Journal of Information Systems Engineering and Management
2025, 10(21s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Refresh Step to Enhance Secure Translation Using Fully Homomorphic Encryption and Sequence-to-Sequence Neural Networks

Michael Lahzi Gaid Ibrahim[1][0000-0001-5030-5169]* and Khaled Shaalan[2][0000-0003-0823-8390]
[1]*Faculty of Engineering and Information Technology, The British University in Dubai, Dubai, United Arab Emirates*
Email: miklahzi@gmail.com
[2]*Faculty of Engineering and Information Technology, The British University in Dubai, Dubai, United Arab Emirates,*
Email: khaled.shaalan@buid.ac.ae

ARTICLE INFO

Received: 16 Dec 2024

Revised: 01 Feb 2025

Accepted: 16 Feb 2025

ABSTRACT

Secure translation has become crucial for governments and organizations to protect the exchange and translation of confidential documents and obtain accurate results in the target language. In a previous study, we discussed how combining fully homomorphic encryption and sequence-to-sequence neural networks could serve this purpose efficiently, which was demonstrated through its application and algorithm verification. The previous proposal presented limitations in terms of the amount of noise it produced, which affected the processing time, along with the limited database considered. Thus, a larger database in both Arabic and English needed to be considered while focusing on reducing time. Based on our previous work, this study found that adding a new "refresh" function to the steps decreases the noise from the multiple created layers; consequently, the computational time is reduced. Furthermore, the developed secure mode of the encryption process did not affect translation quality.

Keywords: Homomorphic Encryption; Fully Homomorphic Encryption; Sequence-To-Sequence; Machine Translation; Information Security; Data Security; Government Cybersecurity; Refresh

INTRODUCTION

Research has shown that the need for secure translation applications is growing rapidly in todays' cyber world. Governments and organizations need highly secured applications than can work on large scale data and within a reasonable time frame. Data security includes data availability, confidentiality, and integrity. Most previous attempts in this regard included a decryption step for the translation, which is the most vulnerable step threatening the hacking and stealing of confidential data [1], known as data cloning. As discussed in literature, these threats have led to a growing need for improving cyber security and making it a part of governmental policies [2].

Data encryption is the conversion of plain text into cipher text. Since its introduc-tion, such conversion was deemed safe, but with advancements in technology, safety is becoming a concern. In particular, data cloning and decryption are possible despite converting text into meaningless words, such as symbols or numeric text, which require a specific key to decode [3]. Over the years, several encryption algorithms have emerged, each outperforming its predecessor to keep up with the fast technological growth. For example, sequence-to-sequence neural network predicts the next element by considering the prior one [4,5]. This approach simply adds an encoder and a decoder to the text. Recurrent neural networks and long short-term memory (LSTM) are among the most recent technologies used in translation and in improving the readability of written text by a computer program [6].

Homomorphic encryption in all its forms works on large-scale statistical analysis, specifically the fully homomorphic encryption (FHE) scheme. Therefore, it has been considered useful in fields like translation applications and highly secure governmental segments and information security [8, 9]. In particular, FHE can deal with additions and multiplications over the underlying algebraic structure; thus, it is more useful in notions of security and cost. FHE provides data security on the cloud, as it can work on encrypted data directly without decrypting them. In this way, data security is achieved [10]. However, with the sequence-to-sequence neural network the layers it forms, and the subsequent application of the FHE scheme, substantial noise is created. This

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

noise affects the outcome and data accuracy, in addition to increasing the time required to obtain the results. Thus, recent research has focused on solving the noise issue. Noise reduction is crucial in all FHE schemes because beyond a certain limit, the ciphertext cannot be decrypted [11].

Several attempts have been made to reduce noise. Nevertheless, this study proposes an approach to obtain accurate results despite the amount of noise created in ciphertext. The novel aspect of the proposed approach is the introduction of a vector with its dimension specified as a parameter of the encryption key [11].

In this study, we extend our previous work [12] by adding a new step to the pro-posed algorithm to reduce noise, which decreases the time required to obtain accurate results. In addition, we worked on a larger database of both Arabic and English languages.

PREVIOUS WORK

In our previous work, we created a database of both Arabic and English words, including 1000 sentences, with 3812 English words and 3112 Arabic words [11]. We used a fixed vocabulary for each language, constituting 3050 most frequently used words from the source language and 2490 from the target language. Two hundred sequences were used in the experimental validation of the normal mode and 40 sequences in the secure mode. Owing to the small size of the database and the random value representation, the translation quality was poor; nevertheless, translation quality was not the main focus of the study. The aim was to reach the same result in both the secure and normal modes. The accuracy and the consistency of the translation was 100% per sentence; however, almost 8 h was required to translate a three-word sentence.

In our previous contribution, we proposed a confidential algorithm for binary classification and provided an experimental validation of the new algorithm. The steps were as follows:

1. First, compute the forget gate as shown in Figure 1. This operation required 15 min. The forget gate can be computed as follows:

$$f_t = \sigma\,(W_f.\,[h_{t-1},x_t] + b_f), \tag{1}$$

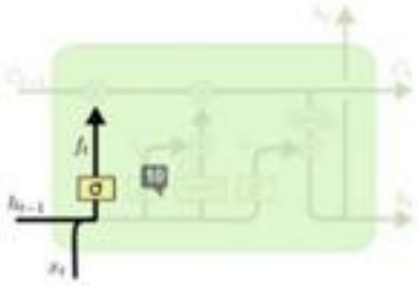

Figure 1. Compute forget gate

2. Input gate layer and tanh layer as follows:

$$i_t = \sigma\,(W_i.[h_{t-1}, x_t] + b_i), \tag{2}$$

$$\acute{C}_t = tanh\,(W_C.[h_{t-1}, x_t] + b_c). \tag{3}$$

This layer has two gates, each required approximately 15 min to be completed, as shown in Figure 2.

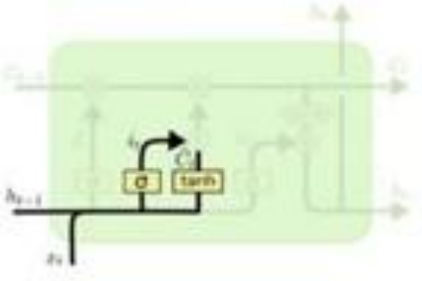

Figure 2. Calculate input gate and cell candidate

3. Update each state value (Figure 3) as follows:

$$C_t = f_t * C_{t-1} + i_t * \acute{C}_t. \quad (4)$$

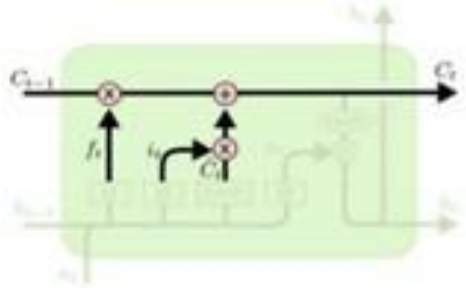

Figure 3. Calculate cell state

4. This fourth gate required approximately 5 min to be completed.

Sigmoid layer and tanh layer (Figure 4):

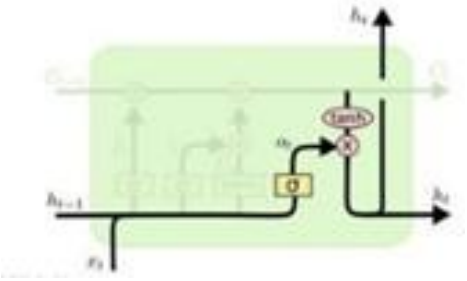

Figure 4. Calculate output gate.

$$o_t = \sigma\, \sigma\, (W_o\, [h_{t-1}, x_t] + b_o), \quad (5)$$

$$h_t = ot^* \tanh (C_t). \quad (6)$$

This layer required more than 2 h to be calculated.
Table 1 presents the time required by each gate using our previously proposed approach.

Table 1. Time required by each gate using our previously proposed approach.

Gate	Time needed to be calculated
Forget gate layer	15 min.
Calculate Input Gate and cell Candidate	This layer has two gates; each gate required approximately 15 min.
Calculate Cell State	5 min.
Calculate Output Gate	More than 2 h.

In the proposed algorithm, words are transformed into vectors, private and public keys are generated, vectors are encrypted using the public key, and encrypted data and the public key are sent to the server, which weights them using the public key to generate the encrypted data. Finally, the output is decrypted using the private key, as shown in Figure 5.

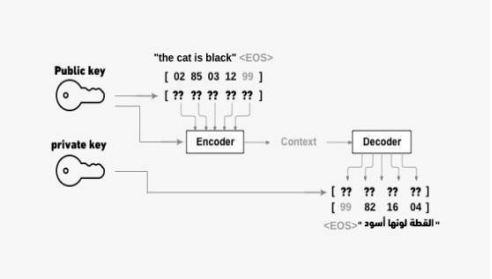

Figure 5. Description process of the proposed approach.

The next section describes the new function proposed herein that decreases the time required to perform these steps. This refresh step can save a significant amount of time.

ENHANCED ALGORITHM

The steps of the process shown in Figure 6 can be summarized as follows:

- First, transfer words to vectors.
- Second, generate both private and public keys.
- Next, encrypt those vectors using the public key.
- Then, send the encrypted data and the public key to the server.
- The server encrypts the weights using the public key and generates encrypted data.
- Add a decryption and encryption function to the client side, named "Refresh." This function is called twice: first by the encoder and then by the decoder.
- Finally, the output to be decrypted is sent using the private key.

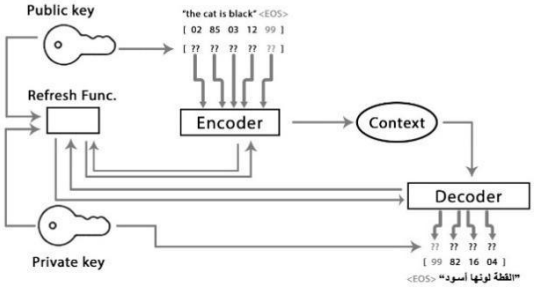

Figure 6. Enhanced algorithm.

ALGORITHM VERIFICATION

Training Part

First, the embedding matrix $E \in R^{L \times M}$ was initialized with random values for simplicity, where L is the word vector length, and M denotes the language vocabularies to be modeled.

We could also initialize the embedding using GloVe or word2vec. However, in our experiment, we used random values owing to the small size of vocabularies [10]. Subsequently, we trained the LSTM encoder–decoder on a training corpus as follows:

For each sentence pair:
Let $\{x_0, x_1, \ldots\ldots, x_s\}$ be the words in the sentence picked from embedding E.
For each $x_t \in R^M$, we compute the et $\in R^N$ encoded vector, where N is the LSTM hidden unit size. Moreover,

$$e_t = \text{LSTM}\ (Xt, e_{t-1}). \qquad (7)$$

The LSTM function is defined as follows:

$$f_t = \text{sigmoid}\,(w_f x_t + u_f e_{t-1} + b_f), \quad (8)$$

$$i_t = \text{sigmoid}\,(w_i x_t + u_i e_{t-1} + b_i), \quad (9)$$

$$o_t = \text{sigmoid}\,(w_o x_t + u_o e_{t-1} + b_o), \quad (10)$$

$$c_t = f_t * c_{t-1} + i_t * \tanh\,(w_c x_t + u_c e_{t-1} + b_c), \quad (11)$$

$$e_t = o_t * \tanh(c_t), \quad (12)$$

where $f_t, i_t, o_t, c_t \in R^N$ are intermediate vectors and w_f, w_i, w_o, $w_c \in R^{N \times M}$, u_f, u_i, u_o, $u_c \in R^{N \times N}$, and b_f, b_i, b_o, $b_c \in R^N$ are weight matrices and bias vector parameters to be trained, respectively.
The output of the encoder is the last e_t output vector of the LSTM, which represents the input sentence. Subsequently, the LSTM decoder is trained similarly but starting with the encoder output.

Let $\{y_e, y_0, y_1, \ldots, y_o\}$ be the output sentence, where ye is the start of the sentence.
For each $y_t \in R^M$, we compute $dt \in R^N$ decoded vector. Finally, we use d_t to predict y_{t+1} using Softmax.

$$d_t = \text{LSTM}(y_t, e_t, d_{t-1}). \quad (13)$$

The Softmax function, which transforms a vector of K real values into a vector of K real values whose sum equals 1, evaluates the entire embedding matrix E based on the output of the decoder and assigns a score to each word vector, indicating its probability of being the output.

The training step focused on maximizing the log probability of a correct translation T given the source sentence S; thus, the training objective is as follows:

$$1/|S| \sum_{(T,S)\in S} \log p\,(T|S), \quad (14)$$

where S is the training set. After training, translations are generated by finding the most likely translation:

$$\check{T} = \arg\max_{T}\ p. \quad (15)$$

Prediction on Encrypted Environment

On the client side, we select the sentence vector representation from the embedding matrix. Then, the selected vectors are encrypted, and the encrypted data are sent to the translation server. In the translation server, we encrypt all the weight matrices and bias vector parameters in both the encoder and decoder with the shared public key of the incoming encrypted message.

While applying the same math operation using homomorphic algorithms, we added a refresh function to the client side, where the encoder and decoder send the encrypted values after calculating gate 5 to the client side, to be decrypted and encrypted again. This step allows us to control the encryption noise and continue calculating gates 6 and 7.

The decoder output is sent to the client, where Softmax can be used in the decrypted mode. An issue with the encrypted LSTM is that we could not compute the sigmoid and tanh functions directly because the homomorphic does not support them. Therefore, we addressed this problem using Taylor series approximation for both sigmoid and tanh functions on the encrypted LSTM.

EXPERIMENTAL RESULTS

Dataset Details

We have improved an existing English-to-Arabic dataset to include more sequences from the United Nations letters, including English and Arabic Translated Letters. We trained our models on a subset of 10,021 sentences comprising

31,521 English and 29,956 Arabic words.
We used a fixed vocabulary for both languages because typical neural language models rely on a random value representation for each word. In particular, we used 30,000 of the most frequent words for the source language and 29,000 of the most frequent words for the target language. Each out-of-vocabulary word was replaced with a special "UNK" token.

Training Details

We used deep LSTMs with 1 layer, nine cells at each layer, an input vocabulary of 1112, and an output vocabulary of 950. Moreover, Softmax was implemented over 950 words at each output.

Parallelization

We used a C#.net implementation of deep LSTM and Seal.dll (the homomorphic library Published by Microsoft) with the configuration from the previous section on a Core i7-4870HQ CPU @ 2.50 GHz, 16 GB RAM. Training

required approximately 2 h for 1000 iterations with the proposed implementation.

Experimental Results

One thousand sequences were used to validate the normal mode, and 200 sequences were used to validate the secure mode. We found that the translation quality enhanced. However, the quality was still poor due to the random value representation and the small size of the used dataset. Nevertheless, the aim was not to improve quality transmission but to obtain similar results in normal and secure modes while decreasing the translation time. In this regard, we could reach the same results in both modes, and the translation time was decreased by less than 5 h for each three-word sentence. Table 2 shows the first 43 results and the consumed time as an example in the normal and secure modes.

Table 2. Forty-three representative results and the consumed time in the normal and secure modes.

#	Source language	Target Language	Time Consumed in Normal Mode	Time Consumed in Secure Mode
1	Welcome to our meeting	مرحباً في الاجتماع	1 s	5 h
2	Towards the whole world	العالم بلتقي	1 s	5 h
3	Congratulations for this action	نهنئكم بالفعل	1 s	4 h
4	War is stupid	الحرب غاشمة	1 s	4 h
5	Best regard	افضل المتعلق	1 s	4 h
6	One calendar year	سنة واحدة على التقويم	1 s	4 h
7	Ugly face	دجاجة الوجه	1 s	4 h
8	See you later	اراك في الخطاب	1 s	4 h
9	I want a glass of water	أريد كأساً من الماء	1 s	5 h
10	What a bad weather!	ما أسوأ الطقس	1 s	5 h
11	My name is	اسم يكون	1 s	4 h
12	Egypt country	غير معروف دولة	1 s	4 h
13	Cat is black	القطة تكون	1 s	5 h
14	Great hope for a peaceful	عظيم امل في السلام	1 s	6 h
15	Free world	العالم حر	1 s	4 h
16	More than ever before	اكثر من قبل	1 s	6 h
17	Human history	تاريخ البشرية	1 s	4 h
18	Share a common destiny	تقاسم مشترك	1 s	5 h
19	Big or small	كبير صغير	1 s	5 h
20	Strong or weak	قوي أسبوع	1 s	5 h
21	Those who advocate	الذين محامي	1 s	5 h
22	Who believe in it	هو يصدق في هذه	1 s	5 h
23	Kind of man	الرجل عطوف	1 s	5 h
24	Regional considerations	اعتبارات إقليم	1 s	5 h
25	Wars begin	الحروب	1 s	4 h

26	The law of love	قانون الحب	1 s	5 h
27	The defenses of peace	دفاع السلام	1 s	5 h
28	To be egoistic	يكون غير معرف	1 s	5 h
29	The reality	الواقع	1 s	4 h
30	It is understandable	هو يكون مفهومة	1 s	5 h
31	Major powers	القوي	1 s	4 h
32	Should pursue objectives	تقديم استعراضاتها	1 s	5 h
33	Sovereign nations	سبباً وجيهاً لتخصيص الدولة	1 s	5 h
34	World leaders	زعماء العالم	1 s	5 h
35	Nelson Mandela's words	غير معروف كلمات	1 s	5 h
36	Decisions we make	القرارات الداعمة	1 s	5 h
37	Fail to make	تفشل في صنع	1 s	5 h
38	When we are ready	متي نحن مستعدون	1 s	5 h
39	Make possible lasting	من الممكن الاستمرار	1 s	5 h
40	Beautiful display resource	مورد جميل	1 s	5 h
41	Inspiring words from current	غير معروف من	1 s	5 h
42	Un secretary-generals	الإعلان العالمي	1 s	5 h
43	Our work will always be rooted	بحث عمل سوف النبات	1 s	5 h

CONCLUSION

In this study, we enhanced a previously proposed algorithm to decrease the time required for translation. We first expanded the database to include 31521 English and 29956 Arabic words. As the calculation time between steps 6 and 7 increased considerably because of noise expansion, by adding the refresh function before the 6th and 7th gates, we were able to control the noise and reduce the time required between each step by 15 min, achieving a more than 3 h reduction in the translation process. However, the algorithm's complexity remains high, necessitating the use of a supercomputer for faster performance, as the increased chains of calculations present significant computational challenges. While we ran the program on our laptop, the process was slow, often requiring the system to run for a week at a time. Additionally, verifying the accuracy of the algorithm and comparing it to real-life applications such as Google Translate requires a large dataset, which our laptop could not efficiently handle and would ideally need a supercomputer.

Author Contributions: Michael Lahzi Gaid Ibrahim: Writing – original draft. Khaled Shaalan: Supervision.
Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflicts of interest.

REFERENCES

[1] Mallaiah, K.; Sirandas, R. Applicability of homomorphic encryption and CryptDB in social and business applications: securing data stored on the third party servers while processing through applications. Int. J. Comput. Appl. 2014, 100, 5–19. DOI: 10.5120/17487-7999.
[2] Riza, A.; Tibben, W.; Win, K. Motives behind cyber security strategy development: A literature review of national cyber security strategy. In Australasian Conference on Information Systems, Wollongong, Australia, 2016.
[3] Manpreet, K.; Kaur, J. Data encryption using different techniques: A review. Int. J. Adv. Res. Comput. Sci. 2017, 8, 252–255.
[4] Lintz, N. Sequence modeling with neural networks (Part 1): Language & Seq2Seq. Indico 2011.
[5] Neubig, G.Neural machine translation and sequence-to-sequence models: A tutorial. 2017, arXiv preprint arXiv:1703.01619
[6] Lipton, Z.C. A critical review of recurrent neural networks for sequence learning. 2015, CoRR, abs/1506.00019.
[7] Armknecht, F.; Boyd, C.; Carr, C.; Gjøsteen, K.; Jäschke, A.; Reuter, C.A.; Strand, M. A guide to fully homomorphic encryption. IACR Cryptology ePrint Archive, 2015, 1192.

[8] Aslett, L.J.; Esperança, P.M.; Holmes, C.C.A review of homomorphic encryption and software tools for encrypted statistical machine learning. 2015, ArXiv, abs/1508.06574.
[9] Tebaa, M.; Hajil, S.E. Secure cloud computing through homomorphic encryption, Int. J. Advanc. Comput. Technol. 2013, 5, arXiv preprint arXiv:1409.0829
[10] Liu, D. Practical fully homomorphic encryption without noise reduction. IACR Cryptology ePrint Archive, 2015, 468.
[11] Gaid, M.L.; Fakhr, M.W.; Selim, G. Secure translation using fully homomorphic encryption and sequence-to-sequence neural networks. In 28th International Conference on Computer Theory and Applications (ICCTA) IEEE, Alexandria, Egypt, 30 October–1 November.
[12] Sepp, H.; Jürgen, S. Long short-term memory. Neural Comput. 1997, 9, 1735–1780. DOI: 10.1162/neco.1997.9.8.1735