewers for their helpful comments and suggestions.

REFERENCES

[1] A. Jung, "A Gentle Introduction to Supervised Machine Learning," Computing Research Repository (CoRR), vol. abs/1805.05052, pp. 6–7, 2018.

[2] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman and A. Wu, "An Efficient K-means Clustering Algorithm: Analysis and Implementation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881–892, 2002.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[4] J. B. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297, University of California Press, 1967.

[5] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc. of 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96), pp. 226–231, 1996.

[6] J. Sander, M. Ester, H. Kriegel and X. Xu, "Density-based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications," Data Mining and Knowledge Discovery, vol. 2, pp. 169– 194, 1998.

[7] M. Gaonkar and K. Sawant, "AutoEpsDBSCAN: DBSCAN with Eps Automatic for Large Dataset," IRD India, vol. 2, no. 2, pp. 11–16, 2013.

[8] G. Salton and M. J. McGill, "Introduction to Modern Information Retrieval," McGraw-Hill Computer Science Series, 1986.

[9] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Proc. of the 1st Int. Conf. on Learning Representations (ICLR 2013), Scottsdale, USA, 2013.

[10] J. Pennington, R. Socher and C. Manning, "GloVe: Global Vectors for Word Representation," Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, Doha, Qatar: Association for Computational Linguistics, Oct. 2014.

[11] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, "Enriching Word Vectors with Subword Information," arXiv preprint arXiv:1607.04606, 2016.

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, "Language Models are Unsupervised Multitask Learners," Proceedings{Radford2019LanguageMA}, [Online], Available: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf, 2019.

[13] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding," Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technol., vol. 1, pp. 4171–4186, 2019.

[14] V. Sanh, L. Debut, J. Chaumond and T. Wolf, "DistilBERT, A Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter," ArXiv, [Online], Available: https://arxiv.org/abs/1910.01108, 2019.

[15] D. Q. Nguyen and A. T. Nguyen, "PhoBERT: Pre-trained Language Models for Vietnamese," Proc. of Findings of the Association for Computational Linguistics (EMNLP 2020), pp. 1037–1042, arXiv: 2003.00744, 2020.

[16] O. Gencoglu, "Deep Representation Learning for Clustering of Health Tweets," Computing Research Repository (CoRR), vol. abs/1901.00439, [Online], Available: https://arxiv.org/pdf/1901.00439, 2019.

[17] L. Pugachev and M. Burtsev, "Short Text Clustering with Transformers," arXiv preprint arXiv:2102.00541, [Online], available: https://arxiv.org/pdf/2102.00541, 2021.

[18] S. Sia, A. Dalmia and S. J. Mielke, "Tired of Topic Models? Clusters of Pretrained Word Embeddings Make for Fast and Good Topics Too!" arXiv preprint arXiv: 2004.14914, [Online], Available: https://arxiv.org/pdf/2004.14914, 2020.

[19] A. Rosenberg and J. Hirschberg, "V-measure: A Conditional Entropy-based External Cluster Evaluation Measure," Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 410–420, Prague, Czech, Jun. 2007.

[20] P. J. Rousseeuw, "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis," Journal of Computational and Applied Mathematics, vol. 20, pp. 53–65, 1987.

[21] A. Vaswani et al. , "Attention Is All You Need," Proc. of Advances in Neural Information Processing Systems, vol. 30, [Online], Available: https://arxiv.org/pdf/1706.03762, Curran Associates, Inc., 2017.

[22] C. Sun, X. Qiu, Y. Xu and X. Huang, "How to Fine-tune BERT for Text Classification?" Proc. of the China National Conference on Chinese Computational Linguistics (CCL 2019), vol. 11856, pp. 194–206, Cham: Springer Int. Publishing, 2019.

[23] J. Chung, C. Gulcehre, K. Cho and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," Proc. of the NIPS 2014 Workshop on Deep Learning, NYU, 2014.

[24] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," Proc. of the 31st International Conference on Machine Learning, Ser. Proceedings of Machine Learning Research, vol. 32, no. 2, pp. 1188–1196, PMLR, Bejing, China, 22–24 Jun. 2014.

[25] A. Joulin, E. Grave, P. Bojanowski and T. Mikolov, "Bag of Tricks for Efficient Text Classification," Computing Research Repository (CoRR), vol. abs/1607.01759, 2016.

[26] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep Contextualized Word Representations," arXiv Preprint arXiv: 1802.05365, [Online], Available: https://arxiv.org/pdf/1802.05365, 2018.

[27] A. Radford and K. Narasimhan, "Improving Language Understanding by Generative Pre-training," [Online], Available: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

[28] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and Stoyanov, "Roberta: A robustly optimized BERT Pretraining Approach," arXiv: 1907.11692, 2019.

[29] R. Sennrich, B. Haddow and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," Proc. of the 54th Annual Meeting of the Association for Computational Linguistics, vol. 1: Long Papers), pp. 1715–1725, DOI: 10.18653/v1/P16-1162, Aug. 2016.

[30] T. Vu, D. Q. Nguyen, M. Dras and M. Johnson, "VnCoreNLP: A Vietnamese Natural Language Processing Toolkit," Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pp. 56–60, New Orleans, Louisiana, 2018.

[31] V.-T. Tran, "Python Vietnamese Toolkit," pyvi 0.1.1, pypi, [Online], Available: https://pypi.org/project/pyvi/, 2020.

[32] V. Anh, B. N. Anh and D. V. Dung, "Open-source Vietnamese Natural Language Process Toolkit," VnCoreNLP, Github, [Online], Available: https://github.com/undertheseanlp/word_tokenize, 2018.

[33] T. Wolf et al., " HuggingFace's Transformers: State-of-the-art Natural Language Processing," Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45, Oct. 2020.

[34] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier and M. Auli, "FAIRSEQ: A Fast, Extensible Toolkit for Sequence Modeling," Proceedings of NAACL-HLT 2019: Demonstrations, pp. 48-53, Minneapolis, Minnesota, 2019.

[35] T. Neeraj, "Feature-based Approach with BERT," Trishala's Blog, Github, [Online], Available: trishalaneeraj.github.io, 2020.

[36] E. Schubert, J. Sander, M. Ester, H. Kriegel and X. Xu, "DBSCAN Revisited: Why and How You Should (Still) Use DBSCAN," ACM Trans. Database Syst., vol. 42, no. 3, pp. 19:1–19:21, DOI: 10.1145/3068335, 2017.

[37] Scikit-learn, "Clustering," Scikit-learn 1.0.2 documentation, [Online], Available: https://scikit-learn.org/stable/modules/clustering.html#k-means, 2011.

[38] N. Rahmah and I. S. Sitanggang, "Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra," Proc. of the IOP Conference Series: Earth and Environmental Science, Workshop and International Seminar on Science of Complex Natural Systems, vol. 31, p. 012012, Bogor, Indonesia, Jan. 2016.

[39] V. Satopaa, J. Albrecht, D. Irwin and B. Raghavan, "Finding a" kneedle" in a Haystack: Detecting Knee Points in System Behavior," Proc. of the 31st IEEE International Conference on Distributed Computing

Systems Workshops, pp. 166–171, DOI: 10.1109/ICDCSW.2011.20, Minneapolis, MN, USA, 2011.
[40] T. H. Nguyen, "Analyze the Effects of Weighting Functions on Cost Function in the Glove Model," arXiv preprint arXiv: 2009.04732, [Online], Available: https://arxiv.org/pdf/2009.04732, 2020.

ملخص البحث:

يتمثّل التّحدي الأكبر في بناء برامج المحادثة في بيانات التّدريب. ويتعيّن أن تكون البيانات المطلوبة واقعيةً وضخمةً بما يكفي لتدريب برامج المحادثة. نقوم ببناء أداةٍ للحصول على بيانات تدريب حقيقية من بريد مراسلات فيسبوك في إحدى صفحات فيسبوك. وبعد خطوات المعالجة الأوّلية للنّصّ، فإنّ مجموعة البيانات التي تمّ الحصول عليها للتّوّ، تولّد مجموعة بيانات (FVnC) ومجموعة بيانات العيّنة. نستخدم إعادة التّدريب لـِ (BERT) من أجل (PhoBERT) باللغة الفيتنامية لاستخلاص سِمات بيانات النّصّ.

تم استخدام خوارزميات (K-Means) و (DBSCAN) للتّجميع للقيام بمهمات التّجميع بناءً على تضمينات المخرج من (PhoBERT). وجرى تطبيق درجة مقياس V ودرجة الظّل (Silhouette) لتقييم أداء خوارزميات التّجميع. كذلك تم عرض فعاليّة (PhoBERT) مقارنة مع أداء نماذج أخرى لاستخلاص السِّمات في مجموعة بيانات العينة ومجموعة بيانات الموسوعة (ويكي). من ناحية أخرى، تمّ اقتراح خوارزمية بحث في الشّبكة (GridSearch) تجمع بين التّقييميْن المتعلّقيْن بالتّجميع بُغية إيجاد المتغيرات المثالية. وبفضل تجميع هذا العدد من المحادثات، فإننا نوفّر الكثير من الوقت والجهد لبناء البيانات من أجل تدريب برامج المحادثة.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

DES22: DES-BASED ALGORITHM WITH IMPROVED SECURITY

Malek M. Barhoush[1], Bilal H. Abed-Alguni[2], Rafat Hammad[3], Mohammad Al-Fawa'reh[1] and Rana N. Hassan[4]

(Received: 8-Oct.-2021, Revised: 19-Dec.-2021, Accepted: 3-Jan.-2022)

ABSTRACT

We live in a world where the Internet has become the backbone of most of our dealings. The Internet has turned this big planet into a small village. The Internet can be reached by everyone, everywhere, at any time. Some authors predict that the number of various types of devices capable of connecting via the Internet will reach 75.44 billion by 2025. These devices vary from low-processing power processors to heavy-processing power processors. It often requires the protection of mobile data between devices. These devices that have limited energy and resources require the protection technology to be adapted. The time it takes to encrypt a message using Data Encryption Standard (DES) is much less than the time it takes to encrypt the same message using Advanced Encryption Standard (AES). The problem with DES is that the key size is small and this makes it vulnerable to brute force attack. This paper gives complete guidelines for adapting the original DES and making it more secure, along with improving its performance compared to the existing standard encryption algorithms, such as AES. The proposed approach improves the original DES security by extending the key size of DES without affecting the cost of DES. The new algorithm is called DES22 and is convenient for low-processing power devices, such as wireless sensors. DES22 has three variants for key size: 128 bits, 256 bits and 512 bits. The paper also proposes another improvement to DES through random permutation and the distribution of the initial permutation and final permutation tables between the encryption and decryption algorithms. The experimental results show that DES22 is more secure and faster than AES.

KEYWORDS

Data encryption standard, DES, Advanced encryption standard, AES, IoT.

1. INTRODUCTION

The Internet has revolutionized the world of communication, where the transmission of various data of large volumes is carried out at a high speed and in less time, making people in contact with each other at any time and from anywhere. This revolution has transformed our planet into a small village. The Internet has transformed society, especially with the presence of Internet of Things (IoT) technologies and cloud computing platforms that connect everything, everywhere at any time. This of course has a positive impact on the progress of mankind, as transactions have become electronic thus turning our life more flexible. On the other hand, the development of technologies and the Internet has facilitated many cybercrimes. The Internet is available to all its users, all over the world, which makes it vulnerable to attack and therefore, it needs protection [1]-[3].

According to [4]-[5], by 2025, about 75 billion devices will be able to communicate with each other *via* the public network. Examples of these devices are: laptops, desktops, IoT, wireless sensors, smartphones, tablets, cars, airplanes, trains, biomedical machines, switches, routers and so on. Some of these devices have processors with high capabilities, while some other devices have processors with medium capabilities and others have processors with very modest capabilities.

Data collected from different types of devices connected to the Internet is a source of intelligent analysis through which decisions are made to improve a particular goal [5]. This data needs to be transferred quickly and needs to be protected from various types of attack.

Encryption is one of the traditional and important means to obtain confidentiality over the public

1. M. M. Barhoush (ORCID: 0000-0002-1146-7293) and M. Al-Fawa'reh (ORCID: 0000-0002-5621-4126) are with Department of Information Technology, Yarmouk University, Irbid, Jordan. Emails: malek@yu.edu.jo and fawareh@yu.edu.jo
2. B. H. Abed-Alguni (ORCID:0000-0002-7481-4854) is with Department of Computer Science, Yarmouk University, Irbid, Jordan. Email: bilal.h@yu.edu.jo
3. Rafat Hammad (ORCID: 0000-0001-9698-7345) is with Department of Information Systems, Yarmouk University, Irbid, Jordan. Email: rafat.hammad@yu.edu.jo
4. R. N. Hassan is with Ministry of Education, Irbid, Jordan. Email: rananaimh77@gmail.com

Internet. It also provides other services, such as authentication, integrity and non-repudiation [2], [6]. Since the Internet allows multiple processes to connect with no physical direct connections, their information may flow among intermediate eavesdropper(s); therefore, it is important to protect their privacy. Various algorithms have been developed to provide encryption services, such as Data Encryption Standards (DESs), Blowfish, Twofish and Advanced Encryption Standards (AESs).

The emergence of cloud computing, mobile computing, big data and the IoT has necessitated the need to develop cryptographic algorithms that help improve their efficiency, speed and confidentiality and reduce battery consumption [7].

Underlying cryptography architecture comprises the following terms: Plain text, cipher text, encryption and decryption algorithms and shared key. The implementation of encryption and decryption algorithms is known to the public, but the secret key is not. In the encryption process, the plain text is converted into cipher text. Meanwhile, in the decryption process, the cipher text is back-converted into the plain text [8].

Encryption algorithms are classified into two main categories: symmetric and asymmetric. In the symmetric cryptography algorithm, the key used for the encryption process is the same as that used for the decryption process. The most important things that distinguish symmetric encryption from asymmetric encryption are the speed of execution and the amount of memory used to hold the key. In the case of symmetric encryption, the execution speed is higher and the amount of memory used for the key is less, compared to asymmetric encryption [9].

DES is one of the symmetric algorithms used for data security between 1977 and 2000, as declared by the National Institute of Standards and Technology [10]. The size of the key in the DES is 56 bits, which means that the process of trying all combinations of the key to decrypt a cipher text that was encrypted with DES is equal to 256 attempts. This is one of the main reasons that led to the cancellation of dealing with DES. In this paper, we propose new models for DES that work with larger key and same block size.

In this paper, the subsequent sections are organized as follows: the literature review is presented in Section 2. A description of DES is introduced in Section 3. In Section 4, the proposed design of the new algorithm DES22 is presented. DES22 security analysis is introduced in Section 5. In Section 6, DES22 performance evaluation and results' analysis are discussed. In Section 7, DES and AES complexity analysis is presented. Finally, the conclusion is introduced in Section 8.

2. LITERATURE REVIEW

Pfitzmannl and Anmann [11] outlined the construction of G-DES algorithm, where they proposed modifications to the DES algorithm and expected that these modifications would increase the speed of DES and make it the fastest among its peers. G-DES allows the user to enter a key with a length of 768 bits and this key is sub-divided into 16 sub-keys of 48 bits each, removing the sub-keys generation activity from the original DES algorithm. In addition, G-DES allows users to enter the substitution boxes created by the user, create initial permutation (IP) and its reverse (FP) or (IP^{-1}) and finally enter the expanding permutation table. These suggestions did not find practical reality, as no implementation of GDES was carried out.

Eli Biham [12] claimed that the speedup of executing his new DES proposal using Alpha-64 architecture compared to the original DES execution is 5. He provided a parallel form of DES implementation running on parallel SIMD model. The author claimed that DES needs 16000 instructions per block and with his new DES implementation, a block can be encrypted within the time it takes to execute 260 instructions. However, Eli Biham's proposal cannot work with all different modes of DES, in addition to that it does not enhance the DES security.

Anderson et al. [13]-[14] proposed an alternative form of the Advanced Encryption Standard (AES) algorithm, which they called Serpent algorithm. Serpent algorithm uses S-boxes as does DES, but the way they designed the S-Box made Serpent more efficient than AES. The size of the data block that the Serpent algorithm handles is 128 bits, the size of the key is 128, 192 or 256 bits and the number of rounds in Serpent algorithm is 32, each round working with four 32-bits in parallel. The data block is represented in little-endian format. Serpent algorithm generates 33 sub-keys of 128-bit length. Figure

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

1 shows the workflow of Serpent Encryption Algorithm. As you can see from Figure 1, Serpent algorithm starts with IP and ends with FP. Each round performs three activities: key mixing, substitution and linear transformation. The Serpent algorithm needs special hardware to work with and it does not find practical reality.

Alani proposed DES96 algorithm in order to enhance the security of DES [15]. The key size of DES96 algorithm is 84-bit. The sub-key is generated *via* S-boxes and XOR process. The IP table contains 84 entries, the IP sub-divides the 84-bits key into 3 parts: 48-bits, 28 bits and 8 bits. The first 48 bits are converted to 32 bits via S-boxes, the next 28 bits are permuted and converted to 16 bits and the last 8 bits are sub-divided into four adjacent 2 bits and then every 2 bits are XORed. As a result, 4 bits are produced. These 3 parts contribute to the creation of the sub-keys. Alani's proposal increases the security of DES a little bit.

In [16], the authors suggested a new technique to enhance the security of DES. They proposed that the content of the following tables or lists should not be static: IP table, the inverse permutation table, substitution boxes, expansion box and shrinking box. In other words, the transposition and substitution processes should not rely on static tables. The key input of their algorithm is tied with a random number, the value of which will determine the rotation value for the above-mentioned tables or lists. The idea is brilliant, but the way they translated their idea was not perfect. The number of possible IP along with corresponding inverse tables is 64, the number of possible expansion and corresponding shrinking tables is 32 and the number of possible S-boxes is 8*64. In this case, their suggested proposal has 64*32*8* 64 possible transposition and substation tables or lists. The brute force attack for this model is $64*32*8*64*2^{56}$ tries.

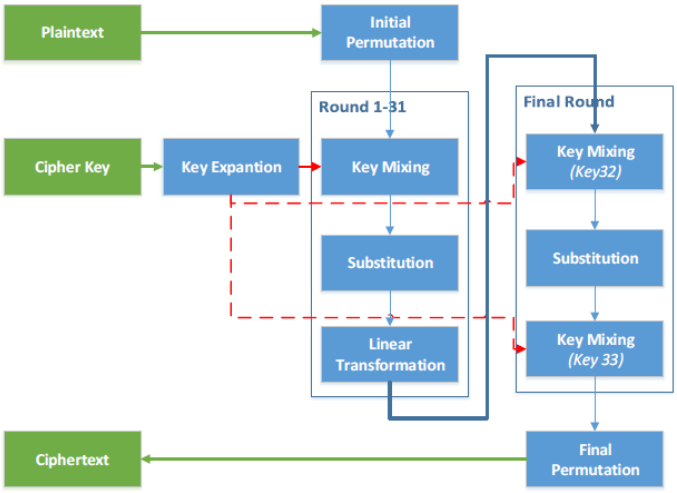


Figure 1. Workflow of serpent encryption algorithm [14].

In [17], the authors changed some components within DES. They replaced the eight S-boxes with one S-box and this change reduced the complexity of the DES hardware design circuit and thus reduced the execution time and energy consumed. Furthermore, the authors apply pipeline technology to 3DES encryption and decryption processes, which leads to reduce the latency of 3DES encryption and decryption processes.

In [18], the authors introduced a new cryptography algorithm based on the AES algorithm, with three rounds and a 128-bit key size. They replaced the AES S-box with their own 16-entries S-box. They claimed that their algorithm is secure. In the world of cryptography, the fewer rounds the algorithms have, the less confusion and diffusion the cipher has. Therefore, the security of [18] algorithm needs justification.

In [19], the authors suggested a semi-AES-128 algorithm, in which they reduced the number of rounds by a half compared to the original AES and as a result, the execution time of the new code was reduced by 35%. The security of their algorithm needs justification.

In [20], the authors proposed a pipelined implementation of AES that could work with multicore processors. Their proposal has 1.7 speedup compared with the original AES implementation. This type of algorithm needs multi-core processors to deliver better performance.

In [21], the authors proposed instruction set extensions to increase the performance and reduce the instructions count of AES implementation. They showed in their experiments that the speedup of their proposal is 10. The authors utilized embedded RISC processor, SPARC V8 architecture and superscalar processor for their test.

In [22], the authors designed a hardware for AES S-box and its inverse using 78 XOR and 36 AND gates, along with a coupled quadratic congruential generator (CQCG). The CQCG generates a random sequence for their design. This design increases the performance of AES algorithm.

In [23], the authors proposed a modified version of AES; they generated S-box table *via* PN Sequence Generator component and the same PIN is used to generate the input key. Their design increased the security of AES; however, it did not improve AES's performance. Table 1 summarizes the literature review section.

Table 1. Literature review summary.

Algorithm	**Description**	**Advantages**	**Disadvantages**
G-DES [11].	A variant of DES algorithm with a key length of 768 bits.	It seems to have more security.	There is no practical reality for the algorithm.
Parallel DES [12].	A parallel DES proposal using Alpha-64 architecture.	Its speedup is 5.	It cannot work with all different modes of DES. Key size is 56 bits.
Serpent algorithm [13]-[14].	The authors replace AES S-box with S-boxes used in DES; Serpent has 32 rounds. Serpent utilizes parallelism.	Serpent is more efficient than AES.	Serpent algorithm needs special hardware to work with.
DES96 algorithm [15].	DES96 has a key length of 84 bits.	The security of DES96 is better than that of DES.	DES96 security is not sufficient.
New DES [16].	The authors suggest that all DES tables should be rotated randomly.	The security of the algorithm is better than that of DES.	The security of this algorithm is not enough.
A variant of DES [17].	The authors replace the eight S-boxes of DES with one S-box	The complexity is reduced compared with DES.	Key size is 56 bits.
AES-based algorithm [18].	The modified AES has three rounds and a 128-bit key size.	The modified AES is faster than the original AES.	The cipher has less confusion and diffusion property.
simi-AES-128 [19].	The number of iterations is reduced to the half compared with AES.	The time latency is reduced to 35% compared with AES.	The security of the algorithm needs justification.
Pipelined- AES [20].	A variant implementation of AES utilizing pipeline.	Pipelined-AES has 1.7 speedup compared with the original AES.	The algorithm needs multi-core processors to deliver better performance.
Instruction set extensions [21]	Adding new instruction set to support AES performance.	The authors claim that the speed up is 10.	The proposal needs a change in the hardware architecture.
Hardware S-box [22].	Designing a hardware S-box using 78 XOR and 36 AND gates.	The design increases the performance of AES algorithm.	The design needs special hardware.
Modified version of AES [23].	The authors generated s-box table *via* PN Sequence Generator component.	The security of AES is increased.	AES performance is a problem.

3. DATA ENCRYPTION STANDARD (DES)

In this section, we will walk through the important points in the life cycle of DES to understand how to improve its security.

3.1 DES Workflow

DES deals either with block or stream cipher; the inputs of DES block encryption algorithm are blocks which a plain text length of 8 bytes or 64 bits and a key length of 56 bits. DES stream cipher works with a byte or many bytes. DES includes many simple operations, such as: substitution, permutation,

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

word expansion, 6 to 4 bytes shrinking and XOR operation. In the block encryption process, the plain text is divided into blocks of 8 bytes long. For each input block, DES algorithm iterates 16 times before producing the 8 bytes cipher block. Each round requires 48-bits sub-key, so the algorithm needs 16 sub-keys of 48-bits length each. These sub-keys are generated out from a 56-bits key in a process called sub-keys generation. Figure 2, Figure 4 and Figure 5 show the DES encryption workflow.

The DES encryption starts with IP to the input block of 8 bytes length, where the bits of the input block are reordered according to the predefined IP table. Figure shows both IP and FP tables. As an example, bit numbers 58, 50, 42, 34, 26, 18, 10, 2 of the input block become the first eight bits of the output block, while bit numbers 60, 52, 44, 36, 28, 20, 12, 4 of the input block become the second eight bits of the output block, and so on.

The output of permutation process is a block of 2 words. It is divided into two halves: left word (L-word) and right word (R-word), on the basis that the word is equal to four bytes. The two words are passed through 16 rounds, where within each round, the same activities are repeated. These activities are summarized as follows: R-word expansion process, round key XORing process, expanding R-word, shirking process, 32-bit permutation process and L-word XORing process. Figure 2, Figure 4 and Figure 5 depict the 16 rounds for DES encryption process.

Figure 5 shows the black box for each round, where the L-word and R-word resulting from round i are used as an input to the next round. Figure 6 depicts the white box for each DES i^{th} round. The R-word is passed through; expansion process, where the expansion process uses a predefined expansion table; this table describes how to expand the 32-bit R-word into 48 bits. The result is XORed with 48 sub-key associated with iteration i^{th} number. Then the result is shrunk into 32 bits using predefined S-boxes. The resulting 32 bits are XORed with L-word. Finally, the L-word is swapped with the R-word, so that the current round L-word becomes the next round R-word and the current round R-word becomes L-word for the next round.

The DES encryption workflow is summarized as follows:

1- Divide the input plain text into blocks of length 8 bytes / 2 words, each word is 32 bits long.
2- If the last block is less than 8 bytes, then pad it with zeros.
3- Generate 16 sub-keys out from 56-bits input key, each sub-key has a length of 48 bits. Name each sub-key: SK (1), SK (2), …, SK (16).
4- While there is a block of plain text not processed yet, do steps 4 to 7. Otherwise, go to step 10.
5- Pass the current block into the permutation process called IP.
6- Name the permutated 2 words: L-word and R-word.
7- Pass the two words L-word and R-word through 16 rounds, where each round performs the same following operations with R-word:
    a. Pass R-word into the expansion process to produce 48 bits. See Figure 6.
    b. XOR the expanded 48 bits with the corresponding SK(i), where "i" is an integer number in the range from 1 to 16 and refers to round number. See Figure 6.
    c. Pass the result from the previous step through predetermined S-box to shrink it into 32 bits. See Figure 6.
    d. Then, pass the Shrunk 32 bits into the permutation process, then the result is XORed with L-word; the result is the new value of L-word. See Figure 5.
    e. Swap the new value of L-word with R-word, so that the current round is finished and the next round is configured with new values of L-word and R-word. See Figure 5. A summary of each round's activity is shown in Figure .
8- After 16 rounds are finished, the final LW and RW are passed into the final permutation called IP-1 and then the current cipher block is ready.
9- Go to step 4.
10- Concatenate all cipher blocks and then the whole encryption process is done.

In the substitution operation, each text pattern is uniquely substituted by cipher pattern, while in the permutation operation, the bits are distributed in an organized and studied way utilizing a table that guides the distribution mechanism [10]. The theory behind substitution and permutation is to hide the properties of the plain text, thus making it difficult for the attacker to break the cipher without knowing the key [27].

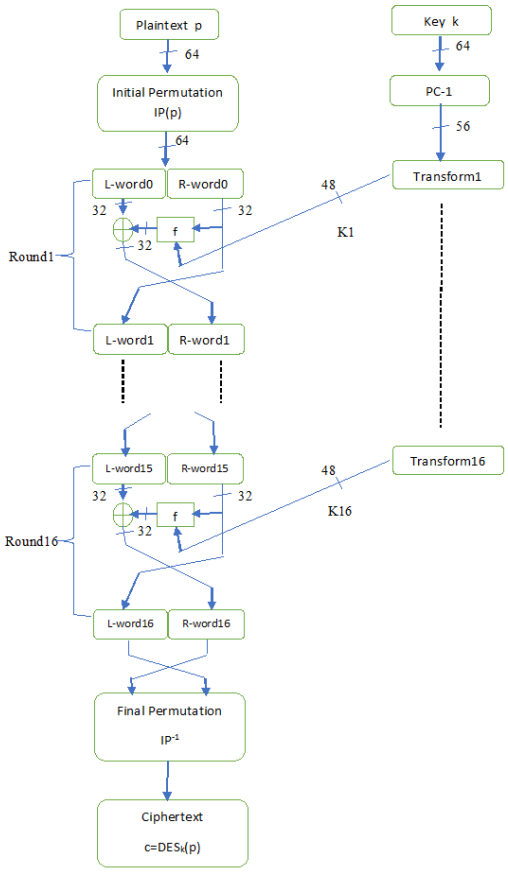


Figure 2. DES encryption process [24].

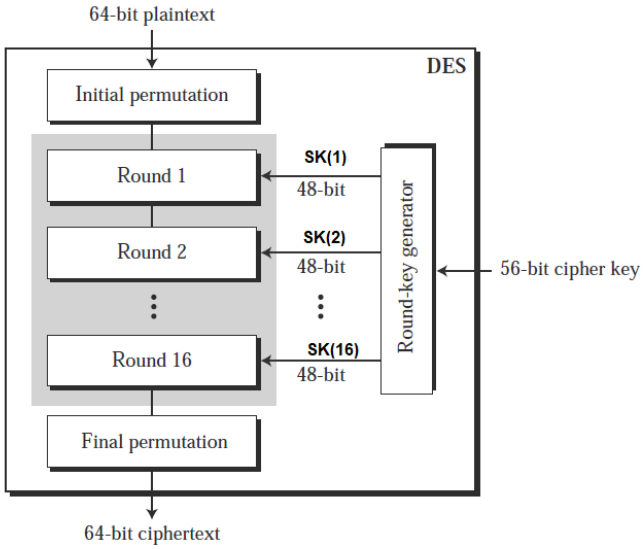


Figure 4. DES general structure [26].

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

***IP*$^{-1}$**

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Figure 3. IP and IP inverse tables [25].

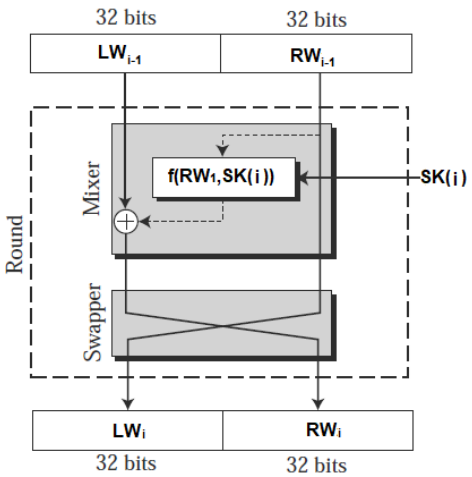


Figure 5. Black box for each round [26].

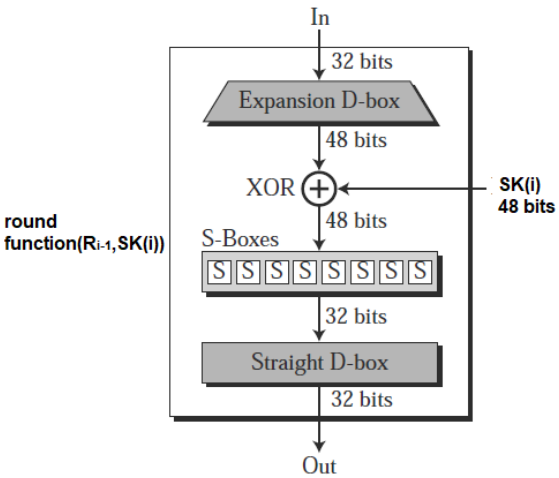


Figure 6. White box for each DES round [26].

In [10], the author states that for a stronger secure algorithm, it is recommended to have a large block size, a large key size, a large number of iterations producing staging product cipher, a strong key expansion and complex operations within each iteration.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

3.2 Sub-keys Generation

The DES encryption and decryption processes use a key with a length of 56 bits. Originally, the entered key is 64 bits / 8 bytes long. The uppermost bit of each byte is discarded; therefore, the remaining bytes are 56 bits long. The DES encryption or decryption processes pass through 16 rounds, where each round uses a sub-key derived from a 56-bits key. A process called sub-keys generation derives 16 sub-keys out from a 56-input key. The process starts by permuting the 56-bits key using a table called PC-1, then the result is broken into two parts; left half (C) and right half (D), where each part has a length of 28 bits. The generator iterates 16 times doing the following activities to generate 16 sub-keys with a length of 48 bits:

a) Rotate left each of the two parts C and D either by one or two bits. For iteration number 1, 2, 9 and 16, the left rotation is one bit. Otherwise, the left rotations are two bits. As a result, the overall left rotation for each C and D is 28 bits [24].
b) Pass a copy of C and D into another permutation process using a table called PC-2, which will generate a sub-key SK(i) with a length of 48 bits.

Figure 8 depicts the key scheduling process. Figure shows the two tables PC-1 and PC-2.

The DES decryption algorithm has the same workflow as the DES encryption algorithm, except that it uses the key scheduling order in reverse and this is why the authors describe this kind of algorithm as Feistel [8].

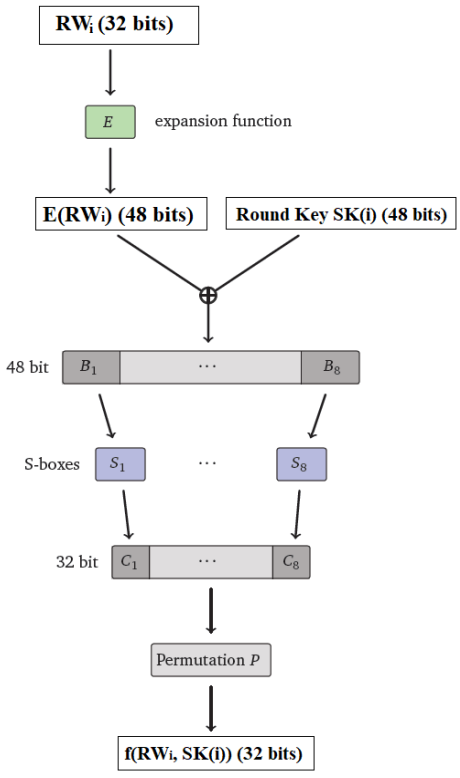


Figure 7. Summary of round activities.

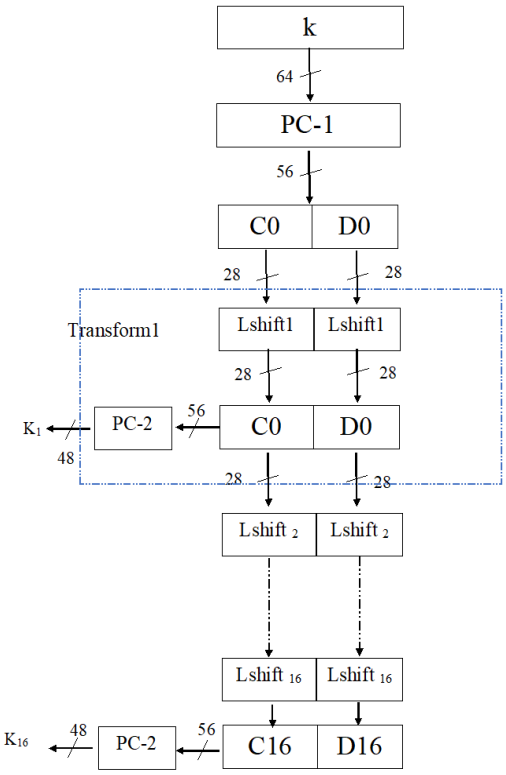


Figure 8. Key schedule for DES encryption [24].

4. THE PROPOSED DESIGN OF DES22

We call the proposed design DES22, where the number 22 refers to the year 2022. DES22 transforms 64-bits plaintext blocks into 64-bits cipher text blocks. DES22 is a substitution transformation with 16 rounds, with the key sizes of 128, 256 or 512 bits. The IP table along with its inverse (IP^{-1}) are dynamically created before the start of the encryption process. Therefore, the parameters for the DES22 algorithm are plain text or cipher text, a key of length (128, 256 or 512) bits and a dynamic initial permutation table (DIP) along with its reverse (DIP^{-1}). The DES22 algorithm is an open-source algorithm, while the key, DIP and (DIP^{-1}) should be only be only shared with the sender and the receiver.

4.1 Dynamic IP and IP Inverse

The DES process starts with IP for each plain text block and terminates with its inverse (IP^{-1}) before producing the final value of cipher text block. The lookup table for IP and (IP^{-1}) are standard and fixed. These two processes do not add any security value to the DES algorithm and may hide a back door threat [24].

We provide a procedure that dynamically creates a table for IP process (DIP) and the corresponding inverse table for IP^{-1} process (DIP^{-1}). The terms DIP and DIP^{-1} can be used to denote processes or tables depending on the context. Since these two tables do not give any secrecy value to DES, we distribute them among the encryption and decryption algorithms. We use the DIP process at the end of the encryption process, while we use the DIP^{-1} at the beginning of the decryption process. The two tables DIP and DIP^{-1} are randomly generated; so, it is necessary that they are available only to the sender and the receiver. This distribution increases the security of DES. The following pseudo-code shows the dynamic creation of the DIP and its corresponding DIP^{-1} tables. Algorithm 1 describes the Dynamic creation of the tables IP and IP inverse.

Algorithm 1. Creating dynamic permutation tables DIP and DIP^{-1}

Input: no input
Output: return two lists DIP and DIP^{-1}
Steps:

1- Create empty lists l1, DIP, DIP^{-1} each with empty 64 entries
2- Initiate the list l1 with integer values range from 1 to 64
3- count = 0, index = 0
4- while count < 64 do
5- index = choose a random integer exists in the list l1
6- DIP [count] = index
7- DIP-1[index] = count
8- Remove the value index from the list l1
9- count = count + 1
10- end while
11- return DIP and DIP^{-1}
12- end

It should be noted that to have better randomness in producing DIP and DIP^{-1} tables from the previous pseudo-code, we need to deal with true random generation. It should be noted that the number of probabilities of the tables generated by the previous pseudo-code is equal to the factorial of 64 (64!); this huge number of probabilities increases the security of the DES algorithm.

4.2 Key of Size 128, 256 & 512 Bits

The original DES deals with a key of size 56 bits; this key is permuted using a lockup table called PC1, where table PC1 is shown in Figure . The proposed DES22 deals with a key-unit of 64-bits, therefore, PC−1 table is modified from 56 entries into 64 entries. We called the new table Modified PC−1 (MPC-1). Round key **permutation table PC−2 [24]. key.** shows the permutation table of 64 input-key; the matrix MPC-1 does not exclude any bit of the input key-units. We believe that the original design of DES excludes 8 entries from the input key because of the need for parity bits in data transfer. In the proposed DES22, we do not need to exclude any bits from the input key. The proposed DES22 is designed to support key lengths of 128 bits, 256 bits or 512 bits. The input key is divided into multiples of 64 bits; we will call them multiple key-units. Therefore, 128-bit key is composed of 2 key-units, 256-bit key is composed of 4 key-units and 512-bit key is composed of 8 key-units. The process of creating sixteen 48-bits sub-keys starts with permuting each key-unit using the MPC−1 matrix.

If the key input is a 128 bits long key, 2 key-units, then each of the two key-units passes through eight rounds (n = 8) to generate sixteen sub-keys. Each key-unit is subdivided into its corresponding two 32-bit parts called C and D. For each key-unit, the round key generator iterates 8 times doing the following activities to generate 8 sub-keys with a length of 48 bits:

a) For the first iteration, rotate left each of the two parts C and D by three bits.

b) For the remaining iterations, rotate left each of the two parts C and D by four bits. As a result, the overall left rotations for each C and D are 31 bits.
c) Pass a copy of C and D into the permutation process using the table called PC-2, which is the same table used in the original DES algorithm. This permutation generates a sub-key SK(i) with a length of 48 bits, where i = 1, 2, 3……16.

$PC-1$							
57	49	41	33	25	17	9	1
58	50	42	34	26	18	10	2
59	51	43	35	27	19	11	3
60	52	44	36	63	55	47	39
31	23	15	7	62	54	46	38
30	22	14	6	61	53	45	37
29	21	13	5	28	20	12	4

$PC-2$							
14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Figure 9. Initial key permutation table PC−1 and Round key permutation table PC−2 [24].

57	49	41	33	25	17	9	64
1	58	50	42	34	26	18	56
10	2	59	51	43	35	27	8
19	11	3	60	52	44	36	32
63	55	47	39	31	23	15	16
7	62	54	46	38	30	22	48
14	6	61	53	45	37	29	40
21	13	5	28	20	12	4	24

Figure 10. The permutation table of 64-input key.

If the key input is a 256 bits long key, 4 key-units, then each of the four key-units passes through four rounds (n = 4) to generate sixteen sub-keys. Each key-unit is subdivided into its corresponding two 32-bit parts called C and D. For each key-unit, the round key generator iterates 4 times to perform the following activities to generate 4 sub-keys with a length of 48 bits:

a) For the first iteration, rotate left each of the two parts C and D by seven bits.
b) For the remaining iterations, rotate left each of the two parts C and D by eight bits. As a result, the overall left rotations for each C and D are 31 bits**.**
c) Pass a copy of C and D into another permutation process using the table called PC-2. This permutation generates a sub-key SK(i) with a length of 48 bits, where i = 1, 2, 3……16.

If the key input is 512 bits long, 8 key-units, then each of the eight key-units passes through two rounds (n = 2) to generate sixteen sub-keys. Each key-unit is subdivided into its corresponding two 32-bit parts called C and D. For each key-unit, the round key generator iterates twice to perform the following activities to generate two sub-keys with a length of 48 bits:

a) For the first iteration, rotate left each of the two parts C and D by fifteen bits.
b) For the second iteration, rotate left each of the two parts C and D by sixteen bits. As a result, the overall left rotations for each C and D are 31 bits**.**
c) Pass a copy of C and D into another permutation process using the table called PC-2. This permutation generates a sub-key SK(i) with a length of 48 bits, where i = 1, 2, 3……16.

Figure 11 depicts the new key scheduling process, where **n** represents the number of iterations to generate 48-bits sub-keys, MPC-1 represents the modified permutation table PC-1, each C and D is with a length of 32 bits. The subscript i in the words key-unit represents the i^{th} key-unit according to the table below.

Table 2 shows the relation between the length of input key and the number of key-units, as well as the number of iterations to generate n 48-bits sub-keys and the number of overall 48-bits sub-keys.

The reason behind applying total left rotation for each key unit by 31 bits is that this process produces unique sub-keys out from each key-unit. Table 3 depicts the values between key-units and their corresponding sub-keys. Table 4 shows the results of encrypting a plain text using different key-units.

4.3 Ignoring Weak Keys

In [8], Forouzan mentioned that some keys are considered weak or semi-weak and should be taken care of. Weak keys are four: all bits that have a value of zero, all bits that have a value of 1 or all bits that have a value of zero in the C part and a value of 1 in the D part, or just the opposite. On the other hand, there are 12 semi-weak keys; there values in the hexadecimal system are: [01FE 01FE 01FE 01FE, FE01 FE01 FE01 FE01, 1FE0 1FE0 0EF1 0EF1, E01F E01F F10E F10E, 01E0 01E1 01F1 01F1, E001 E001 F101 F101, 1FFE 1FFE 0EFE 0EFE, FE1F FE1F FE0E FE0E, 011F 011F 010E 010E, 1F01 1F01 0E01 0E01, E0FE E0FE F1FE F1FE, FEE0 FEE0 FEF1 FEF1]. Forouzan also

mentioned that there are keys called potential weak keys. The use of these keys leads to generate only four distinct sub-keys.

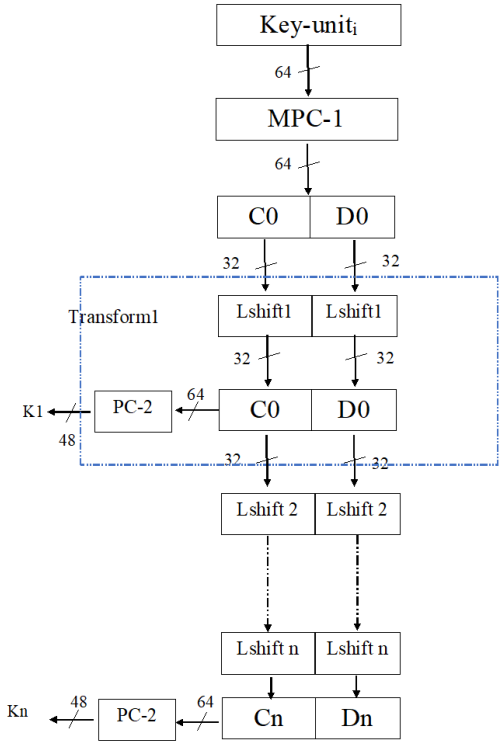


Figure 11. The new key scheduling process.

Table 2. The relation between the length of input key and the number of key-units.

Key length in bits	Number of key-units	Number of iterations to generate (n) 48-bits sub-keys **or** Number (n) of 48-bits sub-keys generated out of each key-unit.	Number of overall 48-bits sub-keys
128	2	8	16
256	4	4	16
512	8	2	16

Table 3. The generated sub-keys based on different key-units.

No. of Key-units	**Key-unit value**	**Sub-keys**
2	KU[1] = 0x123456789ABCDEF0	sub_key[1]= 0x439f4cc6ac1d sub_key[2]= 0xb1ed6a19d3a3 sub_key[3]= 0x719f15ca29dc sub_key[4]= 0x37c4f6731623 sub_key[5]= 0x3a833d047bdc sub_key[6]= 0x270afce38c2b sub_key[7]= 0x9a2f783d51f4 sub_key[8]= 0x439f4cc6ac1d
	KU[2] = 0x22234512987ABB23	sub_key[9]= 0x1c9d64820556 sub_key[10]= 0xadc724504649 sub_key[11]= 0xb9b2718075a8 sub_key[12]= 0x7273d4f24832 sub_key[13]= 0x43617f953050 sub_key[14]= 0x530bab10ac8e sub_key[15]= 0x950ece0b60e3 sub_key[16]= 0x1c9d64820556
	KU[1] = 0x123456789ABCDEF0	sub_key[1]= 0xb1ed6a19d3a3 sub_key[2]=0x37c4f6731623 sub_key[3]=0x270afce38c2b sub_key[4]=0x439f4cc6ac1d
	KU[2] = 0x22234512987ABB23	sub_key[5]=0xadc724504649 sub_key[6]=0x7273d4f24832 sub_key[7]=0x530bab10ac8e sub_key[8]=0x1c9d64820556

4	KU[3] = 0xFEDCBA9876543210	sub_key[9]=0x86b6b30bd8b2 sub_key[10]=0xe4495b1506ad sub_key[11]=0xd7d488ee0133 sub_key[12]=0xe84cb54ebea8
	KU[4] = 0x9988776655443322	sub_key[13]=0x8d15e8a29c2b sub_key[14]=0xf97445c1a41c sub_key[15]=0x6372a78629d1 sub_key[16]=0x8e0bc74c70cc
8	KU[1] = 0x123456789ABCDEF0	sub_key[1]=0x37c4f6731623 sub_key[2]=0x439f4cc6ac1d
	KU[2] = 0x22234512987ABB23	sub_key[3]=0x7273d4f24832 sub_key[4]=0x1c9d64820556
	KU[3] = 0xFEDCBA9876543210	sub_key[5]=0xe4495b1506ad sub_key[6]=0xe84cb54ebea8
	KU[4] = 0x9988776655443322	sub_key[7]=0xf97445c1a41c sub_key[8]=0x8e0bc74c70cc
	KU[5] = 0x123456789ABCDEF0	sub_key[9]=0x37c07c631633 sub_key[10]=0x61af4dc6a81d
	KU[6] = 0x22234512987ABB23	sub_key[11]=0x7373d4a2c8a2 sub_key[12]=0x1cbf64820172
	KU[7] = 0xFEDCBA9876543210	sub_key[13]=0xe4096b1526e5 sub_key[14]=0xe948b55efe28
	KU[8] = 0x9988776655443322	sub_key[15]=0xf97415c1845c sub_key[16]=0x8e0fc35c30cc

What distinguishes DES22 is that when generating a key, it makes sure that none of the above-mentioned keys are used. DES22 makes sure that each key-unit is not equal to the four weak keys and does not equal any of the twelve semi-weak keys. In case that the key-unit is among the list of possible weak keys, this condition is confirmed by checking that all the 16 sub-keys must be distinct.

5. DES22 SECURITY ANALYSIS

DES22 is a security improvement to the DES algorithm. It uses three key sizes (128, 256 and 512 bits), which is a clear improvement in terms of security. Also, it uses dynamic permutation tables DIP and DIP^{-1}. Obviously, in order to break an encryption/decryption algorithm that uses n-bits key using brute force attack, this needs 2^n tries. On the other hand, using dynamic permutation tables of 64 entries, the attack process becomes more difficult, as it needs 64! attempts. Table 5 summarize the number of attempts to break DES22 *via* brute force attack. The time needed for a brute force attack is equal to the number of attempts multiplied by the speed of one attempt by the world's fastest supercomputer. The following formula expresses the time interval for a brute force attack.

Table 4. The results of encrypting a plain text using different key-units.

Key-units in hexadecimal value	Plain text in hexadecimal value	Cipher text in hexadecimal value
KU[1] = 0x123456789ABCDEF0 KU[2] = 0x22234512987ABB23	0x9474b8e8c73bca7d	0x7f7fe23bf6189e77
KU[1] = 0x123456789ABCDEF0 KU[2] = 0x22234512987ABB23 KU[3] = 0xFEDCBA9876543210 KU[4] = 0x9988776655443322	9474b8e8c73bca7d	8e4ea022e7964db0
KU[1] = 0x123456789ABCDEF0 KU[2] = 0x22234512987ABB23 KU[3] = 0xFEDCBA9876543210 KU[4] = 0x9988776655443322 KU[5] = 0x123456789ABCDEF0 KU[6] = 0x22234512987ABB23 KU[7] = 0xFEDCBA9876543210 KU[8] = 0x9988776655443322	9474b8e8c73bca7d	67d8f66f41850f16

$Time_{brute\ force\ attack}$ = No. of attempts * $speed_{one\ attempt}$.

Assume that the world's fastest supercomputer has the speed of 415.5 peta FLOPS and assume that the speed of DES22 encryption process is 415.5 FLOPS, then each brute force attempt takes 415.5 FLOPS; in this case, this supercomputer can perform 10^15 encryption activities each second.

2^128 * 1/ peta flops = 340282366920938463463374.60743177 sec = 340282366920938463463374 years = 3.4 * 10^23 years

2^256 * 1/ 442,010 teraflops = 2.619671257150657121412886247114e+59= 8.306923063009440390071303421848 2e+51 years

64! = 1.268869321858841641034333893351 6e+89

Table 5. The relationship between DES22 key size and brute force attack.

Key size	**No. of attempts using brute force attack**
DES22 using 128-bits key	2^{128} attempts
DES22 using 256-bits key	2^{256} attempts
DES22 using 512-bits key	2^{512} attempts
DES22 using 128-bits key + dynamic permutation	(2^{128} + 64!) attempts
DES22 using 256-bits key + dynamic permutation	(2^{256} + 64!) attempts
DES22 using 512-bits key + dynamic permutation	(2^{512} + 64!) attempts

6. DES22 PERFORMANCE EVALUATION AND RESULTS' ANALYSIS

The DES22 is implemented with three different key lengths, 128, 256 and 512 bits, in addition to using the dynamic permutation table as key extension. For performance evaluation, we ran three algorithms, such as: DES22, DES and AES, to compare the time executions among them. We use the terms data-units and cipher-units to describe a unit of data or cipher with a length of 64 bits. The table below summarizes the execution time for encrypting n data-units and decrypting n cipher-units using DES22, DES and AES implementations. For AES implementation, we run the one with a key of 128 bits. In this research, we get the best DES implementation from the site [28], as well as the best AES implementation from the site [29].

The test was performed on Intel laptop Intel® core™ I7-6500 CPU @ 2.50 GHz 2.60 Hz and 16 GB RAM, Windows 10 professional 64-bits operating system. Table 6 depicts the system we use.

Table 6. System specifications.

Processor	Intel® core™ I7-6500 CPU @ 2.50 GHz 2.60 Hz
RAM	16 GB
System type	64-bit operating system, x64-based processor

Table 7 shows the speed of encrypting a variable number of data-units (8 bytes / 64 bits) using DES, AES, DES22, Parallel DES, Serpent, DES96, New DES and a variant of DES. The abbreviation "enc" in the table header means encryption. DES22-128 means DES22 with 2 key-units (128 bits), 256 means 4 key-units (256 bits) and 512 means 8 key-units (512 bits). It is noticeable that DES22 is close to the speed of DES and almost three times faster than AES. It is also noted that DES22 is faster and more secure than all its peers. Parallel DES is faster -but less secure- than DES22 due to the size of the key.

Table 7. The execution time for encrypting data-units.

Time in milliseconds for enc [No. of data-units]										
No. of data-units	**DES**	**AES**	**DES22 128**	**DES22 256**	**DES22 512**	**Parallel DES [12].**	**Serpent [13] [14].**	**DES96 [15].**	**New DES [16].**	**A variant of DES [17].**
1	.01	.0708	.01	.012	.012	.004	.032	.015	.014	.025
2	.014	.07	.019	.02	.018	.004	.31	.022	.02	.035
1K	12	46	9	9	10	4	26	21	19	28.8
2K	24	90	22	23	21	8	52	35	33	55
4K	41	150	31	33	34	11	87	68	62	94
10K	93	325	79	79	90	19.7	180	144	132	241
20K	274	696	150	169	181	60	550	412	390	680
50K	526	1912	418	420	224	157	1024	811	795	2,782
100K	1339	3481	895	915	925	337	2695	2027	1912	2,678
1M	7887	27610	7113	7201	7386	1985	15,879	12012	11152	16,562

The following chart in Figure shows the execution time to encrypt 1 mega data-units (8 megabytes) using DES, AES, DES22, Parallel DES, Serpent, DES96, New DES and a variant of DES.. From time perspective, AES is 3 times slower than DES and DES22. Since the security of DES22 is far better than those of both AES and DES and DES22 is faster than AES, it is obvious that DES22 has a better performance and security for simple and complex processors.

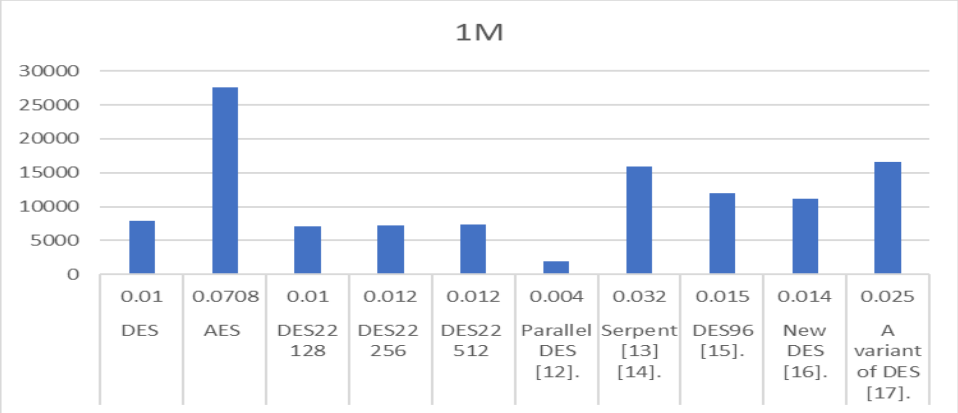


Figure 12. The execution time to encrypt or decrypt 1 Mega data-units.

7. DES AND AES COMPLEXITY ANALYSIS

DES and DES22 are Feistel cipher algorithms, which accept a block of 64-bit plain text and produces a block of 64-bit cipher text, or *vice versa*. The block is passed through an initial permutation and then through 16 stages of product ciphers, in which at each stage the following operations are performed on the block:

1. Splitting 64 bits into two halves.
2. Swapping the two halves.
3. Combining the two halves.
4. 32-bit to 48-bit expansion *via* P-box.
5. Using XOR.
6. Shrinking 48 bits into 32 bits *via* S-boxes.
7. 32-bit permutation.

The DES S-boxes were designed under the supervision of IBM researchers and they considered certain types of attacks in their design; one of these attacks is differential cryptanalysis. Recently, it is discovered is that DES is immune to differential attack due to the S-boxes design [30] and this of course is reflected in the DES22. DES is susceptible to linear and Davies-Murphy attacks [30]-[31] and this also is reflected in the DES22.

On the other side, AES is non-Feistel cipher and sometimes called Rijndael. Rijndael was selected among four other peers, MARS, RC6, Serpent and Twofish. The reason behind the superiority of AES lies in the following characteristics: security, cost, algorithm & implementation. AES is a block cipher, the block size is 128-bit and the key size varies: 128-bit, 192-bit or 256-bit.

In AES, the plain text is divided into 128-bit blocks, where each block is converted into a 4X4 array called state. The state is XORed with round key and then the result is passed through 10, 12 or 14 rounds, noting that the length of the key determines the number of rounds. In each round, the following operations are performed on the block:

1. Substitute each byte of the state with another value obtained from a table called S-box.
2. Rotate each row of the state to the left according to predefined values.
3. Perform matrix multiplication with each column of the state; this is called mix-column.
4. Finally, do XOR operation with the state and the round key.

AES is immune against differential and linear attacks. The reason for this is that the system design is based on Glorias Field ($GF(2^8)$) and uniform distribution of Sbox [10], [26], [30] and [32]. We strongly believe that in case of differential and linear attacks occur using AES, the long key size

makes these attacks unfeasible. The same is true with DES22, where the longer key size makes differential, linear and Davies-Murphy attacks inapplicable to DES22.

The simple operations and product cipher used in DES and DES22 make them faster in execution than AES, since AES contains matrix multiplications.

8. CONCLUSION

What called us to go back to the past and search for what was used in the field of encryption, which was suitable for the speed of devices in that era, is the emergence of the so-called IoT. The world of the IoT may contain sensors with a simple processor that takes readings and sends them to the main processor, which in turn bases its decisions on these readings. It is essential to protect these readings as well as the speed of their transmission and processing. One of the factors that contribute to security and speed is the availability of an encryption algorithm that is fast and secure.

DES is faster than AES; however, it is not secure as AES. In this paper, DES is extended by increasing the key size and the permutation table is adapted to make the new product more secure and faster than both the original DES and AES. Extended DES is called DES22. The results of the experiment show that DES22 is faster and more secure than AES.

ACKNOWLEDGEMENTS

The authors would like to thank Yarmouk University for supporting them in conducting this research.

REFERENCES

[1] M. Barhoush, Persistent Protection in Multicast Content Delivery, PhD Thesis, Concordia University, Canada, 2011.

[2] M. Barhoush and J. W. Atwood, "Requirements for Enforcing Digital Rights Management in Multicast Content Distribution," Telecommunication Systems Journal, vol. 45, no. 1, pp. 3–20, DOI: 10.1007/s11235-009-9231-4, 2010.

[3] M. Michels, W. Fecke, J.-H. Feil, O. Musshoff, F. Lülfs-Baden and S. Krone, "Anytime, Anyplace, Anywhere'—A sample Selection Model of Mobile Internet Adoption in German Agriculture," Agribusiness, vol. 36, no. 2, pp. 192–207, 2020.

[4] M. E. Whitman and H. J. Mattord, Principles of Information Security, 6th Ed., Cengage Learning, 2017.

[5] A. Al Hayajneh, M. Z. A. Bhuiyan and I. McAndrew, "Improving Internet of Things (IoT) Security with Software-defined Networking (SDN)," Computers, vol. 9, no. 1, p. 8, DOI: 10.3390/computers9010008, 2020.

[6] R. Atkinson and S. Kent, "Security Architecture for the Internet Protocol," [Online], Available: https://datatracker.ietf.org/doc/html/rfc4301, 1995.

[7] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," IEEE Communications Surveys and Tutorials, vol. 19, no. 3, pp. 1628–1656, 2017.

[8] B. A. Forouzan, Cryptography and Network Security, Mc Graw Hill India, 3rd Edition, 2015.

[9] M. Agrawal and P. Mishra, "A Comparative Survey on Symmetric Key Encryption Techniques," International Journal of Computational Science and Engineering, vol. 4, no. 5, p. 877, 2012.

[10] W. Stallings, Cryptography and Network Security, 4th Edition, Pearson Education India, 2006.

[11] A. Pfitzmann et al., "More Efficient Software Implementations of (Generalized) DES," Computers & Security Journal, vol. 12, no. 5, pp. 477–500, 1993.

[12] E. Biham, "A Fast New DES Implementation in Software," Proc. of the International Workshop on Fast Software Encryption (FSE 1997), Part of the Lecture Notes in Computer Science Book Series, vol. 1267, pp. 260–272, 1997.

[13] R. Anderson, E. Biham and L. Knudsen, "Serpent: A Proposal for the Advanced Encryption Standard," NIST AES Proposal, vol. 174, pp. 1–23, 1998.

[14] M. Naeemabadi, B. S. Ordoubadi, A. M. Dehnavi and K. Bahaadinbeigy, "Comparison of Serpent, Twofish and Rijndael Encryption Algorithms in Tele-ophthalmology System," Advances in Natural and Applied Sciences, vol. 9, no. 4, pp. 137–150, 2015.

[15] M. M. Alani, "DES96-improved DES Security," Proc. of the 7th IEEE International Multi-Conference on Systems, Signals and Devices, pp. 1–4, Amman, Jordan, 2010.

[16] M. Pranav and A. K. Rajan, "DES Security Enhancement with Dynamic Permutation," Proc. of the IEEE International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), pp. 6–11, Davangere, India, 2015.

[17] Y. Jun, L. Na and D. Jun, "A Design and Implementation of High-speed 3DES Algorithm System," Proc.

of the 2nd IEEE International Conference on Future Information Technology and Management Engineering, pp. 175–178, Sanya, China, 2009.

[18] D. Ma and Y. Shi, "A lightweight Encryption Algorithm for Edge Networks in Software-defined Industrial Internet of Things," Proc. of the 5th IEEE International Conference on Computer and Communications (ICCC), pp. 1489–1493, Chengdu, China, 2019.

[19] J. N. Mamvong, G. L. Goteng, B. Zhou and Y. Gao, "Efficient Security Algorithm for Power-constrained IoT Devices," IEEE Internet of Things Journal, vol. 8, no. 7, pp. 5498–5509, 2020.

[20] M. M. Barhoush, N. A. Kofahi, K. M. O. Nahar, A. M. R. Alsobeh, A. Jaradat and B. Alomari, "Performance Enhancement of the Advanced Encryption Standard *via* Pipelined Implementation," Journal of Theoretical and Applied Information Technology, vol. 97, no. 15, pp. 4213–4226, 2019.

[21] S. Tillich and J. Großschädl, "Instruction Set Extensions for Efficient AES Implementation on 32-bit Processors," Proc. of the 8th International Conference on Cryptographic Hardware and Embedded Systems (CHES'06), pp. 270–284, DOI: 10.1007/11894063_22, 2006.

[22] R. Sornalatha, N. Janakiraman, K. Balamurugan, A. K. Sivaraman, R. Vincent and A. Muralidhar, "FPGA Implementation of Protected Compact AES S--Box Using CQCG for Embedded Applications," Advances in Parallel Computing (Smart Intelligent Computing and Communication Technology), IOS Press, vol. 38, pp. 396-401, 2021.

[23] H. Zodpe and A. Sapkal, "An Efficient AES Implementation Using FPGA with Enhanced Security Features," Journal of King Saud University - Science, vol. 32, no. 2, pp. 115–122, 2020.

[24] C. Paar and J. Pelzl, Understanding Cryptography: A Textbook for Students and Practitioners, Springer Science & Business Media, ISBN-13: 978-3642446498, 2009.

[25] U. S. N. B. of Standard, "Data Encryption Standard," Federal Information Processing Standards Publication, vol. 46, no. January 1977, pp. 1–18, 1977.

[26] B. A. Forouzan, Cryptography & Network Security, 1st Ed., McGraw-Hill, ISBN-13: 978-0073327532, 2007.

[27] C. E. Shannon, "Communication Theory of Secrecy Systems," Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949.

[28] D. Huertas, "DES Algorithm Implementation in C," DES/des.c, [Online], Available: https://github.com/dhuertas/DES, 2020.

[29] ProgrammerSought "AES Encryption Algorithm C++ Implementation," [Online], Available: https://www.programmersought.com/article/66314322796/.

[30] D. R. Stinson, Cryptography: Theory and Practice, 3rd Ed., Chapman and Hall/CRC, ISBN-13: 978-1584885085, 2005.

[31] S. Kunz-Jacques and F. Muller, "New Improvements of Davies-Murphy Cryptanalysis," Proc. of the 11th International Conference on Theory and Application of Cryptology and Information Security (ASIACRYPT'05), pp. 425–442, DOI: 10.1007/11593447_23, 2005.

[32] D. Ganguly, Cryptography and Network Security-Fundamentals and Practices, Mc Graw Hill Education (India), Private Limited New York, NY, ISBN 9781578087556, 2012.

ملخص البحث:

تقدّم هذه الورقة إرشاداتٍ كاملة تتعلّق بتكييف نظام (DES) وجعْله أكثر أماناً، الى جانب تحسين أدائه مقارنةً بأداء خوارزميّات التّشفير الأخرى مثل (AES). إنّ الطّريقة المقترحة في هذه الورقة تزيد من درجة الأمان في نظام (DES) الأصلي، عن طريق توسيع حجم المفتاح دون التأثير على تكلفة النّظام. الخوارزمية الجديدة تسمّى (DES22)، وهي ملائمة للأجهزة ذات القدرة المنخفضة على المعالجة، مثل المجسّات اللاسلكية. ولخوارزمية (DES22) ثلاثة خيارات لحجم المفتاح: 128 و 256 و 512 بِت (bits). كذلك تقترح الورقة تحسيناً آخر على نظام (DES) من خلال تبديل الترتيب على نحوٍ عشوائي وتوزيع جداول تبديل الترتيب الاستهلالية والنهائية بين خوارزمية التّشفير وخوارزمية إزالة (فكّ) التّشفير. وتظهر نتائج التجربة أنّ خوارزمية (DES22) المقترحة في هذا البحث هي أكثر أماناً وأسرع من خوارزمية التّشفير المعتمدة على نظام (AES).

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

AN IN-DEPTH VISION TO HARDWARE DESIGN SECURITY VULNERABILITIES

Zainab Younis and Basim Mahmood

(Received: 29-Oct.-2021, Revised: 20-Dec.-2021, Accepted: 11-Jan.-2022)

ABSTRACT

Hardware plays a major role in our everyday life. Despite the technological thrive, there remain various security issues regarding hardware weaknesses that needed to be addressed carefully. Hence, an in-depth vision of the vulnerabilities that may exist in hardware design is delivered in this study by generating a network model that contains the most common weaknesses reported in common weakness enumeration (CWE). The main goal of the generated network is to deeply analyze the relations between different hardware designs and security weaknesses. Based on the conducted analysis, recommendations and suggestions are given to benefit many parties including hardware security developers. Accordingly, the analysis approach depends on different concepts that are inspired by the field of network science. The generated model is illustrated in a graph, wherein the nodes are the weaknesses and the edges are created if two weaknesses have a relation to each other. Promising findings have been attained and can be observed in the given model. For instance, the weaknesses CWE-441, CWE-1189, CWE-276 and CWE-1304 have not been given enough attention by the CWE and should be highly considered by software developers. Moreover, a rank for the hardware vulnerabilities based on network metrics is provided and compared with the most recently announced list of top hardware weaknesses by CWE. It is found that only two weaknesses are in common between the two lists, which indicates that the CWE list does not highly consider the relations among the weaknesses.

KEYWORDS

Complex networks, CWE vulnerabilities, Data analysis, Hardware vulnerabilities.

1. INTRODUCTION

As is a well-known fact, computers consist of both software and hardware. The term hardware refers to the tangible components and devices a computer is made of. A computer is not a single device, but a system of amalgamated devices working together to achieve the desired tasks [1]. People use these devices in their day-to-day activities for a myriad of purposes, such as at work, for medicine or engineering tasks, for communication, for home activities and entertainment, or for implementing different types of software [2]. In a similar case to software, there are also security issues and risks when it comes to hardware, but the scope is much more loosely defined when discussing hardware security [3]. The physical chips and boards present in electronics, embedded/IoT devices, networks and even cyber-physical systems are also considered hardware components [4].

Security issues stem from vulnerabilities referring to the weaknesses found in the design process and implementation of hardware architecture, which can be exploited by a hacker or perpetrator to mount an attack. The common vulnerabilities enumeration (CVE) is the most accredited source of information about security vulnerabilities. Hardware is usually manufactured before or during software development, but cannot be easily updated like software, yet hardware executes the software that controls a cyber-physical system [5]. Hence, it is often the last line of defense against potential attacks; that is, if the attack reaches the hardware, the damage may be permanent or irreversible. Cyberattacks target software by targeting the flaws in hardware design since they are undetectable and do not leave a software trace in the log files of that system. Hence, attacks like this on hardware formulate dangerous risks on any system using flawed hardware designs. The MITRE corporation defines weakness as a weakness present inside a component of a computer that, "when exploited, results in a negative impact on confidentiality, integrity or availability" [6]. The common weakness enumeration (CWE) is operated by MITRE corporation and is supported by the department of homeland security in the U.S. [7].

CWE is a community-developed list that includes different types of ordinary software and hardware weakness that own various security implications. CWE was established as a support system for people

Z. Younis and B. Mahmood are with Department of Computer Science, College of Computer Science and Mathematics, University of Mosul, Mosul, Nineveh, Iraq. Emails: zainab.younis@uomosul.edu.iq and bmahmood@uomosul.edu.iq

with hardware, software systems, or networks that are vulnerable to attack. Potential attackers would target weaknesses in the form of bugs, faults, or other specific flaws in hardware or software architecture, design, code, or execution. There is what is called the CWE list along with an associated classification taxonomy, in which both act as terminology that can recognize and communicate such weaknesses in CWE terms [8]. The main aim of CWE is to support programmers, hardware architects, and designers on how to avoid, deal with and eliminate common mistakes before products are delivered. This way, vulnerabilities can be stopped at the source, which is the main target of CWE servicing developers and practitioners in security.

In the end, using CWE helps avoid and prevent various kinds of security weaknesses that have haunted the hardware and software industries putting enterprises at risk [9]. CWE has published a catalog of the hardware weaknesses and made it readily available for interested parties assisting individuals and organizations to fully understand the reasons for weaknesses occurrence in applications and other cyber-enabled capabilities. CWE takes pride in originating from the fact that their work is derived from actual real-world examples of various weaknesses that appear in software applications. Conceptual patterns that make software and hardware exploitable by potential attackers are then discovered and generalized out of these weaknesses to assist designers and developers in recognizing and learning them at an early stage in the development lifecycle of software or hardware for either their avoidance completely or their quick identification to address them before the software program is put into operation.

The entries in the CWE system, or what are simply referred to as CWEs, are an amalgamation of the types of weaknesses discovered by exploiting them “in the wild” or as they happen in different situations through investigative testing and examination of software by testers, developers and hackers. There are several specialists, including academics, representatives of government agencies and research institutions, information security tool vendors, and major operating system vendors that constitute the international CWE community. These representatives create the CWEs by discovering a particular weakness in a product and making it general information or through examining software architecture, code, design, or deployed applications and finding flaws that may permit a potential adversary to infiltrate the system and do undesirable things. Discovering these weaknesses early on presents an opportunity to identify how an attacker may leverage a weakness and how a CWE community member or defender can remove the weakness.

If a company or organization has experienced a previous attack on their software or hardware, consequentially making them interested in a particular type of weakness, then the CWE institution can exploit the relationships between CWEs and common attack patterns enumeration and classification (CAPEC) to predict or foresee future infiltrations and hence suggest defense methods. Moreover, CWE is organized by their properties where a list of possible properties and definitions of the properties can be found on MITRE’s website. CWE and SANS institute for security have established themselves as the most prominent organizations providing security and practical information on applications that are unbiased [4]. Open web application security projects (OWASP) and CWE/SANS are also mentioned by [5] as being the most popular entities in the field. There have been several different research works conducted in the past few years on CWE and some of such studies are reviewed in Section 2 of this study. When discussing hardware weaknesses, it is noteworthy to mention that they can be functional (e.g., the core function of the hardware) or nonfunctional (e.g., performance, availability, …etc.) depending on the nature of a system and its usage scenarios. Usually, an adversary identifies a weakness or sometimes even more than one weakness and then exploits that weakness and this is what constitutes a typical attack [5].

Occasionally, a piece of hardware -such as a computer’s memory- might experience unexpected behavior in certain situations, which in turn can be taken as an advantage by cyber attackers [10]. Previous research has revealed different kinds of attacks that have compromised various platforms, such as private computers [11], internet-related browsers [12]-[13], cloud-based virtual systems [14], and smartphones [15]. These attacks have been employed to intensify privileges [15], detect cryptographic keys [16], expose online connected systems [17], or lock down a processor [18]. Safety-critical hardware is also prone to attacks, as it can have weaknesses. An exploitable bug, for instance, was discovered in the Actel ProASIC3 and has been utilized by the military [19], automotive and medical applications [20], and the Boeing 787 aircraft [21].

Consequently, to avoid weaknesses, CWE and professional hardware designers provide best practices

and technical support for hardware weaknesses. Complex networks area is a field in computer science that models problems in the form of a graph with connected nodes and edges. The complex networks method is a multidisciplinary approach that can be used in addressing issues by investigating the relations among nodes and edges [22]. This approach is widely used in the computer security literature for investigating a variety of issues [19, 23]. Hence, this study utilizes the concepts of complex networks to investigate hardware vulnerabilities. This study is organized as follows: Section 2 contains some of the related works explained in a laconic way. Section 3 includes a description of the data collection process and the strategy followed in generating the network. Section 4 presents the visualizations and the obtained results. Finally, the paper's conclusion is given in Section 5.

2. RELATED WORK

Some of the most recent works on CWE weaknesses involved relating software weaknesses using complex networks [24] and making recommendations to software developers turning their attention to some of the neglected weaknesses when designing software. Later, the authors of [25] introduced a method to rank and prioritize weaknesses listed by CWE/SANS and OWASP. This method is usually used as a way for increasing CWSS accuracy. In other studies, such as [26], open-source tools are utilized for discovering, identifying, and classifying possible and emerging weaknesses, in addition to describing their advantages and disadvantages. The authors introduced a new weakness checking tool called *HardVul*, which is supposedly able to run on any kind of architecture, and then they reported what they found in their testbed. Similarly, the performance of different architectures was evaluated using the benchmark suite in [26], which was primarily produced to measure user-friendliness in the proposed operating system-friendly microprocessor architecture (OSFA). The authors ended up making it work on other architectures; therefore, studies that compare performance and reduce impact and overhead can be conducted. CWE has listed the findings in its memory corruption section.

Furthermore, research has also been conducted on both software and hardware security and their related weaknesses. For example, the authors of [27] proposed a contemporary and innovative methodology for an HLS-based security-aware system that creates architectures that are efficient (in terms of resources, energy, and performance) whilst also being secure. By doing so, the researchers highlighted then-emerging challenges that had to be faced and overcome by high-level synthesis (HLS) tools to have secure hardware accelerators. The discovery of these challenges led the authors to commence a discussion around hardware weaknesses mentioned in CWE list1 and they focused on how they can alter accelerator behavior through the exploitation of errors in hardware design. Other research articles have had a dissimilar approach, investigating the threats themselves and how to combat them or be proactive in their avoidance at the stage of software and hardware development. A general threat model is presented in [28] about visual sensor network (VSN) applications and their components exploiting their attack surfaces. The STRIDE taxonomy and CWE were used to classify the outlined threats and their weaknesses, respectively, both being considered as popular taxonomies for security weaknesses. After developing a threat model, which displays the possible methods an adversary can compromise the system, a tool for analyzing the threat is used to quantify the risks of a potential attack vector, after which priorities can be decided upon for the mitigation of the security issue. The authors of [28] bring forth a threat model that is relevant and complimentary for previous research in the fields of wireless sensor networks (WSNs) and the internet of things (IoT).

Related works of a slightly different purpose have investigated the efforts spent by academics and industry professionals in documenting and classifying the existing hardware security landscape. The contributions of [29] lie in the examination of joint efforts in a community in the categorization of hardware weaknesses paying particular attention to the CWE database. The authors discovered at the time that these efforts ranged from making classifications of known weaknesses and presenting them as a taxonomy to suggest best practices for the identification, mitigation, and prevention of these issues when designing a product. This is the aim that most research studies on software and hardware weaknesses have in common. The researchers of [30], for example, shed light on hardware weaknesses particular to the internet of things (IoT) and produced an ontology-driven storytelling framework (OSF). This OSF aims at identifying recurrent patterns revealing weaknesses over time, which in turn can be used to assist in mitigating the negative effects of weaknesses or at the least predict and prevent future weaknesses. To provide a profound analysis of the weaknesses and weaknesses found in IoT within CWE and CVE datasets and to be able to study how they are connected and related, in addition to

providing insight for the prevention of emerging weaknesses and their effects, the authors utilized contemporary natural language processing and machine-learning techniques.

Recent research such as [31] have tackled hardware security more generally and presented a basis or a so-called foundation that was established by industry researchers and academics that support the realization of an eco-system of CAD tools that are security-aware, especially covering hardware security and fault-injection assessment for SoC designs and security assurance standardization for electronic design integration. All this was to encourage the creation and development of what they named CAD tools for design for security and security validation and assurance. To polish off this review of related works, it is worth noting that [32] surveyed existing frameworks for public weakness and weakness sharing, examining their efficacy for hardware, and identifying potential gaps. The authors also intended to address potential risks and present potential benefits through analyzing hardware weakness reporting efforts and discussing how to quantify security for hardware. This work focuses on discovering relationships among hardware weaknesses and weaknesses listed by CWE using a complex network approach. From the reviewed studies, it is noticed that the research opportunities still stand for performing various research works in this field.

Accordingly, there is a severe lack in providing studies that consider the relations among hardware security weaknesses during the design phase. This is important, since a weakness may become a side effect of other weaknesses or cause others, which is not considered in the literature. Moreover, it is believed that the scoring system of CWE should focus more on the relations among weaknesses to have more dimensions about the severity of weaknesses. Hence, this work comes to deal with this issue and delivers an approach that models the hardware security weaknesses reported by CWE and reveals the most dangerous ones based on their relations to other weaknesses. The model depends on the notions of complex networks. Likewise, the proposed model can provide a deep view of the weaknesses and the relations among them. What makes this work unique is that the network science approach is utilized to investigate the relations among CWE hardware weaknesses profoundly and provide recommendations to software developers. Moreover, this procedure can be added to the scoring scheme of CWE aiming at having more dimensions about the severity level of weaknesses, which benefits software developers and hardware architects.

3. RESEARCH METHODOLOGY

3.1 Dataset Formation

In this study, the data was gathered from the most accredited source of security weaknesses, the CWE. The strategy followed in the data collection process was based on collecting all the security weaknesses, including the hardware and software ones. In the CWE, each weakness is classified and weaknesses are categorized under certain classes and categories based on the nature of weakness. According to CWE, the classes can be the following: (a) research concepts that deal with the theoretical aspect of the weaknesses; (b) software development that relates to weaknesses that are frequently faced during software design; (c) hardware design that deals with the weaknesses that are often faced through the design; (e) architectural concepts that relate to weaknesses of the software architectural design. It should be mentioned that many weaknesses in CWE are classified under more than one class (mixed).

This means that the weakness may belong to two or three classes at the same time. Besides, a category, in CWE, includes a group of weaknesses that are similar or highly related to each other. Moreover, the CWE provides detailed information about each weakness in terms of relation to other weaknesses. For instance, a weakness W_i from a particular class C_m and a particular category G_n may have relations to other weaknesses that belong to the same or different classes or categories. These relations can be one of the types of relations defined by CWE as follows: W_i is "*ParentOf*" W_j: W_i represents the parent of W_j; W_i is "*MemberOf*" W_j: W_i is in the same category as W_j; W_i is "*ChildOF*" W_j: W_i is a child of W_j; W_i is "*PeerOf*" W_j: W_i *is* like W_j.

Based on the above kinds of relations, the dataset was created accordingly. Hence, the dataset includes the following information for each weakness: *identifier (ID), CWE code, name, class, category, list of relations* to other weaknesses. The data collection process includes 1013 weaknesses from different classes and categories and 2913 relations connecting them. Furthermore, this work considers the hardware design weaknesses and their related attack patterns. However, the key aim is to investigate

hardware security weaknesses. For these weaknesses, the related attack patterns are given to provide the developers with more information about such weaknesses.

3.2 Network Generation and Metrics

As mentioned in Section 1, this study is inspired by complex networks, that is; in turn, based on graph theory. Using this theory, a given problem can be formalized as nodes connected by edges. Accordingly, each security weakness (hardware and software) is considered as a node. An undirected edge is created between two weaknesses *if and only if* one of the relation types in Section 2.1 is held (see Figure 1). This strategy was followed in [24] and [25] when generating a network of weaknesses. In contrast, if there is no relation between a given pair of weaknesses, then no edge is created for this pair. After considering this strategy, a complex network of weaknesses is generated. This network will be analyzed using network measurements at two levels of node and network. Figure 2 demonstrates the preliminary visualization of the network, in that, different colors reflect different classes considered in the created dataset, and nodes size reflects the frequency of connections with other weaknesses. In Figure 2, the colors of nodes refer to the class type of the weaknesses (Yellow = (Research Concepts-Software Development) class, Pink = Hardware Design class, Green = Software Development class, Blue = Attack Pattern class, Brown = Research Concepts class, Dark Green = Architectural Design class and Light Blue = (Research Concepts-Software Development-Architectural Design) class). Node's size reflects the frequency of relation of a weakness; larger sizes denote a high frequency of relations.

Figure 1. Edge creation between two weaknesses.

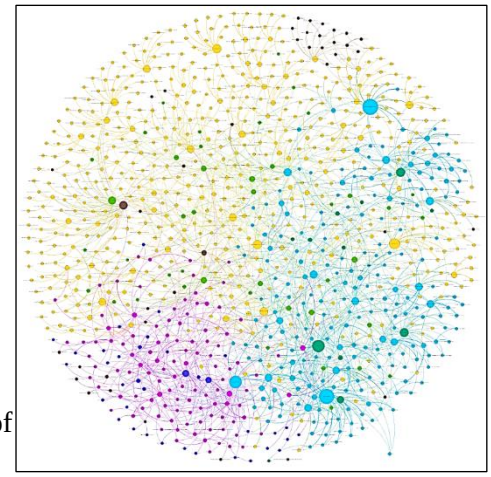


Figure 2. Basic visualization of the CWE network of weaknesses.

After generating the network of CWE weaknesses, the network metrics that are used in the evaluation can be summarized in the context of this work as follows [24]-[25]:

- *Average Degree*: the average number of connections of the weaknesses in the network.
- *Diameter*: the interval between the network's outmost weaknesses (longest connected weaknesses).
- *Density*: the actual number of connections among the weaknesses to the number of all possible connections when reaching a fully connected network.
- *Average Path Length*: the average shortest paths of any given pair of weaknesses in the network.
- *Clustering Coefficient*: the tendency of the weakness to cluster together with other weaknesses in the network and can be calculated as follows [22]:

$$C_i = \frac{2\left|\left\{w_{jk} : w_j, w_k \in N_i, w_{jk} \in R\right\}\right|}{k_i\left(k_i - 1\right)} \quad (1)$$

where R is network relations, w_{jk} is a weakness between the weaknesses w_j and w_k; N_i is the overall count of weaknesses and w_i is the neighbors' weakness. The mean of clustering coefficient (C) in the network is formulated as follows [22]:

$$C_G \frac{\sum_{i=1}^{n} C_i}{N} \quad (2)$$

where C_i is given in Eq. 1; N represents the weaknesses number.

- *Betweenness Centrality*: the number of times that the weakness occurs in the shortest paths of the other weaknesses in the network. In complex networks, this metric reflects the importance of a specific node within a network [22]. Accordingly, it shows how influential a weakness is to be a cause or a side effect of other weaknesses within the network and can be calculated using the following formula [22]:

$$B_c\left(W_j\right) = \sum_{i \neq j \neq k} \frac{\sigma_{i,k}\left(W_i\right)}{\sigma_{i,k}} \tag{3}$$

where $\sigma_{i,k}(W_i)$ is the number of shortest paths between weakness (W_i) and (W_k) passing through the weakness (W_j). The above equation is used for all the available pairs of weaknesses in the network.

- *Closeness Centrality*: reflects how close nodes are to each other [22]. It is an indicator of how close a weakness is to other weaknesses and can be calculated using the following equation [22]:

$$C_c\left(W_j\right) = \frac{N-1}{\sum_k \sigma_{j,k}} \tag{4}$$

where N is the number of weaknesses in the network, $\sigma_{i,k}$ is the shortest path between the weakness j and k.

- *Eigenvector Centrality*: reflects how well-connected a particular node is to the other nodes within a network. This metric shows the degree of connectivity a weakness has to the highly connected weaknesses in the network. If w and z are considered weaknesses, $a_{v,t}$ equals one if they are connected and zero otherwise. The u score for a weakness w is calculated as follows:

$$u(w) = \frac{1}{\lambda} \sum_{z \in M_{(w)}} u_z = \frac{1}{\lambda} \sum_{z \in G_{(w)}} a_{w,z} u_z \tag{5}$$

where G is the network of weaknesses, $M(w)$ is the adjacent of weakness w and λ is the eigenvalue.

4. RESULTS AND DISCUSSION

The first step in the analysis of this work is to visualize the network showing the hardware design weaknesses and attack patterns including the main characteristics of the generated network. Figure 3 shows the visualization of the CWE network of weaknesses with three main types of weaknesses: hardware design weaknesses (pink nodes), attack patterns (blue nodes), and the other weaknesses that belong to other classes (cyan nodes). Table 1 presents the main characteristics of the network.

Table 1. Characteristics of the CWE network of weaknesses.

# of Nodes	# of Edges	Average Degree	Average Clustering Coefficient	Diameter	Density	Average Path Length
1013	2913	5.571	0.155	10	0.006	5.064

Figure 3 reveals how the hardware design weaknesses and attack pattern are highly connected with the other classes of weaknesses (see also Figure 4). This can be considered as an indicator of the impact of non-hardware weaknesses on hardware security design, which is an interesting fact. Based on Table 1, the characteristics of the network also reflect some facts. The average degree of 5.571 shows the weak level of connections among the weaknesses, which is also confirmed when observing the average clustering coefficient that reflects a weak tendency of weaknesses to cluster together. The diameter reflects a long distance from the farthest weaknesses in the network. This is evident since the mean path size is 5.064 with a density level of 0.006. Moreover, the degree distribution of the nodes in the CWE network of weaknesses follows a power-law distribution as demonstrated in Figure 5, in that the x-axis represents the frequency of connections and the y-axis is the number of weaknesses. This means that few weaknesses appear with a high frequency of connections, while many of them have few connections. It can be inferred, according to the Pareto Rule [33], that 20% of the weaknesses dominate the

connections in the network. These metrics indicate that the relations among weaknesses are most likely restricted by the classes and the categories of the weaknesses, as shown in Figure 6. The relations are denser within the same class and less across classes. In Figure 6, the classes are encoded with different colors as follows: Yellow = (Research Concepts-Software Development) class, Pink = Hardware Design class, Green = Software Development class, Blue = Attack Pattern class, Brown = Research Concepts class, Dark Green = Architectural Design class and Light Blue = (Research Concepts-Software Development-Architectural Design) class). Node's size reflects the frequency of relation of a weakness; larger sizes denote a high frequency of relations.

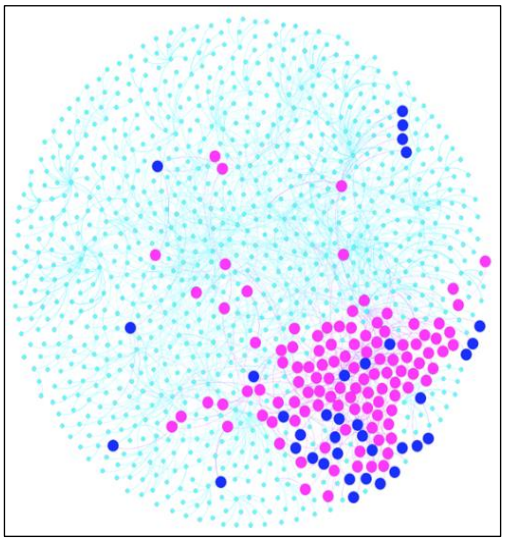


Figure 3. CWE network visualization showing the connections of the hardware design and attack patterns.

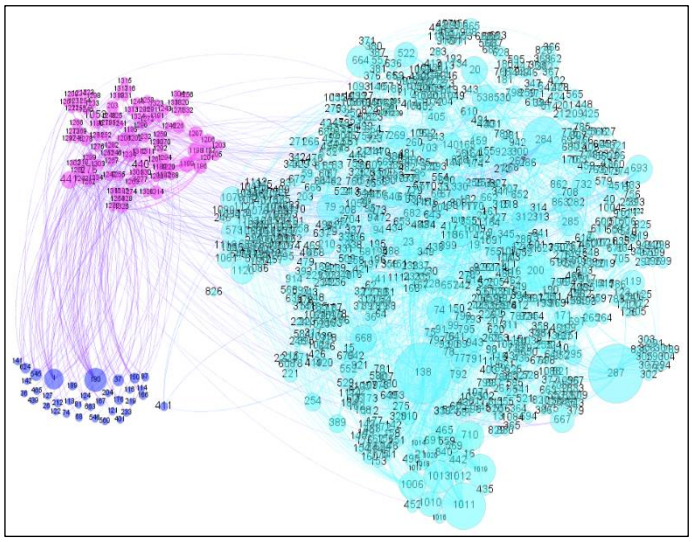


Figure 4. The relations between the hardware design weaknesses (pink color), attack patterns (blue), and other classes' weaknesses (cyan).

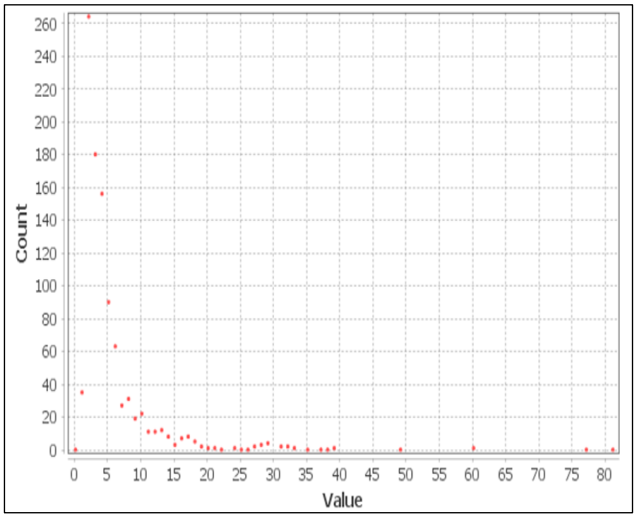


Figure 5. Degree distribution of the CWE network of weaknesses.

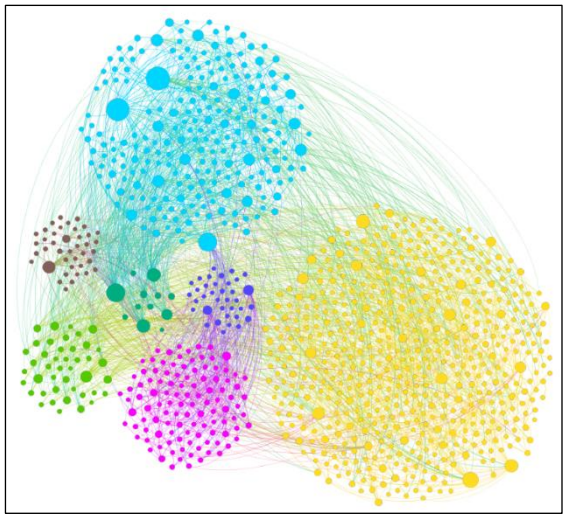


Figure 6. The density of the relations among different kinds of classes of weaknesses.

Now, the next step is to analyze the hardware weaknesses and attack patterns. Hardware weaknesses and attack pattern nodes are extracted from the main network, as shown in Figure 7. Surprisingly, the attack patterns do not have relations to each other, in that all their relationships are with weaknesses from other classes. The reason behind this case is that all the attack patterns are not original, and they are originated from other classes' weaknesses. For instance, the "Improper Resource Locking (CWE-413)" is a ChildOf the "Improper Locking (CWE-667)" weakness that belongs to the research concepts class.

Alternatively, the hardware design weaknesses are better connected compared to the attack patterns. Figure 8 depicts the hardware weaknesses and how they are connected. In this figure, nodes size reflects

the value of betweenness centrality (the larger the size, the higher the value of betweenness centrality). This means that the most influential weaknesses have larger sizes. Moreover, the number of weaknesses in the network is 107 connected by 140 relations (edges). As mentioned, the betweenness centrality measurement reflects how influential a weakness is in a community of weaknesses; that is, it represents the number of times a weakness is positioned in the shortest paths of the other pairs of weaknesses. Therefore, the hardware weaknesses are ranked using their betweenness centrality values as presented in Table 2. Besides, not all hardware weaknesses have appeared in the table, because values have dropped to zero. The table also presented the other metrics for each weakness (degree centrality, closeness centrality, eigencentrality, and clustering coefficient).

Based on Table 2, the CWE-441 (unintended proxy or intermediary ("confused deputy") obtained the highest betweenness value, which means that it is the most influential hardware design weakness. This weakness relates to the access control issues when an unintended proxy is performed. This matter should be given more attention by software developers since it appears more frequently in the shortest paths of the network pairs of weaknesses. In the second rank, the CWE-1208 appears, which is a category of weaknesses that relate to improper protection of hardware. As can be seen, many weaknesses have not been given enough focus in the literature, but they have a significant impact on the security of hardware design.

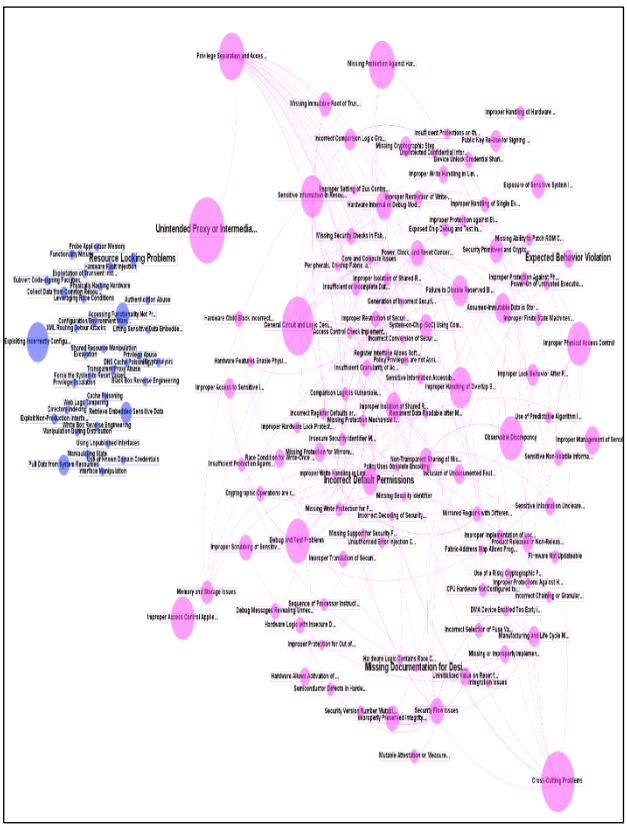


Figure 7. Visualization of the hardware design weaknesses (pink nodes) and attack patterns (blue nodes).

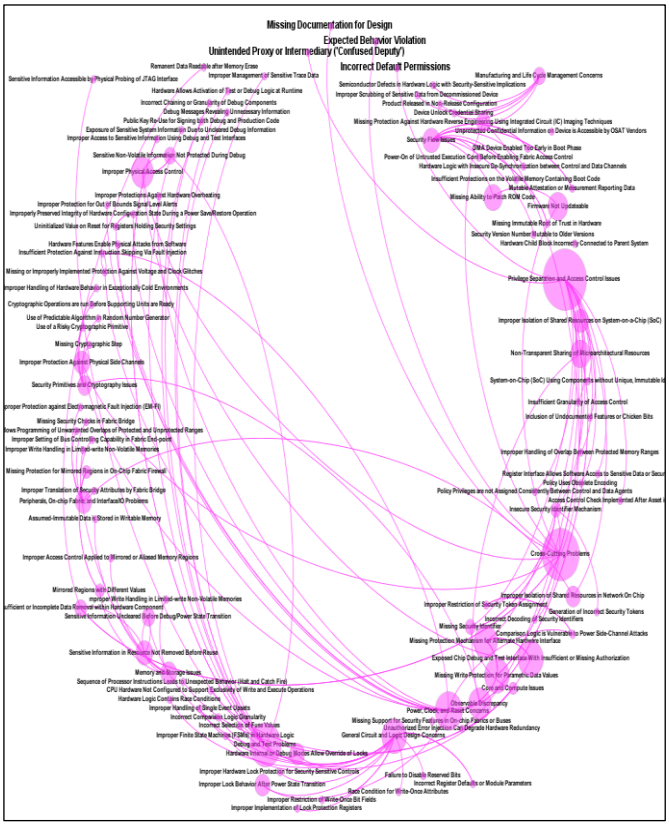


Figure 8. Visualization of the connections among hardware design weaknesses.

Furthermore, in terms of clustering coefficient, three weaknesses have strong tendencies to cluster with other weaknesses as a side effect of a cause. These weaknesses gained the highest levels of clustering coefficient; CWE-1189 (improper isolation of shared resources on system-on-a-chip (SoC)), CWE-276 (incorrect default permissions), and CWE-1304 (improperly preserved the integrity of hardware configuration state during the power save/restore operation). Software developers should be aware of the risk of these three weaknesses since they may impact the whole system in terms of security.

As can be observed in Table 2, the CWE-276 surprisingly has the highest value of eigencentrality. This means that it is connected to the highly connected weaknesses in the network, which makes it more dangerous when compared to the other network weaknesses. From the closeness centrality levels, it can

be noticed that most of the weaknesses in Table 2 have close levels, meaning that they are considered close to most of the weaknesses in the network. Many interesting facts can be extracted from Table 2 since it includes the best-connected CWE weaknesses. It should be mentioned that the values presented in Table2 were extracted from the whole network of weaknesses. In Table2, the CWE code of each weakness is hyperlinked to its web page CWE website. Table 3 presents the most recent list of the most important hardware weaknesses in CWE in 2021, in that only two weaknesses in Table 2 (bold and underlined) are shown in the 2021 most recent list of CWE (Table 3). This means that more attention should be given to the rank in Table 2 and the relations among weaknesses are crucial to be considered by software developers during the design phase.

By looking at the results, visualizations, and network metrics values, a better understanding can be obtained of the weaknesses and their relations to each other by hardware designers. Moreover, the outcomes provided in this work may consume time and effort during the design, testing, and maintenance phases. This is important since they reduce the total software cost. Finally, more analysis is needed of network edges that may hide some unseen patterns about the weaknesses. This can be performed by generating a network of all the weaknesses available in the CWE list.

Table 2. Prioritizing the hardware design weaknesses according to network centrality measurements and ranking them based on their betweenness.

RANK	CWE	Weaknesses	Category	Centrality Measurements				Clustering Coefficient
				Betweenness	Degree	Closeness	Eigen	
1st	441	Unintended Proxy or Intermediary ('Confused Deputy')	Privilege Separation and Access Control Issues	0.0154	16	0.2420	0.0559	0.0250
2nd	1208	Cross-Cutting Problems	Cross-cutting Problems	0.0140	10	0.2213	0.0218	0.0222
3rd	1199	General Circuit and Logic Design Concerns	General Circuit and Logic Design Concerns	0.0135	14	0.2295	0.0296	0.0110
4th	1278	Missing Protection against Hardware Reverse Engineering Using Integrated Circuit (IC) Imaging Techniques	Manufacturing and Life Cycle Management Concerns	0.0106	8	0.2291	0.0282	0.0000
5th	1198	Privilege Separation and Access Control Issues	Privilege Separation and Access Control Issues	0.0104	18	0.2287	0.0677	0.0261
6th	1207	Debug and Test Problems	Debug and Test Problems	0.0096	14	0.2114	0.0417	0.0000
7th	203	Observable Discrepancy	Security Primitives and Cryptography Issues	0.0096	10	0.2082	0.0160	0.0222
8th	1263	Improper Physical Access Control	Cross-cutting Problems	0.0094	7	0.2342	0.0480	0.0000
9th	1257	Improper Access Control Applied to Mirrored or Aliased Memory Regions	Memory and Storage Issues	0.0093	5	0.2350	0.0572	0.0000
10th	226	Sensitive Information in Resource Not Removed Before Reuse	Memory and Storage Issues	0.0090	10	0.2095	0.0232	0.0667
11th	**1260**	**Improper Handling of Overlap Between Protected Memory Ranges**	**Privilege Separation and Access Control Issues**	**0.0088**	**4**	**0.2360**	**0.0544**	**0.0000**
12th	1209	Failure to Disable Reserved Bits	General Circuit and Logic Design Concerns	0.0060	3	0.2145	0.0137	0.0000
13th	1282	Assumed-Immutable Data is Stored in Writable Memory	Memory and Storage Issues	0.0058	5	0.2417	0.0327	0.0000
14th	**1189**	**Improper Isolation of Shared Resources on System-on-a-Chip (SoC)**	**Privilege Separation and Access Control Issues**	**0.0055**	**6**	**0.2316**	**0.0259**	**0.2000**
15th	1234	Hardware Internal or Debug Modes Allow Override of Locks	Belonging to 2 Categories in Hardware Weaknesses.	0.0055	9	0.2113	0.0458	0.0000

17th	1323	Improper Management of Sensitive Trace Data	Debug and Test Problems	0.0053	5	0.2272	0.0441	0.0000
18th	276	Incorrect Default Permissions	Privilege Separation and Access Control Issues	0.0053	11	0.2316	0.1035	0.2182
19th	1196	Security Flow Issues	Security Flow Issues	0.0047	11	0.2041	0.0272	0.0000
20th	1304	Improperly Preserved Integrity of Hardware Configuration State during a Power Save/Restore Operation	Power, Clock and Reset Concerns	0.0047	6	0.2376	0.0517	0.1333
21st	1266	Improper Scrubbing of Sensitive Data from Decommissioned Device	Manufacturing and Life Cycle Management Concerns	0.0046	6	0.2036	0.0137	0.0000
22nd	1206	Power, Clock and Reset Concerns	Power, Clock and Reset Concerns	0.0046	11	0.2144	0.0333	0.0364

Table 3. The 2021 list of the CWE's most important hardware weaknesses.

#	CWE	Title
1	1189	Improper Isolation of Shared Resources on System-on-a-Chip (SoC)
2	1191	On-Chip Debug and Test Interface with Improper Access Control
3	1231	Improper Prevention of Lock Bit Modification
4	1233	Security-Sensitive Hardware Controls with Missing Lock Bit Protection
5	1240	Use of a Cryptographic Primitive with a Risky Implementation
6	1244	Internal Asset Exposed to Unsafe Debug Access Level or State
7	1256	Improper Restriction of Software Interfaces to Hardware Features
8	1260	Improper Handling of Overlap between Protected Memory Ranges
9	1272	Sensitive Information Uncleared before Debug/Power State Transition
10	1274	Improper Access Control for Volatile Memory Containing Boot Code
11	1277	Firmware Not Updateable
12	1300	Improper Protection of Physical Side Channels

5. CONCLUSIONS

In this study, a thorough sight of important hardware weaknesses is provided *via* the creation of a network model that contains the most common weaknesses reported in common weakness enumeration. In addition, the generation of the network considered three main types of weaknesses, which are hardware design, attack patterns, and other classes. The study showed how hardware design and attack patterns are highly connected. Moreover, it demonstrates how the hardware design weaknesses are better connected when compared to the attack patterns. The analysis revealed that the CWE-441is the most influential hardware design weakness. This weakness relates to the access control issues when the unintended proxy is performed. It also showed that the CWE-1189, CWE-276, and CWE-1304 gained the highest levels of clustering coefficient, which means that software developers should be aware of the risk of these three weaknesses, since they may impact the whole system security. As future work, we plan to combine all the weaknesses provided by CWE in one network model and deeply explore and reveal the unseen facts about the weaknesses.

ACKNOWLEDGMENTS

The authors are grateful to the staff and faculty members at the Department of Computer Science at the University of Mosul for supporting them constantly. The authors also would like to thank the contributors of the CWE for their efforts in providing worldwide researchers with data on security weaknesses.

REFERENCES

[1] T. Gaddis, Starting out with Python, ISBN-13: 978-0134444321, Harlow, UK: Pearson, 2018.

[2] A. Sengupta, "Hardware Vulnerabilities and Their Effects on CE Devices: Design for Security against Trojans [Hardware Matters]," IEEE Consumer Electronics Magazine, vol. 6, no. 3, pp. 126–133, 2017.

[3] M. Alenezi, M. Zagane and Y. Javed, "Efficient Deep Features Learning for Vulnerability Detection Using Character N-gram Embedding," Jordanian Journal of Computers and Information Technology, vol. 7, no. 1, pp. 25-38, 2021.

[4] P. A. Wortman, F. Tehranipoor and J. A. Chandy, "Exploring the Coverage of Existing Hardware Vulnerabilities in Community Standards," Proc. of the Silicon Valley Cybersecurity Conference (SVCC2020), pp. 87–97, DOI:10.1007/978-3-030-72725-3_6, 2021.

[5] G. Bloom, E. Leontie, B. Narahari and R. Simha, "Hardware and Security: Vulnerabilities and Solutions," Chapter 12, pp. 305-331, Handbook on Securing Cyber-Physical Critical Infrastructure, Morgan Kaufmann, 2012.

[6] CVE, "Terminology," [Online], Available: https://cve.mitre.org/about/terminology.html, [Accessed: 27-Oct-2021].

[7] B. Martin, "Common Vulnerabilities Enumeration (CVE), Common Weakness Enumeration (CWE) and Common Quality Enumeration (CQE)," ACM SIGAda Ada Letters, vol. 38, no. 2, pp. 9–42, 2019.

[8] CWE, "Common Weakness Enumeration," [Online], Available: https://cwe.mitre.org/index.html, [Accessed: 28-Oct-2021].

[9] S. Bhunia and M. H. Tehranipoor, Hardware Security: A Hands-on Learning Approach, ISBN-13: 978-0128124772, Cambridge, MA: Morgan Kaufmann Publishers, 2019.

[10] C. Li and J.-L. Gaudiot, "Detecting Malicious Attacks Exploiting Hardware Vulnerabilities Using Performance Counters," Proc. of the 43rd IEEE Annual Computer Software and Applications Conference (COMPSAC), pp. 588-597, DOI: 10.1109/COMPSAC.2019.00090, Milwaukee, WI, USA, 2019.

[11] M. Seaborn and T. Dullien, "Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges," Black Hat Briefings, pp. 1-71, [Online], Available: https://www.blackhat.com/docs/us-15/materials/us-15-Seaborn-Exploiting-The-DRAM-Rowhammer-Bug-To-Gain-Kernel-Privileges.pdf, 2015.

[12] E. Bosman, K. Razavi, H. Bos and C. Giuffrida, "Dedup Est Machina: Memory Deduplication As an Advanced Exploitation Vector," Proc. of the IEEE Symposium on Security and Privacy (SP), pp. 987-1004, DOI 10.1109/SP.2016.63, San Jose, CA, USA, 2016.

[13] D. Gruss, C. Maurice and S. Mangard, "Rowhammer.js: A Remote Software-induced Fault Attack in JavaScript," Proc. of the 13th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), arXiv:1507.06955, pp. 300–321, 2016.

[14] Y. Xiao, X. Zhang, Y. Zhang and R. Teodorescu, "One Bit Flips, One Cloud Flops: Cross-VM Row Hammer Attacks and Privilege Escalation," Proc. of the 25th USENIX Security Symposium (USENIX Security 16), pp. 19-35, Austin, TX, USA, 2016.

[15] V. van der Veen, Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H. Bos, K. Razavi and C. Giuffrida, "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms," Proc. of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 1675–1689, DOI: 10.1145/2976749.2978406, 2016.

[16] K. Razavi, B. Gras, E. Bosman, B. Preneel, C. Giuffrida and H. Bos, "Flip Feng Shui: Hammering a Needle in the Software Stack," Proc. of the 25th USENIX Security Symposium (USENIX Security 16), pp. 1-18, Austin, TX, USA, 2016.

[17] A. Tatar, R. K. Konoth, E. Athanasopoulos, C. Giuffrida, H. Bos and K. Razavi, "Throwhammer: Rowhammer Attacks over the Network and Defenses," Proc. of the USENIX Annual Technical Conference (USENIX ATC 18), pp. 213-226, Boston, MA, USA, 2018.

[18] Y. Jang, J. Lee, S. Lee and T. Kim, "SGX-Bomb: Locking Down the Processor *via* Rowhammer Attack," Proc. of the 2nd Workshop on System Software for Trusted Execution, pp. 1-6, DOI: 10.1145/3152701.3152709, 2017.

[19] A. Ferraiuolo, R. Xu, D. Zhang, A. C. Myers and G. E. Suh, "Verification of a Practical Hardware Security Architecture through Static Information Flow Analysis," ACM SIGARCH Computer Architecture News, vol. 45, no. 1, pp. 555–568, 2017.

[20] T. Yaqoob, H. Abbas and M. Atiquzzaman, "Security Vulnerabilities, Attacks, Countermeasures and Regulations of Networked Medical Devices—A Review," IEEE Communications Surveys & Tutorials, vol. 21, no. 4, pp. 3723–3768, 2019.

[21] A. Stander and J. Ophoff, "Cyber Security in Civil Aviation," Imam Journal of Applied Sciences, vol. 1, no. 1, pp. 23-26, 2016.

[22] R. Albert and A.-L. Barabási, "Statistical Mechanics of Complex Networks," Reviews of Modern Physics, vol. 74, no. 1, pp. 47–97, 2002.

[23] C. Li, Securing Computer Systems through Cyber Attack Detection at the Hardware Level, PhD Thesis, University of California, Irvine, 2020.

[24] Z. K. Younis and B. Mahmood, "Towards the Impact of Security Vulnerabilities in Software Design: A Complex Network-based Approach," Proc. of the 6th Int. Engineering Conf. "Sustainable Technology and Development" (IEC), pp. 157-162, DOI: 10.1109/IEC49899.2020.9122923, Erbil, Iraq, 2020.

[25] B. Mahmood, "Prioritizing CWE/SANS and OWASP Vulnerabilities: A Network-based Model," International Journal of Computing and Digital Systems, vol. 10, no. 1, pp. 361–372, 2021.

[26] S. Trecakov, C. Tran, H. Badawy, N. Siddique, J. Acosta and S. Misra, "Can Architecture Design Help Eliminate Some Common Vulnerabilities?" Proc. of the 14th IEEE Int. Conf. on Mobile *Ad Hoc* and Sensor Systems (MASS), pp. 590-593, DOI: 10.1109/MASS.2017.100, Orlando, FL, USA, 2017.

[27] C. Pilato, S. Garg, K. Wu, R. Karri and F. Regazzoni, "Securing Hardware Accelerators: A New Challenge for High-level Synthesis," IEEE Embedded Systems Letters, vol. 10, no. 3, pp. 77–80, 2018.

[28] J. Simonjan, S. Taurer and B. Dieber, "A Generalized Threat Model for Visual Sensor Networks," Sensors, vol. 20, no. 13, p. 3629, 2020.

[29] P. A. Wortman, F. Tehranipoor and J. A. Chandy, "Exploring the Coverage of Existing Hardware Vulnerabilities in Community Standards," Proc. of the Silicon Valley Cybersecurity Conference, Virtual, pp. 87–97, [Online], Available: https://svcc2020.svcsi.org/accepted-papers/Exploring-the-Coverage-of-Existing-Hardware-Vulnerabilities-in-Community-Standards, 2021.

[30] C. Bandi, S. Salehi, R. Hassan, S. M. P D, H. Homayoun and S. Rafatirad, "Ontology-driven Framework for Trend Analysis of Vulnerabilities and Impacts in IoT Hardware," Proc. of the 15th IEEE International Conference on Semantic Computing (ICSC), pp. 211-214, DOI: 10.1109/ICSC50631.2021.00045, Laguna Hills, CA, USA, 2021.

[31] S. Aftabjahani, R. Kastner, M. Tehranipoor, F. Farahmandi, J. Oberg, A. Nordstrom, N. Fern and A. Althoff, "Special Session: CAD for Hardware Security - Automation Is Key to Adoption of Solutions," Proc. of the 39th IEEE VLSI Test Symposium (VTS), pp. 1-10, DOI: 10.1109/VTS50974.2021.9441032, San Diego, CA, USA, 2021.

[32] J. Bellay, D. Forte, R. Martin and C. Taylor, "Hardware Vulnerability Description, Sharing and Reporting: Challenges and Opportunities," Proc. of Annual GOMACTech Conf., pp. 1-7, [Online], Available: http://dforte.ece.ufl.edu/wp-content/uploads/sites/65/2021/05/GOMACTech_conf.pdf, 2021.

[33] A. Clauset, C. R. Shalizi and M. E. J. Newman, "Power-law Distributions in Empirical Data," SIAM Review, vol. 51, no. 4, pp. 661–703, 2009.

ملخص البحث:

تلعب معدّات الحاسوب دوراً رئيسياً في حياتنا اليومية. وعلى الرّغم من النّهضة التكنولوجية، ثمّة مسائل تتعلّق بالأمان في معدّات الحاسوب ترجع الى نقاط ضعف في حاجةٍ الى معالجتها بعناية. لذا تقدم هذه الورقة رؤية متعمّقة لنقاط الضّعف في أمان تصميم معدّات الحاسوب، وذلك عبر تقديم نموذج لشبكة يحتوي على نقاط الضّعف الأكثر شيوعاً الواردة في سجلّ نقاط الضّعف الأكثر شيوعاً (CWE). ويتمثل الهدف الأساسي للشّبكة المقترحة في إجراء تحليل معمّق للعلاقات بين التّصاميم المختلفة لمعدّات الحاسوب ونقاط الضّعف المتعلّقة بالأمان في تلك التصاميم. وبناءً على نتائج التّحليل، تقدم هذه الورقة مجموعةً من التّوصيات التي من شأنها أن تفيد مختلف الأطراف، ومنهم مصمّمو الأمان في معدات الحاسوب. وبناءً على ذلك، يرتكز منهج التّحليل الى مجموعةٍ من المفاهيم المستوحاة من مجال علم الشّبكات. وقد تمّ إظهار النّموذج المقترح في شكل مخطط تكون فيه العُقَد هي نقاط الضّعف، في حين تتشكل الحواف إذا كانت نقاط الضّعف مرتبطة ببعضها البعض. وقد تمّ الحصول على نتائج واعدة من الممكن ملاحظتها في نموذج الدراسة. فمثلاً، نقاط الضّعف CWE-441 وCWE-1189 و CWE-276 و CWE-1304 لم تحْظ بالإهتمام الكافي، وعليه فإنّه يجب أخذها بعين الاعتبار من جانب مطوّري البرمجيات. كذلك تمّ تقديم لائحة تبين ترتيب نقاط الضّعف الخاصة بالمعدّات بناءً على قياسات الشّبكة، ومقارنتها بأحدث اللوائح الصادرة لترتيب نقاط ضعف المعدات عن سجلّ نقاط الضعف الأكثر شيوعاً (CWE). وقد وُجد أنّ هناك نقطتي ضعف مشتركتين فقط بين اللائحتين، مما يؤشر الى أنّ لائحة (CWE) لا تأخذ العلاقات بين نقاط الضّعف بعين الاعتبار.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

MELANOMA SKIN LESION CLASSIFICATION USING IMPROVED EFFICIENTNETB3

Saumya R. Salian and Sudhir D. Sawarkar

(Received: 4-Nov.-2021, Revised: 26-Dec.-2021, Accepted: 12-Jan.-2022)

ABSTRACT

Malignant skin cancer is one the most common and lethal type of skin cancer. Early detection of cancerous skin lesions will increase the possibility of patient survival. In recent years, implementation of models built on deep neural networks in building medical diagnostic imaging systems is quite beneficial to medical experts. In this study, we present an improved and fine-tuned EfficientNetB3 model to classify malignant skin lesions using the concept of fine-tuning transfer learning. We have performed a comparative analysis of different deep learning pre-trained models, like ResNet50, InceptionV3, InceptionResNetV2 and EfficientNet B0-B2 models. The analysis findings signify the ability of utilizing fine-tuned EfficientNetB3 in the mission of melanoma detection and development of a computer-aided diagnostic system. All experimental procedures were carried out on ISBI-ISIC 2017 dataset. To check the efficiency of the proposed model, we compare the proposed model with EfficientNetB3 baseline model and present state-of-art pre-trained methods and approaches. The proposed EfficientNetB3 model obtained an accuracy of 87.12%, a recall of 87.00%, a precision of 87.00% and an F1 score of 85.00%. The proposed model achieved good computational results and efficaciously addressed the problem of model over-fitting and abated false negative labels.

KEYWORDS

Melanoma, Deep learning, Classification, Skin lesions, Computer vision.

1. INTRODUCTION

Cancer disease is minacious to human life. At present, over 100 different cancers are known that affect humans. One of the most fast-expanding and lethal cancers is skin cancer. Malignant melanoma is a class of skin lesion cancer that is considered as most precarious and prevalent type of cancer. Melanoma skin cancer occurs due to abnormal reproduction of pigmented melanocytes cells. The key factor that risks to the expansion of melanoma skin cancer cases is the increased exposure to natural or artificial ultra violet rays, tanning beds and sunburns. According to Skin Cancer Foundation statistics, it is estimated that new cases of melanoma cancer would intensify close to 5.8% and count of fatality rate is likely to intensify by 4.8% [1]. If skin cancer is not diagnosed at an early stage, 80% of cases may result in death. Therefore, it becomes important that identification of melanoma at a very preliminary stage will remarkably escalate the survival chance of a patient. Medical experts or dermatologists usually examine skin surface using dermotoscopy to identify the skin lesions [2]. Visual inspection using dermatoscopy is a subjective technique and its diagnosis often depends on the medical practitioner's experience. For unbiased diagnosis of melanoma skin cancer, an automated and computerized image analysis system is a prime requirement. Computer-aided diagnostic systems will aid medical experts to utilize technological advances, but also have second opinion. To support dermatologists to achieve faster results in diagnosis of skin cancer with lesser computational time, it is necessary to develop computer-aided image processing systems which automatically classify malignant skin lesions. In the last few years, there have been a notable increase in the quest to build computer-aided diagnostic solutions to diagnose malignant skin lesions. Before 2015, research was typically based on conventional machine learning computer vision algorithms to detect skin cancer. Deep-learning models have accomplished outstanding results in building diagnostic computer-aided systems.

Convolutional Neural Network (CNN) has achieved remarkable results on tasks, like melanoma classification, object recognition, image recognition, …etc. In recent years, deep neural networks along with transfer learning models have been used together to work on large datasets [3]. Transfer learning is a technique of reusing knowledge gained by a model trained on a specific domain to build the solutions for related problems [12]. Transfer learning assists in lowering the time required to train the network

S. R. Salian and S. D. Sawarkar are with Department of Computer Engineering, Datta Meghe College of Engineering, Navi Mumbai, India.
Emails: srs.cm.dmce@gmail.com and sudhir_sawarkar@yahoo.com

and reducing out-of-sample error. ImageNet is a huge dataset having more than 15 million labelled images. The diverseness and ordered organization of ImageNet provide exceptional opportunities to explore and further research in the field of computer vision systems. All the deep learning models that we are analyzing are pre-trained on ImageNet [13] dataset consisting of 1000 classes and containing features with respect to their weights and biases. In order to use these models to adapt to our two class skin lesion classification problem, fine-tuning of the network is important. Pre-trained model is fine-tuned when the target dataset is different from the original dataset on which the model was trained. Performance of the pre-trained model can be remarkably enhanced by fine-tuning the model on target dataset instead of training from the beginning. In fine tuning transfer learning, we take underlying weights of a pre-trained model and adjust (fine tune) them to fit to our dataset.

In this work, we report a deep learning-based image classification system that classifies the dermoscopic images into two classes of malignant and benign melanoma. We propose an improved and fine-tuned EfficientNetB3 model for melanoma lesion classification. EfficientNetB3 model belongs to the family of EfficientNet B0-B7 [15] models that are previously trained on huge datasets such as ImageNet. In most of the baseline Convolution Neural Networks (CNNs), initial layers of the model extract much less features and the last layers of the model bring out comprehensive features. Instead of building the CNN architecture from scratch, we have utilized network architecture of EfficientNetB3 model by freezing the initial layers of the model and then fine tuning their weights in order to classify them accurately to our two-class skin lesion classification task. We use various data augmentation techniques and add customized layers of global average pooling, drop out layers to minimize the model over-fitting problem and change the last classification layer with softmax layer that classifies the benign and malignant skin lesions.

Most of the pre-trained CNN architectures suffer from model over-fitting problem. With the purpose to obliterate this drawback, fine tuning of pre-trained network plays an essential role in building an efficient computer-aided diagnostic model for classification objective. In this study, we examine the efficiency of our proposed model in detection of malignant melanoma skin lesion. We compare our fine-tuned EfficientNetB3 model with baseline EfficientNetB3 model without additional layers of data augmentation, global average pooling and dropout. A comparative analysis of the proposed model with other state-of-the-art advanced neural networks, like ResNet50, InceptionV3, InceptionResNetV2 and EfficientNetB0-B2 is carried out in this paper. We also compare our proposed model with deep-learning networks proposed by other researchers. We carry out a comparative analysis of model performance evaluated on metrics like accuracy, recall, precision and F1-score. We have carried out our experiments on dataset available from ISBI-ISIC 2017 Challenge with the purpose to improve the diagnosis of melanoma [4]. Experiment analysis indicates that our proposed EfficientNetB3 model performs better than all other pre-trained models and provides a promising result that tackles the problem of model over-fitting in deep neural pre-trained models. Our model can be used as a decision support solution to help dermatologists in the diagnosis process. The paper is outlined in following manner. Section 2 provides a detailed quintessence of recent works and Section 3 furnishes a detailed description of the proposed methodology. Section 4 contains an analysis and a comparison of the results obtained. We deduce the paper along with future scope of research in Section 5.

1.1 Contribution

- We present an improved EfficientNetB3 model that aids in diagnosis of malignant melanoma with better accuracy.
- We perform a comparative analysis of existing pre-trained models and investigate the model performance on malignant melanoma classification task.
- We employ the concept of fine tuning by adjusting the learned weights of EfficientNetB3 model on our malignant melanoma skin lesion classification task and add custom layers of data augmentation, global average pooling (GAP) layer, dropout layer and dense softmax classification layer during the training and testing phases to achieve better accuracy.
- The proposed model provides promising results to handle the problem of model over-fitting which is usually observed in pre-trained deep neural models.

2. RELATED WORKS

Zabir et al. [5] presented an automatic segmentation and classification model built on deep learning with transfer learning. To scale up training data, different augmentation methods are applied on the training dataset. The authors carry out semantic segmentation using U-Net model on augmented images and further apply various deep convolution neural networks to extract features, such as VGG16, VGG19, ResNet50, InceptionResNetV2, DensetNet201, InceptionV3 …etc. and compare classification results with several classifiers, such as SVM, Random Forest, Decision Tree and AdaBoost. The proposed algorithm achieved an accuracy of 0.92 with DenseNet201 used as feature extraction model and SVM as classification model.

Nils et al. [6] submitted an ensemble-based deep neural network model for lesion classification. They applied cropping techniques to address the problem of images with different resolutions and loss balancing approach to handle the class imbalance dataset. Multi-resolution EfficientNet model coupled with comprehensive data augmentation were researched and an accuracy of 0.928 was achieved for melanoma lesion classification.

Hasib et al. [7] proposed a two-stage unified deep learning framework for automatic skin cancer image classification using adversarial training fused with transfer learning to comprehensively handle inter-class diversity and imbalance class dataset. The classifier is trained by continually decreasing the focal loss function that supports the model in learning. The MelaNet model achieved an F1 score value of 94%, which is better compared to other baseline models, like VGG-Gap and VGG-Gap+Augment.

Jason et al. [8] proposed a melanoma classification deep learning-based approach combined with conventional image processing method to achieve superior results in terms of accuracy. Fusion model with cross-validation achieved a classification accuracy of 0.94, which is better than ResNet-50 classifier and traditional classifiers for image processing. AUC values of 0.87 and 0.90 were observed when ResNet-50 and a traditional image processing-based classifier were applied, respectively.

Arthur et al. [9] presented a computer based diagnostic system employing CNN framework. The proposed diagnostic system classified the lesion images into classes, such as seborrheic keratosis, nevi and melanoma. The authors applied an ensemble and aggregation method along with directed acyclic graph technique on three-class CNN in the context of improving the model performance. The proposed network accomplished an accuracy of 76.6%.

Maria et al. [10] presented a comparative analysis of various deep learning models on ISIC dataset for melanoma classification. The authors investigated the influence of pre-processing stage on varied neural network approaches, like 2D-CNN, self-organizing neural networks and ResNet. Canny edge detector along with morphological techniques, like Ostu thresholding and dilation operation, were used to identify hair artefacts. Further, image pre-processing was carried out using inpainting method to reconstruct the original image without hair artefacts. Skin lesion segmentation to highlight the lesion region was performed using bitwise AND binary operator. ResNet model and achieved an accuracy of 81.5% on ISIC dataset for melanoma classification.

Jasil et al. [11] reported a comparative study for identification of skin lesion type employing transfer learning technique, utilizing deep-learning networks. All experimental analyses were examined on ISIC dataset and pre-prcoessing techniques, like image normalization, image resizing and augmentation, were performed on datasets to raise the preciseness of neural network models. In particular, InceptionV3, VGG16 and VGG19 models were reviewed for skin lesion classification task. From the experimental evaluation, VGG16 furnished superior results with an accuracy of 77%.

3. PROPOSED METHODOLOGY

We provide a comprehensive description of our advanced method for malignant skin lesion classification in the current section. The proposed methodology consists of the following sub-sections: (1) Dataset and data preparation, (2) Data augmentation, (3) EficientNetB3 network, (4) Fine-tuning with global average pooling, (5) Fine-tuning with dropout. (6) Fine-tuning fully connected classification layer.

3.1 Dataset and Data Preparation

We have carried out our research on ISBI-ISIC 2017 dataset [4] accessible from “Skin Lesion Analysis

toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), hosted by the International Skin Imaging Collaboration (ISIC)". The dataset consists of 3297 lesion images classified into 1800 benign class skin lesion images and 1497 malignant class skin lesion images. Training and testing of the pre-trained model are carried out ISBI-ISIC 2017 dataset. The dataset was spilt in the ratio 80:20 training and testing set comprising 2637 training data images and 660 testing data images. Figure 1 depicts sample images of ISBI-ISIC 2017 dataset of malignant class. Images of skin lesion were of varied pixel sizes in the RGB color space. For the model training and testing, we resized the image into 224×224 pixels. The images of skin lesion were resized to 224×224 pixel resolution to make images compatible with EfficientNetB3 model.

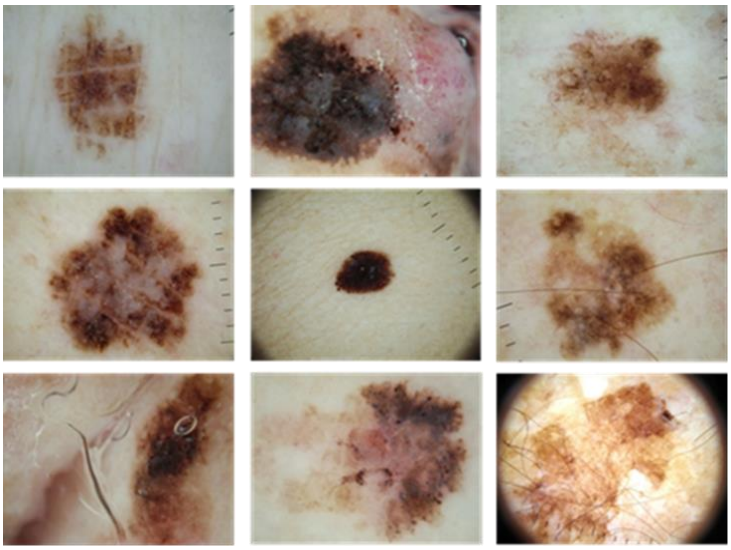


Figure 1. Skin lesion images from ISBI-ISIC dataset.

3.2 Data Augmentation

One of the significant techniques to reinforce the efficiency of neural network method, it is important to train the network with enormous wealth of data. Most of the computer vision tasks consist of lesser dataset, which results in building poor classification models. Pre-trained deep-learning models are trained on a huge dataset such as ImageNet and perform substantially well on huge datasets. To increase the size of our training data and to build an efficient melanoma skin lesion classification model, we use data augmentation technique. Data augmentation is an approach of increasing the number of copies of training data artificially by slight modification on original data without actually gathering new data. Image augmentation is one the most common techniques used to improve the performance of deep-learning models and reduce over-fitting problem.

To increase the efficacy of our, we add an augmentation layer on top of EfficientNetB3 neural network layer. To enhance the size of our training data, we create altered variants of training images that belong to the same class. Training dataset is expanded using different augmentation techniques, like horizontal random flip, rotation range, zoom range, width range and height. In our data augmentation layer, we use various data augmentation transformation techniques with specific range values, as shown in Table 1. To use data augmentation right within our model, we'll create a Keras sequential model consisting of only data pre-processing layers. We then use this sequential model within another functional model. Figure 2 indicates the data augmentation layer created using various techniques.

Table 1. Data augmentation techniques.

Data Augmentation Technique	**Description**
Random Flip	Images are flipped horizontally.
Random Rotation	Images are rotated randomly by 0.2 value.
Random Zoom	Images are Zoomed randomly by 0.2 value.
Random Height	Image height is shifted randomly by 0.2 value.
Random Width	Image width is shifted randomly by 0.2 value.

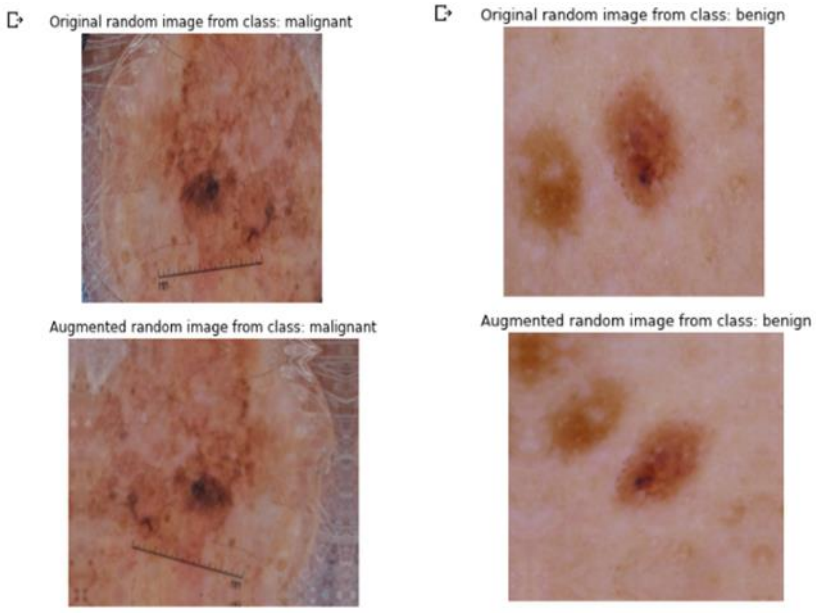


Figure 2. Random augmented images.

3.3 EfficientNetB3 Network Architecture

EfficientNet [15] consists of a family of models from B0 to B7 and is considered as one of the most computationally efficient deep-learning models trained over ImageNet. EfficientNet models are based on compound scaling method which deftly expands a baseline convolution network model size to target model size in an efficient manner, attaining top model accuracy gain. Compound scaling method allows the network to be uniformly scaled across width, depth and resolution. Figure 3 shows the compound scaling strategy in EfficientNet compared to baseline model. The EfficientNet model is comprised of different types of mobile inverted bottleneck convolution blocks MBConv with varied kernel size of 3x3 and 5x5. We expand the network depth, width and resolution uniformly using compound scaling coefficient ∅ in the following manner:

$$d=\alpha^{\emptyset},\ w=\beta^{\emptyset},\ r=\gamma^{\emptyset}\ \ such\ that\ \alpha\beta^2\gamma^2 \approx 2\ ,\ \alpha\geq 1, \beta\geq 1,\ \gamma\geq 1 \quad (1)$$

where d is depth, w is width, r is resolution and α, β, γ are constants.

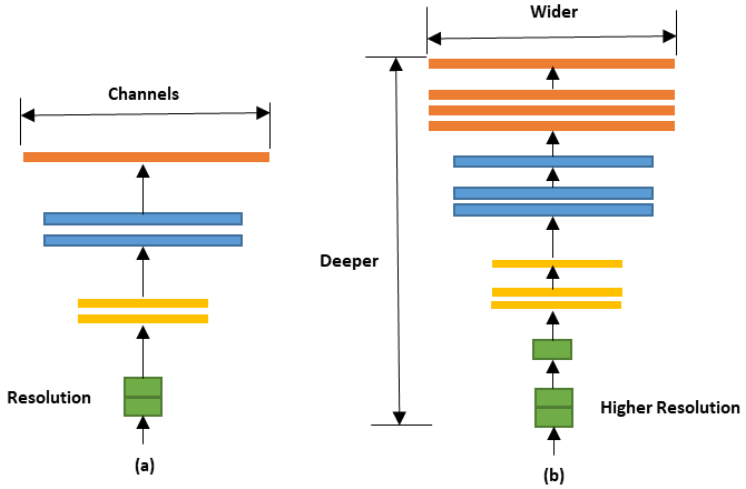


Figure 3. Scaling of EfficientNet model: (a) Baseline model and (b) Compound scaling model.

The value of ∅ is user-specified and helps scale up the network and identify computation resources available for the model. EfficientNetB0 model is built up by using values of ∅= 0, w=1, d=1, r=1 representing the baseline model. EfficientNetB0 is comprised of MBconv1 and MBcon6 layers. Likewise, EfficientNetB3 model is constructed by using values of ∅= 3, w= α^3, d= β^3, r= γ^3, indicating that more resources are available to acquire superlative performance. EfficientNetB3 model consists of

more layers of MBCon6 with inverted residual connection. EfficientNetB3 model consists of deeper network compared to baseline model, which apprehends intricate and richer features and generalizes well on new missions. EfficientNetB3 consists of a wider network that can extract optimal features and patterns that are beneficial for classification task. Along with improved accuracy, EfficientNetB3 model also ameliorates efficiency by decreasing FLOPS (Floating Point Operations per Second) and parameters. Figure 4 shows EfficientNetB3 network architecture.

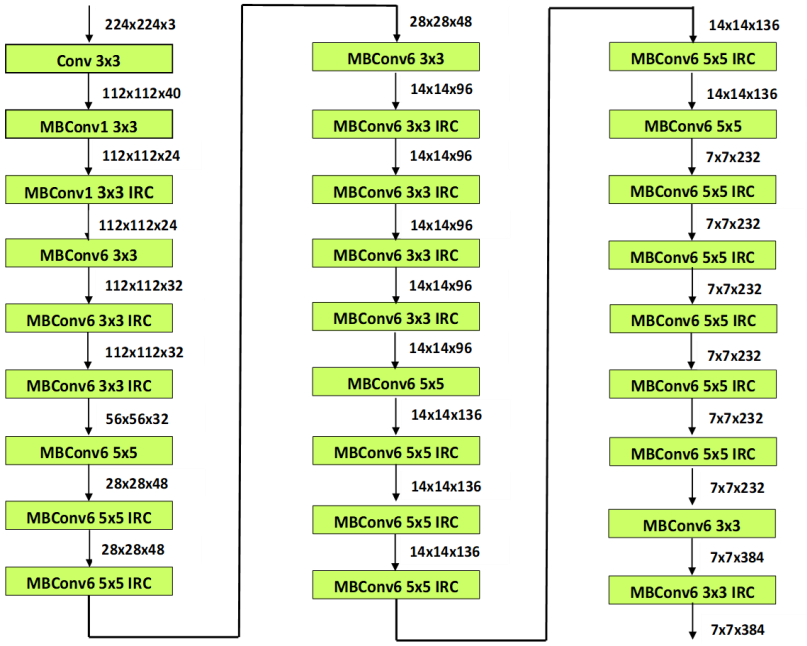


Figure 4. EfficientNetB3 network architecture.

3.4 Fine Tuning with Global Average Pooling

EfficientNetB3 model outputs a list of feature vectors or feature maps which are used to extract patterns. The motive behind using global average pooling is to generate a single feature map for each benign and malignant lesion category of the classification task. Rather than appending fully connected layers over feature maps, we take average of each feature map to generate a feature vector belonging to each class. Feature maps generated through global average pooling layer represent confidence maps of each benign and malignant category. Global average pooling layer diminishes spatial dimensionality of the feature maps, thus shrinking the count of feature parameters in the model and improving model performance by reducing the problem of model over-fitting [19]. We use tf.keras.layers.GlobalAveragePooling2D() layer to transform 4-dimensional tensor into 2-dimensional tensor by condensing the input tensor shape of (1,4,4,3) to (1,3).

3.5 Fine Tuning with Dropout Layer

To train the EfficientNetB3 model faster and to avert the neural network from over-fitting, we add a dropout layer. Due to its large architecture, EfficientNetB3 model produces a large number of parameters which are often slow to use during training time. With dropout technique, one can overcome this problem by randomly dropping units from the neural network during training, thus thinning the neural network. By using dropout layer, it becomes easier for the model during the testing phase to estimate the averaging results of all these thinned network predictions by utilizing unthinned network having smaller weights [18]. We choose 0.2 value of probability p of dropout in our proposed model to achieve a higher accuracy and reduce the over-fitting problem from the proposed neural network. In this study, we show that dropout layer improves the performance of EfficientNetB3 neural network for melanoma skin lesion classification task.

3.6 Fine-tuning Fully Connected Classification Layer

In this stage, we merge together all the feature vectors that we have received from the previous stages. We pass the feature vectors received from the dropout layer further into the network for classification

as an output layer. The top layer of baseline EfficientNetB3 pre-trained model outputs 1000 classes, because it was pre-trained on ImageNet. To adjust to our two-class skin lesion classification problem, we fine-tune the pre-trained network to adapt to only two classes of malignant and benign skin lesions. In order to calculate the class probabilities of each input, we have used softmax layer as an output layer for EfficientNetB3 model classifier that works on our target dataset of two classes of skin lesions. The real and predicted value difference is calculated using loss function. We build an output activation layer using tf.keras.layers.Dense() and meld the inputs with the output using the model tf.keras.Model(). Figure 5 shows the architecture of our proposed methodology.

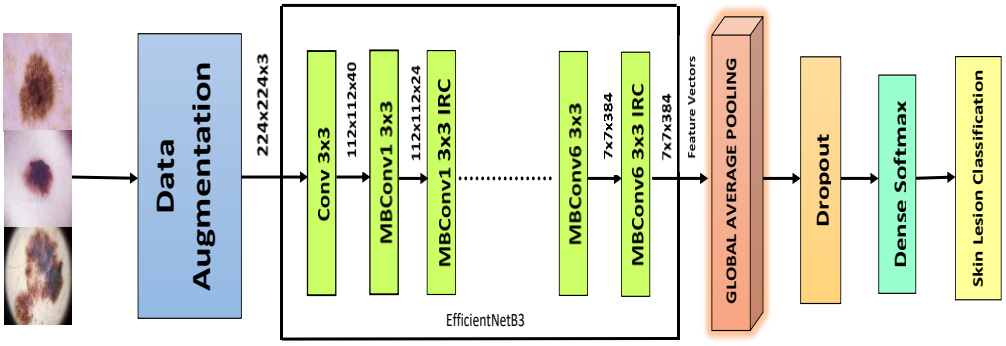


Figure 5. Proposed methodology.

4. Experimental Results and Discussion

To check the efficiency of our model, we carried out our experimental analysis in 3 approaches. We first compare the fine-tuned proposed EfficeinetNetB3 model with the pre-trained base EfficientNetB3 model on ISBI-ISIC 2017 dataset. In the second approach, we perform comparative analysis of the proposed model with present state-of-the-art pre-trained classification methods, such as ResNet50, InceptionV3, InceptionResNetV2 and EfficientNetB0-B2 on ISBI-ISIC 2017 dataset. In the third approach, we carry out comparative analysis of our improved model with models proposed by other authors.

4.1 Experimental Specification

We have performed all our experiments on Google Colab Python notebook which provided access to free 12Gb Tesla K80 NVIDIA GPU of SMI 470.74. We have used Adam as our optimization algorithm with a rate of learning of 0.001 to compile the models. All the methods are trained with a count of 35 epochs with 32 batch size. We investigated the efficiency of our proposed model by computing recall, F1 score, precision and accuracy. The metrics are computed on True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) cases.

True Positive (TP)- correct label is positive and predicted label is positive.
False Positive (FP)- correct label is negative and predicted label is positive.
True Negative (TN)- correct label is negative and predicted label is negative.
False Negative (FN)- correct label is positive and predicted label is negative.

Accuracy: It is an evaluation metric that finds the model performance across all classes. It is the fraction of the sum of correct predictions to the sum of total predictions.

$$\text{Accuracy} = \frac{((TN+TP))}{((TN+TP+FN+FP))} \tag{2}$$

Precision: It is an evaluation metric that calculates the fraction of the total positive samples over the sum of total positive samples either classified precisely or imprecisely.

$$\text{Precision} = \frac{TP}{((TP+FP))} \tag{3}$$

Recall: It is an evaluation metric that calculates the fraction of the total positive samples over the sum of total positive input samples classified correctly.

$$\text{Recall} = \frac{TP}{((TP+FN))} \quad (4)$$

F1-score: It is an evaluation metric that finds the accuracy of the mode by combining both precision and recall by giving more weightage to false positives and false negatives.

$$\text{F1} - \text{score} = \frac{2TP}{((2TP+FP+FN))} \quad (5)$$

4.2 Comparison of Proposed Model with Base Pre-trained EfficientNetB3

In our proposed model, we incorporated data augmentation technique and additional layers of global average pooling (GAP), dropout and dense softmax classification to increase the exactness of skin lesion classification results. We compare our proposed model with baseline EfficientNetB3 model without data augmentation, global average pooling layer and dropout layer. The final classification layer of base pre-trained EfficientNetB3 model is replaced by dense softmax layer with the correct number of output classes (two class) to adjust the model to our classification task. From Table 2, it is observed that baseline EfficientNetB3 model performed much less with an accuracy of 56.97%, precision, F1-score and recall of 58.00%. Our proposed model gave an accuracy of 87.12% and precision, F1-score and recall of 87.00%.

Table 2. Comparative result of proposed model with baseline EfficientNetB3.

Approach	Dataset	Accuracy	Precision	Recall	F1-score
Baseline EfficientNetB3	ISBI-ISIC 2017	56.97	58.00	58.00	58.00
Proposed Model	**ISBI-ISIC 2017**	**87.12**	**87.00**	**87.00**	**87.00**

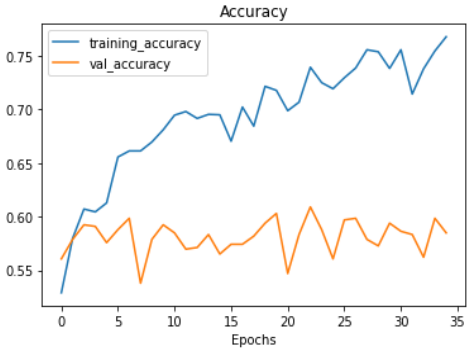


Figure 6. Accuracy curve of baseline EfficientNetB3.

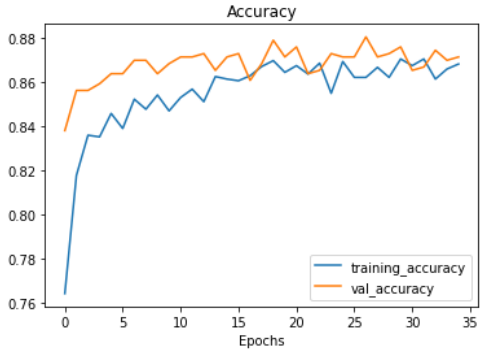


Figure 7. Accuracy curve of the proposed model.

Figure 6 shows the accuracy curve of baseline EfficientNetB3 and Figure 7 shows the accuracy curve of the proposed fine-tuned model. In Figures 6 and 7, x-axis represents the epochs and y-axis represent the accuracy value. From Figure 6, it was observed that baseline EfficientNetB3 gave inferior results and showed a huge gap between training and validation accuracy curves resulting into model over-fitting. From Figure 7, it was noticed that the proposed model provided superior results, as the gap among accuracy curves of training and validation is fairly reduced. In some cases, the training and testing accuracy curves are overlapping, which indicates the proposed model avoids over fitting problem.

4.3 Comparison of Proposed Model with Other Pre-trained Models

We further review the potency of the proposed model by comparing it with present state-of-the-art baseline pre-trained methods. We present comparative analysis of our proposed model with pre-trained models, such as ResNet50, InceptionV3, InceptionResNetV2, EfficientNetB0, EfficientNetB1 and EfficientNetB2. ResNet-50 is a variant of residual neural network with 50 layers pre-trained on ImageNet. ResNet-50 model includes five stages consisting of convolution block and identity block. At each block of convolution and identity, 3 convolutional layers are present, respectively [14]. Applying the concept of skip connections is the strength of ResNet model. Skip connection assists in reducing the vanishing gradient problem in deep neural network by skipping through some of the layers during the

training. InceptionV3 is a deep-learning model pre-trained on ImageNet dataset comprising of recurrent inception modules. Inception module consists of convolution layer of 1×1, 3×3, 5×5, max pooling layer and concatenation layer [16]. InceptionResNetV2 [17] is another deep-learning model trained on ImageNet dataset. It is a hybridized model elicited from residual connection and inception modules. It consists of varied-sized convolutional filters mapped with residual connections with 164 deep layers. Concatenation layer of the inception module is missing in the InceptionResNetV2 model. Combination of inception architecture with residual connection accelerates the training speed. EfficientNetB0-B2 [15] belongs to the family of EfficientNet models based on compound scaling method. These models are considered to attain top accuracy gain and are also computationally efficient deep-learning models pre-trained on ImageNet dataset.

All these models are pre-trained on ImageNet dataset and contain features with respect to 1000 classes. To adjust these models to our ISBI-ISIC 2017 dataset, we use softmax layer as the last classification network layer with two classes. Only the top layer of these pre-trained models is adjusted and the rest of layers remain frozen. All these pre-trained models are trained with a count of 35 epochs with 32 batch size. Adam optimization algorithm with a rate of learning rate of 0.001 is used to compile all the models. We evaluate their performances on the ISBI-ISIC 2017 dataset for melanoma skin lesion classification task. From Table 3, it is observed that ResNet50 achieved the second best result and provided an accuracy of 84.85%, precision, recall and F1-score of 85.00%. Figure 8 indicates the accuracy curve of ResNet50. InceptionV3 and InceptionResNetV2 provided accuracy values of 82.73% and 83.33%, respectively. Figure 9 shows the accuracy curve of InceptionResNetV2. In Figures 8 and 9, x-axis represents the epochs and y-axis represents the accuracy value. EfficientNetB0, EfficientNetB1 and EfficientNetB2 gave accuracy values of only 60.15%, 55.75% and 58.48%, respectively. The proposed model outperformed present state-of-the-art deep-learning methods on ISBI-ISIC 2017 dataset with highest F1-score of 87.00%. To investigate the efficacy of the proposed model, we analyze the confusion matrix of our model with ResNet50, InceptionV3 and InceptionResNetV2 models. Figure 10 shows the confusion matrix of the proposed model having lesser number of false negative and false positive labels as compared to other state-of-the-art advanced neural network methods ResNet50, InceptionV3 and InceptionResNetV2.

Table 3. Comparative results of the proposed model with state-of-the-art pre-trained methods.

Approach	Dataset	Accuracy	Precision	Recall	F1-score
ResNet50	ISBI-ISIC 2017	84.85	85.00	85.00	85.00
InceptionV3	ISBI-ISIC 2017	82.73	83.00	83.00	83.00
InceptionResNetV2	ISBI-ISIC 2017	83.33	83.00	83.00	83.00
EfficientNetB0	ISBI-ISIC 2017	60.15	65.00	60.00	59.00
EfficientNetB1	ISBI-ISIC 2017	55.75	55.00	56.00	55.00
EfficientNetB2	ISBI-ISIC 2017	58.48	66.00	66.00	66.00
Proposed Model	**ISBI-ISIC 2017**	**87.12**	**87.00**	**87.00**	**87.00**

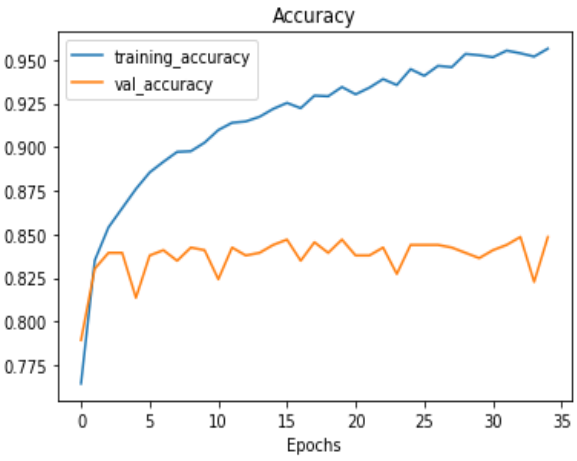


Figure 8. Accuracy curve of ResNet50.

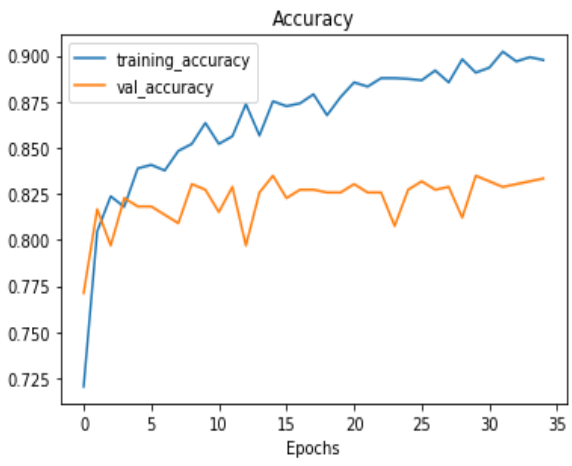


Figure 9. Accuracy curve of InceptionResNetV2.

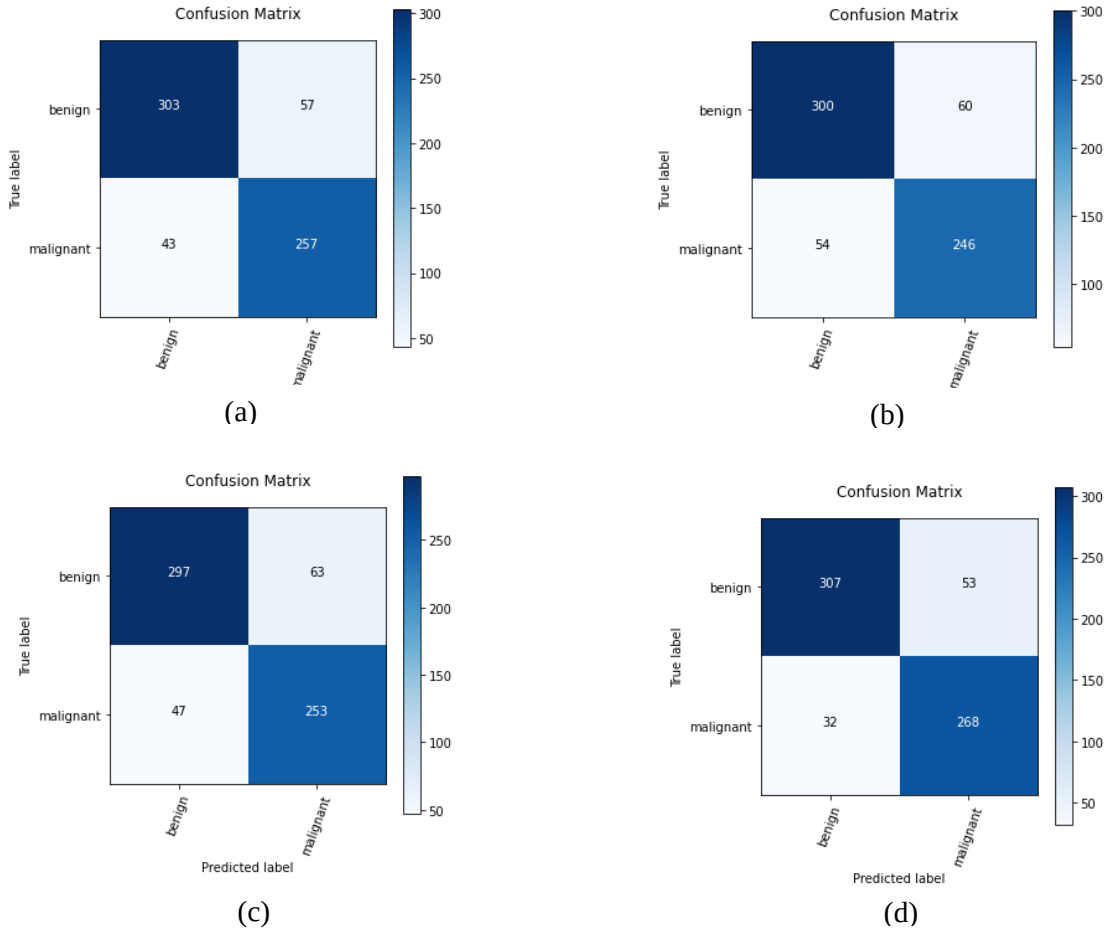


Figure 10. (a) Confusion matrix of ResNet50, (b) Confusion matrix of InceptionV3, (c) Confusion matrix of InceptionResNetV2 and (d) Confusion matrix of the proposed model.

4.4 Comparison of the Proposed Model with Other Approaches

In addition, we analyze the findings of the proposed model with previous works carried out on ISIC dataset. In Table 4, we summarize the evaluation outcomes of the proposed model with those of other approaches. The proposed model accomplished an accuracy of 87.12% on ISBI-ISIC 2017 dataset, whereas Hasib et al. [7] achieved an accuracy of 81.18% using MelaNet deep-learning model with adversial training and transfer learning. Arthur et al. [8] used a combination of decision-directed acyclic graph with VGG_19 CNN deep-learning model and obtained an accuracy of 76.6%. Maria et al. [10] presented a deep neural network-based melanoma classification approach along with pre-processing technique for removal of hair artefacts and lesion segmentation technique. The authors achieved an accuracy of 81.5% on ResNet model and an accuracy of 74.1% on 2D CNN model on ISBI-ISIC 2017 dataset. Jasil et al. [11] inspected the performance of deep-learning networks, like InceptionV3, VGG16 and VGG19 using pre-processing techniques, like image normalization, image resizing and augmentation on ISIC dataset. According to [11], VGG19 achieved an accuracy of 76%. It can be noted that the proposed approach furnished more accurate results compared to other approaches from Table 4.

Table 4. Comparative results of the proposed model with other approaches.

Reference	Dataset	Approach	Accuracy %
Proposed Model	**ISBI-ISIC 2017**	**Fine-tuned EfficientNetB3**	**87.12**
Hasib et al. [7]	ISBI-ISIC 2016	MelaNet	81.18
Arthur et al. [9]	ISIC-2018	DDAG VGG_19_2	76.6
Maria et al. [10]	ISBI-ISIC 2017	ResNet	81.5
		2D CNN	74.1
Jasil et al. [11]	ISIC-2018	VGG16	77
		VGG19	76
		InceptionV3	74

5. CONCLUSIONS

Melanoma skin cancer is fatal for human being and timely diagnosis of skin cancer is needed. Building a computer-aided diagnostics system for classification of melanoma skin lesion is important. In this paper, we propose an improved, comprehensive and fine-tuned EfficientNetB3 deep neural network model to categorize lesions into malignant and benign. We employ fine-tuning transfer-learning concept by utilizing network architecture of EfficientNetB3 model by freezing the initial layers of the model and then fine-tuning their weights in order to classify accurately to our two-class skin lesion classification task. We employ a range of data augmentation approaches to upgrade the correctness of the model in the training phase. We fine-tune EfficientNetB3 model, by adding customized layers of global average pooling (GAP), dropout and dense softmax classification node. In this paper, we provided comparative analysis of pre-trained deep-learning models for malignant melanoma classification. We reviewed the performance of ResNet50, InceptionV3, InceptionResNetV2 and EfficientNet B0-B2 pre-trained models. From the results, it was observed that these pre-trained models suffered model over-fitting problem. We tried to tackle the problem of model over-fitting by utilizing augmentation techniques and adding layers of global average pooling, dropout and dense softmax, which achieved promising results on EfficientNetB3 model. In this paper, we furnished an improved EfficientNetB3 model to categorize lesions into malignant and benign with promising results. The investigational results reflected the capability of employment of our proposed model in the task of malignant melanoma classification. Future scope of our research would be training and testing on more datasets and checking the efficiency of our proposed model aiding to build a reliable computer-aided diagnostic system for melanoma diagnosis. The study can be extended for future work by inspecting larger architectures and building deeper fine-tuned models.

REFERENCES

[1] Cancer.org, "Cancer Facts and Figures 2021," [Online], Available: https://www.cancer.org/ content/ dam/cancer-org/research/cancer-facts-and-statistics/annual-cancer-facts-and-figures/2021/cancer-facts-and-figures-2021.pdf, 2021.

[2] S. Sonthalia, S. Yumeen and F. Kaliyadan, "Dermoscopy Overview and Extradiagnostic Applications," StatPearls [Internet], Treasure Island (FL): StatPearls Publishing; PMID: 30725816, [Online], Available: https://www.ncbi.nlm.nih.gov/books/NBK537131/, Jan. 2022.

[3] K. Munir, H. Elahi, A. Ayub, F. Frezza and A. Rizzi, "Cancer Diagnosis Using Deep Learning: A Bibliographic Review," Cancers (Basel), vol. 11, no. 9, p. 1235, DOI: 10.3390/cancers11091235, 2019.

[4] Isic-archive, "ISIC Challenge Datasets," [Online], Available: https://challenge.isic-archive.com/data/.

[5] A. Nazi, Zabir and Tasnim Azad Abir, "Automatic Skin Lesion Segmentation and Melanoma Detection: Transfer Learning Approach with U-Net and DCNN-SVM," Proc. Int'l conf. on Computational Intelligence, pp. 371-381. Springer, DOI: 10.1007/978-981-13-7564-4_32, Singapore, 2020.

[6] N. Gessert, M. Nielsen, M. Shaikh, R. Werner and A. Schlaefer, "Skin Lesion Classification Using Ensembles of Multi-resolution EfficientNets with Meta Data," MethodsX, vol. 7, no. 100864, p. 100864, DOI:10.1016/j.mex.2020.100864, 2020.

[7] H. Zunair and A. B. Hamza, "Melanoma Detection Using Adversarial Training and Deep Transfer Learning," Phys. Med. Biol., vol. 65, no. 13, p. 135005, DOI: 10.1088/1361-6560/ab86d3, 2020.

[8] J. R. Hagerty, R. J. Stanley, H. A. Almubarak et al., "Deep Learning and Handcrafted Method Fusion: Higher Diagnostic Accuracy for Melanoma Dermoscopy Images," IEEE Journal of Biomedical and Health Informatics, vol. 23, pp. 1385-1391, DOI: 10.1109/JBHI.2019.2891049, 2019.

[9] A. C. Foahom Gouabou, J.-L. Damoiseaux, J. Monnier, R. Iguernaissi, A. Moudafi and D. Merad, "Ensemble Method of Convolutional Neural Networks with Directed Acyclic Graph Using Dermoscopic Images: Melanoma Detection Application," Sensors (Basel), vol. 21, no. 12, p. 3999, DOI: 10.3390/s21123999, 2021.

[10] M. Frasca, M. Nappi, M. Risi, G. Tortora and A. A. Citarella, "A Comparison of Neural Network Approaches for Melanoma Classification," Proc. of the 25th IEEE International Conference on Pattern Recognition (ICPR), pp. 2110–2117, DOI:10.1109/ICPR48806.2021.9412893, Milan, Italy, 2021.

[11] S. P. G. Jasil and V. Ulagamuthalvi, "Deep Learning Architecture Using Transfer Learning for Classification of Skin Lesions," Journal of Ambient Intelligence and Humanized Computing, vol. 2021, DOI: 10.1007/s12652-021-03062-7, 2021.

[12] F. Zhuang, Q. Zhiyuan, D. Keyu, X. Dongbo, Z. Yongchun, Z. Hengshu, X. Hui and H. Qing, "A Comprehensive Survey on Transfer Learning," Proceedings of the IEEE, vol. 109, no. 1, pp. 43–76, DOI: 10.1109/JPROC.2020.3004555, 2021.

[13] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural

Networks," Commun. of the ACM, vol. 60, no. 6, pp. 84–90, DOI: 10.1145/3065386, 2017.

[14] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, DOI: 10.1109/CVPR.2016.90, 2016.

[15] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," Proc. of the Int. Conf. on Machine Learning, ArXiv, vol. abs/1905.11946, pp. 6105-6114, 2019.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826, DOI: 10.1109/CVPR.2016.308, Las Vegas, NV, USA, 2016.

[17] C. Szegedy, S. Ioffe, V. Vanhoucke and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," Proc. of the 31st AAAI Conference on Artificial Intelligence (AAAI'17), vol. 31, no. 1, pp. 4278–4284, 2017.

[18] H. Lim, "A Study on Dropout Techniques to Reduce Overfitting in Deep Neural Networks," Proc. of Advanced Multimedia and Ubiquitous Engineering, Part of the Lecture Notes in Electrical Engineering Book Series, vol. 716, pp. 133-139, DOI: 10.1007/978-981-15-9309-3_20, 2021.

[19] T.-Y. Hsiao, Y.-C. Chang, H.-H. Chou and C.-T. Chiu, "Filter-based Deep-compression with Global Average Pooling for Convolutional Networks," Journal of Systems Architecture, vol. 95, pp. 9–18, DOI: 10.1016/j.sysarc.2019.02.008, 2019.

[20] A. El-Halees and M. Tafish, "Breast Cancer Severity Predication Using Deep Learning Techniques," Jordanian Journal of Computers and Information Technology (JJCIT), vol. 06, no. 01, pp. 94-102, 2020.

[21] S. F. Abuowaida and H. Y. Chan, "Improved Deep Learning Architecture for Depth Estimation from Single Image," Jordanian Journal of Computers and Information Technology (JJCIT), vol. 06, no. 04, pp. 434-445, 2020.

ملخص البحث:

يُعدّ سرطان الجلد الخبيث أحد أخطر أنواع السرطان وأوسعها إنتشاراً. وإنّ الاكتشاف المبكّر لتقرُّحات سرطان الجلد يَزيدُ من احتمال نجاة المريض. وفي السّنوات الأخيرة، فإنّ تطبيق نماذج تقوم على الشّبكات العصبية العميقة في أنظمة التّصوير التّشخيصية يُعدّ أمراً ذا فائدة للخبراء في الحقل الطبي.

في هذه الدّراسة، نقدّم نموذجاً محسَّناً دقيق الضّبط باستخدام (EfficientNetB3) لتصنيف تقرُّحات الجلد الضّارّة باستخدام مفهوم تعليم النّقل ذي الضّبط الدّقيق. كذلك نجري مقارنةً تحليليةً لنماذج مختلفة من نماذج التعلّم العميق المدرّبة مسبقاً، مثل: (ResNet50) و(InceptionV3) و(InceptionResNetV2) و(EfficientNetB0-B2). وقد أظهرت نتائج التّحليل انّ نموذج (EfficientNetB3) دقيق الضّبط من الممكن الاستفادة منه في لكشف عن سرطان الجلد وتطوير أنظمة حاسوبية للتّشخيص. وقد جرى تطبيق كلّ الإجراءات على مجموعة البيانات (ISBI-ISIC2017). ولاختبار فاعلية النّموذج المقترح، تمّت مقارنته بنموذج (EfficientNetB3) الأساسي القائم على الطّرق القائمة المستندة الى التّدريب لمسبق. وقد حقّق النّموذج المقترح دقّة بلغت 87.12%، بينما كانت النّسبة 87% لكلٍّ من الاستعادة والضّبط، و 85% لدرجة F1. كذلك حقّق النّموذج المقترح في هذه الدّراسة نتائج حسابية جيدة، وعالج مشكلة الملاءمة الفائضة، ووضع حدّاً للأوسام السّالبة الخاطئة (FN).

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

ED_{25519}: A NEW SECURE COMPATIBLE ELLIPTIC CURVE FOR MOBILE WIRELESS NETWORK SECURITY

Mausam Das[1] and Zenghui Wang[2]

(Received: 7-Nov.-2021, Revised: 9-Jan.-2022, Accepted: 24-Jan.-2022)

ABSTRACT

Wireless Sensor Networks (WSNs) create various security threats, such as application variance in different sectors along with the model of cryptographic primitivity and necessity. Despite modernistic evolution, the skillful utilization of Elliptic Curve Cryptography (ECC) for WSNs is a very progressive investigation topic and approaches to reduce the time and intensity cost. Security of ECC commits on the hardness of the Elliptic Curve Discrete Logarithm Problem. Many elliptic curve standards are available, such as ANSI X9.62, NIST FIPS 186-2 …etc. Due to some drawbacks in NIST curves associated with security matters, it is important to investigate for secure alternatives. In our work, we will select ED_{25519} (Edwards curve) at the 128-bit security level and contrast it with Weierstraß curve (also known as Weierstrass curve). To complete the field-calculation functions, we utilize a radix-2^4 , which illustrates the operands with MoTE-ECC for Memsic's MICAz motes over Optimal Prime Fields (OPFs) of variable size; e.g. 160, 192, 224 and 225 bits. We take ECDH (Elliptic-curve Diffie–Hellman) key interchange among two nodes where every node needs two scalar multiplications to execute. The scalar multiplication over twisted Edwards curve utilizes a comb technique to establish base point and utilizes extended projective coordinates for point summation. Our implementation shows that an ECDH takes 18.20 mJ energy consumption over 160-bit OPF, which is performing better than AVR-based sensor node. The advantages of our proposed method will grant advance security and power consumption and diminish communication burden through key management.

KEYWORDS

Domain name system security extension (DNSSEC), Secure real-time transport protocol (SRTP), Secure/Multipurpose internet mail extension (S/MIME), Spectrum-aware degree-ranking-based energy-efficient clustering (SDEC), Advance virtual RISC (Reduced instruction set computing (AVR)), Twisted Edwards curve (TE), Weierstrass curve (WEI).

1. INTRODUCTION

Elliptic Curve Cryptography (ECC) was recommended separately by Victor Miller and Neal Koblitz in 1985 and it started to benefit in cryptographic standards [1]. Cryptographic essential protocols (Transport Layer Security (TLS) protocols, key exchange protocols, public key encryption, digital signatures) that use ECC became very famous due to their small key sizes, exceptional computational performance. Using ECC often yields Perfect Forward Secrecy (PFS), as compared to RSA(Rivest, Shamir, Adleman). In this work, we will consider cryptographic primitives with so-called Optimal Prime Fields (which grant for capable modular rebate), where security builds on Elliptic Curve Discrete Logarithm Problem (ECDLP). To obtain the achievement, we also desire for a lightweight application with small amount of RAM and ROM. Different investigators and associations have suggested many elliptic curves (ECs), such as Weierstraß and NIST; those are used mainly for key exchange and digital signatures. Few prominent examples are Elliptic Curve Diffie-Hellman Key Exchange (ECDHE) and Elliptic Curve Digital Signature Algorithm (ECDSA). For different security levels, NIST has suggested a few prime and binary elliptic curves [5]. Nonetheless, the research community raised ambiguity on the security of Weierstraß or NIST recommended curves for complexity on scalar multiplication within Dual Elliptic Curve Deterministic Random Bit Generator [7], [20] and did not replicate the present state-of-the-art of ECC in terms of efficiency. Due to this reason, we select alternative elliptic curves with better performance and greater security level [31]. Some suggest Brain pool curves developed by Teletrust [8]. Bernstein recommended Montgomery curve; Curve25519 [9]. A set of elliptic curves have been proposed by J.-W.-Bos et al. of Microsoft Research with performance and security perspectives [6]. In this work, we have selected a new elliptic curve ED_{25519} at the 128-bit security level and shown

1. M. Das (ORCID: 0000-0002-0259-7440) is with Department of Information Systems, Madda Walabu University, Ethiopia and with School of Computing, University of South Africa, Florida, South Africa. Email: mausamdas2010@gmail.com
2. Z. Wang (ORCID: 0000-0003-3025-336X) (Corresponding Author) is with Department of Electrical Engineering, University of South Africa, Florida, South Africa. Email: wangz@unisa.ac.za

field-calculation through a radix-2^4 that demonstrates the quantities with MoTE-ECC over Optimal Prime Fields (OPFs) of variable size; e.g. 160, 192, 224 and 225 bits. We take ECDH (Elliptic-curve Diffie–Hellman) key interchange among two nodes. Our implementation shows that an ECDH takes much less energy consumption over 160-bit OPF and we compare it to a Weierstraß curve with regard to ECDLP, energy consumption and "ECC security" [16]. Based on our experiment and calculation, we can say that our selected curve may perform better in wireless networks. Our implementation result takes less energy consumption over 160-bit OPF. The remainder of the paper is organized as follows: in Section 2, we explain Elliptic Curve including ED_{25519} and related work. Section 3 introduces motivation, Section 4 shows the methodology and Section 5 provides implementations. Section 6 shows fixed-base comb method for point multiplication. Section 7 performs security analysis and Section 8 exhibits execution time. Section 9 exemplifies energy consumption and performance and Section 10 represents the conclusion.

2. ELLIPTIC CURVE AND RELATED WORK

2.1 Elliptic Curve

According to Euler and Gauss entirety, Edwards popularized ordinary form of elliptic curve in 2007 [2]. The curve is explained as:

$$y^2 + x^2 = a^2(1 + x^2y^2) \tag{1}$$

over the field K, where $a \in K$, such that: $a^5 \neq a$. As Edwards declared in his paper, each curve of the form given in (1) is bi-rationally identical to an elliptic curve in Weierstraß [3]. Because of an established field K of different distinctive and erratic integers c, $d \in K$ so that $cd(1 - dc^4) \neq 0$, the curves are popularized as:

$$y^2 + x^2 = c^2(1 + dx^2y^2) \tag{2}$$

The aforementioned explanation covers higher than $1 = 4$ of entire isomorphism classes of elliptic curves over a restricted field. It is illustrated that each elliptic curve on a non-binary field is birationally equivalent to a curve in Edwards structure over an expansion of the field and in several facts over the innovative field [4]. In [6], Bernstein et al. established a simplification of Edwards curves called twisted Edwards curves. These combine elliptic curve in Montgomery form [10]. As interpreted in [6], the set of twisted Edwards curves is constant under quadratic flourish, whereby a quadratic twist of an Edwards curve is not naturally an Edwards curve. A quadratic flourish of a curve is an isomorphic curve on a field expansion of scale two. In a field K of different distinctive and non-zero components a, $d \in K$, the twisted Edwards curve $E_{T,a,d}(K)$ is described as:

$$E_{T,a,d}(K): ax^2 + y^2 = 1 + dx^2y^2 \tag{3}$$

If $a = 1$, then $E_{T,a,d}$ is an Edwards curve with $c = 1$. Moreover, $E_{T,a,d}$ is a quadratic flourish of the Edwards curve $E_{\mathcal{O},1,d/a} = a$ with the map: $(\bar{x}, \bar{y}) \rightarrow (x, y) = (\frac{\bar{x}}{\sqrt{a}}, y)$ over the field expansion $K(\sqrt{a})$:

$$\bar{x}^2 + \bar{y}^2 = 1 + (d = a)\bar{x}^2\bar{y}^2 \tag{4}$$

Twisted Edwards curves and Montgomery curves are intently relevant. As explained in [4], each twisted Edwards curve $E_{T,a,d}$ on the ground K with char$(K) \neq 2$ is birationally identical to a Montgomery curve $E_{M,A,B} : Bv^2 = u^3 + Au^2 + u$ using the map:

$$(x, y) \rightarrow (u, v) = (\frac{(1+y)}{(1-y)'}\frac{(1+y)}{(1-y)}) \tag{5}$$

where $A = \frac{(a+d)}{(a-d)'}$ and $B = \frac{4}{(a-d)}$

Whether a is a square in K, therefore such curves are isomorphic through K itself. Since the function enumerates of the point computation in [11], it is simple to watch that twisted Edwards curves surpass curves in Weierstraß shape in fast condition (although the binary form of Edwards curve is a bit-delay from compared with its Weierstraß equivalent [10]). Twisted Edwards curve's cluster rules are standardized and perfect; that carries to secure fulfilments over specific kinds of offensives [4]. The best relevant implementation of twisted Edwards curves is Edwards-curve Digital Signature Algorithm (EdDSA). The ED_{25519} is a twisted Edwards curve utilized for EdDSA, elsewhere particular parameters are determined like [12]: $a = -1,\ d = \frac{121665}{121666'},\ p = 2^{255} - 19.$

The respective Montgomery curve of ED_{25519} is Curve25519 that is specified as [13]:

$$y^2 = x^3 + 486662x^2 + x \quad (6)$$

Point propagation is speedy and capable upon Montgomery curves. This successfully utilizes unlike point supplement and point folding [11] as well as regular Montgomery ladder computation to execute a point addition[14]. The constant Montgomery ladder algorithm [40] is executed in fixed rate, which leads to timing attack. Various activities have utilized Curve25519 since its presentation by Bernstein in 2006 [9]. Moreover, because of its 128-bit security stage and effective computation [44], it has also an optimistic application for Internet of Things (IoT) demand. Currently, an amount of hardware fulfilments have been established [15]-[19] over a concentrate on IoT demand. All these functions use FPGA (Field-Programmable Gate Array) DSP (Digital Signal Processing) segments to execute modular factors. High-efficiency cryptographic converters that may be initiated on affordable FPGAs or ASICs (Application Specific Integrated Circuits) are in order in favour of mobile uses similar to the Internet of Things (IoT) and Intelligent Transport Systems (ITSs) [14]. Affordable FPGAs (containing negative-consolidated-established FPGAs) are specifically limited in several hardware funds. Utilization of hardware assets will minimize with movable low energy without wasting achievement [43]. In consequence, we propose a zone-competent, low-energy hardware execution of the ED_{25519} on FPGA. The DSP parts of FPGA assets will not utilize our exploit. We establish a great race interlace modular factor adjusted in favour of such implementation.

2.2 Related Work

Security: Studies specifically associated to the security of ECDSA P-256 and ED_{25519} have been performed before. Nystrom [21] observed through an examination of an initial RFC8080 (Request for Comments) design with no citation or confirmation that ED_{25519} would provide enhanced security assets and enforcement features comparable to RSA and ECDSA algorithims, causing such declaration to be eliminated through RFC8080. However, there are motivations to trust that Ed25519 gives better security compared with ECDSA P-256; e.g. while monitoring Lange and Bernstein's security roster in favour of elliptic curves in comprehensive called security curves [16], it is observed that Weierstraß is examined to be unreliable, while ED_{25519} (which is related to Curve25519 [38]) is considered to be safe. An ECDSA P-256 particular assault has been outlined as well. Brumley et al. [22] discovered that such ECDSA P-256 in the newest form of OpenSSL 1.0.1 (which is OpenSSL 1.0.1u) is exposed to reserve-schedule attacks, permitting themselves to restore the individual for TLS and SSH. That might be appropriate in favour of DNSSEC, whereas DNSSEC package may trust OpenSSL, considering that enforcements of ED_{25519} might be protected against reserve-schedule attacks[23]. The importance of these security inconvenience and achievable alleviates is evaluated in this work.

3. MOTIVATION

In this study, we choose ED_{25519} curve and its extended twisted Edwards coordinates at the 128-bit security level. A famous Weierstraß elliptic curve is presently obtained, though it contains a few disadvantages due to that we select another curve ED_{25519}. Additionally, we focus on high-speed signature validity, achieving SPA (Simple Power Analysis) attack and high-speed scaler computations. Security has become a major concern due to high benefit through IoT equipment which we are using in our daily life. Particular types of IoT equipment are source-compelled; in favour of particular cases, these contain less storage capacity as well as lengthy battery life. Due to this, encoded algorithms such as ECC are appropriate here.

3.1 Some Drawbacks of Weierstraß Curves

Weierstraß mathematical expression is $y^2 + ax + b$ over $\mathbb{F}_p$. The straightforward calculation in Weierstraß curve is hard. Magma supplies small methods in favour of calculation on elliptic curves shown in small as well as in large (difficult) Weierstraß designs [15]. The circumstance is much intricate in the constant case: the majority of quality algorithms may effectively move into exceptions. Weierstraß curve's quantifiable characteristics place into 3 and twisted Edwards curve's quantifiable characteristics place into -1. Two alternative curves are chosen in a fixed way and provide twist-security; this characteristic is benefited from in some works. Clock period of Weierstraß curve's point supplement as well as point multiplication are high over 160-bit OPFs (Optimal Prime Fields). This type of curves is

of less value to assist Twisted Edwards curves over prime fields and might be combined with past equipment by exchanging the curve's variables as well as the field computation.

3.2 Problems in Existing WSN Encryption System

Our current task is relevant to high-speed ECDH key shares and less energy consumption. Attackers may estimate the existing record from encoded media [24]. The problem that we have recognized is that the effective use of PKC(Public-Key Cryptography) is the serious restrict assets of cell-voltage sensor nodes. Also, ECDH key exchange energy ingest is high [41]. For illustration, the predominant MICAz mote from (the ATmega128 [25]) as well as distinctive attributes are 4 KB of RAM and 128 KB flash storage area. Tiny ECC [26] is the presently installed ECC software package for WSNs; this is a favourable arrangement with several times observations to establish famous curves across 160- and 192-bit primary grounds. However, we have noticed in [27] that ECC on compelled equipment is not a self-loading occurrence; for this reason, the advanced function currently does not fulfil the majority of the software. So, the effective performance of ECC on sensor node is still a demandable research matter and new methods are necessary to enhance the performance rate (i.e., energy cost) as well as memory.

3.3 Suitable Encryption Scheme for WSNs

Encoding and decoding expressions are normally asset-vigorous security techniques [42], but wireless equipment is attribute-restricted. Therefore, thin-security algorithms are considered relatively less material- absorbing. In this statement, measured to other non-symmetrical key algorithms, ECC is a more desirable act, because the size of key length is very small as well as it needs a smaller amount of power absorption. Another important thing is that detector node equipment is generally a fixed mechanism which is composed of technique-on-chip, micro devices, memory chips, energy-control ICs and additional similar types of chips. In favour of security-specified jobs, varieties of ICs are deployed in this scheme. Whether a particular concern security plan is preferred for wireless sensor node equipment, for a particular case where ECC arrangement is in favour of key shares, we decide another possibility to diminish energy consumption. Whether only ECC is deployed for together key shares and encoding functions, then the complete security process might be applied through a restricted amount of chips to reduce the quantity of gates over circuits and diminish energy absorption. Through this experiment, we have seen high-speed ECDH key shares with less energy consumption.

4. METHODOLOGY

We carefully choose ED_{25519} curve in relation to EdDSA and extended coordinates aspect. Particularly, this curve's expression is $E: \{(x, y) \in F_q \times F_q : -x^2 + y^2 = 1 + dx^2y^2\}$. ED_{25519}-SHA (Secure Hash Algorithm)-512 is EdDSA accompanied by these arguments: $b = 256$; H is $SHA - 512$; q is the prime $2^{255} - 19$; the 255-bit encrypting of $F_{2^{255}-19}$ is the regular small-endian encoding of $\{0,1\ldots, 2^{255} - 20\}$; ℓ is the prime $2^{252} + 27742317777372353535851937790883648493$ from [28], $d = -121665/121666 \in F_q$; as well as B is the special point $(x, 4/5) \in E$ for which x is optimistic. This Edwards curve is corresponding to $-x^2 + y^2 = 1 - \left(\frac{121665}{121666}\right)x^2y^2$, because -1 is multiplied in F_q. Additionally, the security of ED_{25519}-SHA-512 is not harmed, because r is not able to be seen by the attacker. The most vital job in favour of elliptic prime curve procreation is to select a prime number. Brainpool curves achieve pseudo-random prime digits to produce the prime curves, but due to lack of capabilities, such kind of curve is not as good as Edwards curves. Weierstraß curves make progress on random prime branches. To enhance performance, TinyECC or MICAz restrain better situations in favour of proficiency 128-, 160- and 192-bit fields. We have deployed coordinates-twisted Edwards curve and have selected the additional quantifiable characteristics of the curve in a particular way. In favour of field-computing process, we deploy a radix-2^4 model in relation to MoTE-ECC adjacent to Optimal Prime Fields (OPFs). In MoTE-ECC, RAM size in favour of 256-bit OPFs is 556 bytes and for 160-bit OPFs, RAM size is 380 bytes, which is smaller than in AVR-based sensor nodes. Our selected curve's parameters and other details are explained in the next parts.

4.1 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Elliptic curve discrete logarithm problem (ECDLP) specifies the security of ED_{25519} and Curve25519.

Assume E is the elliptic curve stated on the prime field $\mathbb{F}_p$ and assume the combination of logical dots over curve E indicated through $E(\mathbb{F}_p)$. At the present time, consider a point $P \in 2E(\mathbb{F}_p)$ of order n as well as the closed ring subgroup of $E(\mathbb{F}_p)$ produced by point $\langle P\rangle = \{\mathcal{O}, P, 2P, \ldots, (n-1)P\}$. Acquire a random number $k \in [1, n-1]$ and let $Q = k \cdot P$, where the point Q is stated through joining spot P to emphasize $k-1$ times.

$$Q = k.P = \underbrace{\{P + P \ldots + P.\}}_{k\ times} \quad (7)$$

Specifying particular parameters of an area Q, the problem of establishing a particular number k is specified (ECDLP) [15]. The dot Q is able to be quickly calculated in relation to k deploying particular identical-aspect task $Q = k \cdot P$ (declared elliptic curve point doubling or scalar multiplication). However, to determine analytically k from recognized points Q and P is absolutely complex.

4.2 Parameters of Ed25519

Parameter	Value
p	p of ED_{25519} in [RFC7748] (i.e., 2^{255}-19)
b	256
encoding of	255-bit little-endian encoding of $\{0,1,\ldots,p-1\}$
H(x)	SHA-512(dom2(phflag, context)‖ x) [RFC6234]
c	base 2 logarithm of cofactor of ED_{25519} (i.e., 3)
n	254
d	-121665/121666=37095705934669439343138083508754565189542113898432190163887855330859402835555
a	-1
b	(X(P),Y(P)) of ED_{25519} in [RFC7748] (i.e., 15112221349535400772501151409588531511454012693041857206046113283949847762202, 46316835694926478169428394003475163141307993866256225615783033603165251855960))
L	Order of ED_{25519} in [RFC7748] i.e., 2^252+27742317777372353535851937790883648493).
PH(x)	x (i.e., the identity function)

4.3 Edwards-curve Digital Signature Algorithm (EdDSA)

EdDSA is a digital authorization arrangement. ED_{25519} understands EdDSA authorization.

Algorithm 1. EdDSA key establishment as well as authorization generation

Key setup.
1: Hash k such that $H(k) = (h_0, h_1, \ldots, h_{2b-1}) = (a, b)$
2: $a = (h_0, \ldots, h_{b-1})$ perform with integer in little-endian symbols
3: $b = (h_b, \ldots, h_{2b-1})$
4: Compute public key: $A = aB$
Signature generation.
5. Compute ephemeral private key: $r = H(b, M)$.
6. Compute ephemeral public key: $R = rB$.
7. Compute $h = H(R, A, M)$ and convert to integer.
8. Compute: $S = (r + ha) \bmod \ell$.
9. Signature pair: (R, S).

Establishing the key first four sequences is deployed and implemented through a private key. Element $(x, \ldots, y)$ indicates addition of the constituent part. We specified an individual scalar and $b = (h_0, h_1, \ldots, h_{2b-1})$ the auxiliary key. Particular ephemeral key r is established in Step 5 . To justify an authorization (R, S) over a message M accompanied by public key A, a justifier observes the method explained in Algorithm 2. ECDSA acts this way: it substitutes F_q^* accompanied by an order$-\ell$ subdivision of an elliptic-curve set in contrast with F_q and describes $x(R)$ even though $x -$ is the element of R. ECDSA in addition to substituting A accompanied by $-A$, exchanges the authorizer's

Algorithm 2. EdDSA signature establishment

1: Compute $h = H(R, A, M)$ and convert it into an integer.
2: Check if the group equation $8SB = 8R + 8hA$ in E holds.
3: If the group equation holds, the signature is correct.

calculation directed to sum and acquires the establishment of equation $H(M)B + x(R)A = SR$. ECDSA substitutes specifically three-scalar mathematical expressions accompanied by the function of two-scalar mathematical expressions $S^{-1}H(M)B + S^{1}x(R)A = R$ at the cost of needing S to be altered modulo ℓ.

The $pmuldq/pmuludq$ directions accomplish two quantities of 32-bit numbers, manufacturing 64-bit output in each sequence. The $pmuldq$ direction is explained in [23].

4.4 Elliptic-curve Diffie–Hellman (ECDH)

A direction by Hisil [29] of ECDH on an Edwards curve is accompanied by identical security characteristics to Curve25519. Lin and Scott [30] of ECDH conducted an investigation on an Edwards curve in addition to an endomorphism. Bernstein's Curve25519 programme was used in favour of Diffie-Hellman key shares.

Such curve is described by $\frac{\varepsilon}{F_p}: y^2 = x(x^2 + 48662_x + 1) where\ p = 2^{255} - 19$.

We identify $\#\varepsilon(F_p) = 8r$ and $\#\varepsilon'(F_p) = 4r'$, in which r as well as r' are 253-bit primes. Recent performance of this system uses x-coordinate on a Montgomery display of the curve, for together mathematical reduction, side-channel security and little effort to establish.

4.5 Extended Twisted Edwards Coordinates

The important mathematical relationship in favour of point calculation on twisted Edwards curves was nominated by Hisil et al [29], constituting points in the utmost twisted Edwards coordinates: a point $P = (x, y)$ is appointed through the quadruple $(X: Y: T: Z)$; for example $x = X/Z$, $y = Y/Z$, $xy = T/Z$ and $Z \neq 0$. The additional coordinate T homogeneous coordinates were derived $(X: Y: Z)$ in relation to the multiplication of x as well as y, with a characteristic $T = XY/Z$. The group affinity element is demonstrated through $(0: 1: 0: 1)$, undesirable of an element $(X: Y: T: Z)$ which is $(-X: Y: -T: Z)$. A point in affine elements (x, y) be able to change into extended twisted Edwards coordinates by $X = x$, $Y = y$, $T = xy$ as well as $Z = 1$. To change rear to affine, T is disregarded, in addition to an inversion and two multiplications are needed: $x = X/Z$ and $y = Y/Z$. Likewise, it is possible to convert a point into identical projective coordinates $(X: Y: Z)$ merely *via* dumping T. Hisil et al projected an extensive coordinate scheme to facilitate a supplementary coordinate $t = xy$ [29]. As a substitute of signifying a point over twisted Edwards curve E_T through association with x and y coordinate solitary, we can utilize the extended affine coordinates (x, y, t). The consequent developed coordinates of that point are $(X: Y: T: Z)$, by which the supplementary coordinate T has the assets $T = XY/Z$ through $Z \neq 0$. In appreciation of these coordinates, Hisil et al. invented the proficient point addition method, particularly under the constraint $a = -1$. Behind implementation of clear-cut resources [32], the mathematical calculation rate of an assorted point addition over a curve is done through $a = -1$ total to $7M + 6A$, whereas a doubling needs $3M + 4S + 6A$.

Algorithm 3. Point multiplication in assorted homogeneous and extended twisted Edwards coordinates

Input: $\boldsymbol{P_1 = (X_1; Y_1; Z_1)}$ **in homogeneous projective coordinates.**
Output: $\boldsymbol{P_3 = 2P_1 = (X_3; Y_3; T_3; Z_3)}$ **in extended twisted Edwards coordinates.**
1: $A \leftarrow X_1^2;\ B \leftarrow Y_1^2;\ C \leftarrow 2Z_1^2$
2: $D \leftarrow -A;\ E \leftarrow (X_1 + Y_1)^2 - A - B;\ G \leftarrow D + B$
3: $F \leftarrow G - C;\ H \leftarrow D - B;\ X_3 \leftarrow E.F$
4: $Y3 \leftarrow G.H;\ T_3 \leftarrow E.H;\ Z_3 \leftarrow F.G$

Algorithm 4. Point addition in extended twisted Edwards coordinates

Input: $\boldsymbol{P1 = (X_1, Y_1, T_1, Z_1)}$ **and** $\boldsymbol{P_2 = (X_2, Y_2, T_2, Z_2)}$ **in extended twisted Edwards coordinates; constant** $\boldsymbol{k = -2d}$**, where** $\boldsymbol{d = -121665/121666}$**.**
Output: $\boldsymbol{P_3 = (X_3, Y_3, T_3, Z_3)}$ **in extended twisted Edwards coordinates.**
1: $A \leftarrow (Y_1 - X_1).(Y_2 - X_2);\ B \leftarrow (Y_1 + X_1).(Y_2 + X_2);\ C \leftarrow k.T1.T_2;$
2: $D \leftarrow 2Z_1Z_2;\ E \leftarrow B - A;\ F \leftarrow D - C;$
3: $G \leftarrow D + C;\ H \leftarrow B + A;\ X_3 \leftarrow E.F;$
4: $Y3 \leftarrow G.H;\ T_3 \leftarrow E.H;\ Z_3 \leftarrow F.G;$

4.6 Projective Coordinate Randomization

We put the arbitrary projective coordinates countermeasure to the extended twisted Edward coordinates $(X{:}\,Y{:}\,T{:}\,Z)$ in Joye's Double-Add, Goundar's Signed-digit and FLS (Fuzzy Logic System) algorithms [33]. In Joye's Double-Add and Goundar's Signed-digit algorithms, we arbitrarily produce $\lambda \in F_p \backslash 0$ and execute $X' \leftarrow \lambda x,\ Y' \leftarrow \lambda y,\ T' \leftarrow xY'\ Z \leftarrow \lambda$, where $P = (x; y)$ is the enter position in affine organization and particular consequential point $P' = (X'{:}\,Y', T', Z')$ is worn in position of P within particular rest of algorithms. During FLS algorithm [31], we indiscriminate the coordinates for initial point which is filled from the chart of previous calculated points, $P_0 = (X{:}\,Y{:}\,T{:}\,Z)$, as pursue: produce arbitrary $\lambda \in F_p \backslash 0$ and do $X' \leftarrow \lambda x,\ Y' \leftarrow \lambda Y,\ T' \leftarrow \lambda$ T and $Z' \leftarrow \lambda Z$. Particular consequential point $P'_0 = (X'{:}\,Y'{:}\,T'{:}\,Z')$ is adopted in position of P_0. While this countermeasure is implemented, the ideals of the coordinates of the accumulator point Q are randomized, altering from single implementation of the scalar multiplication to the additional.

Since particular significance of P'_0 is allocated to Q in the foundation of assessment phase, the extended twisted Edwards coordinates of every point $P_i j$ are governed to accumulate in the table, by means of a random λ produced in favour of every point, similar to how P_0 was governed. The authorization generation utilizes the FLS algorithm through $(v = 1; w = 4)$ (8 points, 1 chart) with an exclusive table search for safeguarded and governed coordinates countermeasures. The accomplishment effect of EdDSA-ED_{25519}-SHA512 progress particular state of the art capabilities [34] needs 19047706 sequences for authorizing; an enhancement of 17.9% and 30776942 cycles in favour of authentication; an enhancement of 5.7%. The transparency of the table search security (action taken for threat) as well as governed projective coordinates to particular FLS algorithm is merely 1.0%. Likewise, while these actions are implemented over the signature creation role, transparency is too little (0.9%). To calculate shared secret utility, overhead of the coordinate randomization is only 0.04%. Particular point replication algorithm (Algorithm 2) stands on the enthusiastic doubling principle, most effective in favour of $a = -1$ (the case for ED_{25519}) and charges $4M + 4S$. We examine stand on Faz-Hernandez et al.'s customized LSB (Lower Side Band)-set comb algorithm, called FLS [33].

4.7 Prime Field $\mathbb{F}_2^{255-19}$

A constituent of $\mathbb{F}_p$ is an integer modulo 2^{256-38} through field process. This surplus illustration favourably permits additional proficient decline than sinking straight modulo p. Simply, at the last part of scalar multiplication computation, where an integer is not previously existent in $\mathbb{F}_p$, we deduct p in steady time.

Table 1. Standard outcome on ATmega328P.

Operation Class	Operation/Algorithm	Cycles
Fixed-base ECSM ED_{25519}	FLS(v=1,w=3)	21 553 188
	FLS(v=2,w=4)	26 661 293
	FLS(v=1,w=3) lookup prot.	21 658 857
	FLS(v=1,w=4)	18 119 234
	FLS(v=2,w=3)	19 170 150
	FLS(v=1,w=4)lookup prot.	18 264 710
	FLS(v=2,w=3) lookup+rand. coord.	18298387

4.8 Field Multiplication and Squaring

Field squaring is executed as a 3-level subtractive Karatsuba, in which there is no provisional exclusion of M. The 32-bit multiplier from the multiplication is reprocessed here, along with a function name, at the foundation level.

4.9 Field Inversion

We utilize Fermat's theorem, $x^{-1} \equiv x^{p-2} (mod\ \ p)$, to calculate inversion in $\mathbb{F}_p$ in steady time. Addition sequence is composing of 254 squares at 11 multiplications, although we diminish the amount of provisional field variables; those needed are 10 to just 5.

4.10 Arithmetic Modulo Ed25519 Group Order

EdDSA ED_{25519} authorization method needs accumulation as well as multiplication modulo ED_{25519} cluster order (N). We execute decline modulo N in C utilizing a stable rate edition of the Barret algorithm achieved through unfolding the ultimate subtraction circle into pair facsimiles of its body (the highest digit of computational process) as well as utilizing provisional shifts executed in steady time. We computed the mutual of the modulus $R = \lfloor b^{2n}/N \rfloor = \lfloor 256^{64}/N \rfloor$, where a constraint of the Barret algorithm [15] is accumulated into program memory. The development identifies the 256-bit multiplier and afterword diminishes completely. The addition is also reduced as well as executed in assembly.

4.11 Optimal Prime Fields

The prime fields that we utilize in MoTE-ECC fit in a unique class of restricted fields recognized as Optimal Prime Fields (OPFs) [39]. The OPF library establishes a great extent of resilience as single and an identical role can process to operate every extent. An additional significant characteristic of the library is its flexibility over SPA attacks, since entire mathematics functions are applied in a normal manner and perform forever the identical series of guidelines, irrespective of the real worth of the operands. These grounds are described through primes and mentioned as $p = u.2^k + v$ by which u as well as v are a little contrasted to 2^k; henceforth they robust into single or double registers for targeted podium. MoTE-ECC sustains OPFs through $2^{15} \leq u < 2^{16}$ (i.e. u is 16 bits lengthy) as well as $v = 1$. An actual illustration is $p = 65356.2^{144} + 1$ (i.e., $u = 65356$ and $k = 144$), which occurs through a 160-bit prime and hex notation is as pursued. $p = 0xFF4C0000000000000000000000000000000000001$. Low hamming weight characterized these prime structures, while only extremely important bytes as well as slightest-importance bytes are non-zero; while other bytes are zero. The small power of p permits favourably precise resources of modular calculation, since just non-zero bytes of p require to be progressed in the decline function. We apply the inversion in OPFs founded on Fermat's little theorem $a^{p-2} \equiv a^{-1} \bmod p$, the straight idea of the appearance $u.2^k - 1$, needing n squarings as well as n multiplications, by which n indicates the bit-span of p. To diminish the bit-span, we choose an algorithm in favour of effective enhancement-supported inversion with regard to OPFs.

In appreciation about such an algorithm, it is feasible to reduce the entire number of functions to n squaring addition just $HW(k) + HW(u-1) + 1$ multiplications, whereby $HW(x)$ implies the hamming weight for x. During the initial period, $a^{2^{0.6ex}}$k-1 is measured through the exponentiation technique. During the next stage, a right-to-left square-and-multiply algorithm is conducted.

Algorithm 5. Optimized exponentiation-based inversion for OPFs

Input: Element a of $\mathbb{F}_p$ with $p = u.2^k + 1$.**Output:** $r \equiv a^{u.2^k-1} \equiv a^{-1} \bmod p$.

1: $u' \leftarrow u - 1$
2: $r \leftarrow a$, $b \leftarrow \lceil ld(k) - 2 \rceil$, $i \leftarrow 1$
3: while $b > 0$ do
4: $t \leftarrow r^2$
5: for $j = 1$ to $i - 1$ do
6: $t \leftarrow t^2$
7: end for
8: $r \leftarrow r.t$
9: $i \leftarrow i \ll 1$
10: if $k\&b > 0$ then
11: $r \leftarrow r^2.a$
12: $i \leftarrow i + 1$
13: end if
14: $b \leftarrow b \gg 1$
15: end while
16: $t \leftarrow r.a$
17: $b \leftarrow 1$
18: while $b < 0x8000$ do
19: if $u'\&b > 0$ then
20: $r \leftarrow r.t$
21: end if
22: $t \leftarrow t^2$
23: $b \leftarrow b \ll 1$
24: end while
25: if $u'\&b > 0$ then
26: $r \leftarrow r.t$
27: end if

5. IMPLEMENTATIONS

A radix-2^r illustration a component f within b-bit prime area like $(f0, f1, \ldots, f_{\lceil b/r \rceil - 1})$ is given as:

$$f = \sum_{i=0}^{\lceil \frac{b}{r} \rceil - 1} fi2^{\lceil ir \rceil}$$

This is termed a radix-2^r illustration. We utilize radix-2^4, indication in favour of field components. We diminish intermediary consequences modulo $2^{256} - 38$ within the complete implementation of the scalar propagation and merely diminish the ultimate result modulo $2^{255} - 19$. We completed n- bit quantities, an entire of $\lceil n/2 \rceil$ limited results is produced, where the outcome of extreme altitude for limited output array is $\lceil n/2 \rceil + 1$ components to be combined. A radix-2^4 numeral is ullustrated through numbers, since the particular set $D = \{0,1,2\ldots,14,15\}$ within an identical radix-2^4 illustration utilizing a zero-discharge number set in the shape of $D' = \{\pm 1, \pm 3, \ldots, \pm 13, \pm 15\}$. $\mathbb{F}_p$ implies an OPF established through a prime structure $p = u \cdot 2^k + 1$; therefore, u is within the order $[2^{15}, 2^{16} - 1]$; i.e., u contains extent of 16 bits. Despite the aforementioned topic, the bit range n for primes is a product of 32; e.g. n = 160, 192, 224 or 256 bits. Field components are mentioned as $a \in \mathbb{F}_p$. The proper partitioning application is selected to the equilibrium of the digit of odd as well as uniform directions, diminishing the entire digits of the needed round. We track investigation: odd n-bit numeral k specified by $k = \sum_{i=0}^{n-1} k_i \cdot 2^i$ through $k_i \in \{0,1\}$ for $0 < i < n - 1$ and $k_{n-1} = k_0 = 1$ may be mentioned as conventional Binary Signed-Digit(BSD), since $k = 2^{n-1} + \sum_{i=0}^{n-2} (2k_{i+1} - 1) \cdot 2^i$; i.e., entire numbers of BSD illustration of k are non-null. For confirmation, we monitor:

$$\begin{aligned} k &= 2^{n-1} + \textstyle\sum_{i=0}^{n-2} (2k_{i+1} - 1) \cdot 2^i = 2^{n-1} - \sum_{i=0}^{n-2} 2^i + \sum_{i=0}^{n-2} 2k_{i+1} \cdot 2^i \\ &= 1 + \textstyle\sum_{i=0}^{n-2} k_{i+1} \cdot 2^{i+1} = 1 + \sum_{i=0}^{n-1} k_i \cdot 2^i = \sum_{i=0}^{n-1} k_i \cdot 2^i \ \ with \ \ k_0 = 1 \end{aligned} \tag{8}$$

This formula is utilized to change an odd numeral provided in regular binary shape into a BSD illustration including merely non-zero numerals; specifically -1 and 1.

6. FIXED-BASE COMB METHOD FOR POINT MULTIPLICATION

We transfer the entire binary illustration of k single bit as appropriate and introduce a "1" within empty MSB (Most- Significant Bit) place. Presently, this transferred bit-series is accurately transferred in the structure of BSD of k to clarify all zero bits as -1. Radix-2^4 illustration may be acquired by splitting the bit-series within a cluster of 4-bit numerals, every single one communicating with an odd digit in the area $[-15, 15]$. This way, w indicates the digit of bits (i.e., size of bit-series) treated in every replication of the curve and $d = \lceil n/w \rceil$. Our alternative composes an off-heritage moment (Step 1) as well as an online stage. During the beginning stage, 2^{w-1} items are computed in advance and accumulated, including all straight associations of P. Our execution computes in advance eight points as we utilize $w = 4$ to obtain an equalization among implementation times as well as accumulator demands. A demonstration of the structure $(2a_i - 1)$ in stage 1 output is one way 1 (while $a_i = 1$) or the other -1 (if $a_i = 0$), therefore implementing the numeral-set alteration explained previously. In every recurrence, the online form includes a basic curve that performs duality followed through summation. Anyhow, compared to established comb technique, $w - 1$ bits (in place of w bits) through k are utilized to establish a certain kind of 2^{w-1} advance calculated points, which are to be added, when an additional bit (specifically $K_{(w-1)d+i}$ within stage 5 of Algorithm 6) is specified, since such point is truly joined or deducted. To accomplish a normal implementation, we require an operation that is subject to the significance of a bit, allocating a spot R or the adverse of that spot (i.e., $-R$) to a target. The pessimistic matter of R in prolonged refined coordinates is $-R = (-x, y, -t)$. MoTE-ECC executes the revocation of a component $x \in \mathbb{F}_p$ confiding on the quality of a bit b as pursue. We compute $x' = p - x$ through deductions after we carry bit b to extract a cover m, that is likewise an all-1 byte (if $b = 1$), otherwise an $all - 0$ byte (if $b = 0$). Furthermore, a second mask is required: m' is the bit-smart supplement of m; i.e., m' is 0 when m is an all -1 byte and *vice versa*. Next, we assess $(x'_i \& m) | (x_i \& m')$ for total byte of x' as well as x (whereby & and | express a bit-smart and and or function). Particular field is whether $-x = p - x$ (if $b = 1$; i.e., the negation is truly performed) rather (if $b = 0$; i.e., no contradiction). Comb procedure follows $d - 1$ point addition and $d - 1$ multiplication of the real worth of scalar bits.

Algorithm 6. Regular w-bit comb method for fixed-base scalar multiplication

Input: $\boldsymbol{n - bit\ scalar\ k = (k_{n-1}, \ldots, k_1, k_0)_2\ with\ k_0 = 1, point\ P \in E(\mathbb{F}_p)}$.
Output: $\boldsymbol{Q = k \cdot P}$

1: Pre-compute $R[j] = R[a_{w-2}, \ldots, a_1, a_0] = 2^{dw}P + (2a_{w-2} - 1)2^{(d-1)w}P + \ldots + (2a_1 - 1)2^wP + (2a_0 - 1)P\ for\ all\ bit - strings\ j = (a_{w-2}, \ldots, a_1, a_0)\ of\ length\ w - 1$
2: $Q \leftarrow R[k_{dw}, \ldots k_{2d}, kd]$
3: $for\ i = d - 1\ down\ to\ 1\ do$
4: $Q \leftarrow 2Q$
5: $Q \leftarrow Q + (2k_{(w-1)d+i} - 1) \cdot R[k_{(w-2)d+i}, \ldots, k_{d+i}, k_i]$
6: end for

7. SECURITY ANALYSIS

MoTE-ECC acquires the prolonged coordinate procedure in favour of twisted Edwards curves. It is possible to convert each twisted Edwards curve into a Montgomery curve conversely. ECDH procedure is to utilize the ridiculous comparison among Montgomery and twisted Edwards curves. Every L-detector is pre-installed through a single individual secret key. Next to key setup, every couple of connected L-detectors has various mutual keys. Therefore, yielding L-detector does not influence the security of transmissions within different L-detector. The DH private key is merely calculated among two transmit stakeholders, After that, it is utilized in favour of its progressive communication. Scalar multiplication protocols generally include three instances: established base point (kG), while G is a determined point (generally subset creator) and k is a scalar; varying foundation point (kP), while P is a point which is not previously known. Suppose two detector nodes A and B to determine a mutual private key, while the group domain arguments (a, d, A, B, G, p) are concurred above. This way, a and d are the arguments of twisted Edwards curve E_T, when A and B distinguished to be bi-rationally identical to Montgomery curve E_M. G exists at a point of prime rule over E_T and p specifies the essential OPF. Single turn ECDH key sharing protocol can be split into three phases:

First; node A produces a private key d_A and produces the respective public key $Q = d_A \cdot G$. Such scalar multiplication is completed through twisted Edwards curve E_T utilizing originator G. Afterwards, node A transforms the point $Q = (x_q, y_q)$ into spot $M = (x_m, y_m)$ over the bi-rationally identical Montgomery curve E_M and dispatches x-coordinate x_m of M to node B. Node B executes the identical stairs through private key d_B and transmits particular x-coordinate to A.

Second; thereafter, x-coordinate is obtained by node A from B; it starts to measure the scalar multiplication $S = d_A \cdot M$(M includes merely an x coordinate) over the Montgomery curve E_M. Node B performs similarly through the x-coordinate, obtained from node A. Together node A as well as node B need to perform dual scalar multiplication to acquire the mutual private key $S = d_A \cdot d_B \cdot G$. Considering that the foundation point G is steady and previously aware, we utilize quick scalar multiplication through fixed-base comb procedure utilizing a window diameter $w = 4$ as well as eight points quantified in advance.

Third; ECDH key interchange is mainly established by the calculated energy W_c in favour of two scalar multiplications; the correspondence energy W_t is mainly insignificant.

Table 2. Execution time (clock cycle) of field arithmetic function for operands of a measurement of 160,192, 224 and 256 bits.

Operation	160 bits	192 bits	224 bits	256 bits
mod_add	530	631	732	833
mod_sub	530	631	732	833
mod_mul	3237	4500	5971	7650
mod_sqr	2901	3909	5058	6347
mod_inv	571916	830823	1163655	1491839

Executing a fixed-base scalar multiplication over twisted Edwards curve $E_T: -x^2 + y^2 = 1 - 121665/121666x^2y^2$ is in contrast to $\mathbb{F}_2^{255} - 19$ and the outcome is reversed compared to the Mongomery curve in terms of single inversion. The curve points are represented as $E_T(F_2^{255} - 19)$. A

proficiently quantifiable bi-rational correspondence exists between E_T and E_M, hence the curves exchange similar cluster framework. Twisted Edwards curve is ideal in favour of $a = -1$ (for ED_{25519}), the input point is truly illustrated in twisted Edward complements and point Q outcomes are based on quick scalar multiplication in extended projective coordinates. Alteration of point Q over a twisted Edwards curve E_T within a point M on the birationally-corresponding Montgomery curve E_M can be performed in this method. Initially, we alter the projective point $Q = (X_q, Y_q, T_q, Z_q)$ on E_T associated with affine illustration $Q = (x_q, y_q)$ and work out $M = (x_m, y_m)$ on E_M by the use of $x_m = (1 + y_q)/(1 - y_q)$ along with $y_m = (1 + y_q)/((1 - y_q) \cdot x_q)$. We scamper an inversion within affine-to-projective alteration to acquire $1/Z_q$ as well as other reversal for the element of Edwards-to-Montgomery alteration ($to\ obtain\ 1/[(1 - y_t) \cdot x_t]$). To diminish the computational transparency reasoned through two inversions, we straightforwardly alter the point $Q = (X_q, Y_q, Tq, Z_q)$ to the point $M = (x_m, y_m)$ as follows:

$$x_m = (1 + y_q)/(1 - y_q) = (1 + Y_q/Z_q)/(1 - Y_q/Z_q) = (Z_q + Y_q)/(Z_q - Y_q) \quad (9)$$

$$y_m = (1 + y_q)/(x_q \cdot (1 - y_q)) = (Z_q^2 + Y_q Z_q)/(X_q Z_q - X_q Y_q) \quad (10)$$

Now, we will obtain one inversion to calculate $1/(X_q Z_q - x_q Y_q)$, which is multiplied by X_q to get $1/(Z_q, -Y_q)$.

8. EXECUTION TIME

We applied the OPF inversion from scrape and utilized OPF documentation from additional arithmetic functions.

We utilize the role of ANSI C and establish the performance time of point addition and point multiplication on twisted Edwards curve. The point addition and point multiplication on twisted Edwards curve are quicker than on Weierstraß curve. The supremacy of scalar multiplication through the performance period of all cryptographic activities is clear. Regarding signature authentication performance on ED_{25519}, the implementation time rises dramatically while the marvellous characteristic is activated since this presents an additional multiplication. Particular arithmetic functions replicate the carry transmission sequence even without transmission. Particular regular-time implementation characteristic (not obligatory) makes stronger the execution in contrast to the aggressor's capability to utilize side-channels within the structure of executing timing assault.

Table 3. Implementation time (in clock cycles) of point arithmetic functions over 160-, 192-, 224- and 256-bit OPFs.

Operation	160 bits	192 bits	224 bits	256 bits
TE point add	27355	36903	47907	60367
TE point dbl	25421	33848	43463	54262
WEI Point add	40222	N/A	N/A	N/A
WEI Point dbl	31536	N/A	N/A	N/A

Table 4. Implementation time (in clock series) of scalar multiplication over 160-, 192-, 224- and 256-bit OPFs.

Operation	160 bits	192 bits	224 bits	256 bits
Scalar mul. TE curve	2767454	4412519	6603888	9420788
Scalar mul. WEI curve (R.Int)	7384579	N/A	N/A	N/A
ECDH	9044084	14377068	21460334	30539566

The ECDH protocol is implemented to $2.76 \cdot 10^6$ clock series over a 160-bit OPF, which contains Edwards-to-Montgomery alteration. MoTE-ECC utilizes Montgomery curve with an execution time of $6.27 \cdot 10^6$ cycles over 160-bit OPF. The complete computation cost of an ephemeral ECDH key exchange amounts to about $9.04 \cdot 10^6$ clock cycles while utilizing a 160-bit OPF, with an implementation time of 1.22 s at 7.37 MHz.

9. ENERGY CONSUMPTION AND PERFORMANCE

MoTE-ECC, the slightest ECC is executed in favour of Memsic's MICAz motes as well as other 8-bit AVR-established sensor nodes. Energy is the highest valuable source for battery-driven detector nodes. Thus, it is essential to maximize the achievement of ECC application due to the reason the energy utilization of scalar multiplication increases consecutively over the implementation time. Based on [37], the ATmega128 processor of a MICAz mote relies on a medium current of 8 mA (at a delivered voltage of 3.0 V) while it is operating. Due to the reason that the clock rate for a mote is familiar to be 7.3728 MHz, we may obtain the energy utilization of single scalar multiplication through the execution of a basic computation like $W = U \cdot I \cdot t$, thus U implies the delivered power (i.e., 3 V while utilizing two traditional 1.5 V AA power cells). I is the medium current worn through the processor (i.e., 8 mA in our case), and t is the performance time. In our execution, we get a medium implementation time of 2767454 clock cycles, substantiating the energy charges of computing an individual scalar multiplication figure to $W_c = U \cdot I \cdot t = 3v \cdot 8mA \cdot (2767454/7.3728 \cdot 10^6) = 9.008mJ$. ECDH key interchange needs every node to calculate two scalar multiplications as well as to transmit a message (including the public key) to another node. Based on the energy pattern explained in [35], the energy value of sending an agreement message is $W_t = P \cdot t = 0.185$ mJ. Thus, the entire energy expenditure of ECDH key interchange is mainly rigid by the computation energy W_c for two scalar multiplications; the correspondence energy W_t is basically insignificant. Complete energy utilization to achieve an ECDH key interchange is $W = 2 \cdot W_c + W_t = 18.20$ mJ for each node. Piotrowski et al. declare in [36] that the assessed capability of a 1.5 V AA alkaline power cell is around 2500 mAh and two AA batteries may technically release an energy of 21600 Ws. The node energized by two AA alkaline power cells utilizes just 31.25 % of the entire ability. ECDH key interchanges can execute prior to the delivered voltage of the MICAz mote falling below 2.7 V.

Table 5. Energy utilization of TE curve, WEI curve and ECDH over 160-, 192-, 224- and 256-bit OPFs.

Operation	160 bits	192 bits	224 bits	256 bits
TE Curve	9.00 mJ	14.36 mJ	21.49 mJ	30.66 mJ
WEI Curve	24.03 mJ	N/A	N/A	N/A
ECDH	18.20 mJ	28.91 mJ	43.17 mJ	61.51 mJ

Beyond achievement, execution-time memory utilization is a significant factor for WSN utilization, since a standard AVR-supported detector node characterizes merely 4 kB RAM. Our comb technique in favour of scalar multiplication on a twisted Edwards curve needs to reserve eight points provided in enlarged affine coordinates. Like that, we merely require to shift the point which is needed for the present replication of the comb technique from ROM or flash memory to RAM. Our selection of $w = 4$ with eight previously assessed points illustrates a fair trade-off between performance and code size. The total ROM/flash footprint of MoTE-ECC supporting Montgomery as well as twisted Edwards curves is 14.7 kB, which establishes about 11.5% of 128 kB flash storage which exists on an ordinary AVR-deployed sensor node.

9.1 Signature Verification

The speed of Edwards-curve summation, particularly through -1 twist, makes such methods especially proficient. The prime $q = 2^{255} - 19$ is identical to 5 modulo 8; therefore, every square $\alpha \in F_q$ fulfils $\alpha^2 = \beta^4$, whereby $\beta = \alpha^{(q+3)/8}$, i.e.,$\pm\alpha = \beta^2$. The regular assessment is an individual elaboration to measure β, pursued by a fast propagation of β by $\sqrt{-1}$ if $\beta^2 = -\alpha$. In expansion, α is a percentage u/v, while $u = y^2 - 1$ and $v = dy^2 + 1$. Beginning from u and v requires only some multiplication compared with single exponent.

9.2 Defeating SPA Attacks

Within a model, SPA attacks attempt first to identify the energy utilization of a series of commands performed on a tool identical to the aimed tool. A determined couple (key, data) is refined and replicates for such various pairs oppose a single track acquired from the objective (pattern identical stage). A greatly routine execution of the comb technique for permanent-base scalar multiplication is utilized to diminish the SPA-leakage.

9.3 Fast Scalar Multiplication

To obtain the finest speed, we presume that $(a = -1)$. An n-bit scalar multiplication contains of absolutely $d = \lceil n/w \rceil$ point doublings as well as effectively d point additions. Therefore, w-bit comb technique reduces the amount of point doublings by an element of w in contrast to the binary method in spite of that the respective w bits of k are all 0. We can estimate the value of $\varepsilon^e \leftarrow 2\varepsilon$ as $3M + 4S$ by pressing an additional multiplication to the function measure of $\varepsilon \leftarrow \varepsilon^e + \varepsilon^e$.

10. CONCLUSION

This work suggested a novel ECDH key exchange technique through little energy utilization. First, we select ED25519's extended coordinates through point addition as well as point-doubling algorithm. Second, we determine radix-2^4 in favour of optimal execution through a Fermat-established inversion that is powerful over SPA attacks. Third, we represent the achievement of ECDH key interchange by combining Montgomery as well as twisted Edwards curves. Fourth, we compute and illustrate energy exhaustion of TE Curve, ECDH and contrast it with WEI curve. MoTE-ECC in favour of Memsic's MICAz motes is utilized for quick ECDH key interchange on 160-bit OPF and RAM footmark of OPF 256 bits is 556 bytes. We obtained the implementation time by joining quick $\mathbb{F}_p$ mathematically (grateful to utilizing an OPF) with extremely competent group mathematical twisted Edwards curve. MoTE-ECC assists 160-bit OPF with merely 380 bytes in RAM. We applied multiplication as well as other arithmetic functions required for ECC in a framework model to obtain great expandability and little code size. In favour of additional effective field arithmetics, we utilize quick point addition/doubling equation of twisted Edwards curves. An ECDH key interchange needs just one-third of energy of the ECDH execution. In the future, we will extend our work through Montgomery and Edwards curves for secure monitoring of low-power wireless devices by leveraging the security modules, like ECDH and ECDSA, which will support more ECC security features and will perform better than the existing curves.

ACKNOWLEDGEMENTS

This research is partially supported by the South African National Research Foundation (Grant Nos. 137951 and 132797), South African National Research Foundation incentive grant (No. 114911), and South African Eskom Tertiary Education Support Programme.

REFERENCES

[1] Rehana, Jinat, "Security of Wireless Sensor Network," Seminar on Internetworking, [Online], Available: http://www.cse.tkk.fi/en/publications/B/5/papers/Rehana_final.pdf, 2009.

[2] D. J. Bernstein and T. Lange, "Faster Addition and Doubling on Elliptic Curves," Proc. of the International Conference on the Theory and Application of Cryptology and Information Security, pp. 29-50, Springer, Berlin, Heidelberg, December 2007.

[3] D. J. Bernstein, P. Birkner, M. Joye, T. Lange and C. Peters, "Twisted Edwards Curves," Proc. of the International Conference on Cryptology in Africa, pp. 389-405, Springer, Berlin, Heidelberg, June 2008.

[4] P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization," Mathematics of Computation, vol. 48, no. 177, pp. 243-264, 1987.

[5] A. Verri Lucca, G. A. Mariano Sborz, V. R. Quietinho Leithardt et al., "A Review of Techniques for Implementing Elliptic Curve Point Multiplication on Hardware," Journal of Sensor and Actuator Networks, vol. 10, no. 1, pp. 3-17, 2021.

[6] D. J. Bernstein, T. Lange and R. R. Farashahi, "Binary Edwards Curves," Proc. of the International Workshop on Cryptographic Hardware and Embedded Systems, pp.244-265, Springer, Berlin, Heidelberg, August 2008.

[7] O. Reyad, M. Karar and K. Hamed, "Random Bit Generator Mechanism Based on Elliptic Curves and Secure Hash Function," Proc. of the IEEE International Conference on Advances in the Emerging Computing Technologies (AECT), pp. 1-6, arViv:2002.09239, 2020.

[8] Brainpool, "ECC Brainpool Standard Curves and Curve Generation," v. 1.0, [Online], Available: https://www.teletrust.de/fileadmin/files/oid/oid_ECC-Brainpool-Standard-curves-V1.pdf, October 2005.

[9] D. J. Bernstein, "Curve25519: New Diffie-Hellman Speed Records," Proc. of the International Workshop on Public Key Cryptography, pp. 207-228, Springer, Berlin, Heidelberg, April 2006.

[10] P. Sasdrich and T. Güneysu, "Efficient Elliptic-curve Cryptography Using Curve25519 on Reconfigurable Devices," Proc. of the International Symposium on Applied Reconfigurable Computing, pp. 25-36, DOI:10.1007/978-3-319-05960-0_3, Springer, Cham, April 2014.

[11] P. Koppermann, F. De Santis, J. Heyszl and G. Sigl, "X25519 Hardware Implementation for Low-latency Applications," Proc. of the IEEE Euromicro Conference on Digital System Design (DSD), pp. 99-106, Limassol, Cyprus, August 2016.

[12] P. Koppermann, F. De Santis, J. Heyszl and G. Sigl, "Low-latency X25519 Hardware Implementation: Breaking the 100 Microseconds Barrier," Microprocessors and Microsystems, vol. 52, pp. 491-497, 2017.

[13] F. Turan and I. Verbauwhede, "Compact and Flexible FPGA Implementation of Ed25519 and X25519," ACM Transactions on Embedded Computing Systems (TECS), vol. 18, no. 3, pp. 1-21, 2019.

[14] T. Schütze, "Automotive Security: Cryptography for Car2X Communication," Proc. of Embedded World Conference, vol. 3, pp. 4-24, Nürnberg, Germany, March 2011.

[15] D. Hankerson, A. J. Menezes and S. Vanstone, Guide to Elliptic Curve Cryptography, ISBN: 978-0-387-21846-5, Springer Science & Business Media, 2006.

[16] D. J. Bernstein and T. Lange, "SafeCurves: Choosing Safe curves for Elliptic-curve Cryptography," [Online], available: https://cr.yp.to/talks/2014.01.18/slides-dan+tanja-20140118-a4.pdf, 9 April 2019.

[17] V. Bunimov and M. Schimmler, "Area and Time Efficient Modular Multiplication of Large Integers," Proc. of the IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP), pp. 400-409, The Hague, Netherlands, 2003.

[18] N. Takagi and S. Yajima, "Modular Multiplication Hardware Algorithms with a Redundant Representation and their Application to RSA Cryptosystem," IEEE Transactions on Computers, vol. 7, pp. 887-891, 1992.

[19] M. A. Nassar and L. A. El-Sayed, "Efficient Interleaved Modular Multiplication Based on Sign Detection," Proc. of the IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA), pp. 1-5, Marrakech, Morocco, 2015.

[20] M. Scott, "Backdoors in NIST Elliptic Curves," MIRACL, [Online], Available: https://miracl.com/blog/backdoors-in-nist-elliptic-curves/, 2013.

[21] M. Nystrom, "Last Call Review of draft-ietf-curdle-dnskey-eddsa-02," [Online], Available: https://datatracker.ietf.org/doc/review-ietf-curdle-dnskey-eddsa-02-secdir-lc-nystrom-2016-12-15/, 2016.

[22] C. P. García and B. B. Brumley, "Constant-time Callees with Variable-time Callers," Proc. of the 26th USENIX Security Symposium (USENIX Security 17), pp. 83-98, 2017.

[23] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe and B.-Y. Yang, "High-speed High-security Signatures," Journal of Cryptographic Engineering, vol. 2, pp. 77-89, 2012.

[24] P. Gupta and V. Shmatikov, "Security Analysis of Voice-over-IP Protocols," Proc. of the 20th IEEE Computer Security Foundations Symposium (CSF'07), pp. 49-63, Venice, Italy, July 2007.

[25] Atmel, "8-bit ARV Microcontroller with 128K Bytes In-System Programmable Flash: ATmega128, ATmega128L, Datasheet," [Online], Available: https://datasheet.ciiva.com/26814/atmega128l-8au-26814613.pdf, June 2008.

[26] A. Liu and P. Ning, "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks," Proc. of the 7th IEEE International Conference on Information Processing in Sensor Networks (IPSN 2008), IEEE Computer Society Press, pp. 245-256, St. Louis, MO, USA, 2008.

[27] P. Szczechowiak, L. B. Oliveira, M. Scott, M. Collier and R. Dahab, "NanoECC: Testing the Limits of Elliptic Curve Cryptography in Sensor Networks," Proc. of European Conference on Wireless Sensor Networks (EWSN 2008), Part of the Lecture Notes in Computer Science Book Series, vol. 4913, pp. 305-320, 2008.

[28] J. Großschädl, M. Hudler, M. Koschuch, M. Krüger and A. Szekely, "Smart Elliptic Curve Cryptography for Smart Dust," Proc. of the International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine 2010), vol. 74, pp. 623-634, Springer, Berlin, Heidelberg, 2010.

[29] H. Hisil, K. K.H. Wong, G. Carter and E. Dawson, "Twisted Edwards Curves Revisited," Proc. of the International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2008), vol. 5350, pp.326-343, Springer, Berlin, Heidelberg, 2008.

[30] S. D. Galbraith, , X. Lin and M. Scott, "Endomorphisms for Faster Elliptic Curve Cryptography on a Large Class of Curves," Proc. of Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 518-535, Springer, Berlin, Heidelberg, 2009.

[31] A. Faz-Hernández, P. Longa and A. H. Sánchez, "Efficient and Secure Algorithms for GLV-based Scalar Multiplication and Their Implementation on GLV-GLS Curves," Topics in Cryptology – CT-RSA 2014 Conf., pp.1-27, DOI:10.1007/978-3-319-04852-9_1, Springer, Cham, 2014.

[32] M. Hamburg, "Fast and Compact Elliptic-curve Cryptography," IACR Cryptology ePrint Archive: Report 2012/309, [Online], Available: https://ia.cr/2012/309, 2012.

[33] E. Nascimento, J. López and R. Dahab, "Efficient and Secure Elliptic Curve Cryptography for 8-bit AVR Microcontrollers," Proc. of the 5th International Conference on Security, Privacy, and Applied Cryptography Engineering (SPACE 2015), vol. 9354, pp. 289–309, Springer, Cham, October 2015.

[34] M. Hutter and P. Schwabe, "NaCl on 8-bit AVR Microcontrollers," Proc. of the International Conference on Cryptology in Africa, pp. 156-172, Springer, Berlin, Heidelberg, 2013.

[35] G. De Meulenaer, F. Gosset, F. X. Standaert and O. Pereira, "On the Energy Cost of Communication and Cryptography in Wireless Sensor Networks," Proc. of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, pp. 580-585, Avignon, France, October 2008.

[36] K. Piotrowski, P. Langendoerfer and S. Peter, "How Public Key Cryptography Influences Wireless Sensor Node Lifetime," Proc. of the 4th ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '06), pp. 169-176, DOI: 10.1145/1180345.1180366, October 2006.

[37] Crossbow Technology Inc., "MICAz Wireless Measurement System," Data Sheet, [Online], Available: http://courses.ece.ubc.ca/494/files/MICAz_Datasheet.pdf, April 2015.

[38] S. Ullah and R. Zahilah, "Curve25519 Based Lightweight End-to-End Encryption in Resource Constrained Autonomous 8-bit IoT Devices," Cybersecurity, vol. 4, no. 1, pp. 1-13, 2021.

[39] Z. Liu, E. Wenger and J. Großschädl, "MoTE-ECC: Energy-scalable Elliptic Curve Cryptography for Wireless Sensor Networks," Proc. of International Conference on Applied Cryptography and Network Security (ACNS), Lecture Notes in Computer Sciences, vol. 8479, pp. 361-379, 2014.

[40] Z. Liu, J. Weng, Z. Hu and H. Seo, "Efficient Elliptic Curve Cryptography for Embedded Devices," ACM Transactions on Embedded Computing Systems (TECS), vol. 16, no. 2, pp. 1-18, 2016.

[41] S. Nimbhorkar and L. Malik, "Comparative Analysis of Authenticated Key Agreement Protocols Based on Elliptic Curve Cryptography," Proc. of the International Conference on Information Security & Privacy (ICISP2015), pp. 826-827, Nagpur, India, Elsevier, December 2015.

[42] M. Elhoseny, H. Elminir, A. Riad and X. Yuan, "A Secure Data Routing Scheme for WSN Using Elliptic Curve Cryptography and Homomorphic Encryption," Journal of King Saud University-Computer and Information Sciences, vol. 28, no. 3, pp. 262-275, 2016.

[43] F. De Rango, G. Potrino, M. Tropea and P. Fazio, "Energy-aware Dynamic Internet of Things Security System Based on Elliptic Curve Cryptography and Message Queue Telemetry Transport Protocol for Mitigating Replay Attacks," Pervasive and Mobile Computing, vol. 61, pp. 101-105, 2020.

[44] M. Düll, B. Haase, G. Hinterwälder et al., "High-speed Curve25519 on 8-bit, 16-bit and 32-bit Microcontrollers," Designs, Codes and Cryptography, vol. 77, no. 2, pp. 493-514, 2015.

ملخص البحث:

تَخلق شبكات المجسّات اللاسلكية تهديداتٍ تتعلّق بالأمان، الى جانب مدى بساطة وضرورة نموذج التّشفير. وعلى الرغم من حداثة ظهورها، فإنّ الاستفادة من التّشفير باستخدام المنحنيات البيضوية موضوع بارز للبحث ومنهج يهدف الى تقليل كلفة الوقت والكثافة في تلك الشبكات. ويرتكز أمان التّشفير باستخدام المنحنيات البيضوية على صعوبة مشكلة الخوارزميات المجردة المرتبطة بالمنحنيات البيضوية المعيارية مثل: X9.62 ANSI و 186.2 FIPS NIST وغيرها. وبسبب بعض العيوب في منحنيات NIST فيما يتعلق بالأمان، فإنّ من الضروري استقصاء بدائل آمنة. وفي هذا البحث، نختار ED_{25519}، وهو منحنى إدواردز (Edwards) على مستوى أمان قدرُهُ 128 بِت، ونقارنه بمنحنى وايرستراس (Weierstrass). ولإتمام وظائف الحساب الميداني، نستخدم ($radius\text{-}2^4$) الذي يمثل العوامل الداخلة في الحساب مع [MoTE-ECC] كنظام تشفير على المجالات الأساسية المثالية (OPFs) بأحجام مختلفة مثل: 160 و 192 و 244 و 255 بِت. كما نستخدم نظام (ECDH) لتبديل المفاتيح بين عُقدتيْن، حيث تحتاج كلّ عُقدة الى عمليّتي ضرب بين كمّيات غير متجهة لكي يتمّ تنفيذ العملية المتعلقة بتبديل المفاتيح. وتَستخدم عملية ضرب الكميات غير المتجهة مُنحنى إدواردز المُزاح وتقنية المشط لإنشاء نقطة الأساس، وتَستغل المحاور ذات الإسقاط الممتدّ لجمع النّقط. ويُظهر تطبيقنا على نظام (ECDH) استهلاكاً للطاقة قدرُهُ 18.20 ملي جول (mJ) عند استخدام مجالٍ أساسي مثالي بحجم 160 بِت. وهذا أفضل من عُقد المجسّات المستندة الى نظام (AVR). وتجدر الإشارة الى أن إيجابيات الطّريقة المقترحة تحقّق درجةً متقدمةً من الأمان والطّاقة المستهلكة، مع التّقليل من عوائق الاتصال عبر إدارة المفاتيح.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

A COMPARATIVE STUDY OF DIFFERENT SEARCH AND INDEXING TOOLS FOR BIG DATA

Ahmed Oussous[1] and Fatima Zahra Benjelloun[2]

(Received: 17-Nov.-2021, Revised: 15-Jan.-2022, Accepted: 26-Jan.-2022)

ABSTRACT

The exponential growth of data generated from the Moroccan court makes it difficult to search for valuable knowledge within multiple and huge datasets. Traditional searching methods are not adapted to Big Data context. Indeed, handling the search of specific information on Big Data requires advanced methods and powerful search systems. To contribute to the Court Digital Transformation Strategy, we aim to develop a solution that will leverage the technological advances in this field.The project we propose consists in developing new methods and techniques of artificial intelligence in order to automate the content of a large mass of data produced by the jurisdictions of the Kingdom of Morocco and to design a system capable of analyzing large volumes of complex judicial data. The aim is to discover and explain certain existing phenomena or to extrapolate new knowledge from the information analyzed, to recognize shapes, make predictions and make the necessary adjustments if necessary. For that, the purpose of this first study is to investigate and examine the existing search and indexing technologies for Big Data. It compares the leading solutions used for information retrieval in order to choose one that will serve as the base for our jurisprudential search engine.

KEYWORDS

Big data, Indexation, Search engines, Solr, ElasticSearch, Lucene.

1. INTRODUCTION

Nowadays, the potential of Big Data is recognized by many industries, research laboratories, governmental and private sectors. They exploit Big Data to extract valuable insight and knowledge. In fact, more than thousands of data gigabytes are rapidly generated every day, in different formats and from heterogeneous sources (ex., ICT applications, sensors, social media, mobile devices, logs and so on) [1].

Big Data is raising many challenges [2]. In fact, because of Big Data characteristics (velocity, variety and volume), experts need to process and analyze Big Data rapidly to extract valuable insights, find and analyze patterns within such large data, establish more accurate predictions and get a better understanding of the industrial changes. Big Data analysis is a powerful tool to maintain companies' agility and competitiveness [3].

Thus, to process such huge streams of data generated very rapidly and in different formats, experts need powerful solutions to stock, manage, process and analyze Big Data.

These tools are well explained in our paper as a review that surveys recent technologies developed for Big Data [4].This article offers a broad overview of the major Big Data technologies, as well as comparisons based on system components, such as data storage, data processing, data querying, data access and management. It classifies and examines the primary technological aspects, benefits, limitations and applications. Actually, experts need also advanced search and indexation tools to deal with Big Data that imposes huge volumes and high complexity. In fact, research efforts previously focused on finding efficient massive storage capabilities. But, there was a shift in research to innovate new and efficient solutions for advanced big-data analytics [5].

In fact, extracting useful information from such huge volumes of data requires adapted tools to perform advanced analytics and search operations on big data. Not only solutions should be scalable, efficient and powerful, but they should handle the complexity of the unstructured datasets and to retrieve data from distributed storage [6].

1. A. Oussous is with Department of Informatics, Faculty of Sciences and Techniques of Mohammedia (FSTM), Hassan II University, Casablanca, Morocco. Email: Ahmed.oussous@fstm.ac.ma
2. F. Benjelloun is with Laboratory of Engineering Sciences, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco. Email: fatima.benjelloun1@gmail.com

Thanks to the advancement in information technology, it has become easier to collect and store Big Data. But, there are other challenges, such as how to find new and efficient ways to first index data, then to extract information and valuable knowledge from massive volumes of unstructured data.

To fill this gap, many advanced indexing solutions have been developed and incorporated in big data analysis. The goal is to enhance query execution and optimize operations as well as to improve the efficiency of searching information in large, complex and unstructured datasets.

The process of search engine indexing needs scalable and powerful solutions that can collect, parse and store tremendous data volumes. Such process includes also the creation of indexes to ensure fast, efficient and accurate information retrieval. Indeed, searching aims to process queries and retrieve information based on both queries and the indexes previously created [6].

Indexation and search tools are important, as they play a major role to access and rapidly search data items. These search tools help experts in many data analysis tasks; for instance, to investigate and analyze massive datasets in order to find hidden patterns and discover relationships and other useful information.Those extracted patterns and information enable experts to get a better understanding of the studied phenomenon as well as to monitor sector changes and evolutions (for instance, customer behaviours and preferences). Such cohesive understanding is needed to make timely strategic decisions to take advantage of opportunities, minimize risks and control costs [7].

Big-data search and analytics technologies are used in many sectors (e.g. finance, marketing, research, health, security) and for important applications such as : to monitor disease evolution, adapt medical prescriptions online, detect traffic congestion points, understand the decision process of drivers enhance customer services, predict citizen and customers behaviors, prevent terrorism, detect policy violations and better understand nature evolution [8]-[11].

1.1 Motivation and Methodology

Multiple solutions were designed to tackle the issues related to information retrieval in the case of Big Data. However, because each solution has its advantages and limits, users need experience and deep knowledge to select the best suitable solution for their user case.

As far as we know, there is no published state-of-the-art survey assessing the efficiency and performance of such technologies in the literature. Hence, we try to fill this gap. This article is a continuation of our previous article, which deals with big-data technologies [4].

Information management is one of the main axes on which the Ministry of Justice is committed to modernize and develop the judicial administration with a view to establish the foundations of the digital court. To this end, the Ministry has made several investments to modernize judicial administration methods by intensifying the use of new information and communication technologies.

The transition to digital court implies the generation of raw, semi-structured and information-rich data. It generates extremely large data, which makes the management of this data quite complicated and difficult to process and analyze with classic database management tools or traditional information management tools. This large amount of data motivated us to study new methods and technologies that can be applied in order to understand the structure of any type of data and integrate it into models that can be understood and used by everyone. One of the research axes that we deal with in this article is the indexing and search for judicial information.

Indeed, the objective of this paper is to investigate and examine the existing search and indexing technologies for Big Data. It compares between the leading solutions used for information retrieval. The goal is to offer a detailed overview about such solutions as well as their best use cases in order to choose one that will serve as the base for our jurisprudential search engine.

Several search engines have been compared in terms of multiple criteria, like functionality, productivity, efficiency in searching and indexing, ease of use, speed and safety and so on. Advantages and disadvantages of each search engine have been included.

This paper is structured as follows: An overview of important related works on different search tools for Big Data is discussed in Section 2. Big Data search technologies are presented in Section 3 and an advanced comparison analysis of these different tools is discussed in Section 4. A brief conclusion is given in the last section.

2. RELATED WORKS

Most of Big Data search surveys pertinent to this topic give an overview of Big Data search tools' applications, opportunities and indexing challenges. Others discuss also techniques and methodologies used in big-data search and how they can help improve performance and results' accuracy.

In light of the literature, [12] examined Solr and ElasticSearch in terms of query and indexing speeds, simplicity of use, configuration forms and architectures. [13] compared and contrasted the two most popular platforms for building information retrieval systems, Apache Solr and ElasticSearch. The writers looked at both systems to see what they have to offer and how they are used. They looked at expert comments on both systems as well as real-world examples. They conducted a comparative analysis that looked at a variety of factors, including usability and scalability. Finally, they came to the conclusion regarding whatever system is superior for which application. Solr and ElasticSearch were compared by [14] in terms of productivity, ease of use, speed and safety. In addition, the advantages and disadvantages of either search engines have been included.

[15] utilized Splunk to detect the attack of Distributed Denial of Service. The authors used the data generated from the attacks with the Splunk platform to conduct data analysis to quickly identify attacks and predict potential dangers that could arise. [16] developed SmallClient as an indexing system for huge text data to increase indexing and search performance for large datasets. SmallClient focuses on increasing the volume and speed with which large datasets are processed. As a result, their technology becomes a generic indexing framework with quicker data access by allowing users to choose block size and replication factor. SmallClient's performance improves with increasing data amount, according to tests on small and large datasets.

A short comparison of four search engines, Sphinx, Apache Solr, ElasticSearch and Xapian, is done in a research article released by a group of researchers from Moscow Technological University [17]. The authors recommended to use ElasticSearch to arrange the interface for working with Big Data (search and visualization). [18] presented a system that uses a customized ElasticSearch search engine to successfully solve the problems of real-time analysis. As a consequence, the authors discovered that a suitable configuration of ElasticSearch and Kibana enables real-time analysis of large-scale data and can assist policy makers in seeing the findings instantly to support the decision-making. [19] conducted a functional analysis of well-documented open source forensic tools and search engines. The authors presented also a literature study of publicly accessible forensic datasets. They compared through a benchmarking exercise both ElasticSearch and Solr's indexing as well as full text searching procedures in terms of memory and time usage. [20] outlined the fundamentals of developing and deploying a social-media monitoring and analysis system for cybersecurity. The system is the outcome of a systematic method of gathering, processing and analyzing publicly available data. It is based on the use of information retrieval, data analysis and information flow aggregation methodologies and tools. In their built-in system, Sphinx, a full-text search engine for massive data, is employed as a search engine. [21] described the design and deployment of a CLP tool that compresses unstructured text logs while allowing rapid searches on the compressed material. CLP allows more efficient search and analytics on historical logs as compared to ElasticSearch and Splunk enterprise. [22] examined how the academic infrastructure network SINET was hit by a coronavirus-based cyber assault. They built a data flow pipeline based on ElasticSearch and Splunk to handle massive session traffic data recorded on SINET in order to extract and evaluate the COVID-19 attacker group's traffic patterns. Table 1 covers some of the most current studies on search and indexing technologies.

Table 1. Recent works on search and indexing tools.

Article & Year	Objective	Indexing and Search Tools Used	Obtained Results
[12] 2016	Comparison of big-data tools Solr and ElasticSearch	ElasticSearch and Solr	They are similar tools in terms of technical features. Both tools are rapid search tools. ElasticSearch has a wider range of coding languages than Solr. ElasticSearch performs better with short data, whereas Solr performs better with long data. When compared to the amount of data after indexing, Solr utilizes less disk space.

[13] 2016	Providing an overview of the best options for constructing information retrieval systems to developers and members of the scientific community, as well as offering insight into the best use cases for both technologies	Apache, Solr and ElasticSearch	ElasticSearch's ease of use, flexibility and modular architecture make it an excellent candidate for prototype as well as big, scalable information retrieval applications. ElasticSearch provides considerably superior data analytics and the ELK stack, when paired with Logstash and Kibana, it outperforms Solr in several areas, including preprocessing, analytics and visualization. The present version of ElasticSearch has a drawback in that it lacks a centralized mechanism for managing cluster nodes. Teams with Solr expertise should think twice before switching to a new system, as both systems are almost comparable in most circumstances.
[14] 2016	Comparing and analyzing the security of Solr and ElasticSearch; two popular full-text search engines	Solr and ElasticSearch	It includes powerful filtering, highlighting, multi-dimensional searching, caching, Rest Api and a distributed architecture support engine. The Restful API is a very quick and useful tool. When compared to the Solr search engine, ElasticSearch is less complicated and detailed. It is both durable and adaptable. One of the most significant advantages is that it is distributed and real-time.
[15] 2016	Detecting distributed denial of service attacks	Splunk	During DDoS assaults against firewalls, researchers used Splunk big-data technologies to examine traffic characteristics. The experimental results did certainly aid in the knowledge of various attack types and a warning system might be used to identify security issues prior to an assault.
[16] 2017	Creating a huge text data indexing system to increase indexing and search efficiency for massive datasets	SmallClient	When compared to the Lucene indexing library, SmallClient outperforms in terms of index generation and has the shortest time between data upload and query execution. SmallClient indexes are also lower in size than Lucene indexes, in addition to being faster to create. To get complete records, Lucene requires that all attributes be indexed. SmallClient, on the other hand, is not one of them. Even when just one attribute is indexed, SmallClient allows you to obtain whole data records. With growing data amount, SmallClient improves.
[17] 2017	Analysis of software for full-text search and data visualization	Sphinx, Solr and ElasticSearch	Sphinx has a rapid search and indexing system, but it is slow to update. Only MySql and Postgres are supported by Sphinx. Overall, the ElasticSearch system is the best choice for full-text search and data visualization.
[18] 2018	Suggesting a solution to real-time analytical problems	ElasticSearch and Kibana	ElasticSearch is a real-time storage, pre-indexing, search and query solution for very big datasets. A correct ElasticSearch and Kibana configuration enables for real-time analysis of enormous amounts of data, allowing policy makers to view the results instantly and in a manner that allows for decision-making.
[19] 2018	Comparing the functionality, efficiency and effectiveness of open-source search engines for digital forensic search	Solr and ElasticSearch	Deduplication keyword recommendations and search result clustering are supported by Solr, whereas phonetic search is supported by ElasticSearch. Solr provides many unique capabilities that can help with large-scale dataset search. In terms of index building time, ElasticSearch outperformed Solr.

[20] 2020	Monitoring system for social-media content	Sphinx	The practical importance of the obtained results is to develop a functioning model of a social-media content monitoring and analysis system that can be used as part of information and cyber security decision support systems.
[21] 2021	Building a fast and scalable search tool for compressed text logs	CLP tool	ElasticSearch and Splunk enterprise are equivalent, if not better than the CLP tool when it comes to search performance. The CLP tool exceeds Elastic-search and Splunk enterprise in terms of log ingestion by almost 13 times.
[22] 2021	Extracting and analyzing the traffic patterns of the COVID-19 attacker group	ElasticSearch and Splunk	Some unveiled patterns are informative to handling security operations of the academic backbone network.

3. BIG DATA INDEXING TECHNOLOGIES

With big-data search and indexing technologies, data scientists and others can analyze huge volumes of data that conventional analytics and traditional business intelligence solutions cannot handle. The following sub-sections discuss the finest search tools that provide full featured search engines. Thanks to their scalable and high-performance indexing, these tools are designed for information retrieval in Big Data.

3.1 Apache Lucene

We opted to introduce Apache Lucene [23] before looking into Solr and ElasticSearch. This introduction constitutes the information retrieval library for both systems.

Apache Lucene [24] is developed to address big-data searching needs. Lucene is an open-source, high-performance and full-featured text search engine library that is built completely in Java.

Apache Lucene offers multiple query options and scalable indexing (it indexes almost 150 GB per hour on commodity hardware) with minimal memory requirements. The algorithm offers ranked searching, field searching, data-range searching as well as multiple-index searching [8].

Apache Lucene offers powerful features related to four main categories: analysis of incoming content and queries, indexing and storage, searching and ancillary modules (everything else) [25]. The first three items contribute to Lucene's core, while the last item consists of code libraries that have proven to be useful in solving search-related problems.

A high-level Lucene architecture is presented in Figure 1. Its main components are IndexSearcher, IndexReader, IndexWriter and Directory. The IndexWriter object is used to create the index and add new index entries (i.e., Documents). IndexReader reads the content of indexes in support of IndexSearcher. Directory abstracts out the implementation of index dataset access and provides APIs for manipulating them. Both IndexReader and IndexWriter leverage Directory for access to this data. The standard Lucene distribution contains several Directory implementations, such as filesystem-based and memory-based, Berkeley DB-based (in the Lucene contrib module) and several others [26]. Lucene is one of the most powerful and widely used search engines.

3.2 Apache Solr

Solr [27] is the popular, blazing fast open-source enterprise search platform from the Apache LuceneTM project. Apache Solr is more compatible, its major features include powerful full-text search, hit highlighting, faceted search, nearly real-time indexing, dynamic clustering, database integration, rich document (e.g. Word, PDF) handling and geospatial search.

Thanks to SolrCloud mode [28], Solr provides a highly available, scalable, replication and fault-tolerant environment for distributing the indexed content and requests across multiple servers with the help of ZooKeeper. In fact, SolrCloud uses the information in the ZooKeeper database to figure out which servers need to handle the request. In fact SolrCloud's integration with end-user applications is depicted in Figure 2.

As illustrated in Figure 2, there are four elements. SolrCould is an indexing and search service that runs

independently. Users can shop for items from the online store *via* an end-user application, such as an online store application. The Content Management System gives the shop's employees an internal access to update product information from various data sources. Solr will index the product metadata for end users to consume *via* a simple HTTP request and return format of JSON, XML or CSV.

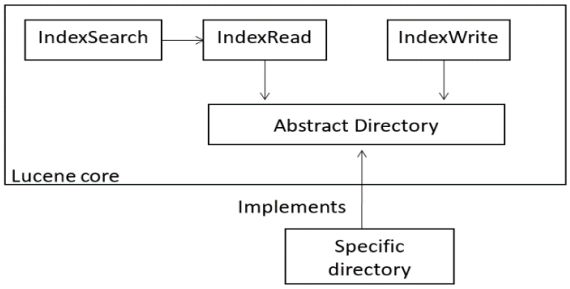


Figure 1. High-level Lucene architecture.

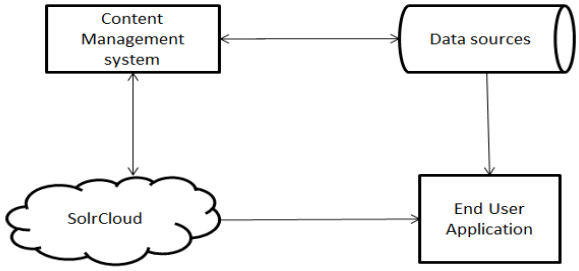


Figure 2. Solr integration with applications.

While Apache Solr can easily handle high-volume traffic, Apache Lucene is used by search-based sites in order to handle reverse index and the related issues.

Contrary to Lucene, Solr is easy to use. It can be installed and used by non-programmers. Solr is a web application (WAR) which can be deployed in any servlet container. Solr is integrated into major distribution of Hadoop (Cloudera, Hortonworks, MapR) as the search engine for their products marketed for Big Data [29].

Unlike Lucene, which is a Java library that can only be used from other Java programs, Solr on the other hand is a wrapper around Lucene that allows using the Lucene functionality from any programming language that can submit HTTP requests.

Solr is used by many largest internet sites across the world. This is because it enhances the search and navigation features. Indeed, it is capable of indexing, efficiently searching multiple websites and returning recommendations for related content. For that, it uses the search query's taxonomy. Furthermore, Solr is a mature solution that has a large user community.

3.3 ElasticSearch

ElasticSearch [30] is an Apache 2.0 licensed open source search solution that is based on JSON. It is an efficient solution used to store, search and analyze structured and unstructured data. Thus, it is suitable for many data types, including system logs, free text, time-series and NoSQL data. ElasticSearch is built on top of Apache Lucene.

ElasticSearch is a distributed, multi-tenant and document-oriented search engine [31]. It supports distributed deployments, by breaking down an index into shards and distributing the shards across the nodes in the cluster.

By integrating it to Logstash and Kibana tools, ElasticSearch can perform tasks for search, analysis and visualization operations. The integration of these three tools is called ELK stack [32].

While both ElasticSearch and Apache Solr use Apache Lucene as the core search engine, ElasticSearch aims to provide a more scalable and distributed solution that is better suited for cloud environments than Apache Solr.

3.4 Splunk

Splunk Enterprise [27] is one of the leading platforms for collecting, analyzing and visualizing machine-generated Big Data. It provides a unified way to organize and extract real-time insights from massive amounts of machine data generated by diverse sources. Spunk is equivalent to ELK Stack that includes ElasticSearch, Logstash and Kibana for storage, analysis and visualization. But, it is mainly used for big-data analysis and can analyze structured or semi-structured data. It is possible to get 15 days free trail of Splunk commercial solutions. The latter was released in 2003.

HunK: Splunk Analytics for Hadoop [33] is a platform for discovering, analyzing and visualizing Hadoop's historical data at rest. Hunk is a full-featured Hadoop exploration, analysis and visualization application. Hunk offers huge improvements in the speed and ease of gaining insights from large data at

rest in Hadoop, based on many years of expertise designing big-data solutions that have been implemented by thousands of Splunk customers. Hunk is compatible with Apache Hadoop and the majority of Hadoop distributions, including MapReduce.

Splunk has three main functionalities, including data collection, data indexing, as well as data search and analysis as follows [34]:

- Data collection: Splunk can gather static data as well as data created by real-time monitoring of modifications and additions to files and directories. Data can also be gathered directly from programs or scripts using network ports. Splunk can also gather, insert and update data from relational databases.
- Indexing: The acquired data is divided into events, which are essentially equal to database entries or simply lines of data. The data is then processed and a high-performance index that points to the stored data is built and updated.
- Search and analysis: Users may use the Splunk Processing Language to search for data and alter it to get the information they need, whether it is in the form of reports or alerts. Individual events, tables and charts can be used to show the findings [35].

3.5 Sphinx Search Server

Sphinx (SQL Phrase Index) [36] is a standalone full-text search engine that gives third-party programs, particularly SQL databases with efficient search capability. This search engine was created in 2001 by Andrew Aksyonoff, a Russian engineer, to ensure (1) good search quality, (2) fast speed and (3) low resource usage (Disk IO, CPU). It's compatible with scripting languages like Python and Java.

Sphinx [37] is an open-source full-text search server, designed from the ground up with performance, relevance (aka search quality) and integration simplicity in mind. It is written in C++ and works on Linux (RedHat, Ubuntu, …etc), Windows, MacOS, Solaris, FreeBSD and a few other systems. Sphinx clusters scale up to tens of billions of documents and hundreds of millions search queries per day, powering top websites, such as Craigslist, Living Social, MetaCafe and Groupon.

Sphinx [38] has been improved. Currently, it is able to handle nearly real-time search among huge volumes of files. In fact, if users need search functions without data visualization and analysis, then Sphinx is a good choice for fast indexing and querying. Sphinx can process 500 queries/sec against 1,000,000 documents with the biggest registered number of indexing estimated at 25+ billion documents. Table 2 provides a general overview for the features of big-data search tools.

Table 2. General overview for the features of big-data search tools.

Feature	**solr**	**ElasticSearch**	**Splunk**	**Sphinx**
Initial release	2004	2010	2003	2001
License	Open-source	Open-source	Commercial	Open-source
Developer	Apache Software Foundation	Elastic	Splunk, Inc.	Sphinx Technologies, Inc.
Format	XML, CSV, JSON	JSON		
Official client libraries	Java	Java, Groovy, PHP, Ruby, Perl, Python, .NET, Javascript		C++
Community client libraries	PHP, Ruby, Perl, Scala, Python, .NET, Javascript, Go, Erlang, Clojure	Clojure, Cold Fusion, Erlang, Go, Groovy, Haskell, Java, JavaScript, .NET, OCaml, Perl, PHP, Python, R, Ruby, Scala, Smalltalk, Vert.x	C# ,Java, JavaScript, PHP, Python Ruby	C++, Java, Perl, PHP, Python, Ruby
Server operating systems	All OS with a Java VM	All OS with a Java VM	Linux OS X Solaris Windows	FreeBSD Linux NetBSD OS X Solaris Windows

4. COMPARATIVE ANALYSIS OF BIG DATA INDEXING TECHNOLOGIES

This section compares the various tools we discussed in Section 3, notably Solr, ElastiSearch, Splunk and Sphinx, in terms of a variety of criteria.

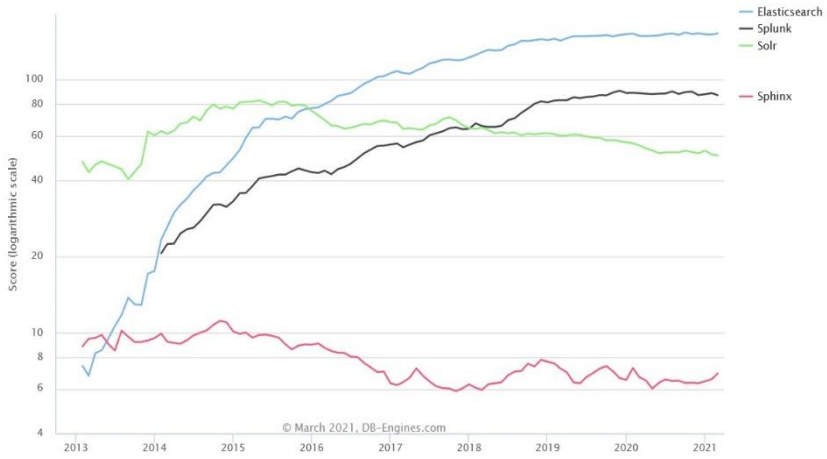


Figure 3. DB-engines ranking.

4.1 Ranking of Search Engines

Regarding the popularity (Figure 2), ElasticSearch is well ranked in comparison to other big-data tools, as it is considered the most popular search engine since 2016. In fact, while ElasticSearch is ranked number one, Sphinx is ranked number two and Solr is ranked number three, as confirmed by DB-Engines that ranks database management systems and search engines according to their popularity. On the contrary, Solr has become popular among the first ten years of its initial release.

4.2 Data Sources

Both ElasticSearch and Solr can handle various types of data sources. Since ElasticSearch is totally based on JSON, it supports data ingestion from different sources by using Logstach and Beasts family.

On the contrary, Solr is based on request handlers to ingest data from multiple sources, including CSV files, databases, XML files, Microsoft Word documents and PDFs. Solr is capable of supporting extraction and indexing from over one thousand file types. This is because of the native support for the Apache Tika library.

Splunk, on the other hand, provides tools for setting a variety of data sources, including those specific to application requirements. Splunk also has capabilities for configuring input forms for any type of data. Files and folders, data from system log files and any other application that uses the TCP protocol can all be used as Splunk inputs. Data may be indexed by Splunk Enterprise from any network port. It can also index data sent over UDP. Splunk software also supports a variety of data sources, including Windows Event Log data, Active Directory data and data from performance monitoring.

When it comes to Sphinx, the data to be indexed can originate from a variety of places, including SQL databases, plain text files, HTML files, e-mails and more. The data that Sphinx indexes is a collection of structured documents, each with the identical set of fields and characteristics. This is comparable to SQL, in which each row represents a document and each column represents a field or property. Different code is necessary to get the data and prepare it for indexing depending on what source Sphinx should obtain the data from. Data source driver is the name of the program (or simply driver or data source for brevity).

4.3 Use Cases

Both Solr and ElasticSearch are document-oriented search engines [13]. But, Solr is more focused on enterprise-directed text searches with advanced information retrieval (IR). It is a good choice for use cases that need to search within a massive volume of static data or to handle Rich Text Format (RTF) documents. Solr is also recommended for Enterprise applications that are based on Big Data ecosystem, like Spark or Hadoop. To be competitive, Solr has implemented new features, including Parallel SQL Interface and streaming expressions.

On the contrary, ElasticSearch is adapted for many use cases. For instance, it is a powerful and flexible solution for full text search. Indeed, it is a good choice not only for Enterprise search and E-commerce [39], but also for other use cases, such as fraud detection, security and collaboration [40]-[41].

ElasticSearch is known for its scalability and easiest way to implement powerful logging solutions. It is capable to grab and index different remote large data sources. It can handle easily time-series data, such as application events and metrics. ElasticSearch is more convenient for modern web applications where data is in JSON format.

Sphinx is a full-text search engine with the advantages of fast indexing and searching, as well as integration with existing database management systems (MySQL, PostgreSQL) and an API for common web programming languages (officially supports PHP, Python and Java; community-implemented APIs for Perl, Ruby, .NET and C++). For Russian and English languages, sophisticated search features, such as ranking and stemming, are supported. The Delta index technique can be used to accelerate indexing for huge amounts of data. Sphinx also offers Real-Time indexes, search result filtering and sorting and wildcard searching. In comparison to ElasticSearch, Sphinx uses fewer memory and compute resources.

Splunk began as a machine-generated data analytics platform, but it has now extended into a number of different domains, including the fields of IT operations and application delivery, security compliance, fraud management, business analytics and the internet of Things. It continues focused on becoming "the omnipresent machine data platform, the standard in every enterprise."This was accomplished through the development of a number of products and version updates, including Splunk Enterprise 8.2.4, a new version of Splunk Cloud Platform, Splunk Enterprise Security 7.0.0 and so on. Event sequencing, a new use case library to speed investigations, rules to strengthen insider threat detection models and solutions to give a threat intelligence-centered view for investigations are among the new features of the revised products.

Both ElasticSearch [3] and Splunk [5] are two of the industry's biggest players right now. Elastic claimed sales of $428 million with 11,300 clients in their most recent fiscal year [13], whereas Splunk reported a revenue of $2.359 billion with 19,400 customers [21]. Furthermore, the products ElasticSearch and Splunk Enterprise are employed by a number of major corporations, including eBay, Verizon and Netflix.

4.4 Searching

Currently, ElasticSearch and Solr support (nearly real-time) searches as well as JSON-based Query DSL. They both take advantage of Lucene's search capabilities.

Unlike ElasticSearch, Solr enables users to write complex search queries. Solr's Standard Query Parser allows users to create a variety of structured queries, but the probability of syntax errors is higher.

On one hand, Solr provides a search user interface named Velocity Search. The latter has robust features. In addition to searching, users can exploit faceting, highlighting, autocomplete and Geo Search. On the other hand, ElasticSearch has a native DSL and a robust aggregation framework with a better caching. It is noticed that the last releases of ElasticSearch ensure a better memory management.

Splunk Assistant [42] is a search function that appears as users input their search parameters into the Search application. The Search Assistant is similar to autocomplete, but with a lot of more features. Matching queries are also returned by the Search Assistant, which are based on recent searches. When users wish to rerun a search from yesterday or a week ago, they can use the Matching Searches list. When they log out, their search history is saved.

Splunk's Search Processing Language (SPL) [43] contains commands and functions for creating searches. Sphinx treats full-text searches as simple "bags of words" by default and all keywords in a document must match in order for the query to succeed. To put it another way, users may do a rigorous Boolean AND on all keywords by default. Text queries, on the other hand, are significantly more versatile and Sphinx has its own full-text query language to reveal that versatility.

4.5 Indexing Performance

Earlier, Solr was based on a defined schema. But currently, both ElasticSearch and Solr supports schemaless mode. As a result, both are flexible and can be used to index data and dynamic fields. So, users do not need to define in advance the schema of the index.

Both ElasticSearch and Solr write indexes in Lucene. But, they have different architecture, files and different mechanisms for sharding and replication. Moreover, while Solr has a powerful Standard Query Parser that is compatible with Lucene syntax, ElasticSearch has native DSL (Domain Specific Language)

support. Both solutions support synonym-based indexing, stemming, custom analyzers and various tokenization options.

Sphinx has a quick search and indexing system [44]; however, it is slow to update due to the lack of an automated index updating mechanism. It only works with MySql and Postgres, which is a huge limitation. It is incompatible with the work at hand, since it is unable to update or remove documents in the index.

Apache Solr features a fast indexing and searching performance, one of the lowest index sizes and a lot of adaptability. It can also be used as a storage facility. Solr comes with a slew of extra features, like imprecise search and the capacity to scale right out of the box. The drawback is that it is a Java server in a servlet container that has been turned into a web service with XML, JSON and CSV interfaces.

ElasticSearch, which is built on Apache Lucene, has somewhat slower indexing and searching speeds than Sphinx, but it also has other features in addition to search and storage (visualization, log collector, encryption system, …etc.). It has the ability to scale and can sample highly complicated forms, making it an excellent choice for an analytical platform. This engine is not the most user-friendly, but it has a lot of extra functions. The main benefit is that this engine consumes very little memory and incremental indexing is as quick as indexing several articles at once. ElasticSearch is substantially quicker than Solr for indexing, as demonstrated by [19].

As a result, ElasticSearch, a search engine and full-text search system, is ideally suited for searching and visualizing enormous volumes of clustered data resulting from users' interactions with diverse information resources.

On the other hand, ElasticSearch and Splunk Enterprise work by creating external indexes on log messages as they are being ingested. These tools may then swiftly search the indexes corresponding to the logs in response to a query, decompressing just the chunks of data that may include logs matching the search term. For example, ElasticSearch is based on Lucene, a general-purpose search engine. This strategy, however, comes with a high cost in terms of storage space and memory use. Despite the fact that these methods compress the logs lightly, the indexes typically take up the same amount of space as the raw logs; moreover, to be completely effective, these indexes must be maintained largely in memory or on fast random access storage.

Thus, users of Splunk Enterprise and ElasticSearch who have a lot of data may only afford to keep their indexed logs for a few weeks [21].

4.6 Clusters, Sharding and Rebalancing

Both search engine solutions support sharding. However, while SolrCloud enables further splitting of an existing shard, ElasticSearch does not offer this option. So, shards cannot increase once they've been created in ElastiSearch. But, shards of an index can be reduced in ElasticSearch based on a shrink API, but it is not possible using SolrCloud.

For cluster coordination, ElasticSearch provides built-in Zen Discovery module. Instead, SolrCloud needs an additional service that is Apache Zookeeper.

When there is a shard or node failure, Elastisearch rebalances clusters automatically. It is rare when manual intervention is required. But, SolrCloud has a complex rebalancing mechanism that is hard to manage [45].

Indexer clusters are groups of Splunk Enterprise indexers set to duplicate each other's data, allowing the system to store multiple copies of all data. Index replication is the name for this procedure. Clusters reduce data loss while enhancing data availability for searches by retaining several, identical copies of Splunk Enterprise data.

Automatic failover from one indexer to the next is a characteristic of indexer clusters. This implies that even if one or more indexers fail, incoming data is still indexed and searchable.

Sphinx offers distributed search capabilities, which helps it scale effectively. In multi-server, multi-CPU or multi-core setups, distributed searching can help reduce query latency (i.e., search time) and throughput (max queries/sec). This is critical for apps that must sift through large volumes of data (i.e., billions of records and terabytes of text). It also allows you to create an arbitrary cluster architecture, clustering and sharding over several agent servers.

4.7 Data Visualization

A user-friendly interface (Graphical user interface (GUI)) is essential for users. For that, Splunk has improved its GUI by integrating a new dashboard and its controls. It offers also the possibility to export the dashboards to pdf version *via* simple features [46].

On the contrary, ElasticSearch does not offer its own GUI. Therefore, users need to install Kibana for visualization [18]. Kibana has various cool background themes that Splunk does not offer. It offers also different controls to manipulate dashboards. Thus, the dashboard in Kibana is slightly better than in Splunk.

The Banana project [47], which was forked from Kibana and works with all types of time series (and non-time series) data saved in Apache Solr, has been integrated into Apache Solr's data visualization capabilities. It makes use of Kibana's extensive dashboard configuration capabilities, adapts important panels to work with Solr and adds a slew of new features. The objective is to offer a rich and flexible user interface that allows users to quickly design end-to-end applications that take advantage of Apache Solr's capability.

4.8 Machine Learning

Solr offers machine learning as a free module that runs on top of the streaming aggregations architecture [48]. Users may employ machine-learned ranking models and feature extraction on top of Solr with the help of the additional libraries in the contrib module, whilst the streaming aggregation-based machine learning is focused on text categorization using logistic regression.

ElasticSearch, on the other hand, offers a commercial solution called X-Pack [49], which includes a Kibana plugin that enables machine-learning techniques for anomaly and outlier identification in time-series data. It is a fantastic set of tools with professional services wrapped in, but it is rather costly.Through Splunkbase, users of Splunk Enterprise and Splunk Cloud Platform can use the Machine Learning Toolkit (MLTK).The Machine Learning Toolkit adds additional Search Processing Language (SPL) search commands, macros and visualizations to the Splunk platform. More than 30 algorithms are supported by MLTK, which are the most extensively used machine-learning algorithms. Anomaly Detection, Classifiers, Clustering Algorithms, Cross-validation, Feature Extraction, Preprocessing, Regressors, Time Series Analysis and Utility Algorithms are all categorized by algorithm type.

4.9 The Community

ElasticSearch is driven more by its company. Indeed, even though those contributors can access and change the code, the final changes are confirmed by the employee of the company. Most of the code is open-source, but there are non-open premium features. For Solr, users can contribute directly to its open-source code. New Solr developers or code committers are selected based on merit. It has a large community.

Splunkbase is a Splunk-hosted community where users can find applications and add-ons for Splunk that can enhance its capability and usefulness, as well as providing a quick and easy interface for certain use-cases and/or vendor products. There are currently over 2,512 applications on the framework [50].

4.10 Documentation

ElasticSearch improved its website and its documentation. As a result, user can find easily clear configuration instructions and multiple examples. Furthermore, because of the ElasticSearch popularity, the internet is full of its books and guides. On the contrary, Solr documentation is not well-maintained. In fact, following its release, it was easy to find well documentation about API's use cases and good examples. But currently, Solr documentation is not complete as many gaps were noticed by users. APIs coverage is not sufficient and it is not easy to find good technical examples and tutorials.Sphinx is the same way, with only a few pages of documentation and no technical examples.

Splunk documentation, on the other hand, comes in a number of formats and topic kinds. Step-by-step instructions, conceptual information, reference manuals, troubleshooting pages, use cases and product tutorials are all included in the Splunk documents collection. The easiest approach for users to achieve their goals using Splunk products is to read the documentation provided by Splunk.

4.11 Summary

After carefully analyzing all systems and reviewing relevant papers and publications, we have come to the conclusion that all of the systems given are viable options for document indexing and searching.

It is not easy to define a winner between those advanced big-data solutions. For that, it is necessary not only to understand their features, ease of maintenance, scaling options, but also to analyze their use cases.

To summarize the comparison, Solr and ElasticSearch have common advantages. For instance, both technologies are quite easy to install and to begin working with. Furthermore, both engines are documented and have matured codebase and large ecosystem. However, they are different. For instance, while Solr offers many functionalities in the field of information retrieval, ElasticSearch is easier for production and scalability.

Depending on their case study and requirements, users can select between the two.Solr is the ideal option for consumers who want a text-based search. ElasticSearch, on the other hand, is the perfect solution if they need distributed and scalable features with analytical queries.

Sphinx is an excellent tool for searching structured data (predefined fields and non-text attributes). Sphinx, on the other hand, needs a lot of effort and time to configure for unstructured material, like MP3s, PDFs and DOCs.As a result, compared to its competitors, it is more difficult to use.

ElasticStack (ELK Stack) and Splunk are the two most popular enterprise log analytics platforms. Splunk is a software tool for monitoring, analyzing and visualizing data. ElasticSearch is a database search engine, while Splunk is a software tool for monitoring, analyzing and visualizing data. Splunk is used to search, monitor and analyze machine data, whereas ElasticSearch stores and analyzes data. Splunk has a number of drawbacks, one of which is that it is a paid and pricey tool, whereas ElasticSearch is a free tool.

In terms of data transfer and user administration, Splunk is a simple and dependable solution, although ElasticSearch is rapidly gaining these capabilities.

Table 3 compares the performance and setup capabilities of several big-data search tools. The comparison is based on data processing techniques and direct or indirect access, data storage, data processing, distributed architectural features, search and indexing capabilities and tool performance.

Table 3. Technical comparison of big data-search tools.

Technical Specifications	**Sub-specifications**	**Solr**	**ElasticSearch**	**Splunk**	**Sphinx**
Access and Data Processing	SQL	Solr Parallel SQL Interface	SQL-like query language	No	SQL-like query language (SphinxQL)
	APIs and other access methods	Java API REST-ful HTTP/JSON API	Java API REST-ful HTTP/JSON API	HTTP REST	Proprietary protocol
	Data Import	DataImportHandler CSV, XML, Tika, URL, Flat File	Rivers modules, ActiveMQ, Amazon SQS, CouchDB, Drop-box, DynamoDB, FileSystem, Git, GitHub, , JDBC, JMS, Kafka, MongoDB, neo4j, Redis, RSS , Twitter, … etc.	Event logs, web logs, live application logs, network feeds, system metrics, archive files, …etc	SQL databases, plain text files, HTML files, mailboxes and so on
Distributed Architecture	Master-slave replication	Only in non-SolrCloud	Not an issue, because shards are replicated across nodes	Multi-source replication	None
	Partition Tolerance	Yes	No	Yes	Yes
	Shard replication	Yes	Yes	Yes	Yes
	Consistency	Eventual Consistency: Indexing requests that are synchronous with replication	By default consistent; Replication between nodes is set to synchronous	Eventual Consistency	
	Web Admin interface	Bundled with Solr	Marvel or Kibana apps	Splunk Web	JamDocs: a web interface

Indexing and Searching	Indexing and Searching	Text-oriented	Better performance of analytical queries	A scalable and reliable platform for investigating, monitoring, analyzing and acting on data	Better ranking relevance
	Real-time Sarch/Indexing	Yes	Yes	Yes	Yes
	Performance	High	High	High	High
	Visualization of data	Banana (Port of Kibana)	Kibana	Splunkbase	No
Characteristics		Highlighters, spell check, autocomplete, filter queries, geospatial, synonyms	index, cross-cluster search, Highlighters, Query DSL, Typeahead, corrections (spell check)	autocomplete suggestions, multifield	autocomplete suggestions, spell checker, faceted, synonyms, highlighting

5. CONCLUSIONS AND FUTURE WORK

Nowadays, large data volumes are daily generated at unprecedented rates from heterogeneous sources. However, traditional technologies lack scalability and performance needed in big-data context. Indeed, traditional indexing solutions are not adapted for big-data. This is because of the large and increasing size of indexes that requires more processing time and optimized index scheme.

This paper reviews and compares the main searching and indexing tools developed to handle big-data challenges. Those solutions are different, but most of them integrated advanced technologies to be scalable and powerful with high-performance indexing.

Furthermore, we compare their features. We notice that most of them optimize indexing and queries to ensure a real-time searching and indexing. They support full-text search by many ways. In addition, they offer shard replication, eventual consistency and methods to process data coming from distributed storage or in the Cloud with some differences. Some offer also the possibility to create plug-in APIs. For an easy usage, they also integrate options for visualization and other features. In addition to this comparison, users need experience to select among big-data searching and indexing solutions according to their needs, because each of them has advantages and limitations.

Overall, the ElasticSearch system is the best choice for full-text search and data visualization applications (free, open-source, simple interface, web-based data processing). It is suggested that the interface for working with big-data be organized using ElasticSearch's capabilities (search and visualization).

In the future work, we aim to build a legal search engine for the Moroccan Ministry of Justice, based on ElasticSearch, which was recommended after conducting this study.

ACKNOWLEDGEMENTS

This paper is part of a project of the AL KHAWARIZMI research program funded by CNRST and ADD: "Elaboration d'un système numérique robuste et intelligent dans le domaine de la justice".

REFERENCES

[1] T. J. Ma, R. J. Garcia, F. Danford, L. Patrizi, J. Galasso and J. Loyd, "Big Data Actionable Intelligence Architecture," Journal of Big Data, vol. 7, no. 1, pp. 1–19, 2020.

[2] V. V. Kolisetty and D. S. Rajput, "A Review on the Significance of Machine Learning for Data Analysis in Big Data," Jordan. Jou. of Comp. and Inf. Technol. (JJCIT), vol. 6, no. 01, pp.41-57, 2020.

[3] J. Wang, Y. Yang, T. Wang, R. S. Sherratt and J. Zhang, "Big Data Service Architecture: A Survey," Journal of Internet Technology, vol. 21, no. 2, pp. 393–405, 2020.

[4] A. Oussous, F.-Z. Benjelloun, A. A. Lahcen and S. Belfkih, "Big Data Technologies: A Survey," Journal of King Saud University-Computer and Information Sciences, vol. 30, no. 4, pp. 431–448, 2018.

[5] H. Hu, Y. Wen, T.-S. Chua and X. Li, "Toward Scalable Systems for Big Data Analytics: A Technology Tutorial," IEEE Access, vol. 2, pp. 652–687, 2014.

[6] A. Gani, A. Siddiqa, S. Shamshirband and F. Hanum, "A Survey on Indexing Techniques for Big Data: Taxonomy and Performance Evaluation," Knowledge and Inf. Systems, vol. 46, no. 2, pp. 241–284, 2016.

[7] V. Jatakia, S. Korlahalli and K. Deulkar, "A Survey of Different Search Techniques for Big Data," Proc.

of the IEEE International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), pp. 1–4, Coimbatore, India, 2017.

[8] T. Lee, H. Lee, K.-H. Rhee and U. S. Shin, "The Efficient Implementation of Distributed Indexing with Hadoop for Digital Investigations on Big Data," Computer Science and Information Systems, vol. 11, no. 3, pp. 1037–1054, 2014.

[9] T. H. Davenport and J. Dyché, "Big Data in Big Companies," International Institute for Analytics, vol. 3, pp. 1–31, 2013.

[10] R. V. Zicari, "Big Data: Challenges and Opportunities," Big Data Computing, vol. 564, p. 103, 2014.

[11] H. Ma, W. Du, S. Xu and W. Li, "Searching Tourism Information by Using Vertical Search Engine Based on Nutch and Solr," Proc. of the 17th IEEE International Conference on Software Engineering Research, Management and Applications (SERA), pp. 128–132, Honolulu, HI, USA, 2019.

[12] M. A. AKCA, T. Aydoğan and M. İlkuçar, "An Analysis on the Comparison of the Performance and Configuration Features of Big Data Tools Solr and ElasticSearch," International Journal of Intelligent Systems and Applications in Engineering, vol. 6, no. Special Issue (2016), pp. 8–12, 2016.

[13] N. Luburić and D. Ivanović, "Comparing Apache Solr and ElasticSearch Search Servers," Proc. of the 6th International Conference on Information Society and Technology (ICIST 2016), pp. 287-291, 2016.

[14] U. Kılıç and K. Aksakalli, "Comparison of Solr and ElasticSearch among Popular Full Text Search Engines and Their Security Analysis," Proc. of 6th International Conference on Future Internet of Things and Cloud Workshops, pp. 163–168, DOI: 10.13140/RG.2.2.24563.32803, 2016.

[15] T.-J. Su, S.-M. Wang, Y.-F. Chen and C.-L. Liu, "Attack Detection of Distributed Denial of Service Based on Splunk," Proc. of the IEEE International Conference on Advanced Materials for Science and Engineering (ICAMSE), pp. 397–400, Tainan, Taiwan, 2016.

[16] A. Siddiqa, A. Karim and V. Chang, "Smallclient for Big Data: An Indexing Framework towards Fast Data Retrieval," Cluster Computing, vol. 20, no. 2, pp. 1193–1208, 2017.

[17] A. Voit, A. Stankus, S. Magomedov and I. Ivanova, "Big Data Processing for Full-text Search and Visualization with ElasticSearch," Int. J. of Advanced Comp. Sci. and Appl., vol. 8, no. 12, p. 18, 2017.

[18] N. Shah, D. Willick and V. Mago, "A Framework for Social Media Data Analytics Using ElasticSearch and Kibana," Wireless Networks, vol. 2018, pp. 1–9, DOI: 10.1007/s11276-018-01896-2, 2018.

[19] J. Hansen, K. Porter, A. Shalaginov and K. Franke, "Comparing Open Source Search engine Functionality, Efficiency and Effectiveness with Respect to Digital Forensic Search," Norsk Informasjonssikkerhetskonferanse (NISK), pp. 1-14, 2018.

[20] D. Lande, I. Subach and A. Puchkov, "A System for Analysis of Big Data from Social Media," Information & Security, vol. 47, no. 1, pp. 44–61, 2020.

[21] K. Rodrigues, Y. Luo and D. Yuan, "CLP: Efficient and Scalable Search on Compressed Text Logs," Proc. of the 15th USENIX Symposium on Operating Systems Design and Implement., pp. 183–198, 2021.

[22] R. Ando, Y. Kadobayashi, H. Takakura and H. Itoh, "Understanding Traffic Patterns of Covid-19 Ioc in Huge Academic Backbone Network Sinet," International Journal of Network Security & Its Applications (IJNSA), vol. 13, no. 6, pp. 23-36, 2021.

[23] D. Shahi, Apache Solr: A Practical Approach to Enterprise Search, ISBN: 978-1-4842-1070-3, 2016.

[24] R. Gao, D. Li, W. Li and Y. Dong, "Application of Full Text Search Engine Based on Lucene," Advances in Internet of Things, vol. 2, no. 4, DOI:10.4236/ait.2012.24013, 2012.

[25] A. Białecki, R. Muir, G. Ingersoll and L. Imagination," Apache Lucene 4," Proc. of SIGIR Workshop on Open Source Inf. Retrieval, [Online], Available: http://opensearchlab.otago.ac.nz/paper_10.pdf, 2012.

[26] B. Lublinsky, K. T. Smith and A. Yakubovich, Professional Hadoop Solutions, ISBN: 978-1-118-61193-7, John Wiley & Sons, 2013.

[27] R. Kuć, Apache Solr 4 Cookbook, ISBN-13: 978-1782161325, Packt Publishing, Ltd., 2013.

[28] B. Abu-Salih, P. Wongthongtham, D. Zhu, K. Y. Chan and A. Rudra, Social Big Data Analytics: Practices, Techniques and Applications, ISBN: 978-981-33-6652-7, Springer Nature, 2021.

[29] B. Abu-Salih, P. Wongthongtham, D. Zhu et al., "Introduction to Big Data Technology," Ch. 2 in Book: Social Big Data Analytics: Practices, Techniques and Applications, pp. 15–59, 2021.

[30] C. Gormley and Z. Tong, ElasticSearch: The Definitive Guide - A Distributed Real-time Search and Analytics Engine, ISBN: 9781449358549, O'Reilly Media, Inc., 2015.

[31] S. Bhandarkar and N. BN, "A Full-text-based Search Algorithm *vs* ElasticSearch," Studies in Indian Place Names, UGC Care Journal, vol. 40, no. 74, pp. 2168–2171, 2020.

[32] Y. Gupta and R. K. Gupta, Mastering Elastic Stack, ISBN-13: 978-1786460011, Packt Pub., 2017.

[33] L. Belcastro, F. Marozzo, D. Talia and P. Trunfio, "Big Data Analysis on Clouds," In Book: Handbook of Big Data Technologies, pp. 101–142, DOI:10.1007/978-3-319-49340-4_4, Springer, 2017.

[34] P. Zadrozny and R. Kodali, Big Data Analytics Using Splunk: Deriving Operational Intelligence from Social Media, Machine Data, Existing Data Warehouses and Other Real-time Streaming Sources, ISBN-13: 978-1430257615, Apress, 2013.

[35] B. P. Sigman and E. Delgado, Splunk Essentials, 2nd Ed., ISBN: 9781785882135 1785882139, Packt Publishing, Ltd., 2016.

[36] T. Hryhorova and O. Moskalenko, "Use of Information Technologies to Improve Access to Information in E-learning Systems," Proc. of the 18th International Conference on Data Science and Intelligent Analysis of Information (ICDSIAI 2018), vol. 836, pp. 206–215, Springer, 2018.

[37] A. Aksyonoff, Introduction to Search with Sphinx: From Installation to Relevance Tuning, ISBN: 9780596809553, O'Reilly Media, Inc., 2011.

[38] A. Ali, Sphinx Search Beginner's Guide, ISBN-13: 978-1849512541, Packt Publishing, Ltd., 2011.

[39] R. Maski, "Using Apache Solr for Ecommerce Search Applications," Happiest Minds, IT Services, pp. 1-12, [Online], Available: https://www.happiestminds.com/whitepapers/using-apache-solr-for-ecommerce-search-applications.pdf, 2013.

[40] V.-A. Zamfir, M. Carabas, C. Carabas and N. Tapus, "Systems Monitoring and Big Data Analysis Using the ElasticSearch System," Proc. of the 22nd IEEE International Conference on Control Systems and Computer Science (CSCS), pp. 188–193, Bucharest, Romania, 2019.

[41] J. Hamilton, B. Schofield, M. G. Berges and J.-C. Tournier, "SCADA Statistics Monitoring Using the Elastic Stack (ElasticSearch, Logstash, Kibana)," Proc. of the Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS2017), pp. 451-455, Barcelona, Spain, 2017.

[42] S. P. Chamarthi S. Prasad and S. Magesh, "Application of Splunk towards Log Files Analysis and Monitoring of Mobile Communication Nodes," International Journal of Applied Science and Engineering Research, vol. 3, pp. 478-483, 2014.

[43] D. Mehta, "Splunk Search Processing Language," In Book: Splunk Certified Study Guide, pp. 27–52, Springer, 2021.

[44] A. Chaudhary, K. Akshatha, K. Kodlekere and S. J. Prasad, "Keyword Based Indexing of a Mmultimdia File," IEEE International Symposium on Multimedia (ISM), pp. 573–576, Taichung, Taiwan, 2017.

[45] P. Kumar, P. Kumar, N. Zaidi and V. S. Rathore, "Analysis and Comparative Exploration of ElasticSearch, Mongodb and Hadoop Big Data Processing," in Book: Soft Computing: Theories and Applications, pp. 605–615, Springer, 2018.

[46] P. Zadrozny and R. Kodali, "Visualizing the Results," in Book: Big Data Analytics Using Splunk, pp. 63–96, Springer, 2013.

[47] K. Venkatesh, M. J. S. Ali, N. Nithiyanandam and M. Rajesh, "Challenges and Research Disputes and Tools in Big-data Analytics," Int. J. of Eng. and Advanced Technol., vol. 6, pp. 1949–1952, 2019.

[48] V. Prajapati, Big Data Analytics with R and Hadoop, ISBN 978-1-78216-328-2, Packt Pub., Ltd., 2013.

[49] F. A. Vadhil, M. L. Salihi and M. F. Nanne, "Toward a Secure ELK Stack," International Journal of Computer Science and Information Security (IJCSIS), vol. 17, no. 7, pp. 139-143, 2019.

[50] Splunkbase, "Home | Splunkbase," [Online], Available: http:splunkbase.splunk.com, [Accessed: Dec. 2021].

ملخص البحث:

إنّ النّمو المتصاعد للبيانات المتولّدة من المحكمة المغربية يجعل من الصّعب البحث عن المعرفة القيّمة في مجموعات البيانات المتعدّدة والضّخمة. وإنّ طرق البحث التقليدية غير منسجمة مع سياق البيانات الضّخمة. وفي الحقيقة، فإنّ البحث عن معلوماتٍ معيّنة في البيانات الضّخمة يتطلّب طرقاً متقدمة وأنظمة بحثٍ عالية الفعالية. وللإسهام في استراتيجية التّحول الرّقمي للمحكمة، نهدف الى تطوير حلّ من شأنه أن يعزّز التّطورات التكنولوجية في هذا المجال. ويتمثّل المشروع الذي نحن بصدد القيام به في تطوير طرقٍ وتقنياتٍ جديدة تتعلّق بالذّكاء الاصطناعي من أجل أتمتة محتوى هائلٍ من البيانات التي ينتجها النّظام القضائي في المملكة المغربية، وتصميم نظامٍ قادرٍ على تحليل كمّ هائلٍ من البيانات القضائية. وهدفنا هو كشف ظواهر معينة قائمة وشرحها واستباط معرفةٍ جديدةٍ من المعلومات التي يتمّ تحليلها؛ من أجل تمييز الأشكال وعمل التّوقعات وإجراء التعديلات اللازمة عند الضّرورة. لذا، فإنّ هدفنا من هذه الدراسة -الأولى من نوعها- هو استقصاء تقنيات البحث والفهرسة القائمة في مجال البيانات الضّخمة وفحصها. ويقارن البحث أبرز الحلول المستخدمة لاسترجاع المعلومات بغية اختيار الحلّ الأمثل من بينها، الذي يصلح لأن يكون أساساً لمحرّك البحث الذي ننوي تطويره للبحث عن البيانات الضّخمة القضائية وفهرستها وتحليلها.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

AN IMPROVED FRACTIONAL TWO-DIMENSIONAL PRINCIPAL COMPONENT ANALYSIS FOR FACE RECOGNITION

Falah Alsaqre

(Received: 25-Nov.-2021, Revised: 15-Jan.-2022, Accepted: 26-Jan.-2022)

ABSTRACT

Two-dimensional principal component analysis (2DPCA) is a subspace technique used for facial image representation and recognition. Standard 2DPCA may be unable to extract informative features to adequately describe the inherent structural information of the original facial images with the presence of irrelevant variations, such as lighting conditions, facial expressions and so on. To deal with this, an improved fractional two-dimensional principal component analysis (IF2DPCA) is proposed in this paper. It is an extension of fractional 2DPCA (F2DPCA), which was developed based on the concept of fractional covariance matrix (FCM). IF2DPCA employs the same principle as F2DPCA for learning a projective matrix, but further extends the use of fractional transformed 2D images throughout the entire recognition task. As a result, the feature subspace modeled by IF2DPCA maintains the most informative content of the 2D face images and is relatively insensitive to irrelevant variations. Experimental results on three face datasets confirm the effectiveness of the suggested IF2DPCA method in facial recognition.

KEYWORDS

Face recognition, Feature extraction, Fractional covariance matrix, 2DPCA, F2DPCA.

1. INTRODUCTION

Face recognition has earned much popularity because of its wide applications in the areas of video surveillance, machine learning and pattern recognition. Among the vast approaches introduced over the years [1]–[4], the most attractive ones are those based on subspace learning techniques [5]. A common paradigm in subspace-based face recognition is to find a subset of features maintaining the informative content of a training set consisting of facial images from distinct classes to be able to correctly assign a class membership to an unknown facial image with the aid of a classifier. Of the subspace learning techniques, the earliest and most widely used is probably the principal component analysis (PCA) [6]–[9]. The PCA procedure consists of mapping high-dimensional input image vectors into a small set of principal components (eigenfaces), describing the most representative content of the input images. Besides, the construction of the eigenfaces is fundamentally dependent on the predominant eigenvectors of the covariance matrix determined from the training image vectors.

Although PCA-based facial recognition methods have exhibited satisfactory recognition accuracy, their formulation requires a preliminary step that unfolds the 2D training images into 1D vectors, inducing high computational cost and loss of inherent structural characteristics of the facial images. To circumvent the implications of image vectorization, two-dimensional principal component analysis (2DPCA) [10] has been developed, wherein the facial images are treated as matrices instead of vectors. The primary purpose of 2DPCA is to create a projective matrix in which the columns are the leading eigenvectors of the covariance matrix evaluated from the row directions of the training instances. In other words, 2DPCA converts each input image to a much smaller feature matrix. This can offer both low computational cost and preservation of the facial structure, so that 2DPCA performs markedly better in most cases than PCA. Following the success of 2DPCA in the representation and recognition of face images, a number of 2DPCA variants have been suggested to improve its performance. Some of them include the bilateral 2DPCA (B2DPCA) [11], horizontal and vertical 2DPCA-based discriminant analysis (HVDA) [12], two-directional two-dimensional 2DPCA ($(2D)^2$PCA) [13], incremental $(2D)^2$PCA (I$(2D)^2$PCA) [14], block-wise $(2D)^2$PCA (B$(2D)^2$PCA) [15] and sequential row-column 2DPCA (RC2DPCA) [16]. The key concept underlying these methods is the projection of face images

F. Alsaqre is with Department of Computer Engineering Techniques, Al Hikma Uni. College, Baghdad, Iraq, Email: alsaqre@ieee.org ORCID: 0000-0002-9510-5264.

onto two (row-wise and column-wise) projection matrices, simultaneously. While this indeed correlates the row-column information and produces far fewer coefficient features than 2DPCA, it generally yields only a slight improvement in recognition accuracy.

Other efforts have concentrated on the adoption of alternative reconstruction error criteria instead of the L2-norm employed in 2DPCA. The representative ones are L1-norm 2DPCA [17], Lp-norm 2DPCA [18], F-norm 2DPCA [19], nuclear-norm 2DPCA [20], R1-norm [21] and Angle-2DPCA [22]. One major advantage of such type of methods is that they perform quite well in image compression. Nonetheless, their solution relies on iteratively evaluating the projective matrices and as such reducing the flexibility of facial recognition. On the other hand, methods like in [23] and [24] extend classical 2DPCA to class-wise 2DPCA (CW2DPCA) to increase the recognition performance. Instead of establishing a holistic projection matrix from the entire training dataset, CW2DPCA builds multiple projective matrices according to the number of classes that constitute the training dataset. However, this results in a longer computational time when dealing with a large training dataset.

Another approach based on the theory of the fractional covariance matrix (FCM) has been introduced to improve the recognition performance of PCA and 2DPCA. The original idea was presented by Gao et al. [25], who replaced the typical covariance matrix in PCA and 2DPCA with an FCM computed from the fractional transformed training images. The new versions of PCA and 2DPCA are named fractional PCA (FPCA) and fractional 2DPCA (F2DPCA), respectively. This approach has three interesting properties. First, adequate selection of the fractional-order to establish the FCM plays a crucial role in the recognition performance. Second, both FPCA and F2DPCA share the same computational complexity as their classical counterparts. However, third, the features subspace is defined in terms of the dominant eigenvectors of the FCM and original training images, which may deteriorate the spatial quality of the captured information. Within this context, to define better projection while avoiding the curse of dimensionality when dealing with image-as-vector, De Carvalho et al. [26] extended FPCA to fractional eigenfaces (FE), where the feature vectors are generated by applying the eigenface technique to the fractional transformed image vectors. Although FE has demonstrated a recognition advantage over FPCA, it still suffers from drawbacks similar to those of 1DPCA-like methods.

It is important to point out that neither [25] nor [26] have provided a clear justification for why the applications of FCM theory in PCA, eigenfaces and 2DPCA could improve face recognition accuracy. In fact, since in these subspace learning techniques, the learned projection matrix maximizes the overall scatter of the entire training samples, they often retain undesirable variations caused by lighting conditions, shadows, facial expressions and so on [6], [27]. Due to this, it makes sense to employ the FCM theory for scaling down, to some extent, the weights of such variations, thereby mitigating their adverse influence on the performance of face recognition. More details can be found in [28], but regarding the modeling of fractional-order singular value decomposition.

Furthermore, Gao et al. [25] demonstrated the superiority of F2DPCA over 2DPCA in terms of the attained recognition accuracy. Later, F2DPCA is performed in the frequency domain to extract texture information [29]. Despite this endeavor, the main drawback with the F2DPCA model is that the original 2D images are directly involved in the calculation of feature matrices. This could pose a remarkable obstacle towards achieving high recognition performance, as the existence of unwanted variations in the original images may affect the facial appearances and may be substantially high in weight, which can result in a large level of uncertainty in the feature matrices. A practical remedy to this drawback is to extract the feature matrices by means of FCM and fractional transformed 2D images.

In this paper, an extension of F2DPCA, termed improved F2DPCA (IF2DPCA), is proposed to enhance the capability of F2DPCA in face recognition tasks. The proposed IF2DPCA is largely inspired by the fractional transformation in FE and the fractional-order covariance matrix in F2DPCA. In mathematical terms, the IF2DPCA determines a projective matrix, based on the FCM theory, to map the fractional transformed 2D images from the image space to the features subspace, such that the measure of the total scatter in the new subspace is maximal. As a consequence, the feature matrices obtained *via* the IF2DPCA model are not only explained by the structural information of the facial images, but also relatively insensitive to the irrelevant variations, leading to better recognition accuracy than either FE or F2DPCA alone.

In the remainder of this paper, background information and related work are presented in Section 2. Section 3 describes the proposed IF2DPCA method for face recognition. Then, experimental results

demonstrating the performance of the IF2DPCA method are reported in Section 4. Finally, conclusions are given in Section 5.

2. BACKGROUND AND RELATED WORK

2.1 PCA

The computational goal of standard PCA [30] is to identify a set of principal component vectors (eigenvectors) from the covariance matrix of the input dataset such that this set characterizes the variations across the dataset samples, in our case the facial images. Formally, let $\boldsymbol{A} = \{\boldsymbol{A}_i\}_{i=1}^{s}$, $\boldsymbol{A}_i \in \mathbb{R}^{m\times n}$, be a face training dataset with s images and its vectorized version is $\boldsymbol{V} = \{\boldsymbol{a}_i\}_{i=1}^{s}$, $\boldsymbol{a}_i \in \mathbb{R}^d$ ($d = mn$). By defining $\overline{\boldsymbol{a}} = 1/s(\sum_{i=1}^{s} \boldsymbol{a}_i)$ as the total mean of the image vectors, the covariance matrix ($\boldsymbol{C}_{PCA}$) of $\boldsymbol{V}$ can be evaluated as:

$$\boldsymbol{C}_{PCA} = \frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{a}_i - \overline{\boldsymbol{a}})\,(\boldsymbol{a}_i - \overline{\boldsymbol{a}})^T \in \mathbb{R}^{d\times d} = \frac{1}{s}\boldsymbol{H}\boldsymbol{H}^T, \tag{1}$$

where $\boldsymbol{H} = \{\boldsymbol{a}_i - \overline{\boldsymbol{a}}\}_{i=1}^{s} \in \mathbb{R}^{d\times s}$. In practice, applying eigenvalue decomposition to the $\frac{1}{s}\boldsymbol{H}\boldsymbol{H}^T$ matrix is infeasible for typical images. As noted in [31], a simpler alternative solution is determining the eigenvalues and eigenvectors of the matrix $\frac{1}{s}\boldsymbol{H}^T\boldsymbol{H} \in \mathbb{R}^{s\times s}$. Now, suppose that $\frac{1}{s}\boldsymbol{H}^T\boldsymbol{H}$ has the eigenvalue-eigenvector pairs $\{(\lambda_i, \boldsymbol{w}_i) : i = 1, 2, \cdots, s\}$, where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_s$ and that $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{s\times k}$ is formed by keeping the k top eigenvectors. It follows that the PCA projection matrix is $\boldsymbol{W}_{PCA} = \boldsymbol{H}\boldsymbol{W} \in \mathbb{R}^{d\times k}$, in which the column vectors are indeed the first k eigenfaces of $\boldsymbol{V}$. With $\boldsymbol{W}_{PCA}$, the image vectors of $\boldsymbol{V}$ can be simply transformed into a set of reduced training feature vectors written as:

$$\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^{s}, \quad \boldsymbol{y}_i = \boldsymbol{W}_{PCA}^T(\boldsymbol{a}_i - \overline{\boldsymbol{a}}) \in \mathbb{R}^k. \tag{2}$$

Furthermore, the classification of a test image vector $\boldsymbol{b} \in \mathbb{R}^d$ is carried out by comparing its corresponding feature vector $\boldsymbol{x} = \boldsymbol{W}_{PCA}^T(\boldsymbol{b} - \overline{\boldsymbol{a}}) \in \mathbb{R}^k$ with the training feature vectors and $\boldsymbol{b}$ ascribes to the class that displays the maximum similarity score. This is usually accomplished through the Euclidean minimum distance procedure.

2.2 Fractional PCA

Fractional principal component analysis (FPCA) [25] is a modified version of PCA. From a formulation perspective, the single difference between standard PCA and FPCA is that the former uses a typical covariance matrix, while the latter utilizes the fractional (r −order) covariance matrix, where $0 < r \leq 1$. Note that when $r = 1$, FPCA becomes equivalent to PCA. In this sense, PCA can be viewed as a special case of FPCA.

Under the FPCA assumptions, the FCM ($\boldsymbol{C}_{FPCA}$) of the training dataset $\boldsymbol{V}$ is defined by:

$$\boldsymbol{C}_{FPCA} = \frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{a}_i^r - (\overline{\boldsymbol{a}})^r)\,(\boldsymbol{a}_i^r - (\overline{\boldsymbol{a}})^r)^T \in \mathbb{R}^{d\times d} = \frac{1}{s}(\boldsymbol{H}^r)(\boldsymbol{H}^r)^T, \tag{3}$$

where $\boldsymbol{a}_i^r = (a_{i1}^r, a_{i2}^r, \cdots, a_{id}^r)^T$ and $\boldsymbol{H}^r = \{\boldsymbol{a}_i^r - (\overline{\boldsymbol{a}})^r\}_{i=1}^{s} \in \mathbb{R}^{d\times s}$.

Like standard PCA, FPCA constructs the projection matrix $\boldsymbol{W}_{FPCA} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{d\times k}$ by staking the k leading eigenvectors of the $\boldsymbol{C}_{FPCA}$. The training feature vectors $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^{s}$ are subsequently obtained using Eq. (2), but replacing $\boldsymbol{W}_{PCA}$ with $\boldsymbol{W}_{FPCA}$ and likewise for the test samples.

As discussed earlier, with respect to Eq. (1), due to the high dimensionality of the typical images, PCA is unable to directly perform the eigenvalue decomposition to $\boldsymbol{C}_{PCA}$. Unfortunately, this intrinsic limitation is also present in FPCA. One way to address this limitation is by employing the fractional eigenfaces (FE) technique [26], which follows the same procedure as the eigenfaces technique, but assumes fractional transformed image vectors. To be specific, let the column vectors of $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{s\times k}$ be the k leading eigenvectors of the matrix $\frac{1}{s}(\boldsymbol{H}^r)^T(\boldsymbol{H}^r) \in \mathbb{R}^{s\times s}$. In this way, one can obtain $\boldsymbol{W}_{FE} = \boldsymbol{H}^r\boldsymbol{W} \in \mathbb{R}^{d\times k}$ as a projection matrix composed by the first k fractional eigenfaces of the dataset $\boldsymbol{V}$. With $\boldsymbol{W}_{FE}$ in hand, the training feature vectors can be calculated by the

following fractional transformation:

$$\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^{s}, \quad \boldsymbol{y}_i = \boldsymbol{W}_{FE}^{T}(\boldsymbol{a}_i^r - (\overline{\boldsymbol{a}})^r) \in \mathbb{R}^k. \tag{4}$$

Moreover, $\boldsymbol{x} = \boldsymbol{W}_{FE}^{T}(\boldsymbol{b}^r - (\overline{\boldsymbol{a}})^r) \in \mathbb{R}^k$ is the feature vector of a particular test sample $\boldsymbol{b} \in \mathbb{R}^d$.

Apart from facilitating the application of FPCA, the FE provides a noticeable enhancement in face recognition accuracy. This can mainly be attributed to the projection of the fractional transformed images in place of the original images in FPCA.

2.3 2DPCA

In [10], Yang et al. introduced 2DPCA, which, unlike PCA, evaluates the covariance matrix using the 2D images without going through the image-vectorization step. As a result, 2DPCA guarantees appropriate preservation of the facial statistical information with low computational cost, hence benefiting the representation and recognition of the facial images.

In the basic formulation, 2DPCA learns a projection matrix such that the overall scatter of the projected training samples is maximized. More concretely, for the training dataset $\boldsymbol{A} = \{\boldsymbol{A}_i\}_{i=1}^{s}$, $\boldsymbol{A}_i \in \mathbb{R}^{m\times n}$, 2DPCA first determines the image covariance matrix ($\boldsymbol{C}_{2DPCA}$) as:

$$\boldsymbol{C}_{2DPCA} = \frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{A}_i - \overline{\boldsymbol{A}})^T(\boldsymbol{A}_i - \overline{\boldsymbol{A}}) \in \mathbb{R}^{n\times n}, \tag{5}$$

where $\overline{\boldsymbol{A}} = 1/s(\sum_{i=1}^{s}\boldsymbol{A}_i)$ denotes the mean image of $\boldsymbol{A}$. After that, the projective matrix $\boldsymbol{W}_{2DPCA} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{n\times k}$ is made with the k top orthonormal eigenvectors of $\boldsymbol{C}_{2DPCA}$. It follows that the projection of each training sample $\boldsymbol{A}_i$ onto $\boldsymbol{W}_{2DPCA}$ makes a set of feature matrices according to the number of training samples; that is:

$$\boldsymbol{Y} = \{\boldsymbol{Y}_i\}_{i=1}^{s}, \quad \boldsymbol{Y}_i = \boldsymbol{A}_i\boldsymbol{W}_{2DPCA} \in \mathbb{R}^{m\times k}. \tag{6}$$

For classification, the feature matrix $\boldsymbol{X} = \boldsymbol{B}\boldsymbol{W}_{2DPCA} \in \mathbb{R}^{m\times k}$ of a test image $\boldsymbol{B} \in \mathbb{R}^{m\times n}$ is matched with the training feature matrices and is given a class membership of its nearest neighbor.

2.4 Fractional 2DPCA

F2DPCA [25] is similar to 2DPCA, except that the covariance matrix is calculated using the fractional transformed 2D images. This implies that, in addition to the inherited properties from 2DPCA, F2DPCA preserves the facial structural information with less impact from the unwanted variations. In more detail, for the training dataset $\boldsymbol{A}$, the FCM ($\boldsymbol{C}_{F2DPCA}$) is computed as:

$$\boldsymbol{C}_{F2DPCA} = \frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{A}_i^r - (\overline{\boldsymbol{A}})^r)^T(\boldsymbol{A}_i^r - (\overline{\boldsymbol{A}})^r) \in \mathbb{R}^{n\times n}. \tag{7}$$

Here, $\boldsymbol{A}_i^r = \left(a_{jl}^r\right)_i$, where $j = 1,2\cdots,m$ and $l = 1,2\cdots,n$. As in 2DPCA, suppose that $\boldsymbol{W}_{F2DPCA} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{n\times k}$ is the F2DPCA projection matrix. And along similar lines, the training feature matrices can be produced using Eq. (6) with $\boldsymbol{W}_{2DPCA}$ replaced by $\boldsymbol{W}_{F2DPCA}$ as follows:

$$\boldsymbol{Y} = \{\boldsymbol{Y}_i\}_{i=1}^{s}, \quad \boldsymbol{Y}_i = A_i\boldsymbol{W}_{F2DPCA} \in \mathbb{R}^{m\times k}. \tag{8}$$

This also applies to the test images. So, for a given test image $\boldsymbol{B}$, the feature matrix is obtained as $\boldsymbol{X} = \boldsymbol{B}\boldsymbol{W}_{F2DPCA} \in \mathbb{R}^{m\times k}$.

3. IMPROVED F2DPCA

Essentially, the proposed IF2DPCA method can be regarded as an appearance-based modeling problem. In the training phase, IF2DPCA generates compact representations of the facial appearances from a set of fractional transformed 2D images. During the testing phase, given an unknown face image, the face identity can be revealed from the compact representations with the aid of a classifier.

3.1 IF2DPCA Formulation

As aforementioned, in the formulation of the F2DPCA model, both the selected eigenvectors of fractional ($r-$order) covariance matrix and the original training samples have participated in the computation of the feature matrices. In the majority of cases, assuming the original images, there is still

a potential for retaining a high level of unwanted information in the projected subspace. Further to this, according to the 2DPCA theory, it is supposed that the projection matrix maximizes the overall scatter of the projected training samples. Arguably, F2DPCA lacks this property. For the sake of dealing with these two concerns, the proposed IF2DPCA method considers the fractional transformed images rather than the original images in the formation of feature matrices. In other words, IF2DPCA and F2DPCA are identical with regard to the definition of the FCM, but differ in the way that they compute the feature matrices. More specifically, IF2DPCA projects the fractional transformed images while F2DPCA projects the raw images.

Let $\boldsymbol{A}^r = \{\boldsymbol{A}_i^r\}_{i=1}^s$, $\boldsymbol{A}_i^r \in \mathbb{R}^{m\times n}$, be the fractional transformed version of the training data $\boldsymbol{A}$ and let $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ be a projection matrix. The projection of each fractional transformed sample into $\boldsymbol{W}$ yields the following projected feature matrices:

$$\boldsymbol{Y}_i = \boldsymbol{A}_i^r \boldsymbol{W} \in \mathbb{R}^{m\times n}, \qquad i = 1, 2, \cdots, s. \tag{9}$$

As indicated in [10], the criterion of the total scatter, $J(\boldsymbol{W})$, can be modeled by means of the trace of the covariance matrix, $\boldsymbol{C}$, of the feature matrices; that is:

$$J(\boldsymbol{W}) = tr(\boldsymbol{C}). \tag{10}$$

In our case, the covariance matrix $\boldsymbol{C}$ is defined by

$$\boldsymbol{C} = \frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{Y}_i - \overline{\boldsymbol{Y}})(\boldsymbol{Y}_i - \overline{\boldsymbol{Y}})^T \in \mathbb{R}^{m\times m} == \frac{1}{s}\sum_{i=1}^{s}[(\boldsymbol{A}_i^r - (\overline{\boldsymbol{A}})^r)\boldsymbol{W}]\,[(\boldsymbol{A}_i^r - (\overline{\boldsymbol{A}})^r)\boldsymbol{W}]^T, \tag{11}$$

where $\overline{\boldsymbol{Y}}$ denotes the mean of the feature matrices. Therefore,

$$tr(\boldsymbol{C}) = \boldsymbol{W}^T\left[\frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{A}_i^r - (\overline{\boldsymbol{A}})^r)^T(\boldsymbol{A}_i^r - (\overline{\boldsymbol{A}})^r)\right]\boldsymbol{W} \in \mathbb{R}^{n\times n} = \boldsymbol{W}^T\boldsymbol{C}_{IF2DPCA}\boldsymbol{W}, \tag{12}$$

where $\boldsymbol{C}_{IF2DPCA} \in \mathbb{R}^{n\times n}$ is the fractional covariance matrix of $\boldsymbol{A}^r$ and it is by default positive semi-definite. Given this, the goal is now to find a set of orthonormal projection vectors, $\boldsymbol{w}_1, \boldsymbol{w}_2 \cdots, \boldsymbol{w}_k$, maximizing $J(\boldsymbol{W})$; that is:

$$\begin{aligned} &\{\boldsymbol{w}_1, \boldsymbol{w}_2 \cdots, \boldsymbol{w}_k\} = \text{argmax}J(\boldsymbol{W}) \\ &s.t.\ \boldsymbol{w}_i^T\boldsymbol{w}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad i,j = 1,2,\cdots,k\,. \end{aligned} \tag{13}$$

This says that these vectors are the k predominant orthonormal eigenvectors of $\boldsymbol{C}_{IF2DPCA}$. Having thus obtained the projection vectors, the IF2DPCA projective matrix can be formed as follows:

$$\boldsymbol{W}_{IF2DPCA} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{n\times k}. \tag{14}$$

The $\boldsymbol{W}_{IF2DPCA}$ is then used to transform each fractional transformed image into a features matrix, creating a set of training feature matrices:

$$\boldsymbol{Y} = \{\boldsymbol{Y}_i\}_{i=1}^s, \quad \boldsymbol{Y}_i = \boldsymbol{A}_i^r\boldsymbol{W}_{IF2DPCA} \in \mathbb{R}^{m\times k}. \tag{15}$$

3.2 Face Classification

After the training phase, the extracted feature matrices are employed for classification. During testing, upon computing the feature matrix $\boldsymbol{X} = \boldsymbol{B}^r\boldsymbol{W}_{IF2DPCA} \in \mathbb{R}^{m\times k}$ of the fractional transform test image $\boldsymbol{B}^r$, the minimum distance between $\boldsymbol{X}$ and $\boldsymbol{Y} = \{\boldsymbol{Y}_i\}_{i=1}^s$is the evidence that the test image belongs to any of the c classes of the training dataset. More specifically, let D_j denote the minimum distance between $\boldsymbol{X}$ and the feature matrices in the jth class, calculated as follows:

$$D_j = \min_{i\in j}(\|\boldsymbol{X} - \boldsymbol{Y}_i\|_F), \qquad j = 1, 2, \cdots, c. \tag{16}$$

Here, $\|\cdot\|_F$ stands for the standard Frobenius norm. Accordingly, the test image is assigned to the class j for which D_j is the minimum among all the classes.

4. EXPERIMENTAL RESULTS

In this section, a set of experiments is presented to confirm the utility of the proposed IF2DPCA model in face recognition. Three public facial datasets (ORL [32], Yale [6] and Georgia Tech [33]) are used to

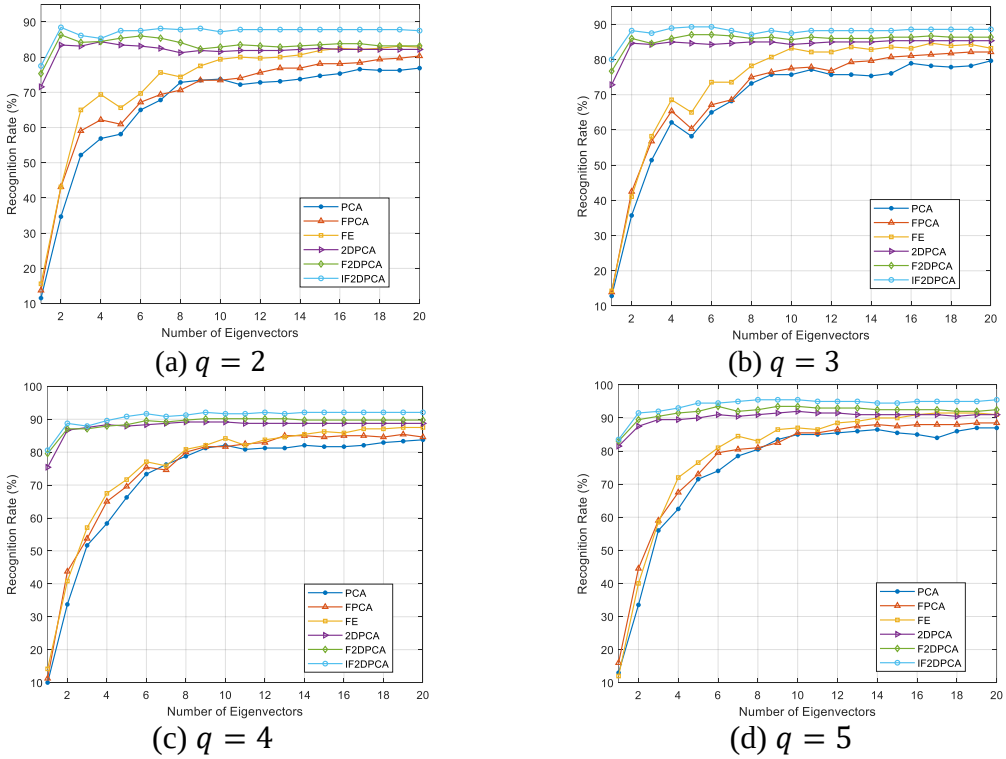
evaluate the recognition performance of IF2DPCA and compare it against PCA [31], FPCA [25], FE [26], 2DPCA [10] and F2DPCA [25]. Throughout experiments, the nearest neighbor classifier is deployed to carry out the classification task. Note that this classifier is based on the Euclidean distance and the Frobenius norm for the 1D and 2D methods, respectively.

4.1 Results on ORL

The ORL facial dataset is made up of 40 classes, each with 10 grayscale images. The samples of a distinct class are collected under different lighting conditions, facial expressions, poses and facial details (such as glasses or no glasses). Within this dataset, all images are resized from 112×92 to 28×23 pixels [25]. Figure 1 shows the samples of one class in the ORL.

Figure 1. Samples of one class in the ORL dataset.

In the experiments, the first q ($q = 2, 3, 4, 5$) images per class are kept to act as the training set and the leftover images compose the testing set. In addition, the value of r is set to 0.01, as this value exhibits the best recognition performance for FPCA-like methods [25] and FE [26] on the ORL dataset. For a certain q, the number of eigenvectors (k) increases form 1 to 20. Under this setting, the size of the learned projection matrices for PCA, FPCA and FE is $644 \times k$ and for 2DPCA, F2DPCA and IF2DPCA is $23 \times k$. Figure 2 depicts the recognition rates of the six methods. Figure 2 a, b, c and d show the results when q is set to 2, 3, 4 and 5, respectively. As observed, compared with other methods, the recognition rate of the IF2DPCA method is the most dominant when considering the same number of eigenvectors in all cases.



(a) $q = 2$ (b) $q = 3$

(c) $q = 4$ (d) $q = 5$

Figure 2. The recognition rates of the IF2DPCA and competitor methods on the ORL dataset.

The experimental results of each method in terms of maximal recognition rate (MRR) and average recognition rate (ARR) are presented in Table 1. Note that the number of eigenvectors corresponding to the achieved MRR is displayed in parentheses. Based on the results shown in Table 1, the IF2DPCA model consistently produced the MRR at the lowest dimension among the tested methods. For example, with $q = 3$, IF2DPCA achieved MRR of 89.28% ($k = 5$), whereas the MRRs for F2DPCA, 2DPCA, FE, FPCA, and PCA are 87.07% ($k = 5$), 85.35% ($k = 15$), 84.64% ($k = 17$), 82.14% ($k = 19$), and 79.64% ($k = 20$), respectively. From this table, it can be seen again that IF2DPCA outperformed the competitors regarding the ARR with the same number of eigenvectors. In the comparison with the FCM-based methods, IF2DPCA showed improvement over F2DPCA, FE, and FPCA by about 2.5%, 14%, and 17.5%, respectively. This set of results emphasizes the benefits of using the fractional transformed images instead of the original ones in F2DPCA.

4.2 Results on Yale

The Yale faces dataset is composed of 165 frontal-view grayscale images representing 15 different classes, where for each class, there are 11 images acquired with various variations in lighting conditions, face expressions and face details. In this group of experiments, the head part of each image is manually cropped and normalized to 40×40 pixels. The cropped images of one class are shown in Figure 3.

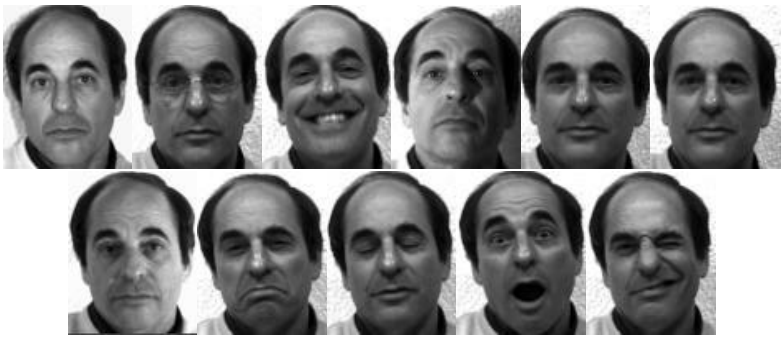
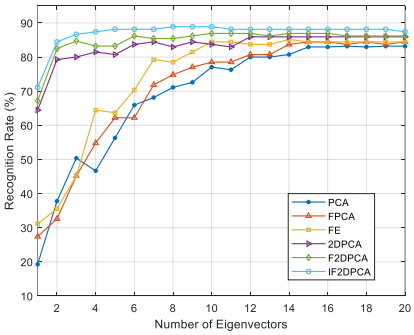


Figure 3. Cropped samples of one person in the Yale dataset.

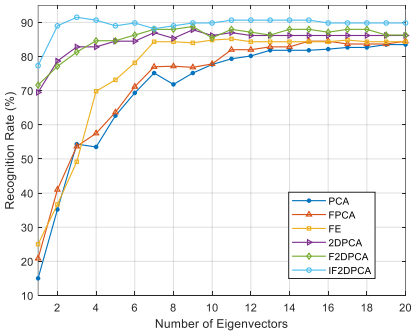
Table 1. The MRR (%) and ARR (%) on the ORL dataset.

	$q = 2$		$q = 3$		$q = 4$		$q = 5$	
Method	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**
PCA	76.87 (20)	65.70	79.64 (20)	67.55	83.75 (20)	71.62	87.00 (19)	74.77
FPCA	80.31 (20)	68.53	82.14 (19)	70.32	85.41 (19)	74.06	88.50 (19)	77.25
FE	83.12 (19)	72.53	84.64 (17)	73.50	87.50 (19)	75.41	91.50 (17)	79.07
2DPCA	84.37 (4)	81.87	85.35 (15)	84.32	89.16 (8)	87.89	92.00 (10)	90.22
F2DPCA	86.31 (2)	83.48	87.07 (5)	85.75	90.16 (9)	88.87	93.50 (6)	91.85
IF2DPCA	88.43 (2)	87.07	89.28 (5)	87.91	92.08 (9)	90.75	95.50 (8)	94.00

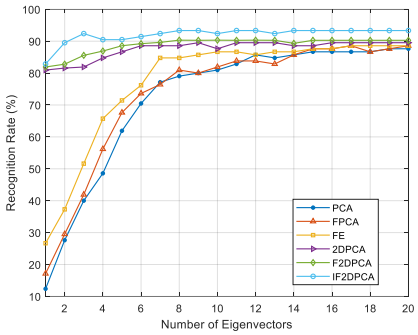
In the following experiments, for each class, the first q ($q = 2, 3, 4, 5$) images are chosen to constitute the training set and the rest are used for testing purposes. For the FCM-based methods, the value of r is assigned to be 0.2 [25]. Further, with each q, the recognition performance of each method is tested by changing the number of eigenvectors (k) from 1 to 20. This implies that the size of the projective matrices for the 1D methods is $1600 \times k$, whereas for the 2D methods, it is $40 \times k$.



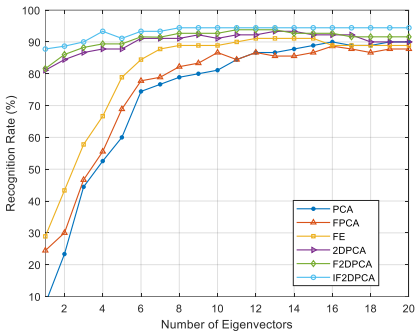
(a) $q = 2$



(b) $q = 3$



(c) $q = 4$



(d) $q = 5$

Figure 4. The recognition rates of the IF2DPCA and competitor methods on the Yale dataset.

The obtained recognition rates of the PCA, FPCA, FE, 2DPCA, F2DPCA and IF2DPCA methods are shown in Figure 4. More specifically, Figures 4a, b, c and d display the recognition rates in the cases that $= 2, 3, 4\ and\ 5$, respectively. Table 2 lists the MRRs and ARRs of the six methods.

Table 2. The MRR (%) and ARR (%) on the Yale dataset.

	$q = 2$		$q = 3$		$q = 4$		$q = 5$	
Method	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**
PCA	83.18 (17)	69.03	83.50 (19)	70.46	87.61 (19)	71.95	90.00 (16)	73.07
FPCA	84.44 (15)	70.77	84.50 (15)	72.54	88.57 (17)	73.40	88.66 (16)	75.10
FE	85.18 (14)	73.88	85.16 (11)	75.70	88.57 (17)	77.30	91.11 (12)	80.67
2DPCA	85.92 (12)	83.07	87.83 (9)	84.58	89.52 (9)	87.60	93.33 (13)	90.11
F2DPCA	86.92 (10)	84.82	88.80 (9)	85.44	90.31 (8)	88.86	93.82 (11)	91.09
IF2DPCA	88.88 (8)	87.03	91.50 (3)	89.33	93.33 (8)	92.04	94.44 (8)	93.27

4.3 Results on Georgia Tech

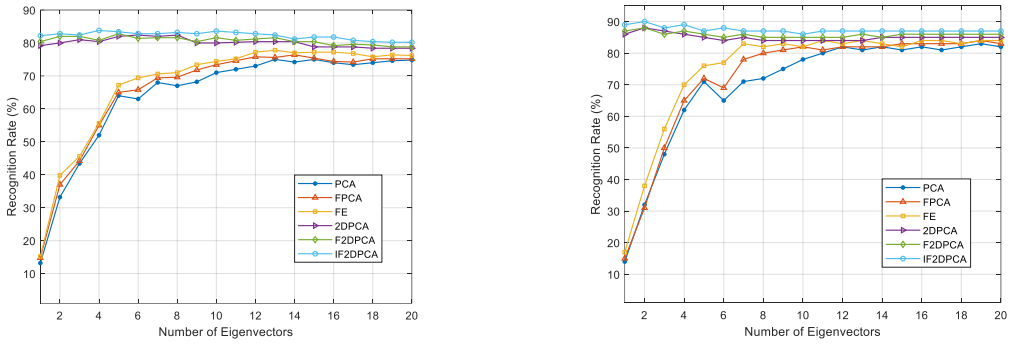
The Georgia Tech face dataset comprises color images of 50 persons, each with 15 facial images taken under different illumination conditions, facial expressions, details and viewpoints. In the experiments, all the images are converted into grayscale and the head part of each image is manually cropped into a size of 50×40 pixels. The cropped images of one person are shown in Figure 5.

For evaluation purposes, the first $q = 10$ and 13 images per person are employed to construct the training set and the others served as testing samples. With this dataset, following the suggestion in [25], the value of r is set to 0.01 based on the cumulative contribution rate of the dominant eigenvalues. As before, for each q, the number of the eigenvectors (k) is increased from 1 to 20. Accordingly, the sizes of the resulting projective matrices for the 1D and 2D methods are $2000 \times k$ and $40 \times k$, respectively.

Figure 5. Cropped images of one person in Georgia Tech face dataset.

The results of the experiments on this dataset are displayed in Figures 6 a and b when $q = 10$ and 13, respectively. As shown, IF2DPCA consistently gives a better recognition rate than any of the five competitor methods. Table 3 reports the achieved MRRs and ARRs by the six methods. As can be seen in the table, with $q = 10$, IF2DPCA reached an MRR of 83.80% at the lowest dimension ($k = 4$) among all the methods. Results from Table 3 also show that IF2DPCA achieved ARRs surpassing those of F2DPCA, 2DPCA, FE, FPCA and PCA by about 1.5%, 2%, 15%, 16% and 18%, respectively.

Furthermore, as shown in the same table, when $q = 13$, the reported MRRs and ARRs of the six methods exhibit the same tendencies as with the previous case. These results further affirm the potential utility of IF2DPCA as an alternative to F2DPCA.



(a) $q = 10$ (b) $q = 13$

Figure 6. The recognition rates of the IF2DPCA and competitor methods on the Georgia Tech dataset.

Table 3. The MRR (%) and ARR (%) on the Georgia Tech dataset.

	$q = 10$		$q = 13$	
Method	**MRR (k)**	**ARR**	**MRR (k)**	**ARR**
PCA	75.00 (13)	64.15	83.00 (19)	70.20
FPCA	76.40 (14)	65.90	84.00 (19)	72.45
FE	77.80 (13)	67.45	84.00 (11)	74.95
2DPCA	82.40 (6)	80.12	88.00 (2)	85.00
F2DPCA	82.80 (5)	80.74	88.00 (2)	85.85
IF2DPCA	83.80 (4)	82.23	90.00 (2)	87.40

4.4 Discussion

As stated in the introduction, the primary objective of this paper is to utilize the theory of FCM and fractional transformed 2D images in order to develop IF2DPCA as an extension of F2DPCA. The key difference between IF2DPCA and F2DPCA lies in the fact that the former uses fractional transformed images throughout the entire recognition task. Judging by the results of experiments on two facial datasets, the recognition rates are consistently better with the feature matrices extracted by IF2DPCA, so it can be considered as a suitable alternative to F2DPCA. The same remark can also be made when comparing FE with FPCA. In other words, among all the competing methods, IF2DPCA offered the best recognition performance in terms of MRR and ARR, whilst FE outperformed its counterpart methods; i.e., FPCA and PCA. Such results strengthen the significance of fractional transformation in the face recognition task as a way to reduce the negative impact of undesirable variations that are present in facial images.

According to the theory of FCM, the value of the order r in FCM-based methods can affect their recognition performances. Moreover, the optimal value of this parameter depends on the characteristics of a particular facial dataset. However, in this work, the values of r for the ORL, Yale and Georgia Tech face datasets are set following the procedure in [25], where these values are empirically derived based on the cumulative contribution rate of the first k eigenvalues.

5. CONCLUSIONS

In this paper, a direct extension of F2DPCA for face recognition, the IF2DPCA, is proposed. The IF2DPCA and F2DPCA methods are conceptually similar, but differ in that the former uses the fractional transformed 2D images not only for evaluating the fractional (r −order) covariance matrix, but also for extracting the feature matrices. With this formulation, the redefined feature matrices capture the inherent characteristics of facial images and are relatively insensitive to undesirable variations. The experiments on the ORL, Yale and Georgia Tech face datasets demonstrate the utility of the suggested method and exhibit that in all cases, IF2DPCA outperforms F2DPCA, 2DPCA, FE, FPCA and PCA in terms of maximal and average recognition rates.

Obviously, the idea of the proposed IF2DPCA method can be easily adapted for other versions of 2DPCA. Another future work may focus on developing a sophisticated algorithm for identifying the optimal value of the fractional order.

REFERENCES

[1] W. Zhao, R. Chellappa, P. J. Phillips and A. Rosenfeld, "Face Recognition : A Literature Survey," ACM Computing Surveys, vol. 35, no. 4, pp. 399–458, 2003.
[2] Y. Kortli, M. Jridi, A. Al Falou and M. Atri, "Face Recognition Systems: A Survey," Sensors (Switzerland), vol. 20, no. 2, pp. 1–36, 2020.
[3] R. Jafri and H. Arabnia, "A Survey of Face Recognition Techniques," Journal of Information Processing Systems (JIPS), vol. 5, pp. 41–68, Jun. 2009.
[4] Z. Mortezaie and H. Hassanpour, "A Survey on Age Invariant Face Recognition Methods," Jordanian Journal of Computers and Information Technology (JJCIT), vol. 05, no. 02, pp. 87–96, 2019.
[5] A. Rao and S. Noushath, "Subspace Methods for Face Recognition," Computer Science Review, vol. 4, no. 1, pp. 1–17, 2010.
[6] P. N. Belhumeur, J. P. Hespanha and D. J. Kriegman, "Eigenfaces *vs.* Fisherfaces: Recognition Using Class Specific Linear Projection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 711–720, Jul. 1997.
[7] M. üg. Çarıkçı and F. Özen, "A Face Recognition System Based on Eigenfaces Method," Procedia Technology, vol. 1, pp. 118–123, 2012.
[8] R. Kaur and E. Himanshi, "Face Recognition Using Principal Component Analysis," Proc. of the IEEE International Advance Computing Conference (IACC), Banglore, India, pp. 585–589, 2015.
[9] M. Slavkovic and D. Jevtic, "Face Recognition Using Eigenface Approach," Serbian Journal of Electrical Engineering, vol. 9, no. 1, pp. 121–130, 2012.
[10] J. Yang, D. Zhang, A. F. Frangi and J. Y. Yang, "Two-dimensional PCA: A New Approach to Appearance-based Face Representation and Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 1, pp. 131–137, 2004.
[11] H. Kong, L. Wang, E. K. Teoh, X. Li, J.-G. Wang and R. Venkateswarlu, "Generalized 2D Principal Component Analysis for Face Image Representation and Recognition," Neural Networks, vol. 18, no. 5, pp. 585–594, 2005.
[12] J. Yang and C. Liu, "Horizontal and Vertical 2DPCA-based Discriminant Analysis for Face Verification on a Large-scale Database," IEEE Transactions on Information Forensics and Security, vol. 2, no. 4, pp. 781–792, 2007.
[13] D. Zhang and Z.-H. Zhou, "(2D)2PCA: Two-directional Two-dimensional PCA for Efficient Face Representation and Recognition," Neurocomputing, vol. 69, no. 1, pp. 224–231, 2005.
[14] Y. Choi, T. Tokumoto, M. Lee and S. Ozawa, "Incremental Two-dimensional Two-directional Principal Component Analysis (I(2D)2PCA) for Face Recognition," Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1493–1496, Prague, Czech Republic, 2011.
[15] A. Mashhoori and M. Z. Jahromi, "Block-wise Two-directional 2DPCA with Ensemble Learning for Face Recognition," Neurocomputing, vol. 108, pp. 111–117, 2013.
[16] W. Yang, C. Sun and K. Ricanek, "Sequential Row: Column 2DPCA for Face Recognition," Neural Computing & Applications, vol. 21, no. 7, pp. 1729–1735, Oct. 2012.
[17] X. Li, Y. Pang and Y. Yuan, "L1-Norm-based 2DPCA," IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), vol. 40, no. 4, pp. 1170–1175, 2009.
[18] J. Wang, "Generalized 2-D Principal Component Analysis by Lp-Norm for Image Analysis," IEEE Transactions on Cybernetics, vol. 46, no. 3, pp. 792–803, 2016.
[19] Y. Wang and Q. Li, "Robust 2DPCA with F -Norm Minimization," IEEE Access, vol. 7, pp. 68083–68090, 2019.
[20] F. Zhang, J. Yang, J. Qian and Y. Xu, "Nuclear Norm-based 2-DPCA for Extracting Features from Images," IEEE Trans. on Neural Networks and Learning Systems, vol. 26, no. 10, pp. 2247–2260, 2015.
[21] Q. Gao, S. Xu, F. Chen, C. Ding, X. Gao and Y. Li, "R1-2-DPCA and Face Recognition," IEEE Transactions on Cybernetics, vol. 49, no. 4, pp. 1212–1223, 2019.
[22] Q. Gao, L. Ma, Y. Liu, X. Gao and F. Nie, "Angle 2DPCA : A New Formulation for 2DPCA," IEEE Transactions on Cybernetics, vol. 48, no. 5, pp. 1672–1678, 2018.
[23] C. Guzel Turhan and H. S. Bilge, "Class-wise Two-dimensional PCA Method for Face Recognition," IET Computer Vision, vol. 11, no. 4, pp. 286–300, 2016.
[24] F. Alsaqre, "Human Face Recognition Using Class-wise Two-dimensional Principal Component Analysis," Int. J. Comput. Digit. Syst. (IJCDS), vol. 9, no. 2, pp. 335–343, 2020.
[25] C. Gao, J. Zhou and Q. Pu, "Theory of Fractional Covariance Matrix and Its Applications in PCA and 2D-PCA," Expert Systems with Applications, vol. 40, no. 13, pp. 5395–5401, 2013.
[26] T. B. A. De Carvalho, M. A. A. Sibaldo, I. R. Tsang et al., "Fractional Eigenfaces," Proc. of the IEEE

International Conference on Image Processising (ICIP 2014), no. 1, pp. 258–262, Paris, France, 2014.
[27] X. Yang, W. Wang, L. Liu, Y. Shao, L. Zhang and N. Deng, "Robust 2DPCA by Tℓ Criterion Maximization for Image Recognition," IEEE Access, vol. 9, pp. 7690–7700, 2021.
[28] J. Liu, S. Chen and X. Tan, "Fractional Order Singular Value Decomposition Representation for Face Recognition," Pattern Recognition, vol. 41, no. 1, pp. 378–395, 2008.
[29] T. H. Le, H. P. Truong, H. T. T. Do and D. M. Vo, "On Approaching 2D-FPCA Technique to Improve Image Representation in Frequency Domain," Proc. of the 4th Symposium on Information and Communication Technology (SoICT '13), pp. 172–180, DOI: 10.1145/2542050.2542061, 2013.
[30] A. Tharwat, "Principal Component Analysis: A Tutorial," International Journal of Applied Pattern Recognition, vol. 3, no. 3, p. 197, 2016.
[31] M. Turk and A. Pentland, "Eigenfaces for Recognition," Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, Jan. 1991.
[32] F. S. Samaria and A. C. Harter, "Parameterisation of a Stochastic Model for Human Face Identification," Proc. of the IEEE Workshop on Appl. of Computer Vision, 1994, pp. 138–142, Sarasota, FL, USA, 1994.
[33] Georgia Tech, "Georgia Tech Face Database," [Online], Available: http://www.anefian.com/research/face_reco.htm.

ملخص البحث:

تعدّ طريقة تحليل المكونات الرئيسية ذي البُعديْن من التّقنيات ذات الحيّز الفرعي المستخدمة في تمثيل الوجوه وتمييزها. ولربّما كان النّمط المعياري من هذه التقنية (2DPCA) غير قادرٍ على استخلاص سِمات المعلومات لِيَصِفَ على نحوٍ كافٍ المعلومات البنيوية الكامنة في الصّور الأصلية للوجوه في ظلّ وجود تغيّراتٍ في ظروف الإضاءة وتعابير الوجوه وما الى ذلك.

ولمعالجة ذلك القصور، تقترح هذه الورقة طريقة محسَّنة تقوم على التّحليل الجزئي ذي البُعديْن المستند الى تحليل المكوّنات الرئيسية (IF2DPCA)، وهي بمثابة امتداد للتّحليل الجزئي ذي البُعدْين باستخدام تحليل المكونات الرئيسية (F2DPCA) الذي تمّ تطويره بناءً على مصفوفة التّغيّرات المشتركة الجزئية (FCM). وتستخدم الطريقتان (IF2DPCA) و (F2DPCA) المبدأ نفسه لتعلُّم مصفوفة إسقاط، إلاّ أنّ الطريقة المقترحة توسّع من استخدام الأجزاء المتحولة من الصّور ذات البُعدْين عبر مهمة التّمييز بكاملها. ونتيجةً لذلك، فإنّ الحيّز الفرعي المُنَمْذَج في الطّريقة المحسّنة المقترحة يحافظ على المحتوى الذي ينطوي على أهمّ المعلومات في صُور الوجوه ذات البُعدْين، وتصبح عملية التّمييز أقلّ حساسية للتّغيُّرات في ظروف الإضاءة وتعابير الوجه وغيرها.

وقد تمّت تجربة الطّريقة المحسّنة المقترحة على ثلاثٍ من مجموعات البيانات؛ إذ أثبتت فعاليتها في تمييز الوجوه.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

DESIGN METHODOLOGY FOR NARROW-BAND LOW NOISE AMPLIFIER USING CMOS 0.18 μM TECHNOLOGY

Raya O. Jaradat[1], Fadi R. Shahroury[2], Hani H. Ahmad[2] and Ibrahim Abuishmais[2]

(Received: 22-Nov.-2021, Revised: 6-Jan.-2022 and 19-Jan.-2022, Accepted: 28-Jan.-2022)

ABSTRACT

This paper presents a design methodology for a fully integrated narrow-band low noise amplifier (LNA). To demonstrate the effectiveness of the proposed methodology, an LNA for Wi-Fi and Bluetooth standards at 2.4 GHz is conducted. The design circuity is implemented using 0.18 μm TSMC CMOS technology; however, the methodology can be equally applied to any process node. Optimum transistor sizing and biasing to achieve minimum noise figure (NF) and maximum power gain without violating the specified power budget are attained by this methodology. It also specifies the criteria for choosing the on-chip RF inductors based on the quality factor, self-resonance frequency and area. The demonstrated LNA design achieves a power gain (S_{21}) of 22.75 dB, an input return loss (S_{11}) of –30.11 dB, a reverse isolation (S_{12}) of –60.49 dB and an output return loss (S_{22}) of –11.23 dB. The linearity parameters of the $P_{1\text{-}dB}$ compression point and IIP_3 are –19 dBm and –13.5 dBm, respectively. It produces an NF of 1.75 dB while consuming 6.16 mW from a 1.8 V power supply.

KEYWORDS

Design methodology, Front-end receiver, LNA, Wireless network, Bluetooth, IEEE 802.11 b, IEEE 802.15.1, 0.18 μm CMOS.

1. INTRODUCTION

Wi-Fi and Bluetooth are ubiquitous among almost all modern devices, as they are used to connect users to the Internet and other peripheral devices. There is a need for high-performance, low-power and low-cost devices to keep up with the customers' need for battery-operated devices. CMOS technology has been very popular in building these devices due to its superior capability of integration, low cost, low power and accessibility. The design of Wi-Fi and Bluetooth front-end receivers is challenging due to several opposing requirements. One of the main building blocks of the front-end receiver is the low noise amplifier (LNA), as it determines the noise figure (NF) and hence the overall sensitivity of the receiver.

The key performance metrics in designing the LNA include NF, power gain, linearity, input-output impedance-matching circuits, stability and reverse isolation. Indeed, all these design parameters are equally critical, but there are unavoidable trade-offs among them when attempting to build an optimum LNA [1]-[3]. Due to this complex inter-relationship among all these entities, there is a need for a simple and precise methodology to obtain optimal performance for a given LNA.

There is a wealth of prior art that has been published optimizing the design of LNA and its performance parameters using different topologies and different techniques. However, there is a lack of a simple and comprehensive design methodology that guides the RF designer through different steps, starting from a set of design specifications to the optimal design parameters [4]-[9].

This paper presents a simple, concise and comprehensive design methodology for narrow-band LNA. The proposed design methodology specifically explains the topology selection with detailed analysis. In addition, the methodology uses g_m/I_D technique to select the optimum transistor sizing and biasing to achieve minimum NF and maximum power gain without violating the specified power budget. It also specifies the criteria for choosing the on-chip RF inductors based on the quality factor, self-resonance frequency and area. Although the methodology utilized the 0.18 μm process node, it can be easily extended to any other process node.

1. Raya O. Jaradat is an Electronics Engineer, Princess Sumaya University for Technology, Amman, Jordan. Email: Ray20178004@std.psut.edu.jo
2. F. R. Shahroury (ORCID:0000-0001-8502-3946), H. H. Ahmad and I. Abuishmais are with Department of Electrical Engineering, Princess Sumaya University for Technology, Amman, Jordan. Emails: fadi@psut.edu.jo, h.ahmad@psut.edu.jo and i.abuishmais@psut.edu.jo

This paper is organized as follows: Section 2 presents the design methodology for narrow-band LNA operated at 2.4 GHz. In Section 3, the simulation results are presented and compared with those of some recently reported works. Finally, conclusions are given in Section 4.

2. DESIGN METHODOLOGY

This section depicts the proposed design methodology demonstrated with LNA for Wi-Fi and Bluetooth. The flowchart illustrated in Figure 1 summarizes the steps to be followed in designing a narrow-band LNA with optimal performance parameters. The methodology starts with the targeted design specifications and ends up with optimal design parameters. The main steps of the flow charts are given in the following sub-section and Section 3.

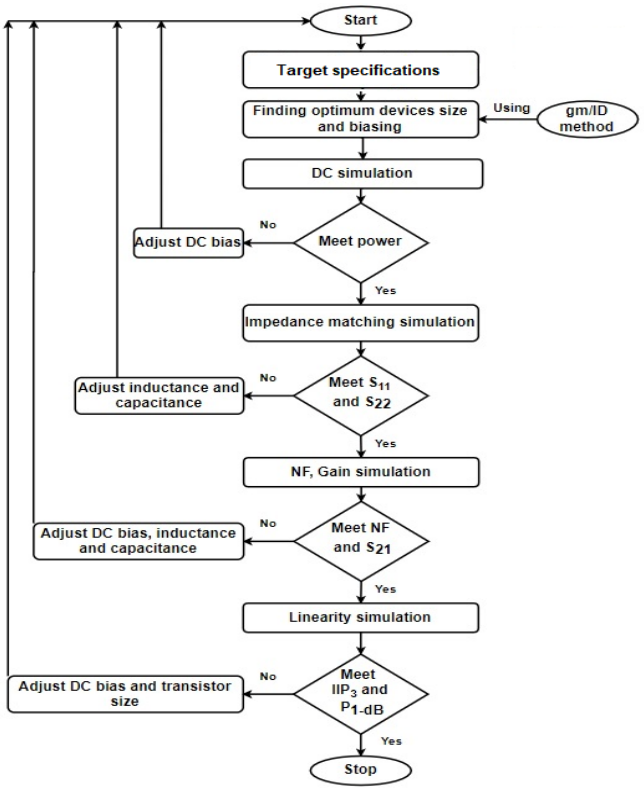


Figure 1. Proposed design methodology flowchart.

2.1 Target Specifications

As the first order of business, when designing an LNA, it seems appropriate to know the target specifications. This is done in terms of several various parameters, like noise, gain, linearity, input and output reflection coefficients. These parameters differ from one application to another. The demonstrated LNA is targeting Wi-Fi, Bluetooth and Zigbee applications. The target design specifications for each application are summarized in Table 1 below based on [4], [10]-[14].

Table 1. LNA design specifications.

Performance parameters	**Bluetooth @ IEEE 802.15.1**	**Wi-Fi @ IEEE 802.11 b**
Noise Figure (NF)	< 3.5 dB	< 3.5 dB
Power Gain (S_{21})	> 15 dB	> 15 dB
1dB Compression Point	> -20 dBm	> -20 dBm
IIP_3	> -15 dBm	> -15 dBm
Input Reflection Coefficient (S_{11})	< -10 dB	< -10 dB
Output Reflection Coefficient (S_{22})	< -10 dB	< -10 dB
Reverse Isolation Coefficient (S_{12})	< -40 dB	< -40 dB
Power	< 10 mw	< 10 mw

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

2.2 Topology Selection

The second step is to choose the topology that best suits the design specifications for the targeted applications. The IDCCS topology has been chosen for several reasons that will be discussed in detail in this section.

2.2.1 Inductively Degenerated Cascoded Common Source (IDCCS) LNA

As a starting point, a simple CS LNA is selected, as shown in Figure 2. The input impedance is dominated by the gate-source capacitance (C_{gs}). So, it is hard to achieve a pure real impedance without an input impedance matching network.

One of the input impedance matching networks that may be used is a parallel resistor at the input to match with the source resistor, but this is not an optimal solution. Although it may provide the real part of impedance, the imaginary part still exists due to CMOS parasitic capacitance. Also, it adds thermal noise that increases the overall LNA NF.

One of the best solutions to achieve real impedance matching without using a resistor and improve the LNA NF is IDCS topology [4], [15]. IDCS topology uses gate and source inductor to match input impedance, as shown in Figure 3. Gate inductor (L_g) dominates the input capacitance of CMOS (C_{gs}) by forming a series LC circuit at the desired frequency; source inductor (L_s), input transistor transconductance (g_m) and C_{gs} of IDCS produce a real impedance to match the source impedance without using a resistor.

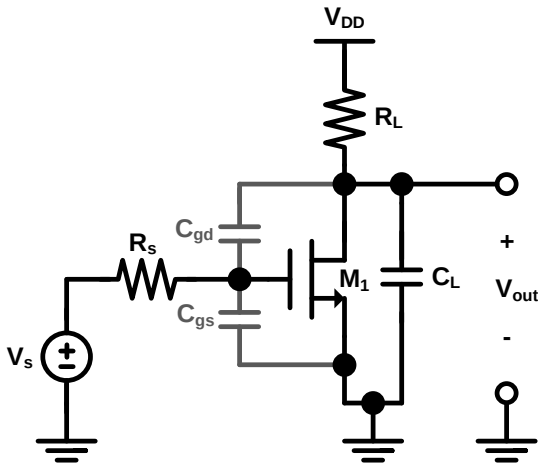


Figure 2. Simple common source CMOS LNA.

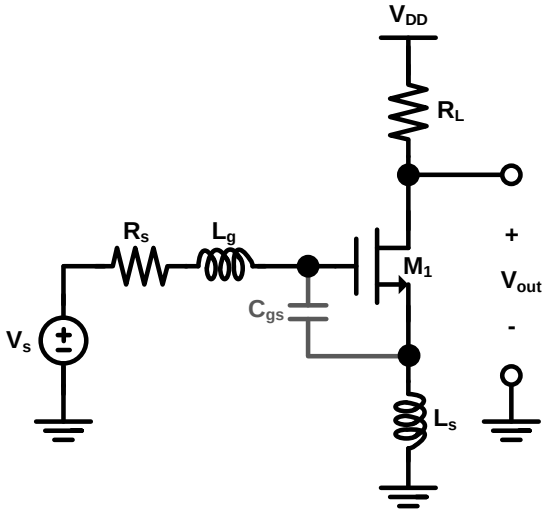


Figure 3. IDCS LNA topology.

2.2.2 Inductively Degenerated Cascoded Common Source (IDCCS) LNA

A small-signal equivalent circuit of the input stage of IDCS CMOS LNA is shown in Figure 4, where the input impedance is Z_{in}.

Applying KVL in the input loop of the circuit in Figure 4, Z_{in} can be found by the following Equations:

$$V_{in} = I_{in} * SL_g + \frac{I_{in}}{SC_{gs}} + (g_m * V_{gs} + I_{in})SL_s \quad (1)$$

where:

$$V_{gs} = \frac{I_{in}}{SC_{gs}} \quad (2)$$

By substituting Equation (2) into Equation (1), we have:

$$V_{in} = I_{in} * SL_g + \frac{I_{in}}{SC_{gs}} + (g_m * \frac{I_{in}}{SC_{gs}} + I_{in})SL_s \quad (3)$$

Then, the input impedance can be found as:

$$Z_{in} = \frac{V_n}{I_{in}} = S(L_s + L_g) + \frac{1}{SC_{gs}} + \frac{g_m * L_s}{C_{gs}} \quad (4)$$

By replacing $s = j\omega$, we have:

$$Z_{in} = \frac{V_n}{I_{in}} = j\omega(L_s + L_g) + \frac{1}{j\omega C_{gs}} + \frac{g_m * L_s}{C_{gs}} \quad (5)$$

The real and imaginary parts of the input impedance are shown in Equation (6) and Equation (7), respectively:

$$Re(Z_{in}) = \frac{g_m * L_s}{C_{gs}} \quad (6)$$

$$Im(Z_{in}) = j\omega(L_s + L_g) + \frac{1}{j\omega C_{gs}} \quad (7)$$

The input impedance of IDCS LNA is similar to the series RLC circuit shown in Figure 5, where the first term and second term shown in Equation (5) are inductive and capacitive, respectively, while the third term is a resistive impedance.

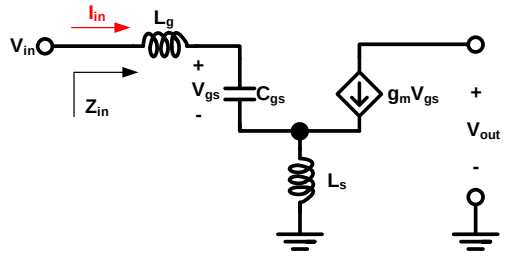


Figure 4. Small-signal equivalent circuit of the input stage of IDCS LNA.

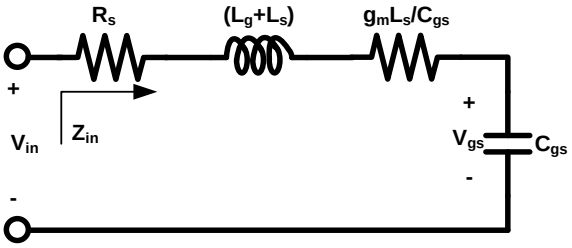


Figure 5. Input equivalent circuit of IDCS LNA.

So, IDCS LNA achieves a resistive input impedance without using a resistor. Generating a resistive impedance using a degenerative source inductor does not generate as much noise as a real resistor; it helps minimize the LNA NF.

To match the input impedance with the source impedance (Z_o) that is usually equal to 50Ω, the real part of Z_{in}, $Re(Z_{in})$ should be equal to 50 Ω. It depends on transconductance, source inductance and C_{gs}. Also, the imaginary part of Z_{in}, $Im(Z_{in})$, should be equal to zero, which occurs only at a particular frequency. This frequency is called the resonance frequency $\omega_\circ$, where $\omega_\circ$ is expressed as in Equation (8) [16].

$$\omega_\circ = \frac{1}{\sqrt{(L_s + L_g)(C_{gs})}} \quad (8)$$

At other frequencies, we can't get pure resistive matching. It is worthy to note that the desired matching impedance can be achieved at a particular frequency. Thus, the IDCS topology is proposed for NB LNA.

2.2.3 IDCS Output Impedance Analysis

Generally, in the NB topology, LNA has an inductive load instead of a resistive load, as shown in Figure 6. There are several reasons to use an inductor instead of a resistor on the load. The first one is to minimize the NF, noting that the resistor will add thermal noise. The second one is to maximize the voltage swing, because in DC behaviour, the frequency is zero and the inductor will act as a short circuit which maximizes the voltage swing, while in the resistor case, the voltage swing will be decreased by the amount of DC voltage drop across the resistor. The third one is that, in AC behaviour, the LNA will act as a Low Pass Filter (LPF) in the resistor case, while in the inductor case, it will act like a Band Pass Filter (BPF) at certain frequencies.

To explain how the IDCS LNA with load inductor (L_d) acts as a BPF, we should mention that L_d has a parasitic resistance modelled as a small series resistor (R_{ds}), as shown in Figure 7. a, where the value of R_{ds} depends on the inductor value and its quality factor. For further discussion, the quality factor (Q) must be introduced.

Q is the parameter used to discover the characteristics and performance of the circuit. The most basic definition is embodied in Equation (9) [17]-[18].

$$Q = \omega \times \frac{\text{Energy Stored}}{\text{Average Power Dissipated}}, \text{ where } \omega \text{ is the angular frequency.} \quad (9)$$

Also, Q can be used as a measure of how lossy the component is, as shown in the Equations below:

$$Q_L = \frac{X_L}{R} = \frac{\omega * L}{R} \quad (10)$$

$$Q_c = \frac{|X_c|}{R} = \frac{1}{\omega C R} \quad (11)$$

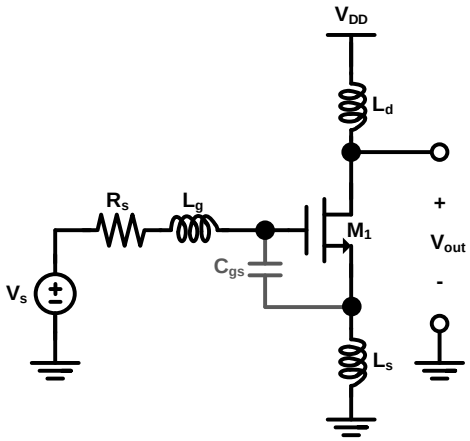


Figure 6. IDCS LNA with load inductor.

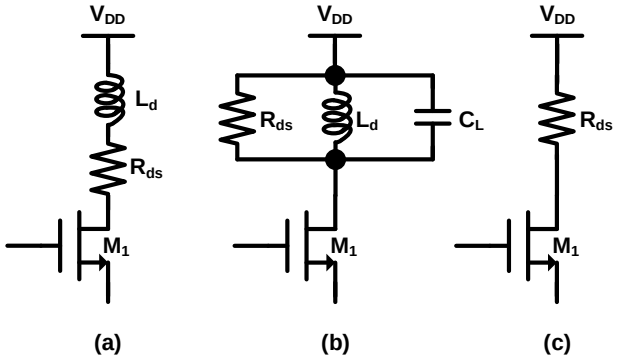


Figure 7. LNA with load inductor (L_d). a) L_d with its series resistor (R_{ds}). b) L_d as a part of parallel RLC circuit. c)LNA load at the resonance frequency (ω_o).

From Equation (10), The value of R_{ds} can be found as expressed in Equation (12).

$$R_{ds} = \frac{L_d * \omega}{Q_{Ld}} \quad (12)$$

where R_{ds} is a series resistor, as shown in Figure 7. (a), L_d is the drain inductor and Q_{Ld} is the quality factor of the inductor L_d.

As shown in Figure 7(a), we have a series (RL) circuit on the load, where the series (RL) circuit can be converted into a parallel (RL) circuit using Equation (13) that shows the quality factor for a parallel (RL) circuit.

$$Q_{Ld} = \frac{R_{dp}}{L_d * \omega} \text{ , then } R_{dp} = Q_{Ld} * \omega * L_d \quad (13)$$

where R_{dp} is the parallel resistor, as shown in Figure 7 (b).

The value of R_{dp} depends on the L_d value and its quality factor and angular frequency. After the conversion, we have a parallel RLC circuit in the load, as shown in Figure 7 (b), L_d and R_{dp} have formed a parallel tuned circuit with C_L, where C_L represents the LNA parasitic capacitance and the input capacitance for the next circuit after the LNA, which might be a mixer or a buffer circuit.

To match the output impedance to 50 Ω, R_{dp} should be equal to 50 Ω and L_d should resonate with C_L at the operating frequency (output resonance frequency), which can be expressed as shown in Equation (14):

$$\omega_o = \frac{1}{\sqrt{L_d C_L}} \quad (14)$$

2.2.4 Cascode IDCS

IDCS topology has bad isolation between input and output due to the Miller capacitor. To solve this problem, cascode IDCS LNA, as shown in Figure 8, has been used to improve the reverse isolation between input and output by reducing the effect of the Miller capacitor, which may cause instability. Transistor M_2 has a minor influence on the noise behaviour of the LNA and its contribution to the total noise can be disregarded [19], because the source of the cascode transistor is connected to a large resistance (M_1 output resistance).

The cascode IDCS has been selected for the designed LNA, since it can achieve narrow-band matching at the input and output at the operating frequency. Also, it can achieve a high gain, minimum noise and high reverse isolation.

2.3 Selecting Device Size and Biasing Voltages

The next step is to find the optimum device size and biasing voltages to obtain the minimum NF and maximum gain by considering the power specification for the demonstrated LNA. The g_m/I_D method is used to find the optimum device size and biasing voltages, because there's a disconnection between the actual transistor behavior and the simple square-law model [20]. Any square law-driven design optimization will be far from the simulation results.

The solution to solve this problem is to begin the design with precomputed simulation data using hand calculations. To achieve our goal in maintaining a systematic design methodology in the absence of a set of functional compact MOS equations, a strategy of using lookup tables or charts is followed. These tables or charts are obtained from the technology characterization via DC Sweep simulation of the transistor model, as shown in Figure 9.

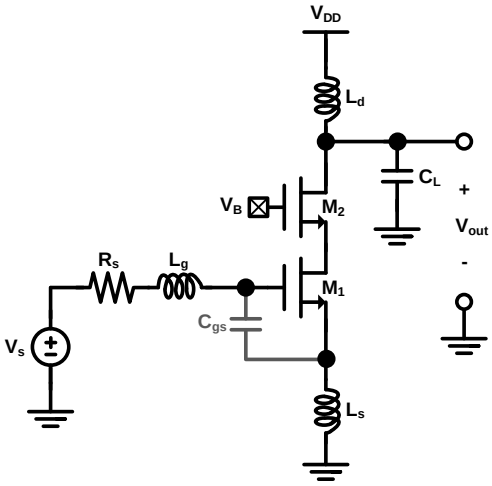


Figure 8. Cascode IDCS LNA.

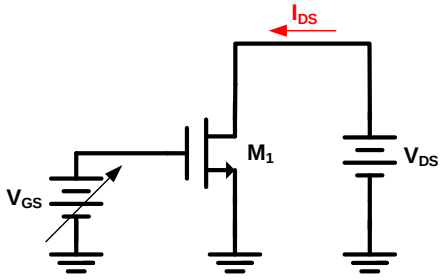


Figure 9. CMOS test circuit for acquiring g_m/I_D charts.

To find the optimum device size and biasing voltages for minimum NF and maximum gain from the resulting charts, we should discuss Equation (15) that expresses the noise factor in general [2], [10], [21].

$$F = F_{min} + \frac{R_n}{G_s}\left|Y_s - Y_{opt}\right|^2 \tag{15}$$

where F_{min} is the minimum noise factor as expressed in Equation (16), R_n is the noise resistance, G_s is the source conductance, Y_s and Y_{opt} are the source admittance and the optimum admittance, respectively.

$$F_{min} = 1 + \frac{2}{\sqrt{5}}\frac{\omega}{\omega_t}\sqrt{\gamma\delta(1-|c|^2)} \tag{16}$$

To achieve minimum NF, we should satisfy the two parts of Equation (15). First, we should achieve minimum noise from the core transistor by finding the optimum value for biasing voltages and sizing. Second, F_{min} can be achieved by the input matching circuit by making the source admittance (Y_s) equal to the optimum source admittance (Y_{opt}). To obtain minimum NF from the core transistor, we should increase the value of cut-off frequency (f_t). From the first resulting chart between f_t and g_m/I_D shown in Figure 10, we see that as long as g_m/I_D is low, f_t will be high and NF is low, but as long as g_m/I_D is high, the gain will be high as well. There is always a trade-off between NF and gain, noting that we need optimum values for both of them.

The y-axis in Figure 10, shows the cut-off frequency ($f_t=g_m/Cgs$) and the x-axis shows the gm-over-ID value (g_m/I_D).

Any increase in the V_{od} value will cause an increase in f_t and a decrease in the NF value, since $V_{od}=2/(g_m/I_D)$, but at the same time, it will cause the power to be increased and the gain to be decreased.

By multiplying f_t with g_m/I_D, the resulting peak value will be the best point to choose the optimum value of V_{od} for maximum gain and minimum noise without increasing the power, as shown in Figure 11, where the y-axis shows the multiplication between cut-off frequency and g_m/I_D (FOM=ft*gm_over_ID) and the x-axis shows the overdrive voltage (V_{OD}).

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

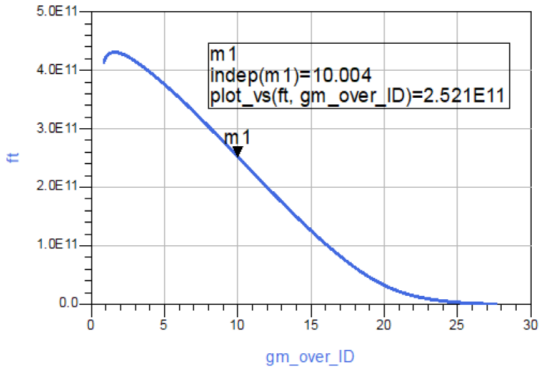


Figure 10. f_t *versus* g_m/I_D for an nMOS transistor.

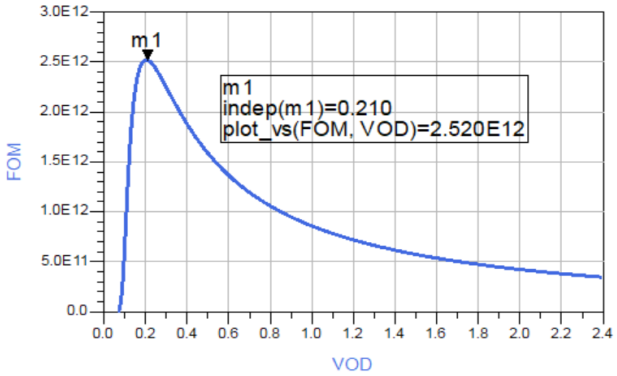


Figure 11. (f_t g_m/I_D) *versus* V_{od} for an nMOS transistor.

From the value of V_{od}, the optimum value of g_m/I_D can be found, where $g_m/I_D = 2/ V_{od}$. After finding the optimum value of g_m/I_D, the optimum values of V_{gs} and current density (I_D/W) can be found in Figures 12 and 13, respectively. Figure 13 shows that the optimum values for g_m/I_D, V_{gs} and I_D/W are 10, 0.65 V and 29.73, respectively.

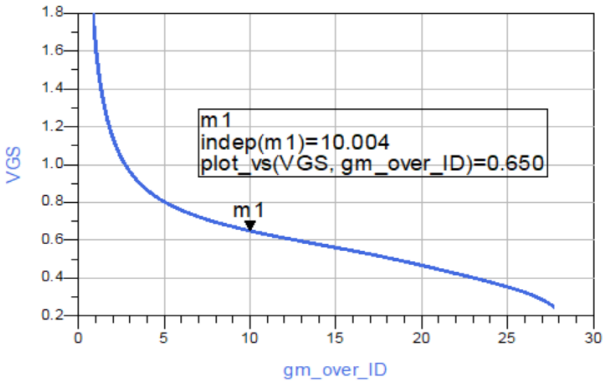


Figure 12. V_{gs} versus gm/I_D for an nMOS transistor.

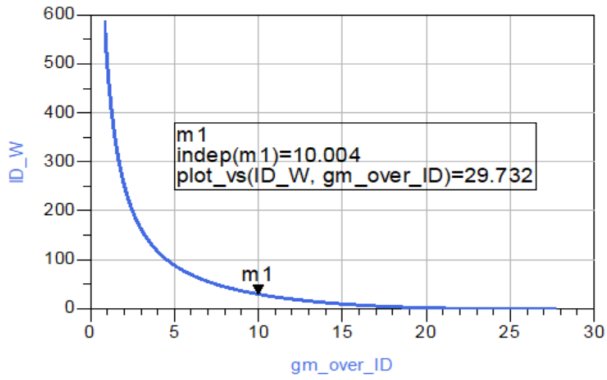


Figure 13. (I_D/W) versus g_m/I_D for an nMOS transistor.

The value of I_D has been selected based on the power budget for each standard. For example, for Wi-Fi and Bluetooth standards (when operated at 2.4 GHz), the maximum power is 10 mW, where power $=I_D*V_{DD}$. In the design circuit, the value of V_{DD} that has been used is 1.8V and the maximum value of I_D is 5.5 mA. After finding the I_D value, from the optimum I_D/W value that was found previously, we can now obtain the optimum device size for each standard for minimum NF, maximum gain and specific power.

2.4 Selecting RF Inductor

The values of RF inductors (Lg, Ls and Ld) should be chosen carefully based on several factors:

1) Inductor value at the operating frequency (2.4 GHz).
2) Inductor quality factor at the operating frequency, as it affects NF and power gain.
3) Inductor self-resonance frequency (SRF). It should be far from the operating frequency, because at SRF, the inductor resonates with its parasitic capacitance [16]. Figure 14 shows how the inductor acts at its SRF while the input impedance is at its peak and the effective inductance is zero, since the negative capacitance reactance ($X_C=1/j\omega c$) cancels the inductive reactance ($X_L = j\omega L$).

All the TSMC 0.18-µm PDK inductors were being tested at the operating frequency and the results are summarized in Table 2. To choose the best inductor with the highest quality factor, the SRF should be far away from the operating frequency, w is the inductor width, rad is the inductor radius, nr is the number of turns and Q is the inductor quality factor.

Figures 15 and 16 show the tested inductor value and its quality factor at the operating frequency.

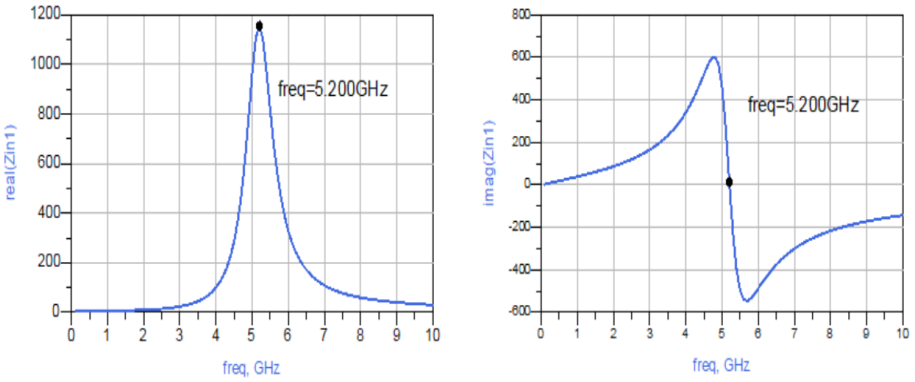


Figure 14. Representing how the inductor acts at its self-resonance frequency (SRF).

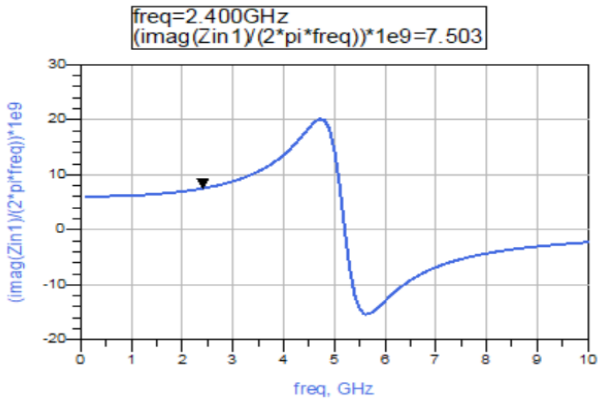


Figure 15. Representing the tested inductor value at the operating frequency.

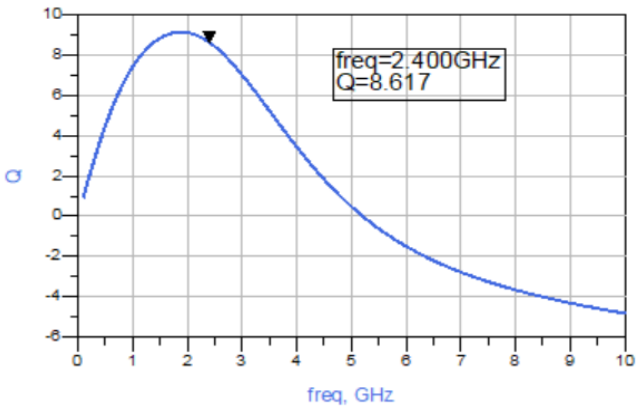


Figure 16. Representing the tested inductor quality factor at the operating frequency.

Table 2. TSMC 0.18-µm PDK inductors.

L1	L (nH)	W (µm)	rad (µm)	nr	Q	SRF (GHz)
Minimum inductance @ 2.4 GHz	0.22	15	30	0.5	7	0.1
Maximum inductance @ 2.4 GHz	25.6	15	125	5.5	5.3	3
Med. @ 2.4 GHz	9.9	15	64	5.5	1.4	5.5
L2	**L (nH)**	**W (µm)**	**rad (µm)**	**nr**	**Q**	**SRF (GHz)**
Minimum inductance @ 2.4 GHz	0.48	30	30	1.5	7.86	0.1
Maximum inductance @ 2.4 GHz	14.64	30	70.3	5.5	1.15	3
Med. @ 2.4 GHz	9.6	30	37	5.5	2.9	4.5
L3	**L (nH)**	**W (µm)**	**rad (µm)**	**nr**	**Q**	**SRF (GHz)**
Minimum inductance @ 2.4 GHz	0.23	15	40	1	9.3	0.1
Maximum inductance @ 2.4 GHz	26.6	15	120	5	4.16	3
Med. @ 2.4 GHz	9.5	15	76	5	7.6	5.5
L4	**L (nH)**	**W (µm)**	**rad (µm)**	**nr**	**Q**	**SRF (GHz)**
Minimum inductance @ 2.4 GHz	0.54	30	65	1.5	7.24	0.1
Maximum inductance @ 2.4 GHz	28.54	30	117	5	0.74	2.5
Med. @ 2.4 GHz	9.26	30	65	4.9	5.34	4

3. SIMULATION RESULT

The demonstrated LNA has been designed using the TSMC CMOS 0.18 µm technology and simulated using ADS RF circuit simulator. Figure 17 shows the design circuity schematic for Wi-Fi and Bluetooth standards with all on-chip components.

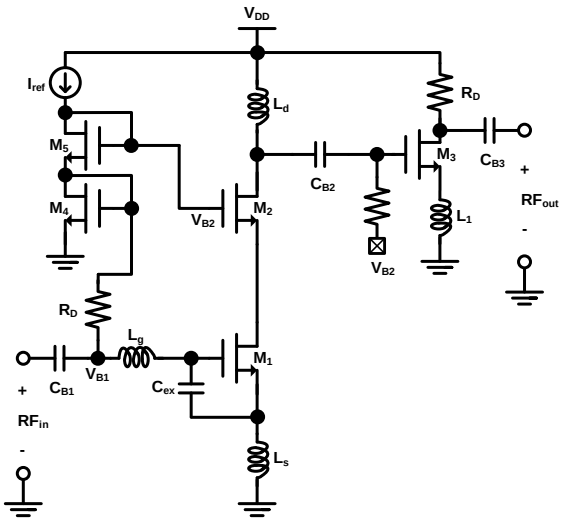


Figure 17. The designed IDCS LNA schematic targeted for Wi-Fi and Bluetooth standards.

The designed LNA uses the source-degenerative cascode topology. The input stage is composed of L_s, L_g, M_1 and C_{ex}. External capacitor (C_{ex}) has been added between gate and source of the input transistor M_1 to minimize design area and NF at low power consumption [19], [21].

These inductors and external capacitors are used for input matching. Transistors M_4 and M_5 are used as LNA biasing circuits, forming a current mirror to set bias current with R_B. The output matching network consists of M_2 and L_D. The drain inductor L_D should resonate with the total drain capacitance and provide a high enough impedance to obtain a decent gain, hence achieving the desired frequency.

The last stage in the design of the design circuity is the output buffer. This buffer is incorporated into the overall LNA design to match the output impedance to a network analyzer for measurement purposes. The common source buffer translates the high impedance path at the gate of M_3 to the low impedance path at the drain of M_3. In addition, coupling capacitances C_{B1}, C_{B2} and C_{B3} are at the input and output of the cascode stage and the buffer stage, respectively. All design values of the components and biasing voltages are summarized in Tables 3 and 4.

Table 3. The designed IDCS LNA component values.

Component	Value
Transistor M_1 width, W_1	163.5 µm
Transistor M_2 width, W_2	96 µm
Transistor M_3 width, W_3	160 µm
Transistor M_4 width, W_4	9 µm
Transistor M_5 width, W_5	22.5 µm
Transistors' length, L	0.18 µm
L_g	9.42 nH
L_s	1.25 nH
L_D	9.42 nH
L_1	2.2 nH
C_{ex}	188 fF
C_{B1}, C_{B2} and C_{B3}	5 pF
R_{B1}, R_{B2}	50 kΩ
R_D	50 Ω
V_{dd}	1.8 V
I_{ref}	200 µA

Table 4. The designed IDCS LNA biasing voltages.

Transistor	Biasing voltages
M_1	V_D =548 mV, V_G=629 mV, V_S=4.14 mV
M_2	V_D =1.8 V, V_G=1.34 V, V_S=548 mV
M_3	V_D =1.56 V, V_G=629 mV, V_S=8.4 mV
M_4	V_D =629 mV, V_G=629 mV, V_S=0 mV
M_5	V_D =1.34 mV, V_G=1.34 V, V_S=629 mV

Figure 18 shows the achieved simulation results for power gain (S_{21}) and maximum power gain. The demonstrated LNA has achieved more than 22 dB S_{21} at the desired frequency (2.4 GHz) and is very close to the maximum gain value that can be achieved at the desired frequency. As shown in Figure 19, the design circuity achieved an NF of 1.75 dB, almost equal to the minimum NF that can be achieved at the desired frequency. Figure 20 shows the achieved reverse transmission coefficient (S_{12}), input reflection coefficient (S_{11}) and output reflection coefficient (S_{22}) at 2.4 GHz. The stability simulation result is shown in Figure 21, where the stability coefficient is greater than one, which means that the designed LNA is unconditionally stable at the desired frequency.

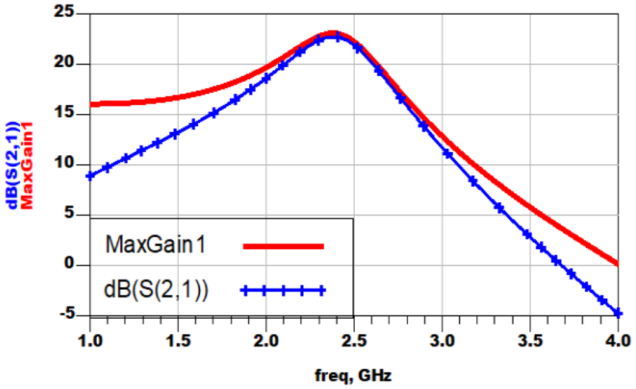


Figure 18. Maximum gain and S_{21} simulation results of IDCS (Wi-Fi and Bluetooth) LNA.

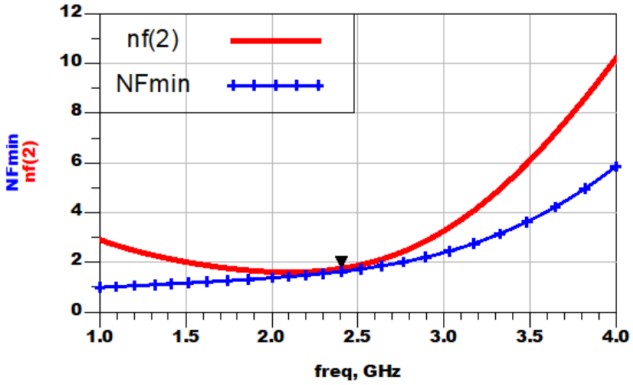


Figure 19. NF and minimum NF simulation results of IDCS (Wi-Fi and Bluetooth) LNA.

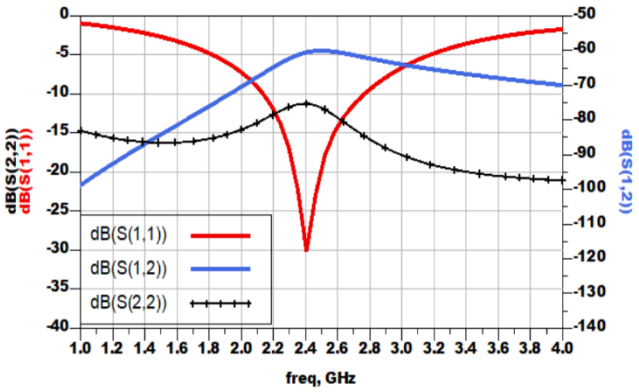


Figure 20. S_{11}, S_{22} and S_{12} simulation results of IDCS (Wi-Fi and Bluetooth) LNA.

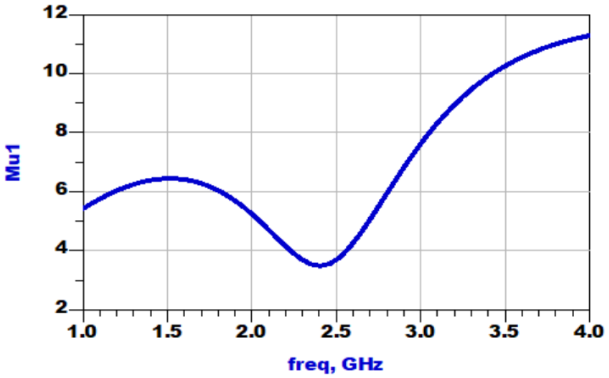


Figure 21. Stability factor simulation result of IDCS (Wi-Fi and Bluetooth) LNA.

Simulation results of $P_{1\text{-}dB}$ compression point and IIP_3 as shown in Figure 22 and Figure 23, are –19 dBm and – 13.5 dBm, respectively. All targeted and achieved performance parameters of (Wi-Fi and Bluetooth) IDCS LNA are summarized in Table 5.

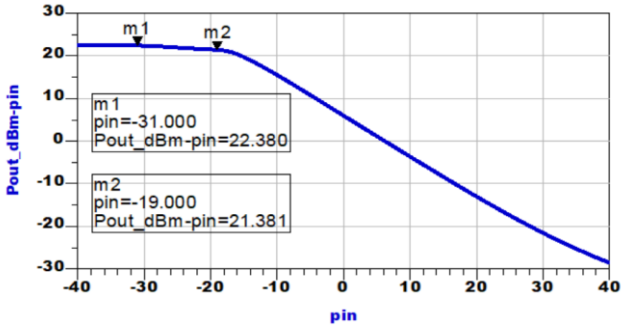


Figure 22. Output power gain *versus* input power p_{in} of IDCS (Wi-Fi and Bluetooth) LNA.

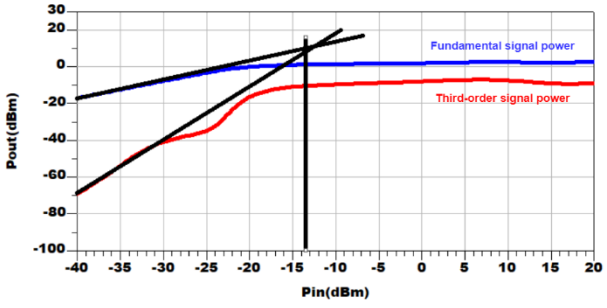


Figure 23. IIP_3 simulation result of IDCS (Wi-Fi and Bluetooth) LNA.

The demonstrated LNA performance has been simulated for different process corners, supply voltages and temperatures at 2.4 GHz. The Figures below (Figure 24–Figure 28) show the corners' simulation

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 01, March 2022.

(Typical-Typical corner (TT), Fast-Fast corner (FF) and Slow-Slow corner (SS)) results of LNA performance parameters when the LNA operates for Wi-Fi and Bluetooth standards over temperature and voltage variations. As shown from the figures below, the designed LNA has a good performance over process, voltage and temperature (PVT) variations.

Table 5. All targeted and achieved performance parameters of (Wi-Fi and Bluetooth) IDCS LNA.

Performance parameters	Targeted specifications	Achieved results
NF	< 3.5 dB	1.75 dB
S_{21}	> 15 dB	22.75 dB
S_{11}	< -10 dB	-30.11 dB
S_{22}	< -10 dB	-11.23 dB
S_{12}	< -40 dB	-60.49 dB
IIP_3	> -15 dBm	-13.5 dBm
$P_{1\text{-}dB}$	> -20 dBm	-19 dBm
Power consumption	< 10 mw	6.16 mW

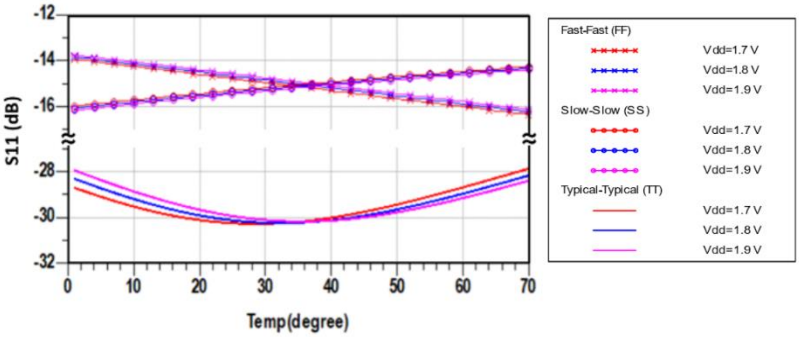


Figure 24. S_{11} simulation results over temperature and voltage variations of the demonstrated LNA at TT, FF and SS corners.

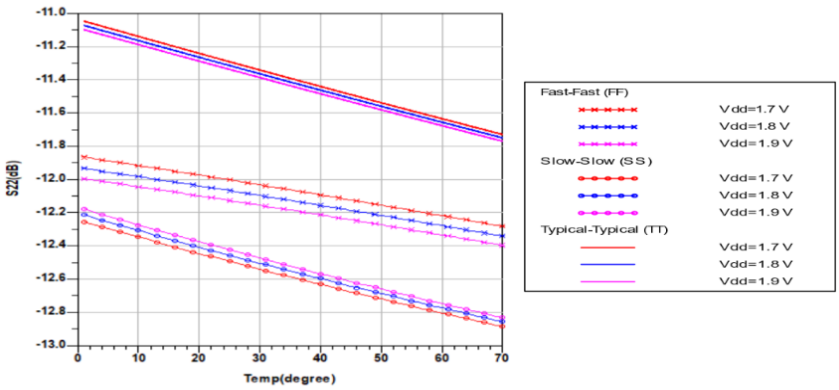


Figure 25. S_{22} simulation results over temperature and voltage variations of the demonstrated LNA at TT, FF and SS corners.

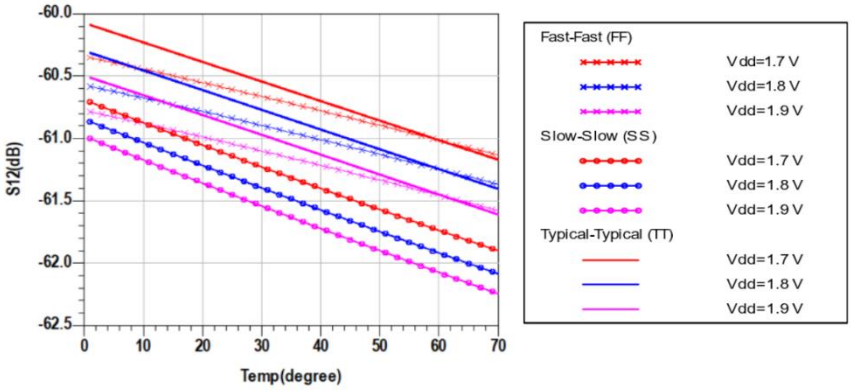


Figure 26. S_{12} simulation results over temperature and voltage variations of the demonstrated LNA at TT, FF and SS corners.

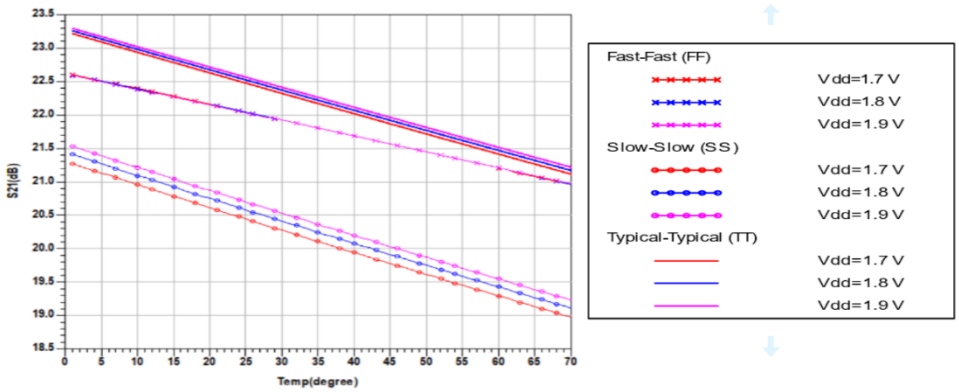


Figure 27. S_{21} simulation results over temperature and voltage variations of the demonstrated LNA at TT, FF and SS corners.

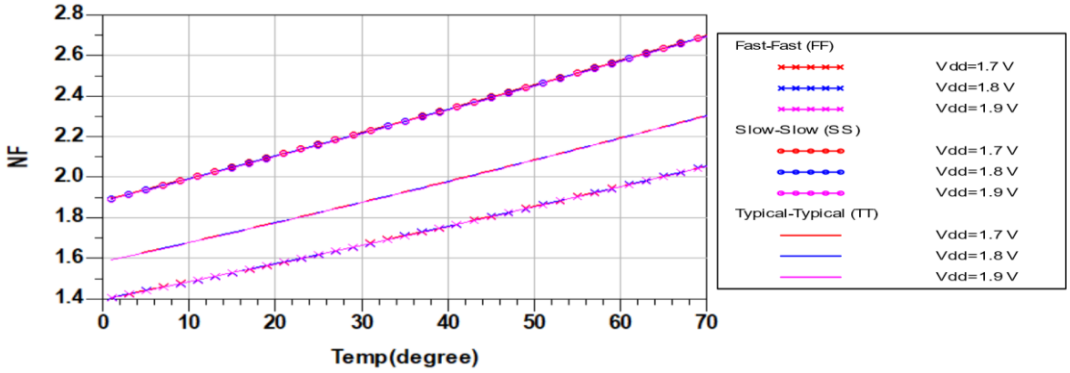


Figure 28. NF simulation results over temperature and voltage variations of the demonstrated LNA at TT, FF and SS corners.

Table 6 shows a comparison of state-of-the-art LNA performances. Because the LNA performance depends on several parameters, like NF, gain, linearity and power consumption, the Figure of Merit (FOM) equation has been used to compare the demonstrated LNA performance with other LNA circuit performances. The FOM equation includes Gain, NF, IIP_3, operation frequency (f_o) and power consumption (P_{DC}) as follows [22]-[23]:

$$\text{FOM[GHz]} = \frac{\text{Gain[Lin.]} * \text{IIP}_3\text{[mW]} * \text{f}_o\text{[GHz]}}{(\text{F}-1) * \text{P}_{DC}\text{[mW]}} \quad (17)$$

The demonstrated LNA has the best FOM among the published NB CMOC LNAs shown in Table 6.

Table 6. A comparison of state-of-the-art LNA performances.

Ref.	**[This work]**	**[4] 2015**	**[5] 2018**	**[6] 2020**	**[7] 2016**	**[8] 2017**	**[9] 2017**
f_o [GHz]	2.4	2.4	2.4	2.4	2.4	2.4	2.4
Tech. [nm]	180	180	180	180	90	180	130
Integration	Fully	Partially	Partially	Fully	Fully	Fully	Fully
Type of results	Simulation	Simulation	Simulation	Measurement	Measurement	Simulation	Simulation
P_{DC} [mW]	6.16	9.68	48	2	3	NA	NA
NF [dB]	1.75	1.2	2.62	8.7	1.8	3.14	10.2
S_{21} [dB]	22.75	14.55	18.24	14.1	13	12.68	10.97
S_{11} [dB]	−30.11	−14.15	−15.95	−14	−10	−13.5	−5.56
S_{22} [dB]	−11.23	−10.6	−13.89	NA	−10	−10	NA
S_{12} [dB]	−60.49	−19.46	−46.05	NA	−20	−33.85	NA
IIP_3 [dBm]	−13.5	−22.41	NA	NA	−8.9	NA	17
$P_{1\text{-}dB}$ [dBm]	−19	−16.43	NA	−14.6	NA	NA	NA
FOM [GHz]	8.07	0.12	NA	NA	3.75	NA	NA

4. CONCLUSIONS

This paper presented a simple and comprehensive design methodology for narrow-band CMOS LNA. The methodology is applied in realizing the IDCCS LNA for Wi-Fi and Bluetooth standards. The g_m/I_D method had been used to optimize the device values for both maximum power gain and minimum NF without exceeding the specified power budget. The circuit is implemented with TSMC CMOS 0.18µm technology using Advance Design System (ADS) RF simulation toolkit. Following the proposed methodology, the design has achieved a power gain (S_{21}) of 22.75 dB, a reverse isolation (S_{12}) of -60.49 dB, input and output reflection coefficients (S_{11} and S_{22}) of -30.11 dB and -11.23 dB, respectively. An NF of 1.75 dB has been achieved with the linearity metrics ($P_{1\text{-}dB}$ and IIP_3) of (-19 dBm and -13.5 dBm).

Note that with a stability factor of 3.48, the circuit is unconditionally stable at the operating frequency of 2.4 GHz. The design power consumption is 6.16 mW from a voltage supply of 1.8V. A superior FOM of 8.07 was obtained using this design methodology. With PVT variations, the design performance metrics still fall within the design specifications with minimal deviations from the typical values obtained. Future work will include an integrated circuit fabrication of the used LNA.

ACKNOWLEDGMENTS

The authors would like to thank EUROPRACTICE for their kind support by providing the technology files.

REFERENCES

[1] C. Y. Wu and F. R. Shahroury, "A Low-voltage CMOS LNA Design Utilizing the Technique of Capacitive Feedback Matching Network," Proc. of the 13th IEEE International Conference on Electronics, Circuits and Systems, pp. 78-81, Nice, France, 2006.

[2] F. R. Shahroury and C. Y. Wu, "A 1-V RF-CMOS LNA Design Utilizing the Technique of Capacitive Feedback Matching Network," NTEGRATION, TheVLSI Journal, vol. 42, no. 1, pp. 83-88, 2009.

[3] F. R. Shahroury, "A 1.2-V Low-power Full-band Low-power UWB Transmitter with Integrated Quadrature Voltage-controlled Oscillator and RF Amplifier in 130nm CMOS Technology," Jordanian Journal of Computers and Information Technology (JJCIT), vol. 2, no. 3, pp. 165-178, 2016.

[4] S. B. Patil and R. D. Kanphade, "Differential Input Differential Output Low Power High Gain LNA for 2.4 GHz Applications Using TSMC 180nm CMOS RF Process," Proc. of the IEEE International Conference on Computing Communication Control and Automation, pp. 911-916, Pune, India, 2015.

[5] P. Bhalse and R. Khatri, "Design of CMOS Differential LNA at 2.4 GHz for RF Front End Receiver," Proc. of the 4th IEEE Int. Conf. for Convergence in Techno. (I2CT), pp. 1-6, Mangalore, India, 2018.

[6] A. R. A. Kumar, A. Dutta and B. D. Sahoo, "A Low-power Reconfigurable Narrow-band/Wide-band LNA for Cognitive Radio-wireless Sensor Network," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 1, pp. 212 - 223, 2020.

[7] C. M. Chou and K. W. Cheng, "A Sub-2 dB Noise-figure 2.4 GHz LNA Employing Complementary Current Reuse and Transformer Coupling," Proc. of the IEEE International Symposium on Radio-frequency Integration Technology (RFIT), pp. 1-3, Taipei, Taiwan, 2016.

[8] N. Shrivastava and R. Khatri, "Design of a 2.4-GHz Differential Low Noise Amplifier Using 180 nm Technology," Proc. of the IEEE International Conference on Recent Innovations in Signal Processing and Embedded Systems (RISE), pp. 494-497, Bhopal, India, 2017.

[9] Aditi and M. Bansal, "A High Linearity and Moderate Gain LNA for Receiver Front-end Applications in 2.4GHz ISM Band," Proc. of the IEEE International Conference on Innovations in Control, Communication and Information Systems (ICICCI), pp. 1-5, Greater Noida, India, 2017.

[10] S. Sattar and T. Z. A. Zulkifli, "A 2.4/5.2-GHz Concurrent Dual-band CMOS Low Noise Amplifier," IEEE Access, vol. 5, pp. 21148 - 21156, 2017.

[11] Y. C. Wang, Z. Y. Huang and T. Jin, "A 2.35/2.4/2.45/2.55 GHz Low-Noise Amplifier Design Using Body Self-biasing Technique for ISM and LTE Band Application," IEEE Access, vol. 7, pp. 183761-183769, 2019.

[12] M. Cimino, H. Lapuyade, Y. Deval, T. Taris and J. B. Begueret, "Design of a 0.9V 2.45GHz Self-testable and Reliability-enhanced CMOS LNA," IEEE Journal of Solid-State Circuits, vol. 43, no. 5, pp. 1187-1194, 2008.

[13] A. Taibi, A. Slimane, M. T. Belaroussi, S. A. Tedjini and M. Trabelsi, "Low Power and High Linear Reconfigurable CMOS LNA for Multi-standard Wireless Applications," Proc. of the 25th IEEE International Conference on Microelectronics (ICM), pp. 1-4, Beirut, Lebanon, 2013.

[14] H. Khosravi, A. Bijari, N. Kandalaft and J. Cabral, "A Low Power Concurrent Dual-band Low Noise

Amplifier for WLAN Applications," Proc. of the IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conf. (IEMCON), pp. 1118-1123, Vancouver, Canada, 2019.
[15] L. H. Lu, H. H. Hsieh and Y. S. Wang, "A Compact 2.4/5.2-GHz CMOS Dual-band Low-noise Amplifier," IEEE Microwave and Wireless Components Letters, vol. 15, no. 10, pp. 685 - 687, 2005.
[16] B. Leung, VLSI for Wireless Communication, 2nd Edition, ISBN: 978-1-4614-0986-1, Boston, MA: Springer, 2011.
[17] G. Gonzalez, Micowave Transistor Amplifiers: Analysis and Design, 2nd Edition, ISBN-13: 978-0132543354, New Jesresy: Prentice Hall, 1997.
[18] B. Razavi, RF Microelectronics, 2nd Edition, ISBN-13: 978-0137134731, Prentice Hall, 2012.
[19] P. Andreani and H. Sjoland, "Noise Optimization of an Inductively Degenerated CMOS Low Noise Amplifier," IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 48, no. 9, pp. 835 - 841, 2001.
[20] P. Jespers, The gm/ID Methodology: A Sizing Tool for Low-voltage Analog CMOS Circuits, ISBN: 978-0-387-47101-3, Springer Science & Business Media, 2009.
[21] T. K. Nguyen, C. H. Kim, G. J. Ihm, M. S. Yang and S. G. Lee, "CMOS Low-noise Amplifier Design Optimization Techniques," IEEE Transactions on Microwave Theory and Techniques, vol. 52, no. 5, pp. 1433 - 1442, 2004.
[22] Z. Li, Z. Wang, L. Chen, C. Wu and Z. Wang, "A 2.4 GHz Ultra-low-power Current-reuse CG-LNA with Active Gm-boosting Technique," IEEE Microwave and Wireless Components Letters, vol. 24, no. 5, pp. 348 - 350, 2014.
[23] R. Ramzan, F. Zafar, S. Arshad and Q. Wahab, "Figure of Merit for Narrow-band, Wide-band and Multi-band LNAs," International Journal of Electronics, vol. 99, p. 1603–1610, 2012.

ملخص البحث:

تعرض هذه الورقة طريقةً لتصميم مكبّر متكامل ضيّق النّطاق منخفض الضّجيج. ولبيان فاعلية الطريقة المقترحة، تمّ تصميم مكبّر منخفض الضّجيج وفق معايير (واي فاي) و (بلوتوث) عند ترددٍ مقداره 2.4 جيجاهيرتز، وقد تمّ تنفيذ دارات التّصميم باستخدام تكنولوجيا سي موس 0.18 ميكروميتر؛ ومع ذلك، فإنّ الطريقة المقترحة يمكن تطبيقها بشكلٍ مكافيء على أي عُقدة عمليات أخرى. وبهذه الطريقة، يمكن الحصول على الحجم الأمثل والانحيازات المثلى للترانزستورات المستخدمة دون خرْق لميزانية القُدرة.

كذلك، تحدد الطريقة المقترحة معايير اختيار ملفات التّرددات الراديوية المستخدمة على الرقاقة بناءً على معامل الجودة، وتردد الرنين الذاتي، والمساحة. ويحقق التّصميم المقترح كسب قُدرة مقداره 22.75 ديسيبل، بينما يبلغ فقْد الرجوع للمدخل 30.11 ديسيبل، والعزل العكسي -60.49 ديسيبل، وفقد الرجوع للمخرج -11.23 ديسيبل. كما تبلغ المتغيرات الخطية لكلٍ من نقطة الانضغاط P1-dB و IIP3: (-19dBm) و (-13.5dBm)، على الترتيب. أما فيما يتعلق برقم الضّجيج فقد بلغ 1.75 ديسيبل، بينما يستهلك المكبّر قُدرة مقدارها 6.16 ميلي واط من مصدرٍ للقُدرة يعطي فرقاً في الجهد يبلغ 1.8 فولت.

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات (JJCIT) مجلة علمية عالمية متخصصة محكمة تنشر الأوراق البحثية الأصيلة عالية المستوى في جميع الجوانب والتقنيات المتعلقة بمجالات تكنولوجيا وهندسة الحاسوب والاتصالات وتكنولوجيا المعلومات.

تحتضن وتنشر جامعة الأميرة سمية للتكنولوجيا (PSUT) المجلة الأردنية للحاسوب وتكنولوجيا المعلومات، وهي تصدر بدعم من صندوق دعم البحث العلمي في الأردن. وللباحثين الحق في قراءة كامل نصوص الأوراق البحثية المنشورة في المجلة وطباعتها وتوزيعها والبحث عنها وتنزيلها وتصويرها والوصول إليها. وتسمح المجلة بالنسخ من الأوراق المنشورة، لكن مع الإشارة إلى المصدر.

الأهداف والمجال

تهدف المجلة الأردنية للحاسوب وتكنولوجيا المعلومات (JJCIT) إلى نشر آخر التطورات في شكل أوراق بحثية أصيلة وبحوث مراجعة في جميع المجالات المتعلقة بالاتصالات وهندسة الحاسوب وتكنولوجيا المعلومات وجعلها متاحة للباحثين في شتى أرجاء العالم. وتركز المجلة على موضوعات تشمل على سبيل المثال لا الحصر: هندسة الحاسوب وشبكات الاتصالات وعلوم الحاسوب ونظم المعلومات وتكنولوجيا المعلومات وتطبيقاتها.

الفهرسة

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات مفهرسة في كل من:

فريق دعم هيئة التحرير

المحرر اللغوي

حيدر المومني

ادخال البيانات وسكرتير هيئة التحرير

إياد الكوز

عنوان المجلة

الموقع الإلكتروني: www.jjcit.org
البريد الإلكتروني: jjcit@psut.edu.jo
العنوان: جامعة الاميرة سمية للتكنولوجيا، شارع خليل الساكت، الجبيهة، عمان، الأردن.
صندوق بريد: 1438 عمان 11941 الأردن
هاتف: 5359949-6-962+
فاكس: 7295534-6-962+

جامعة الأميرة سميّة للتكنولوجيا
Princess Sumaya University for Technology

صندوق دعم البحث العلمي
Scientific Research Support Fund

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات

آذار ٢٠٢٢ | المجلد ٨ | العدد ١
ISSN 2415 - 1076 (Online)
ISSN 2413 - 9351 (Print)

JJCIT

www.jjcit.org
jjcit@psut.edu.jo

مجلة علمية عالمية متخصصة محكمة
تصدر بدعم من صندوق دعم البحث العلمي