Evaluating Federated Learning with Homomorphic Encryption for Medical Named Entity Recognition Using Compact BERT Models

Marcos F. Pontes [1], **Rodrigo C. Pedrosa** [1], **Pedro H. Lopes** [1], **Eduardo J. Luz,** [1]

[1] Departamento de Computação – Universidade Federal de Ouro Preto (UFOP)

marcos.rezende@aluno.ufop.edu.br,{rodrigo.silva,silvap,eduluz}@ufop.edu.br

Abstract. *Medical Named Entity Recognition (NER) identifies and categorizes medical entities from unstructured texts, crucial for health monitoring tasks. Despite advancements with Large Language Models (LLMs), medical NER faces challenges due to limited and dispersed labeled data across institutions, protected under privacy regulations. Federated Learning (FL) offers a solution by enabling decentralized model training while preserving data privacy, but it is vulnerable to byzantine attacks. This research proposes a simple and secure FL protocol using Homomorphic Encryption (HE), called FedHE, that removes the need of trust between the federations and the training coordinator. Encrypted FL imposes significant constraints regarding resources consumption and performance, making the state-of-the-art language models impractical. This research aims to assess how well compact BERT representations work in federated medical NER tasks in comparison to the state-of-the-art approaches. The results showed that compact BERT representations, such as $BERT_{mini}$ are competitive with the state-of-the-art, and are feasible to use in FedHE. However, resource consumption overheads remain a challenge, particularly when the number of clients increase.*

1. Introduction

Medical named entity recognition (NER) aims to identify medical entities (e.g., drug names, adverse reactions and symptoms) from unstructured medical texts and classify them into different categories. It can be used in many intelligent healthcare tasks such as pharmacovigilance and health monitoring [Tang et al. 2013]. With the recent advancements in the field [Peng et al. 2024] the problem of NER has seen significant improvements. However, in the specific context of medical NER, there are significant challenges in the learning process due to the sensitive nature of the data. First, the available labeled data of a single healthcare institution might not be representative enough to adjust a NER model with good predictive accuracy. Second, collaborative training with data sharing is frequently impractical considering the regulations prohibitions and the security risks associated to the data sensitiveness and trust between the parties.

To leverage massively distributed data and enhance model generalizability, federated learning (FL) was introduced in [Konečnỳ et al. 2016] as a novel learning framework. In an FL training loop, clients collaboratively train a shared global model by exchanging model weights or gradients while keeping their data stored locally. By bringing the model to the data, FL avoids data transfer and achieves competitive performance compared to models trained with pooled data.

Recently, [Peng et al. 2024] provided an in-depth evaluation of federated learning in biomedical natural language processing, demonstrating that the BlueBERT ($BERT_{blue}$) model, particularly its larger variant ($BERT_{large_blue}$), trained using FL outperforms both its version trained on data from a single client and GPT-4 when applied in a few-shot prompt setting. Clearly, FL, combined with variations of BERT, stands out as an effective approach for NER.

Although FL's primary focus is on maintaining rigorous privacy protections by preventing data sharing, [Zhu et al. 2019] introduced a new vulnerability in the form of inference attacks, showing that private training data can be extracted from the publicly shared gradients. To mitigate this risk, one approach is to incorporate an encryption step into the federated learning framework. Specifically, employing Homomorphic Encryption (HE) [Yi et al. 2014] within FL allows clients to encrypt their gradients, enabling the central coordinator to aggregate model updates directly on ciphertexts, thereby eliminating the need for decryption.

While HE is often considered the gold standard for data-in-use encryption, it imposes significant performance overheads in terms of both computation and communication. As a result, deploying state-of-the-art natural language models becomes impractical due to the large number of trainable parameters. Therefore, to implement a more secure FL system using HE, smaller models must be selected. In this context, this paper addresses two key questions: (i) What is the computational cost of applying HE in FL for NER applications? (ii) How much predictive accuracy might be sacrificed by choosing a more secure FL+HE approach with a smaller model? The obtained results showed that compact models, like $BERT_{mini}$, can perform competitively with state-of-the-art NER models in a FL+HE setting for different corpora. However, resource overheads — particularly communication bandwidth and memory utilization—continue to pose significant challenges.

2. Federated Learning

The prototypical FL setting consists of a central server S and a set of K distributed clients C, such that $|C| = K$, that jointly cooperate to solve a standard supervised learning task. Each client $c \in C$ has access to it's own private training set $\mathfrak{D}_c = \{x_{c,i}, y_{c,i}\}_{i=1}^{n_c}$. The goal of FL is to train a global predictive model whose architecture and parameters $\theta* \in \mathfrak{R}_d$ are shared amongst all the clients and found to minimize $min_\theta \sum_{c=1}^{K} p_c \mathfrak{L}_c(\theta; \mathfrak{D}_c)$, where $\mathfrak{L}_c$ is the local objective and $p_c \geq 0$ specifies the individual contribution of the client c such that $\sum_{c=1}^{K} p_c = 1$. Two possible configurations for p_c are $p_c = \frac{1}{K}$ or $p_c = \frac{n_c}{n}$, where $n = \sum_{c=1}^{K} n_c$.

The local objective function $\mathfrak{L}_c$ usually is defined as the empirical risk calculated over the training set $\mathfrak{D}_c$ sampled from the client's local data distribution $\mathfrak{L}_c(\theta; \mathfrak{D}_c) = \frac{1}{n_c} \sum_{i=1}^{n_c} \mathfrak{l}(\theta; (x_{c,i}, y_{c,i}))$, where $\mathfrak{l}$ is an instance-level loss (e.g., cross-entropy loss or squared error in the case of classification or regression tasks, respectively).

In Federated Learning, to generate a global model θ from locally trained models with parameters θ_c, an aggregation step is necessary to combine the updates from all clients. One of the most widely used methods for aggregation is called FedAvg. In each round t, clients perform local training steps on their private datasets $\mathfrak{D}_c$ to minimize their respective objective functions $\mathfrak{L}_c$. After completing the local updates, clients send their

updated parameters $\theta_c^{(t)}$ back to the central server S. The server then aggregates these updates by computing a weighted average, typically defined as $\theta^{(t+1)} = \sum_{c=1}^{K} p_c \theta_c^{(t)}$, and uses this to update the global model for the next training round.

2.1. Federated Learning with Fully Homomorphic Encryption

Homomorphic Encryption (HE) allows certain computations (e.g., addition) to be performed directly on ciphertexts, without decrypting them first. The intuitive idea is that a third party can compute data without actually getting to know that data. This problem is solved with key-based encryption where the encryption process preserves algebraic operations. For the addition operator, for example, we would have $e(k, a)+e(k, b) = e(k, a+b)$ for an encryption scheme $e(., .)$, encryption key k, and plaintexts a and b. A third party could thus compute the ciphertext of the value of the addition $a + b$ from the ciphertexts of a and b , and return this to the owner who could decrypt this to get the computation result on the plaintext [Al Badawi and Polyakov 2023].

In the context of FL, the viability of HE is particularly constrained when encrypting local model updates. The use of large language models, such as BERT with $110M$ parameters, becomes nearly infeasible given the bandwidth and computational overhead associated with processing encrypted gradients. This limitation underscores the need for more efficient encryption techniques, model compression strategies, and the adoption of more compact architectures— the latter being the focus of this paper's assessment

3. Methodology

This section describes this work's proposal for making federated medical NER secure with HE. The proposed solution, called FedHE, aims to be conceived as a generic framework on how HE can be used in FL, making them compliant with data privacy regulations and enabling scenarios such as medical NER to work without the risk of inference attacks.

The FedHE protocol uses HE encryption to protect the gradients data. Thus, even if byzantine attackers compromise the computing server, they don't have access to the information of the gradient data from each learning client. In addition, it is impossible for byzantine attackers to use these encrypted gradient data to train shadow models.

In this work, the cryptographic scheme CKKS (Cheon-Kim-Kim-Song) [Marcolla et al. 2022] is used to encrypt clients' gradients preserving the arithmetic operations of addition and multiplication by a scalar plaintext number. CKKS is an asymmetric cryptographic scheme that requires key pairs, so a key management service (KMS) is required. Notice that this work does not aim to detail neither the encryption scheme nor the KMS protocol, but we rely on strategies and algorithms publicly defined in the literature.

The coordinator algorithm orchestrates the federated network (See Algorithm 1). Usually, it defines how the protocol work, establish mechanisms to define the architecture, guarantee trust between the clients, and aggregate the locally generated gradients.1

The FedAvg algorithm is executed homomorphically, without decryption of clients updates. Additionally, although the KMS strategy is not specified in this paper, we assume the coordinator has only access to the public key.

Algorithm 1: *FedHE Coordinator*. The K clients are indexed by $c \in C$, T is the total of federated learning rounds and L is the loss function. The goal is to obtain $\theta*$ that minimizes the clients' loss function.

State: Local model with parameters θ_i.
Function `ServerTrain`**:**
  initialize $\theta^{(0)}$
  request public key from KMS
  for *each round t* = $0, \ldots, T$ **do**
    number of clients: $m \leftarrow max(C \cdot K, 1)$
    client selection: $S_t \leftarrow$ (random set of m clients)
    for *each client* $c \in S_t$ ***in parallel*** **do**
      $\nabla enc(L_c^{(t+1)}) \leftarrow ClientTrain(c)$
    end
    homomorphic *FedAvg:* $\nabla enc(L^{(t+1)}) \leftarrow \sum_{c=1}^{K} \frac{n_c}{n} \nabla enc(L_c^{(t+1)})$
    for *each client* $c \in C$ ***in parallel*** **do**
      $ClientUpdate(c, \nabla enc(L^{(t+1)})$
    end
  end

The FedHE client training algorithm is where the actual train happens (See Algorithm 2). Each client trains on their own private training dataset and share only the encrypted gradient updates with the coordinator for model aggregation.

Algorithm 2: *FedHE Client*. X represents the training samples while Y represents the training labels. I, ϵ, η represents the number of local epochs, the tolerance and the learning rate, respectively. The goal is to obtain $\theta*$ that minimizes the loss function L.

State: Local model with parameters θ_i.
Function `ClientTrain`**:**
  request public key from KMS
  for *each epoch i* = $0, \ldots, I$ **do**
    forward propagation: $\hat{Y}_i = forward(X, \theta_i)$
    compute loss: $L_i = loss(Y, \hat{Y}_i)$
    if $L_i < \epsilon$ **then**
      break
    end
    else
      back propagation: $\nabla L_i = backprop(X, \theta_i, L_i)$
      gradients encryption: $\nabla enc(L_i) = encrypt(\nabla L_i, PublicKey)$
      return $\nabla enc(L_i)$
    end
  end
Function `ClientUpdate(`$\nabla enc(L_{agg})$`)`**:**
  request private key from KMS
  gradients decryption: $\nabla L_{agg} = decrypt(\nabla enc(L_{agg}), PrivateKey)$
  update: $\theta_{i+1} = \theta_i - \eta \nabla L_{agg}$

In conclusion, the FedHE [1] protocol is adaptable to a range of model architectures and can be seamlessly integrated into established FL platforms like Flower [Beutel et al. 2020], TensorFlow Federated[2], FATE[3], among others.

4. Results

In this section, we present the key findings of our analysis on FedHE, focusing on two practical aspects: (1) the performance of FedHE trained with compact BERT models compared to state-of-the-art models, and (2) the performance and resource consumption overheads associated with FedHE.

Named Entity Recognition Corpora

We compared FedHE with alternative training schemes on two biomedical NLP datasets and one news dataset, focusing on NER tasks. In NER, the objective is to identify and classify named entities, such as diseases and genes, from a given sequence of tokens.

The selected corpora were chosen based on two main criteria: they are publicly available, ensuring the reproducibility of results, and they are commonly used in well-cited papers, which helps guarantee the quality of the data. A summary of the selected datasets can be found in Table 1.

Corpus	Entity/Relation Type	Corpora Type	Train	Dev	Test
CONLL-2003	General	News articles	14987	3466	3684
BC2GM	Gene	Medline abstract	26006	3251	3251
BC4CHEMD	Drug/Chem	PubMed abstract	94170	11772	11771

Table 1. List of NER corpora and their statistics

What Are the Performance and Resource Overheads of FedHE?

In FedHE, the encryption of gradients introduces a substantial increase in data size, which can significantly impact bandwidth. Table 4 provides a comparative analysis of the size overhead associated with different BERT models when using the CKKS encryption scheme. For instance, $BERT_{tiny}$, which has a plaintext gradient size of 16 MB, increases to 340 MB when encrypted. Similarly, $BERT_{mini}$'s gradient size grows from 42 MB to 864 MB under the same scheme. The most pronounced effect is seen with $BERT_{blue}$ and $BERT_{large\_blue}$, where the gradient size were 20 times and more than 50 times bigger, respectively. While local training remains unaffected, these increases in ciphertext size lead to significant bandwidth overheads. As such, models like $BERT_{blue}$ become impractical for FedHE due to the prohibitive size of encrypted gradients, emphasizing the need for more bandwidth-efficient approaches or smaller models to maintain feasibility in federated settings.

Table 4 highlights the impracticality of Large Language Models in FedHE due to their exponential growth of memory and bandwidth requirements. Table 3 shows that

[1]Source code: `https://github.com/marcosfpr/fedhe` and `https://github.com/marcosfpr/sealy`.

[2]`https://www.tensorflow.org/federated`

[3]`https://fate.fedai.org/`

Model	Scheme	# Params	Size
$BERT_{tiny}$	Single/Central/Federated	4M	16 MB
	FedHE	4M	340 MB
$BERT_{mini}$	Single/Central/Federated	11M	42 MB
	FedHE	11M	864 MB
$BERT_{blue}$	Single/Central/Federated	108M	415 MB
	FedHE	108M	8 GB
$BERT_{large\_blue}$	Single/Central/Federated	344M	1GB
	FedHE	344M	$>$50 GB

Table 2. Model Parameters and Size for Different Schemes

while operations on encrypted gradients, particularly encryption, become more costly with increased parameter sizes, these do not generally pose a bottleneck in training. However, if federated training involves frequent aggregation rounds and infrequent local training epochs, these operations could become a significant bottleneck when the number of parameters is sufficiently large. Typically, clients perform extensive local training with less frequent aggregations, which aligns with both performance and operational efficiencies in FedHE.

Due to $BERT_{large\_blue}$'s excessive memory demands—over 50 GB in FedHE and 1 GB in plaintext—along with significant bandwidth and processing requirements, this work will focus on comparing the effectiveness only with the base $BERT_{blue}$ model instead. Future work can be done to address the comparsion with larger versions of BERT such as $BERT_{large\_blue}$.

Model	Type	Mean (s)	Std. Dev. (s)	99th Percentile (s)
$BERT_{tiny}$	Encrypt	8.016	0.133	8.484
	Decrypt	2.179	0.053	2.337
$BERT_{mini}$	Encrypt	21.877	0.251	22.671
	Decrypt	5.884	0.095	6.144

Table 3. Summary Statistics for Encryption and Decryption Times of BERT Models

While Table 4 highlights significant bandwidth constraints on the server-side in FedHE, it is also essential to evaluate the resource and time costs associated with aggregation operations. Figure 4 sheds light on the performance of aggregation as the number of clients increases for the $BERT_{tiny}$ model using the $BC2GM$ corpus. The analysis indicates that the homomorphic FedAvg aggregation time does not present an efficiency issue in the training process; specifically, $BERT_{tiny}$ completes aggregation in approximately 20 seconds with 22 clients, whereas $BERT_{mini}$ requires about 50 seconds with 14 clients. However, it is important to note that as the number of clients grows, the memory required to store ciphertext gradients increases significantly. For instance, aggregation for $BERT_{mini}$ with 16 clients led to a coordinator crash due to memory insufficiency. While such issues can be mitigated using external memory strategies, these solutions introduce additional performance overhead.

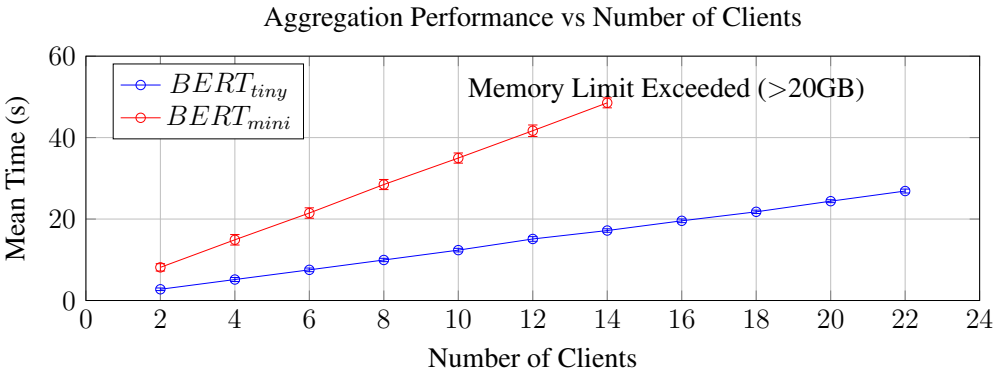

Figure 1. Aggregation performance for BERT models on BC2GM.

How Does FedHE with Compact BERT Models Compare to State-of-the-Art BERT Models For Medical NER?

Another fundamental research question for this work is to understand how far compact BERT models are from the state-of-the-art models for federated medical NER. In particular, we would also like to understand if the introduction of HE can skew the overall results. The table 4 shows an F1 comparison of the approaches tested with the state-of-the-art models.

Another critical research question addressed in this work is evaluating the performance gap between compact BERT models and state-of-the-art models for federated medical NER. Additionally, we investigate whether the integration of HE affects the overall performance outcomes. Table 4 provides a comparative analysis of F1 scores for the evaluated approaches against the state-of-the-art models.

Model	Method	CONLL-2003		BC2GM		BC4CHEMD	
		Train	*Eval*	*Train*	*Eval*	*Train*	*Eval*
$BERT_{tiny}$	Single	0.865 ± 0.005	0.618 ± 0.006	0.804 ± 0.002	0.593 ± 0.001	0.537 ± 0.003	0.391 ± 0.047
	Central	0.953 ± 0.001	0.728 ± 0.002	0.841 ± 0.000	0.645 ± 0.003	0.605 ± 0.011	0.460 ± 0.022
	Federated	0.624 ± 0.001	0.464 ± 0.005	0.726 ± 0.010	0.613 ± 0.016	0.598 ± 0.010	0.448 ± 0.008
	FedHE	0.816 ± 0.000	$\mathbf{0.650 \pm 0.014}$	0.744 ± 0.005	0.604 ± 0.010	0.621 ± 0.004	$\underline{\mathbf{0.464 \pm 0.005}}$
$BERT_{mini}$	Single	0.961 ± 0.002	0.690 ± 0.002	0.802 ± 0.000	0.594 ± 0.002	0.802 ± 0.000	0.538 ± 0.004
	Central	0.990 ± 0.000	0.787 ± 0.005	0.995 ± 0.001	0.763 ± 0.010	0.859 ± 0.001	0.613 ± 0.002
	Federated	0.993 ± 0.000	0.758 ± 0.013	0.958 ± 0.001	0.703 ± 0.012	0.833 ± 0.000	0.584 ± 0.000
	FedHE	0.994 ± 0.000	$\mathbf{0.781 \pm 0.001}$	0.998 ± 0.000	$\mathbf{0.739 \pm 0.004}$	0.877 ± 0.000	$\mathbf{0.590 \pm 0.001}$
$BERT_{blue}$	*Central*	0.999 ± 0.000	0.791 ± 0.009	0.992 ± 0.001	0.758 ± 0.041	0.968 ± 0.001	0.683 ± 0.000

Table 4. F1 Score comparison of FedHE with various BERT models on medical NER datasets. Standard deviations are shown in parentheses. Bold indicates FedHE surpasses Federated, while underscored indicates it surpasses Centralized evaluation.

Training Setup

In all experiments we ran 50 epochs for training the models. The centralized and single-client learning, we conducted 50 local epochs on their private dataset. The single-client data were obtained splitting the dataset in two parts, and taking only one from the corpora.

The federated and FedHE approaches ran 5 aggregation rounds with 10 local epochs each on client data. Effectiveness tests, shown in Table 4, were conducted with

2 clients. Standard deviations for federated and FedHE were calculated from all clients, while single-client and centralized deviations were from 2 runs.

For training the models, we used Adam optimizer with an initial learning rate of $2e-5$ and weight decay of 0.1. All experiments were performed on a system equipped with an NVIDIA A100 GPU and at least 32GB RAM available.

Discussion

The results in Table 4 highlight the performance of different BERT models in 4 different configurations (centralized, single client, federated and FedHE) for medical NER tasks. The centralized $BERT_{blue}$ model is used as a baseline, representing the state-of-the-art in BERT-based models for medical named entity recognition. This model sets a high standard for comparison, demonstrating its accuracy across the datasets.

Importantly, the application of HE does not skew model effectiveness. On the contrary, the encryption noise introduced by the encryption and decryption processes does not damage model accuracy. Instead, it sometimes even improves performance, as reflected in the bolded values in the table. The strongest hypothesis for this fact is that the small noise added by the ciphertext operations helped the model to generalize better. This indicates that HE can be effectively integrated without compromising, and potentially enhancing, the model's performance.

The analysis also reveals that single-client learning models, such as $BERT_{tiny}$ and $BERT_{mini}$, often achieve higher training accuracy, but generalize with less effectively compared to federated and FedHE approaches. Federated learning and FedHE models exhibit superior generalization in all corpora evaluated.

$BERT_{tiny}$ shows lower performance compared to $BERT_{mini}$, with signficant differences in all 4 methods tested for all corpora. $BERT_{mini}$, using only 11M parameters, presented satisfactory results even when compared to the more complex $BERT_{blue}$ with 108M parameters. This suggests that we can achieve results closer to state-of-the-art using compact BERT representations without making FL+HE impractical. This work also suggests that evaluating other slightly more complex BERT variants, such as $BERT_{small}$, could provide additional insights and potential improvements in model performance.

5. Conclusion

Overall, FedHE shows a generic framework for integrating HE in a FL protocol as a strong alternative for federated medical NER tasks. FedHE offers robust performance and practical advantages, making it a compelling choice for scenarios where data privacy and model effectiveness are critical. The results underscore the viability of FedHE in maintaining high performance while incorporating encryption techniques. For future work, we highlight (1) study scenarios where the number of clients is higher and the data is non-IID; (2) assess the feasibility of other compact BERT variants such as $BERT_{small}$ and (3) test the models against other LLM-based baselines such as $BERT_{large\_blue}$.

6. Acknowledgements

The authors would also like to thank the Universidade Federal de Ouro Preto (PROPPI/UFOP) for supporting the development of this study.

References

Al Badawi, A. and Polyakov, Y. (2023). Demystifying bootstrapping in fully homomorphic encryption. *Cryptology ePrint Archive*.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., Li, K. H., Parcollet, T., de Gusmão, P. P. B., et al. (2020). Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F. H., and Aaraj, N. (2022). Survey on fully homomorphic encryption, theory, and applications. *Proceedings of the IEEE*, 110(10):1572–1609.

Peng, L., Luo, G., Zhou, S., Chen, J., Xu, Z., Sun, J., and Zhang, R. (2024). An in-depth evaluation of federated learning on biomedical natural language processing for information extraction. *npj Digital Medicine*, 7(1):127.

Tang, B., Cao, H., Wu, Y., Jiang, M., and Xu, H. (2013). Recognizing clinical entities in hospital discharge summaries using structural support vector machines with word representation features. In *BMC medical informatics and decision making*, volume 13, pages 1–10. Springer.

Yi, X., Paulet, R., Bertino, E., Yi, X., Paulet, R., and Bertino, E. (2014). *Homomorphic encryption*. Springer.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems*, 32.