IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 30, 2021, accepted April 8, 2021, date of publication April 20, 2021, date of current version April 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074268

A Novel Dynamic Attack on Classical Ciphers Using an Attention-Based LSTM Encoder-Decoder Model

EZAT AHMADZADEH, HYUNIL KIM, ONGEE JEONG, AND INKYU MOON, (Member, IEEE)
Department of Robotics Engineering, DGIST, Daegu 42988, South Korea

Corresponding author: Inkyu Moon (inkyu.moon@dgist.ac.kr)

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant by the Korean Government through MSIT (Research on AI-based Cryptanalysis and Security Evaluation) under Grant 2020-0-00126.

ABSTRACT Information security has become an intrinsic part of data communication. Cryptanalysis using deep learning–based methods to identify weaknesses in ciphers has not been thoroughly studied. Recently, long short-term memory (LSTM) networks have shown promising performance in sequential data processing by modeling the dependencies and data dynamics. Given an encrypted ciphertext sequence and corresponding plaintext, by taking advantage of sequential processing, LSTM can adaptively discover the decryption function regardless of the complexity level, which substantially outperforms traditional methods. However, a lengthy ciphertext sequence causes LSTM to lose important information along the sequence, leading to a decrease in network performance. To tackle these problems, we propose adding an attention mechanism to enhance the LSTM sequential processing power. This paper presents a novel, dynamic way to attack classical ciphers by using an attention-based LSTM encoder-decoder for different ciphertext sequence lengths. The proposed approach takes in a sequence of ciphertext and outputs a sequence of plaintext. The effectiveness and flexibility of the proposed model were evaluated on different classical ciphers. We got close to 100% accuracy in breaking all types of classical ciphers in character-level and word-level attacks. We empirically provide further insights into our results on two datasets with short and long ciphertext lengths. In addition, we provide a performance comparison of the proposed method against state-of-the-art methods. The proposed approach has the potential to attack modern ciphers. To the best of our knowledge, this is the first time an attention-based LSTM encoder-decoder has been applied to attack classical ciphers.

INDEX TERMS Cryptanalysis, classical ciphers, attention-based LSTM encoder-decoder, recurrent neural network.

I. INTRODUCTION

Security is a major concern in all fields in which information is protected by various encryption methods. Cryptosystems use various techniques to convert the original message into an encrypted message of non-understandable text. Therefore, a systematic method for modifying or transforming the message is needed. Researchers have proposed many different cryptosystems to improve the information security level by transforming plaintext into secure ciphertext. There are numerous ciphers with various functionalities, specifications, and strengths [1]–[3]. Cryptanalysis includes exploring the weaknesses of cryptosystems to ensure the security level of the encryption algorithm. Researchers have conducted various attacks to analyze the vulnerabilities of different cryptosystems [4]–[8]. Despite their simple structure, classical ciphers are well-suited to illustrating the effectiveness of different attack approaches [4], [7]. Several approaches investigated the application of computational intelligence (CI) to cryptographic problems. Those techniques include the Genetic Algorithm (GA) [9], [10], Simulated Annealing (SA) [11], and Tabu search [12], [13] which mainly rely on searching for decryption key in evolutionary manner. Evaluating hundreds or thousands of solutions might require a long time. Machine learning approaches have been applied to cryptographic problems including artificial neural networks (ANNs) [5], [14], [15]. However, typical machine learning–based approaches to cryptanalysis have

The associate editor coordinating the review of this manuscript and approving it for publication was Jun Wang.

generalized limitations. Taking advantage of the high computational power of deep learning–based techniques has attracted many researchers at a growing rate, and has produced interesting results in many different fields, including medical image analysis [16], speech recognition [17], natural language processing (NLP) [18], and cryptanalysis [6], [19]. However, there are many open problems in different fields where deep learning applications should be investigated. Recently, Gomez *et al.* [20] introduced the cipher cracking model that utilizes general adversarial networks (GANs) to attack classical ciphers. The proposed approach was applied to attack the classical shift and Vigenere ciphers and obtained 98.7% accuracy for the shift cipher and 75.7% for the Vigenere cipher. The fundamental constraint of the ANN-based or convolutional neural network (CNN)-based models is a lack of sequential computation ability. Recurrent neural networks (RNNs), long short-term memory (LSTM), and the gated recurrent unit (GRU) in particular have been established for sequence modeling [21]. RNNs suffer from the vanishing gradient problem for deep networks. LSTM is a class of RNN, and is a powerful sequence-to-sequence learning architecture that was proven capable of learning algorithmic tasks [22]. However, the limitation with LSTM is that a lengthy input sequence causes LSTM to forget important information along the sequence. To tackle this problem, the attention mechanism was introduced, which substantially improves the LSTM capability by adding special attention to important information along the sequence [23], [24].

Inspired by this, we present a new approach to classical cipher attacks by using an attention-based LSTM encoder-decoder model. Unlike previous work, our model can be applied to any type of cipher, regardless of the complexity level. Our approach exploits language models for sequence prediction through the attention mechanism. The approach proposed in this paper fundamentally improves prior work. The sequence-to-sequence decryption ability of the proposed method can preserve the sequential nature of the language model (LM) and the alignment between the input and output sequence, which is an important step in conducting an efficient attack on different ciphers. The proposed attention-based LSTM encoder-decoder model takes in sequences of ciphertext with varying lengths and outputs sequences of plaintext. In particular, our proposed approach can be adopted to crack modern ciphers. Our approach was evaluated by attacking the classical Caesar (shift), Vigenere, and substitution ciphers. We attained nearly 100% accuracy in breaking all types of the aforementioned ciphers. Furthermore, the impact of hyperparameter tuning (the word embedding dimension, the number of hidden LSTM units, the mini-batch size) on the network training performance was investigated. To sum up, our contributions are as follows: 1) we demonstrated an efficient and fast ciphertext attack using the attention-based LSTM encoder-decoder model, 2) the model computes attention weights relying on both the importance of each ciphertext word and the position, 3) compared with several state-of-the-art methods, we conducted qualitative experiments, and the results evaluated the effectiveness of our proposed approach, 4) we conducted our experiments on two different datasets with short and long ciphertext sequence lengths, and 5) we conducted attacks at both the character level and the word level.

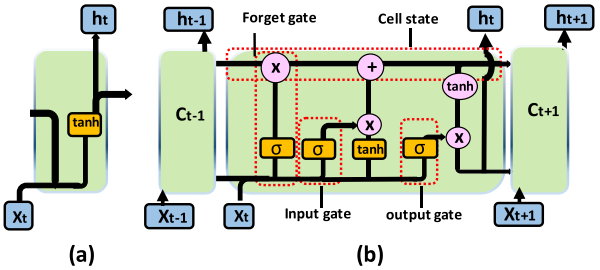

FIGURE 1. (a) the structure of the RNN unit, and (b) an LSTM unit that maps input vector x_t to output hidden state vector h_t.

To the best of our knowledge, this is the first attack on classical ciphers based on the LSTM sequential LM with an attention mechanism. The structure of this paper is as follows. In Section II, we discuss the prerequisites for RNNs and LSTMs and for the LSTM encoder-decoder model and classical ciphers methods. In Section III, we discuss the details of our proposed system, including attention-based LSTM, word embedding, and the architecture of the proposed method. In Section IV, we discuss the experimental results and offer a quantitative analysis. Section V contains concluding remarks.

II. BACKGROUND

A. RECURRENT NEURAL NETWORKS

The RNN is a type of neural network capable of forming a state history of previous input that is suitable for learning algorithmic tasks, and which has become a cornerstone for many NLP applications. The RNN structure makes it suited for variable-length input such as sequential data (see Fig. 1(a)). For sequence data $(x_1, x_2, x_3, \ldots x_T)$, hidden state h_t of the RNN is then updated via the following equation:

$$h_t = f(h_{t-1}, x_t), \tag{1}$$

where f denotes the activation function. However, the limitation of the RNN is that it suffers from vanishing gradients in deep networks. A small gradient value does not contribute very much to learning [25].

B. LONG SHORT-TERM MEMORY (LSTM)

The RNN is unable to memorize what has been seen along the sequences. The LSTM neural network is a special RNN. The difference lies in an LSTM structure that includes the input gate, the output gate, the forget gate, and the memory cell as the hidden layer, which makes it suitable for learning long-range dependencies through its internal memory cells (see Fig. 1(b)) [22], [26].

LSTMs are capable of learning long-term dependencies. They are designed to remember information for a long period

using a memory unit called the cell. Three gates control the information flow in to and out of the neuron's memory cell (the input, output, and forget gates) which each possess an activation function. The LSTM updates its hidden state sequentially, but the updates highly depend on memory cells. For example, if the input gate notes a high activation, the input will be stored in the memory cell [27].

The LSTM network computes the mapping from input sequence $\boldsymbol{x} = (x_1, x_2 \ldots x_n)$ to output sequence $\boldsymbol{h} = (h_1, h_2 \ldots h_m)$ by iteratively calculating network unit activations $(t = 1, 2, \ldots, T)$ as follows:

$$i_t = \sigma(w_{xi}^T x_{(t)} + w_{hi}^T h_{(t-1)} + b_i), \quad (2)$$
$$f_t = \sigma(w_{xf}^T x_{(t)} + w_{hf}^T h_{(t-1)} + b_f), \quad (3)$$
$$o_t = \sigma(w_{xo}^T x_{(t)} + w_{ho}^T h_{(t-1)} + b_o), \quad (4)$$
$$g_t = \tanh(w_{xg}^T x_{(t)} + w_{hg}^T h_{(t-1)} + b_g), \quad (5)$$
$$c_t = f_t \otimes c_{(t-1)} + i_t \otimes g_t, \quad (6)$$
$$h_t = o_t \otimes \tanh c_{(t)}, \quad (7)$$

where i, f, o, c, and h are the input gate, forget gate, output gate, intermediate gate, and the cell memory output; w denotes the layer weight representing input x, and b represents the threshold of the output gate; σ is the sigmoid activation function, and tanh is the tangent activation function, as seen in Fig. 1(b).

C. LSTM ENCODER-DECODER MODEL

The LSTM encoder-decoder is a common framework in the deep learning model, and is normally used for natural language processing tasks to solve the problem of sequence-to-sequence (seq2seq) modeling [28]. The LSTM encoder-decoder model is one of the many-to-many models that take in a sequence of data and output a new sequence of data. The structure of the encoder-decoder architecture is shown in Fig. 2. The encoding process is performed in a sliding window manner.

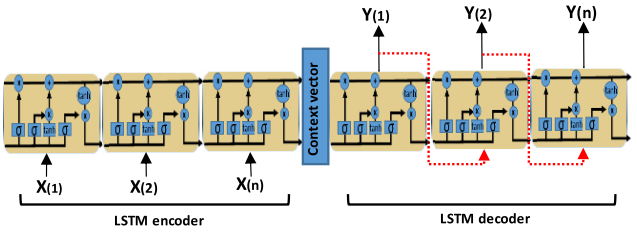

FIGURE 2. The structure of an LSTM-based encoder-decoder network model.

The encoder transforms input sequence $\boldsymbol{x} = (x_1, x_2 \ldots x_n)$ into hidden state sequence $\{h_1, h_2, h_3, \ldots, h_n\}$, which are features extracted from the input sequence. If the input and output sequences have the same length, the extracted features can be served to directly predict the target sequence [22]; otherwise, the hidden states are used to generate context vector C as follows:

$$C = f(h_1, h_2, \ldots . h_t), \quad (8)$$

where f represents the mapping function. The context vector includes abstract information about the entire sequence. In the next step, the context vector is used by the decoder to predict a target output sequence. However, a lengthy input sequence causes the LSTM to lose important information along the sequence, leading to a decrease in network performance. To overcome this limitation, the attention mechanism was introduced.

The attention-based LSTM has shown remarkable results with difficult sequence prediction problems like text translation sorting and long additions, and quickly became the dominant approach [29].

D. CLASSICAL CIPHERS

We now briefly explain the classical ciphers (shift cipher, Vigenere, and substitution) [14]. A shift (or Caesar) cipher is a substitution cipher where each letter in the original message is replaced with a letter corresponding to a certain number of letters up or down in the alphabet. In this way, the original plaintext that was readable is converted into unintelligible ciphertext. The amount of shifting is known to the intended receiver, who can decode the message by shifting each letter in the encrypted message back. Note that the number of different keys is limited to between 0 and 25. A Vigenere cipher is well-known as a poly-alphabetic cipher that maps a letter of the alphabet into a set of different letters. The number of possible keys is 26^m. A simple substitution cipher employs any permutation of the 26 letters as a key, so there are $26! \approx 2^{88}$ possible keys.

III. PROPOSED APPROACH

A. ATTENTION- BASED LSTM ENCODER-DECODER

The attention mechanism is part of a neural architecture that is capable of dynamically highlighting important features of the input data [30]. Therefore, a decoder would have to pay close attention to the specific state of the encoder, which probably has a lot of information. Attention-based systems were applied to NLP after their successes in computer vision, speech recognition, and text reconstruction. However, the original LSTM network does not have an explicit attention ability. There are different types of attention mechanisms proposed in the literature [30]–[33]. In our method, we utilize the Luong method [34] for attention score calculation. The structure of the attention-based LSTM encoder-decoder is shown in Fig. 3.

The attention function takes as input the previous hidden state of the LSTM, C_{t-1}, and the annotations $h_1, \ldots, h_T$ as follows:

$$e_{ti} = f(c_t - 1, h_i), \quad (9)$$

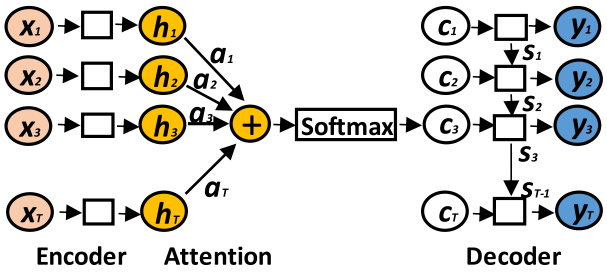

FIGURE 3. The structure of an attention-based LSTM encoder-decoder model.

in which the attention values are fed into a *softmax* function for normalization so the weights will be between 0 and 1:

$$a_{ti} = \frac{\exp(e_{ti})}{\sum_{j=1}^{T} \exp(e_{tj})}. \tag{10}$$

The updated context vector (c_t) is used to predict the target output (y_t) in sampling state t. The attention weight (a_{ti}) is assigned to each hidden state (h_i). The context vector at the time (t) can be calculated as

$$c_t = \sum_{i=1}^{t} a_{ti} h_i, \tag{11}$$

where a_{ti} is the attention weight of the i^{th} hidden state, h_i.

B. CIPHERTEXT ENCODING

Word embedding is a set of language modeling and feature learning techniques in NLP where words are mapped to vectors of real numbers (*Word2vec*) [35]. In other words, embedding is a method of converting word tokens into machine-readable vectors. *Word2vec* is a two-layer neural net that converts the text words into a vectors. The input is a text corpus and the output is a set of vectors. The advantage of *word2vec* is that it can train large-scale corpora to produce low-dimensional word vectors.

Tokenization is taking the words out of sentences and converting a sentence of words into single words (also called word segmentation). We preprocess datasets of the ciphertext so that punctuation is treated as separate tokens, and we ignore any non-English characters and make a sequence of characters with spaces removed, and then converted to numerical vectors using word embedding techniques (see Fig. 4). Given a sentence consisting of n words ($x_1, x_2, x_3, \ldots, x_{n-2}, x_{n-1}, x_n$), every word x_i is converted into a real-value vector, e_i, represented as:

$$e_i = [w_1, w_2, w_3, \ldots., w_{n-2}, w_{n-1}, w_n] \in R^{n \times d}, \tag{12}$$

where w is a word, and d is the size of the word embedding.

C. DROP OUT

To prevent the network from overfitting, we use the dropout technique. Dropout is a regularization technique used in neural networks to prevent overfitting. The dropout operation randomly selects some neurons within a network layer with a specific dropout parameter, and sets the input and output features to 0. In this way, dropout can capture more randomness. In our proposed network the output of the LSTM layer is fed into a dropout layer [36]. The cross-entropy loss function is a commonly used method to measure the difference between the two probabilities. In our proposed scheme, the cross-entropy loss function is utilized to evaluate the prediction loss of the model, which reflects the gap between the real plaintext and the predicted plaintext [37]:

$$loss = -\sum_{i=1}^{T} y_i \log \bar{y}_i, \tag{13}$$

where i is the index number of the sentence, y is the predicted plaintext, and $\bar{y}$ is the corresponding actual plaintext and T represents length of the window. Basically, during the training process, the training data are split into mini-batches. All of the sequences in the mini-batches have the same length as the longest sequence in the mini-batch. To prevent network overfitting, we sort ciphertext sequences by their lengths and padded them to the equal in size. Figures 5(a) and 5(b) show the data sequence sorting of the Brown and the company report datasets before sorting and after sorting respectively. Figure 5(c) illustrates the maximum and minimum data sequence lengths in characters for both datasets.

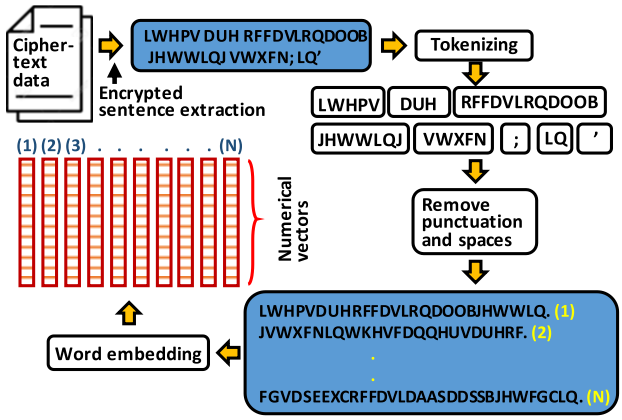

FIGURE 4. The structure of ciphertext tokenizing and converting the sequence of characters (with spaces and punctuations removed) to numerical vectors using the word embedding technique.

D. SYSTEM EQUIPMENTS & PARAMETER SETTINGS

Training of the machine learning model is the most computationally intensive task which system equipment would substantially increase or decrease the processing time. A model hyperparameter is a special configuration of the designed model which cannot be estimated directly from data and controls the learning process that affects the network convergence time. The model hyperparameters are key to machine learning algorithms to be tuned for a specific problem. The system equipment and the parameter settings are given

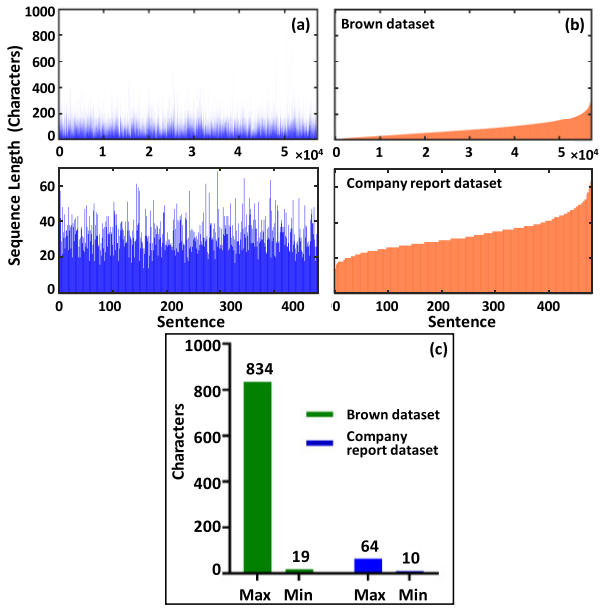

FIGURE 5. Data sequence sorting before the training process: (a) the data sequence before sorting, (b) the sequence after sorting, and (c) the maximum and minimum lengths for each dataset.

TABLE 1. System equipment.

Component	Description
CPU	Intel Cori7-7700k
GPU	GTX1050
OS type	Windows 10 / 64-bit
Language	Matlab
Optimizer	Adam

TABLE 2. Parameter settings.

Parameter	Value
Word embedding dimension	100, 200, 300
Number of hidden units	100, 200, 300
Dropout	0.3
Mini-batch size	10, 50, 100
Learning rate	0.002

TABLE 3. Datasets.

Dataset	Number of sentences	Amount of training data	Amount of testing data
Brown	57,340	80%	20%
Company report	480	80%	20%

in Tables 1 and 2, respectively. As mentioned earlier, to evaluate the efficiency and dynamics of the proposed method, we tested our approach by using different parameter settings on a company report dataset and a dataset called Brown, with short and long ciphertext sequence lengths, respectively. The dataset descriptions and the divisions for training and testing are explained in Table 3.

The steps for training the proposed attention-based LSTM encoder-decoder model are in the algorithm above.

Algorithm 1 Attack on a Classical Cipher With the Attention-Based LST

Input: Dataset (ciphertext sequence) $D = (x_t, y_t), t = 1, 2, 3, \ldots k$ ***Learning rate, hidden neurons, batch size, training epoch***

Output: Predicted plaintext sequence $D = y_t, t = 1, 2, 3, \ldots k$

Steps
Ciphertext sequence preprocessing.
For $i = 1$ *to* k *do*
Input: ciphertext of machine-readable vector
Encoder:
For $m = 1$ *to* t
  Enter the word vector into the LSTM
  Calculate attention weights
  Attention normalization
  Encode hidden state
  Context vector $c_t = \sum_{i=1}^{t} a_{ti} h_i$
Decoder:
For $j = 1$ *to* t
  Input context vector
  Decode hidden state
  Concatenate
  Predict plaintext (y_t)
Calculate network loss: $loss = \sum_{i=1}^{T} y_i \log \bar{y}_i$
Output: ***Plaintext*** $(\mathbf{y_t})$

IV. EXPERIMENTAL RESULTS

In this section, we describe multiple experiments that show the efficiency of the attention-based LSTM encoder-decoder in attacks on classical ciphers. To speed up the training procedure, our proposed network schemes are implemented using a deep learning library written in Matlab, which can be executed on a Graphics Processing Unit (GPU). The GPU generally increases execution speed by five to 10 times, compared with a Central Processing Unit (CPU).

The structure of the proposed attention-based LSTM encoder-decoder model is shown in Fig. 6. In the encoder and the decoder, the model contains four LSTM layers each. First, we vectorize the ciphertext sequences using the word embedding technique and get a vector representation of the ciphertext input. Then, we feed embedded words through the four LSTM layers. Afterwards, the attention score for each ciphertext sequence is calculated, and the output is fed into a softmax function to be normalized. Therefore, the attention scores will be normalized to between 0 and 1, and the context vector will be updated using the normalized attention scores. In a different time, step t in the decoding stage, the attention

TABLE 4. Training dataset plaintext and the corresponding ciphertext encrypted with different classical ciphers.

Plaintext	Ciphertext	Encryption method
Furthermore as an encouragement to revisionist thinking it manifestly is fair to admit that any fraternity has a constitutional right to refuse to accept persons it dislikes	IXUWKHUPRUH DV DQ HQFRXUDJHPHQW WR UHYLVLRQLVW WKLQNLQJ LW PDQLIHVWOB LV IDLU WR DGPLW WKDW DQB IUDWHUQLWB KDV D FRQVWLWXWLRQDO ULJKW WR UHIXVH WR DFFHSW SHUVRQV LW GLVOLNHV	Caesar cipher (3 shifts to the right)
	JBAMBIYVHLI HB TH IULHOVHPXGIUC MI VLEBMMVWBMX AQBHOPWZ CX TJGCJLBMFC PB YUMY CH UHTRM NLHC THC MATNIYWBNC OJL U GVWLNMADMCSUJE LMNQM NS YNYOWL CH UGJNIN TLALIRZ RM XMZUBEIZ	Vigenere cipher (Key: DGIST)
	MPNIZGNTUNG DC DV GVSUPNDHGTGVI IU NGLWCWUVWCI IZWVOWVH WI TDVWMGCIRE WC MDWN IU DJTWI IZDI DVE MNDIGNVWIE ZDC D SUVCIWIPIWUVDR NWHZI IU NGMPCG IU DSSGYI YGNCUVC WI JWCRWOGC	Substitution cipher.

TABLE 5. Experimental results from breaking a caesar cipher.

Ciphertext	Expected (plaintext)	Achieved (plaintext)
WKHXQLWDULDQFOHUJBZHUHDQ HAFOXVLYHFOXERIFXOWLYDWHG JHQWOHPHQDVWKHWHUPZDVWKHQ XQGHUVWRRGLQWKHDFNEDBDQG SDUNHUZDVGHILQLWHOBQRWD JHQWOHPDQ HLWKHU LQ WKHRORJB RULQPDQQHUV	TheUnitarianclergywereanexclusivecl ubofcultivatedgentlemenasthetermwas thenunderstoodintheBackBayandParke rwasdefinitelynotagentlemaneitherinth eologyorin manners	TheUnitarianclergywereanexclusivecl ubofcultivatedgentlemenasthetermwas thenunderstoodintheBackBayandParke rwasdefinitelynotagentlemaneitherinth eologyorin manners
HYHQVRJDQQHWWMXGLFLRXVOB DUJXHGWKHDVVRFLDWLRQFRXOG OHJLWLPDWHOBGHFLGHWKDW SDUNHUVKRXOGQRWEEH HQFRXUDJHGRUDVVLVWHGLQ GLIIXVLQJKLVRSLQLRQVEBWKRVH ZKRGLIIHU IURP KLP LQ UHJDUG WR WKHLUFRUUHFWQHVV	EvensoGannettjudiciouslyarguedthe Associationcouldlegitimatelydecide thatParkershouldnotbeencouragednora ssistedindiffusinghisopinionsbythosew hodifferfromhiminregardtotheircorrect ness	EvensoGannettjudiciouslyarguedthe Associationcouldlegitimatelydecide thatParkershouldnotbeencouragednora ssistedindiffusinghisopinionsbythosew hodifferfromhiminregardtotheircorrect ness

weights are different. Finally, the updated context vector is fed into the decoder. The decoder layer converts this vector into the plaintext sequence through the four LSTM layers. The experimental results show that the attention-based LSTM encoder-decoder model can adaptively learn the decryption function, regardless of the sequence length and key length from the provided data, and thus, has the potential to decrypt complex ciphers.

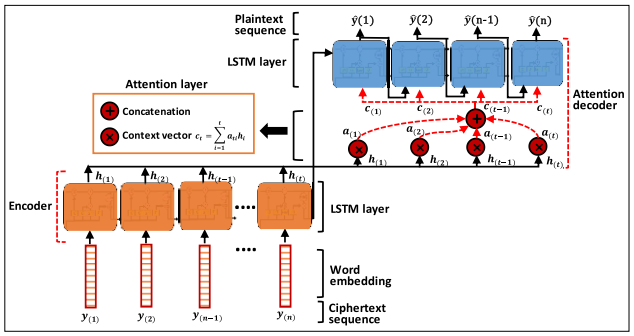

FIGURE 6. The architecture of the attention-based LSTM encoder-decoder model for attacking classical ciphers. In the first step, the ciphertext sequence is converted into numerical vectors using the word embedding technique and is then fed into the encoder LSTM as numerical words. The LSTM encoder converts the input sequence features to the hidden states. The attention mechanism is further used to help better capture the key features for each ciphertext sequence and then feeds them to the softmax function for normalization. Then, the context vector is updated using attention scores. Afterwards, the updated context vector is passed to the decoder. Finally, the decoder converts the ciphertext to plaintext sequentially. Note that for different time steps t in the decoding stage, the attention weights are different.

A. CLASSICAL CIPHER ATTACK

To demonstrate the performance of the proposed method on different sequence lengths, we applied it to two different datasets with short and long sequence lengths (the company report dataset and the Brown dataset, respectively) for both character-level and word-level attacks. The curves for prediction accuracy and loss versus the number of iterations for both datasets during the character-level attack are shown in Fig. 7. We observed that our model's performance is similar for both short and long ciphertext lengths, with faster convergence for shorter sequences. The advantages of our proposed model, in comparison to a state-of-the-art model, can be considered the following: 1) our model is not provided with any prior knowledge of character frequencies, 2) no information about the cipher key is provided, 3) it can overcome a significantly large ciphertext length, and 4) our methodology can easily be adapted to attack different complex ciphers. Our model is capable of solving ciphers with 834 character elements and 184 words.

B. CAESAR CIPHER BREAKING

The Caesar cipher is the earliest known message encryption method. Using the Caesar cipher method, examples of the plaintext message and the corresponding ciphertext are shown in Table 4. The message encrypted with the Caesar cipher method can be easily deciphered when the intended recipient knows the shift number used. As we previously pointed out, in the Caesar cipher method has only 26 keys and can be overcome easily with brute force by trying all

TABLE 6. Experimental results from breaking a vigenere cipher (key: DGIST).

Ciphertext	Expected (Plaintext)	Achieved (plaintext)
ALCHXEFJKYRVCXHXPCEYHAX XRGVABUXLQHHIZCFYRDQH VISRXPIKYTLOLAMIFLMHQRYRZBX WNKHMJRHOWHWWQLVKTLVLM MBIPAIOPWRMMENJBHWAQBM HLVTHHAXIIMZXGNLLVBHHZXY NLLRKWSUPKYKHCBIRZ	Wetodayarenotentitledtoexcoriatehone stmenwhobelievedParkertobedownrig htperniciousandwhobarredtheirpulpits againsthisdemandtopoisonthemindsoft heir congregations	Wetodayarenotentitledtoexcoriatehone stmenwhobelievedParkertobedownrig htperniciousandwhobarredtheirpulpitsa gainsthisdemandtopoisonthemindsofth eir congregations
XONBLHLVTHH HPTCRZCMBI JJEPMURLNSYCAIHVGRZSY RGNISUXWXBJEFMINKNCOJWHICNK GIHWMNLHCMBIFFHOPKOHFPVF YLILRGKYPARNSAQXYBAAXGIVO ILSJUTCQPWZWLYRLNMHWBNCH WTNYYJELISRZCSU	TheirdemandagainsttheCalvinistOrtho doxyforintellectuallibertyhadnevermea ntthattheywouldfollowfreeinquirytothe extremeofproclaimingChristianityanat uralreligion	TheirdemandagainsttheCalvinistOrtho doxyforintellectuallibertyhadnevermea ntthattheywouldfollowfreeinquirytothe extremeofproclaimingChristianityanat uralreligion

TABLE 7. Experimental results from breaking a substitution cipher.

Ciphertext	Expected (Plaintext)	Achieved (plaintext)
GLGVCUHDVVGIIXPJWSWUPCRE DNHPGJIZGDCCUSWDIWUVSUPRJ RGHWIWTDIGREJGSWJGIZDI YDNOGNCZUPRJVUIVGGVSUPNDHGJ VUNDCCWCIGJWVJWMMPCWVH ZWCUYWVWUVCVEIZUCGFZU JWMMGNMNUTZWTWVNGHDNJIU IZGWNSUNNGSIVGCC	evensogannettjudiciouslyarguedthe associationcouldlegitimatelydecidethat parkershouldnotbeencouragednorassist edindiffusinghisopinionsbythosewhodi fferfromhiminregardtotheircorrectness	evensogannettjudiciouslyarguedthe associationcouldlegitimatelydecidethat parkershouldnotbeencouragednorassist edindiffusinghisopinionsbythosewhodi fferfromhiminregardtotheircorrectness
UVGSDVGLGVDNHPGIZUPHZIZWC WCDJGRWSDIGTDIIGNIZDIGLGNE XPCIWMWSDIWUVGQWCIGJMUN IZGWNNGIPNVWVH IZGYPVRWS RGSIPNGIUIZGMWNCISZPNSZDVJCU IUCPYYNGCCWINDIZGNIZDVRGI YDNOGNPCGWIDCDCUPVJWVH VUDNJMUNZWCYNUYDHDVJDFZGV ZWCIPNVCZUPRJSUTGIUUSSPYEWI	onecanevenarguethoughthisisadelicate matterthateveryjustificationexistedfort heirreturningthepubliclecturetothefirst churchandsotosuppressitratherthanletp arkeruseitasasoundingboardforhisprop agandawhenhisturnshouldcometooccu pyit	onecanevenarguethoughthisisadelicate matterthateveryjustificationexistedfort heirreturningthepubliclecturetothefirst churchandsotosuppressitratherthanletp arkeruseitasasoundingboardforhisprop agandawhenhisturnshouldcometooccu pyit
MWVDRREWIJWJCGGTSRGDNDCJDE IUIZGCGSRGNHETGVDCHDVVGIIC CUVGQYRDWVGJWVIZG VWUHNDYZEUMZWCMDIZGNIZGE ZDJDRFDECSUVIGVJGJMUNIZG YNUYNWGIEUMIZGWNSRDWTIUIZG IWIRGUMSZNWCIWDVC	finallyitdidseemclearasdaytotheseclerg ymenasgannettssonexplainedinthebiog raphyofhisfathertheyhadalwaysconten dedfortheproprietyoftheirclaimtothetitl eofchristians	finallyitdidseemclearasdaytotheseclerg ymenasgannettssonexplainedinthebiog raphyofhisfathertheyhadalwaysconten dedfortheproprietyoftheirclaimtothetitl eofchristians

possible keys. However, an automated method is desirable in order to recover the plaintext without human intervention. The experimental result from breaking the Caesar ciphertext, the expected plaintext, and the achieved plain text using the proposed method are illustrated in Table 5.

C. VIGENERE CIPHER BREAKING

The Vigenere cipher was introduced to hinder the use of frequency analysis to specify the cipher mapping. Instead, the Vigenere cipher method uses a separate shift cipher for each element, and the key is tiled to match the length of the plaintext. A Vigenere cipher adds a password of length L to each plaintext block of length L. Increasing the key length significantly increases the number of possible combinations, and thus, prevents frequency analysis. The idea is exactly to contrast the problem of preserving the letter frequencies. The experimental results from the prediction of the plaintext from ciphertext encrypted using the Vigenere cipher method are in Table 6.

D. SUBSTITUTION CIPHER BREAKING

The substitution cipher presents an extremely large key-space where keys are permutations of the alphabet. Substitution ciphers can be solved by exhaustively searching through an extremely large keyspace for the key that produces the decrypted text. In our scheme, we used the cipher mapping key "DVSJGMHZWXORTVUYANCIPLFQEK" (a $\rightarrow$ D, b $\rightarrow$ V, c $\rightarrow$ S, d $\rightarrow$ J. . . .). The experimental results of substitution cipher breaking are shown in Table 7.

In the next step, we conducted a word-level attack on a classical cipher by preserving the space character between consecutive words. The network accuracy for word-level attacks on different classical ciphers is shown in Fig. 8. The maximum lengths for each ciphertext sequence in Brown dataset was 834 characters and 184 words. Since the ciphertext lengths are long, it would be difficult for classical LSTM networks to learn. However, using the attention mechanism and network fine-tuning, it is possible for LSTM networks to fit the training set. The quantitative results from attacks

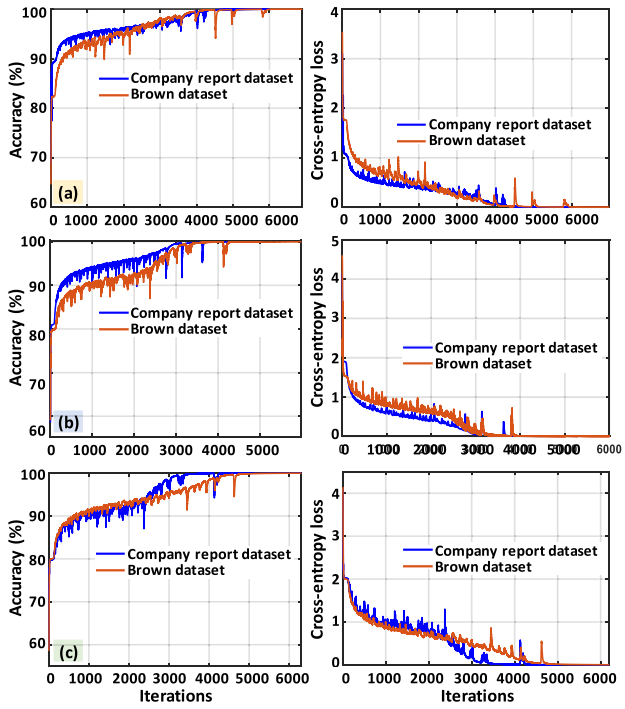

FIGURE 7. The network training performance from attacking different types of classical ciphers at the character level. The first and second rows illustrate network training accuracy and the corresponding loss, respectively, for (a) a Caesar cipher, (b) a Vigenere cipher, and (c) a substitution cipher.

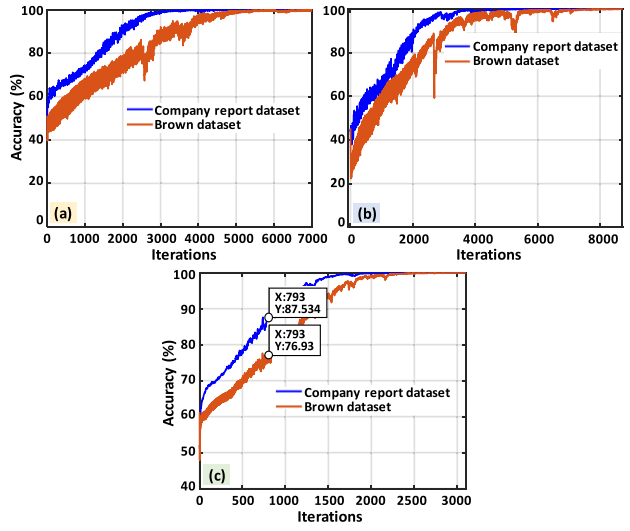

FIGURE 8. Network training performance for word-level cipher cracking of (a) a Caesar cipher, (b) a Vigenere cipher, and (c) a substitution cipher.

on classical cipher using the proposed method at both the character level and the word level are given in Table 8.

To correctly train the LSTM, many hyperparameters need to be tuned. These hyperparameters will affect the performance of the network during the time to convergence. To evaluate the impact of different hyperparameters on network training, we conducted multiple experiments. The investigated parameters include the number of hidden units for LSTMs for both the encoder and the decoder, the word embedding dimension, and the mini-batch size.

The results in Table 8 demonstrate the accuracy of training and testing the proposed method with both datasets. The obtained results indicate the success of the proposed method due to LSTM's capability to learn decryption functions for different types of ciphers. From the learning curves, we can see how the proposed method can adaptively learn the decryption function, regardless of the cipher complexity or key length.

TABLE 8. Network performance from attacking different classical ciphers for training and test data.

Method		Network accuracy	
		Brown: train/test (%)	Company report: train/test (%)
Caesar cipher	Character-level	100/100	100/100
Vigenere cipher		~100/99.9	~100/99.9
Substitution cipher		~100/99.9	~100/100
Caesar cipher	Word-level	100/100	100/100
Vigenere cipher		~100/99.8	~100/99.9
Substitution cipher		~100/99.8	~100/99.9

E. THE IMPACT OF THE NUMBER OF LSTM HIDDEN UNITS

The number of hidden units in an LSTM refers to the dimensionality of the hidden states [27]. Figure 9(a) shows the impact of the number of hidden LSTM units on the network's training progress. Changing the number of hidden units affects the training of LSTMs. Experimental results demonstrated that increasing the number of LSTM hidden units in both the encoder and the decoder will increase training convergence time. This is because, for long sequences, LSTMs are sensitive to the hyperparameters. The experimental results for the impact of different numbers of LSTM hidden units are shown in Table 9.

TABLE 9. Impact of LSTM hidden units on network training performance.

Number of LSTM hidden units	Network loss	Iterations
100	~0.0	5900
200	~0.0	2800
300	~0.0	3000

F. THE IMPACT OF WORD EMBEDDING DIMENSION

Word embedding is a useful tool, which is utilized as a key to many fundamental problems in NLP research. As a critical hyperparameter, the choice of dimensionality for word vectors has a profound influence on network training performance. Word embedding is usually a linear or quadratic function of dimensionality, which directly affects training time and computational costs. The smaller dimensionality

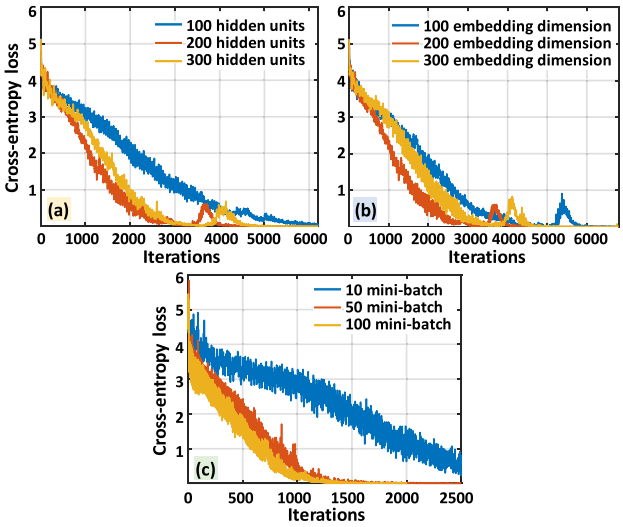

FIGURE 9. The impact of different hyperparameters on training performance: (a) the number of LSTM hidden units, (b) the word embedding dimension, and (c) the Mini-batch size.

of word embedding is normally not enough to capture all possible word relations, while very large embedding dimensionality suffers from overfitting, tends to increase model complexity, and slows down training, which are constraints that can potentially limit model applicability and deployment. The influence of different word embedding dimensions on network training performance is shown in Fig. 9(b). The quantitative results are shown in Table 10.

TABLE 10. Impact of word embedding dimension on network traning performance.

Word embedding dimension	Network loss	Iterations
100	~0.0	4500
200	~0.0	2800
300	~0.0	3500

G. THE IMPACT OF MINI-BATCH SIZE ON TRAINING

The mini-batch stochastic gradient descent (SGD) algorithm is widely used in training machine learning models, and deep learning models in particular [38]. One of the important hyperparameters that need to be tuned is the batch size, which is the amount of data used in every epoch to train the network. A too-large batch size leads the network takes too long to achieve convergence. However, if the mini-batch size is too small, it will make the network fluctuate without achieving acceptable performance. In the second set of experiments, we examined the effect from using different mini-batch sizes on network training performance (see Fig. 9(c)). The performance results in Table 11 show that larger mini-batch sizes produce fewer errors because they come closer to a full-batch gradient, and thus, have a lower bias. On the other hand, a mini-batch size as small as 10 produces a high error rate.

TABLE 11. Impact of mini-batch size on network training performance.

Mini-batch size	Network loss	Iterations
10	~0.0	3000
50	~0.0	1400
100	~0.0	1150

The experimental results for the effects of the number of LSTM hidden units, of the word embedding dimension, and of the mini-batch size on network training performance are given in Tables 9, 10, and 11, respectively.

In Fig. 10, we show a performance comparison between the proposed method and state-of-the-art methods when attacking different classical ciphers. Figure 10 demonstrates how the proposed method outperformed the state-of-the-art methods when breaking different types of classical ciphers.

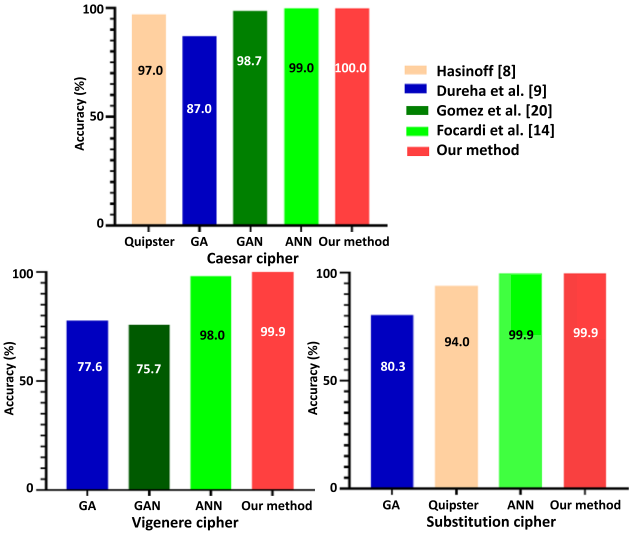

FIGURE 10. Performance comparison of the proposed method against state-of-the-art methods on classical cipher attack.

H. DISCUSSION

Table 8 shows a quantitative measurement of the experimental results on different cipher types from character-level and word-level cipher breaking. The results show that the proposed attention-based LSTM encoder-decoder model was able to break all types of classical ciphers with near-flawless accuracy. The proposed method performed extremely well on all types of classical ciphers, achieving excellent results for both character-level and word-level cipher breaking with a sequence length of 834 characters and 184 words. In comparison to state-of-the-art methods, the attention-based LSTM encoder-decoder model architecture was found to be stable for all ciphers. Besides, we investigated the impact of different hyperparameter tunings on network performance. The above experimental results also elucidate the fact that by using optimal hyperparameters the network can converge in fewer iterations. The results also illustrated that network

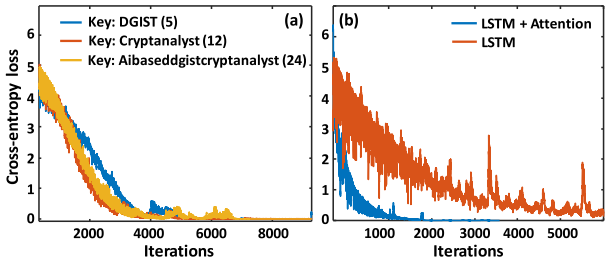

FIGURE 11. **Network performance on Min/MAX sequence length analysis by different key lengths and the impact of the attention mechanism on network performance. (a) network performance of different key lengths of Vigenere cipher, (b) the impact of attention mechanism on network performance.**

performance is stable, regardless of the cipher's complexity level. Therefore, the proposed method has the potential to crack modern ciphers. To analyze the network performance for recovering maximum and minimum sequence length, we conducted several experiments. As mentioned earlier, in Caesar and Substitution ciphers, each character in plaintext is encrypted independently to another character. Therefore the network can recognize the encryption pattern from a minimum sequence of a single character. But in the case of the Vigenere cipher, the pattern in the ciphertext depends on the Key. Therefore the minimum sequence length that the network can recognize the pattern is equal to the key length. To investigate key length impact on the network performance to recognize the pattern in ciphertext, We encrypted the dataset with different key length of (5, 12, 24) characters and analyzed network performance. Figure 11(a) demonstrates the network performance on different key lengths for the Viginere cipher method. As it can be seen network could recognize the pattern in data for different key lengths. The maximum sequence length used in this experiment using the Brown dataset is 834 characters.

We mentioned previously the RNNs have a serious problem when working with long sequences. Since all the input sentence in the encoder side is encoded in one vector as context vector, it is a challenging task to encode all information in a long sequence. Consequently, network performance decreases with long sentences [24]. Attention allows the model to focus on the particular parts of the input sequence which has important information. We analyzed the network performance with and without the attention mechanism (Luong attention mechanism). As shown in Fig. 11(b), the results suggest the attention module improves the network performance and decreases the net loss of the LSTM model.

V. CONCLUSION

The task of attacking a classical cipher was carried out using an attention-based LSTM encoder-decoder network model that provided significant performance in the field of machine learning. In this paper, we proposed a novel, attention-based encoder-decoder network model for attacking ciphertexts with dynamic sequence lengths. Lengthy ciphertext data sequences cause LSTM to lose important features, which gradually decreases network performance. Therefore, we added an attention mechanism to tackle LSTM's problem. The state-of-the-art methods mostly consider the ciphertext as fixed-length data, while the proposed approach needs to be dynamic against different ciphertext lengths. We carried out experiments on two different datasets with long and short ciphertext lengths for character-level and word-level ciphertext breaking. The experimental results evaluated on both datasets with different ciphertext lengths demonstrated nearly 100% accuracy for all types of classical ciphers. The experimental results demonstrate that the proposed method can adaptively learn the decryption function providing the ciphertext and the corresponding plaintext. The proposed approach has the potential to be applied to all types of ciphers, regardless of the complexity level. This is mainly because of the common natural language output structure from which we could exploit sequential structures to learn the decryption function. In addition, the impact of hyperparameter tuning (mini-batch size, number of LSTM hidden units, and word embedding dimension) was investigated.

REFERENCES

[1] P. Yang, N. Xiong, and J. Ren, "Data security and privacy protection for cloud storage: A survey," *IEEE Access*, vol. 8, pp. 131723–131740, 2020.
[2] J. H. Kong, L.-M. Ang, and K. P. Seng, "A comprehensive survey of modern symmetric cryptographic solutions for resource constrained environments," *J. Netw. Comput. Appl.*, vol. 49, pp. 15–50, Mar. 2015.
[3] X. Jing, Y. Hao, H. Fei, and Z. Li, "Text encryption algorithm based on natural language processing," in *Proc. 4th Int. Conf. Multimedia Inf. Netw. Secur.*, Nov. 2012, pp. 670–672.
[4] J. Chen and J. S. Rosenthal, "Decrypting classical cipher text using Markov chain Monte Carlo," *Statist. Comput.*, vol. 22, no. 2, pp. 397–413, Mar. 2012.
[5] N. D. Truong, J. Y. Haw, S. M. Assad, P. K. Lam, and O. Kavehei, "Machine learning cryptanalysis of a quantum random number generator," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 2, pp. 403–414, Feb. 2019.
[6] J. So, "Deep learning-based cryptanalysis of lightweight block ciphers," *Secur. Commun. Netw.*, vol. 2020, Jul. 2020, Art. no. 3701067.
[7] M. Russell, J. A. Clark, and S. Stepney, "Using ants to attack a classical cipher," in *Genetic and Evolutionary Computation—GECCO* (Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 2723. Berlin, Germany: Springer, 2003, pp. 146–147.
[8] S. Hasinoff, "Solving substitution ciphers," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2003.
[9] A. Dureha and A. Kaur, "A generic genetic algorithm to automate an attack on classical ciphers," *Int. J. Comput. Appl.*, vol. 64, no. 12, pp. 20–25, Feb. 2013.
[10] R. Toemeh, "Breaking transposition cipher with genetic algorithm," *Elektron. Elektrotechnika*, vol. 79, no. 7, pp. 75–78, 2015.
[11] M. Din, S. K. Pal, S. K. Muttoo, and S. Madan, "A hybrid computational intelligence based technique for automatic cryptanalysis of playfair ciphers," *Defence Sci. J.*, vol. 70, no. 6, pp. 612–618, Oct. 2020.
[12] I. Polak and M. Boryczka, "Tabu cryptanalysis of VMPC stream cipher," *Tatra Mountains Math. Publications*, vol. 73, no. 1, pp. 145–162, Aug. 2019.
[13] I. Polak and M. Boryczka, "Tabu search in revealing the internal state of RC4+ cipher," *Appl. Soft Comput.*, vol. 77, pp. 509–519, Apr. 2019.
[14] R. Focardi and F. L. Luccio, "Neural cryptanalysis of classical ciphers," in *Proc. CEUR Workshop*, vol. 2243, 2018, pp. 104–115.
[15] M. M. Alani, "Applications of machine learning in cryptography: A survey," 2019, *arXiv:1902.04109*. [Online]. Available: http://arxiv.org/abs/1902.04109
[16] R. J. S. Raj, S. J. Shobana, I. V. Pustokhina, D. A. Pustokhin, D. Gupta, and K. Shankar, "Optimal feature selection-based medical image classification using deep learning model in Internet of medical things," *IEEE Access*, vol. 8, pp. 58006–58017, 2020.

[17] R. A. Khalil, E. Jones, M. I. Babar, T. Jan, M. H. Zafar, and T. Alhussain, "Speech emotion recognition using deep learning techniques: A review," *IEEE Access*, vol. 7, pp. 117327–117345, 2019.
[18] S. Jaf and C. Calder, "Deep learning for natural language parsing," *IEEE Access*, vol. 7, pp. 131363–131373, 2019.
[19] M. M. Rahaman, C. Li, X. Wu, Y. Yao, Z. Hu, T. Jiang, X. Li, and S. Qi, "A survey for cervical cytopathology image analysis using deep learning," *IEEE Access*, vol. 8, pp. 61687–61710, 2020.
[20] A. N. Gomez, S. Huang, I. Zhang, B. M. Li, M. Osama, and L. Kaiser, "Unsupervised cipher cracking using discrete GANs," 2018, *arXiv:1801.04883*. [Online]. Available: http://arxiv.org/abs/1801.04883
[21] H. M. Lynn, S. B. Pan, and P. Kim, "A deep bidirectional GRU network model for biometric electrocardiogram classification based on recurrent neural networks," *IEEE Access*, vol. 7, pp. 145395–145405, 2019.
[22] T. Wang, P. Chen, K. Amaral, and J. Qiang, "An experimental study of LSTM encoder-decoder model for text simplification," 2016, *arXiv:1609.03663*. [Online]. Available: https://arxiv.org/abs/1609.03663
[23] F. Long, K. Zhou, and W. Ou, "Sentiment analysis of text based on bidirectional LSTM with multi-head attention," *IEEE Access*, vol. 7, pp. 141960–141969, 2019.
[24] J. Zeng, X. Ma, and K. Zhou, "Enhancing attention-based LSTM with position context for aspect-level sentiment classification," *IEEE Access*, vol. 7, pp. 20462–20471, 2019.
[25] Y. Zhang, F. Xiao, F. Qian, and X. Li, "VGM-RNN: HRRP sequence extrapolation and recognition based on a novel optimized RNN," *IEEE Access*, vol. 8, pp. 70071–70081, 2020.
[26] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using LSTM for region embeddings," in *Proc. 33rd Int. Conf. Mach. Learn.*, vol. 2, 2016, pp. 794–802.
[27] H. Xiao, M. A. Sotelo, Y. Ma, B. Cao, Y. Zhou, Y. Xu, R. Wang, and Z. Li, "An improved LSTM model for behavior recognition of intelligent vehicles," *IEEE Access*, vol. 8, pp. 101514–101527, 2020.
[28] Q. Wang, R.-Q. Peng, J.-Q. Wang, Z. Li, and H.-B. Qu, "NEWLSTM: An optimized long short-term memory language model for sequence prediction," *IEEE Access*, vol. 8, pp. 65395–65401, 2020.
[29] J. Fan, H. Wang, Y. Huang, K. Zhang, and B. Zhao, "AEDmts: An attention-based encoder-decoder framework for multi-sensory time series analytic," *IEEE Access*, vol. 8, pp. 37406–37415, 2020.
[30] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "Erratum: 'ABCNN: Attention-based convolutional neural network for modeling sentence Pairs,'" *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 566–567, Dec. 2016.
[31] L. Huang, Y. Ma, S. Wang, and Y. Liu, "An attention-based spatiotemporal LSTM network for next POI recommendation," *IEEE Trans. Services Comput.*, early access, May 27, 2019, doi: 10.1109/TSC.2019.2918310.
[32] D. Deng, L. Jing, J. Yu, and S. Sun, "Sparse self-attention LSTM for sentiment lexicon construction," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 11, pp. 1777–1790, Nov. 2019.
[33] G. Zhu, L. Zhang, L. Yang, L. Mei, S. A. A. Shah, M. Bennamoun, and P. Shen, "Redundancy and attention in convolutional LSTM for gesture recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1323–1335, Apr. 2020.
[34] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2015, pp. 1412–1421.
[35] A. Almuhareb, W. Alsanie, and A. Al-Thubaity, "Arabic word segmentation with long short-term memory neural networks and word embedding," *IEEE Access*, vol. 7, pp. 12879–12887, 2019.
[36] N. Yang, H. Tang, J. Yue, X. Yang, and Z. Xu, "Accelerating the training process of convolutional neural networks for image classification by dropping training samples out," *IEEE Access*, vol. 8, pp. 142393–142403, 2020.
[37] X. Li, D. Chang, T. Tian, and J. Cao, "Large-margin regularized softmax cross-entropy loss," *IEEE Access*, vol. 7, pp. 19572–19578, 2019.
[38] X. Qian and D. Klabjan, "The impact of the mini-batch size on the variance of gradients in stochastic gradient descent," 2020, *arXiv:2004.13146*. [Online]. Available: http://arxiv.org/abs/2004.13146

EZAT AHMADZADEH received the M.S. degree in software engineering from IAU University, Science and Research Branch of Tehran, Iran, in 2014, and the Ph.D. degree in computer engineering from Chosun University, South Korea, in 2020. He is currently a Postdoctoral Researcher with the Department of Robotics Engineering, DGIST, South Korea. His research interests include medical image analysis, artificial intelligence, AI-based cryptanalysis, deep learning, and machine learning.

HYUNIL KIM received the B.S. degree in applied mathematics, and the M.S. and Ph.D. degrees in information security from Kongju National University, South Korea, in 2014, 2016, and 2019, respectively. He is currently a Postdoctoral Researcher with the Department of Robotics Engineering, DGIST, South Korea. His research interests include AI-based cryptanalysis, artificial intelligence, deep learning, blockchain, and federated learning.

ONGEE JEONG received the B.S. degree in biomedical engineering from KonKuk University, South Korea, in 2019. She is currently pursuing the integrated M.S. and Ph.D. degree with the Department of Robotics Engineering, DGIST. Her research interests include deep learning, information security, image processing, and digital holography.

INKYU MOON (Member, IEEE) received the B.S. degree in electronics engineering from SungKyunKwan University, South Korea, in 1996, and the Ph.D. degree in electrical and computer engineering from the University of Connecticut, Storrs, CT, USA, in 2007. From 2009 to 2017, he was a Faculty Member with the Department of Computer Engineering, Chosun University, South Korea. He joined DGIST, South Korea, in 2017, where he is currently a Professor with the Department of Robotics Engineering. He has more than 100 publications, including peer-reviewed journal articles, conference proceedings, and invited conference papers. His research interests include image cryptography, digital holography, digital image processing, deep learning, and optical information processing. He is a Senior Member of OSA and a member of SPIE.