Privacy-preserving Data Publishing Using Deep Learning Techniques

by

Tanbir Ahmed

A thesis submitted to
The Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
December 2020

Thesis advisor

Noman Mohammed

Author

Tanbir Ahmed

Privacy-preserving Data Publishing Using Deep Learning Techniques

Abstract

According to a recent study, around 99 percent of hospitals across the United States now use electronic health record systems (EHRs) [1]. One of the most common types of EHR data is the unstructured textual data [2], and unlocking hidden details from this data is critical for improving current medical practices and research endeavors. However, these textual data contain sensitive information, which could compromise our privacy. Therefore, medical textual data cannot be released publicly without any privacy protection. De-identification is a process of detecting and removing all sensitive information present in EHRs, and it is a necessary step towards privacy-preserving EHR data sharing. Since 2016, we have seen several deep learning-based approaches for de-identification, which achieved over 98% accuracy. However, these models are trained with sensitive information and can unwittingly memorize some of their training data, and a careful analysis of these models can reveal patients' data. This thesis presents two contributions. First, We introduce new methods to de-identify textual based on self-attention mechanism and stacked Recurrent Neural Network. Experimental results on three different datasets show that our model performs better than all state-of-the-art mechanisms irrespective of the dataset. Addi-

tionally, our proposed method is significantly faster than existing techniques. We also introduced three utility metrics to judge the quality of the de-identified data. Second, we propose a differentially private ensemble framework for de-identification, allowing medical researchers to collaborate through publicly publishing the de-identification models. Experiments in three different datasets showed competitive results compared to the state-of-the-art methods with guaranteed differential privacy.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to begin by thanking my advisor, my committee, my parents, my significant other, and all the people who have supported me along the way.

This thesis is dedicated to my father, who I wish were with me today.

Chapter 1

Introduction

The advent in computational power and low storage cost has created the avenue for a systematic electronic collection of health information about patients. These records mostly contain patient information such as medical history, medication orders, vital signs, laboratory results, radiology reports, physician and nursing notes, etc. Recent studies suggest that these integrated electronic health records can improve patient safety against medication error and adverse drug events, help with the short-term preventive care and near-term chronic disease management, and save the overall healthcare cost [3]. In hope of utilizing these benefits, the Health Information Technology for Economic and Clinical Health (HITECH) Act was signed as a law in 2009, designed to ensure access to health care facilitated by Electronic Health Record (EHR) systems. In 2016 about 95% of all eligible hospitals and 62% of all office-based physicians were reported to use Electronic Health Record systems [4]. This widespread adoption of the Electronic Health Record system created valuable medical data worth studying.

However, medical researcher and organizations are not free to use EHRs as they contain sensitive patient identifying information, which is commonly private. To preserve the patients' confidentiality, the U.S. Health Insurance Portability and Accountability Act (HIPAA) requires that 18 re-identifying categories of information to be removed from medical records before they are disseminated [5]. These confidential information categories are referred to as the Protected Health Information (PHI) which include patients name, their profession, unique identifying numbers like social security or medical number etc. In this Thesis work, we addressed the problem of publicly releasing EHR and models built upon those while withholding the all PHI instances outlined by HIPPA.

For our first task, we studied de-identificaiton, the process of detecting and removing the PHI instances from the textual EHRs. De-identification is the necessary step for making the EHRs available for research or public usage, and it has attracted substantial interest in the research community. Over the last decades, a considerable amount of effort has been devoted to this problem, including the use of human annotators to manually de-identify the EHRs. In 2005, Douglass *et al.* [6] reported a cost of $50/hour where human annotators read around 20000 words/hour. At this rate, a dataset having 100 million words would cost an astonishing amount of $250,000. In 2008, Nematullah [7] reported the recall value ranged from 63% to 94% depending on the annotators when he asked 14 annotators to de-identify approximately 130 patient notes. Therefore, manual de-identification, which is also prone to errors, is deemed a costly and time-consuming approach for de-identification.

The shortcomings of manual de-identification formed a natural progression to-

wards automated systems, and there are several proposed approaches to de-identify EHRs automatically. These existing automated de-identification systems are predominantly rule-based [6] or employ machine learning algorithms [8]. The rule-based systems rely primarily on the word patterns captured using pre-defined regular expressions and dictionary lookups and do not require any labeled data. Although these systems are easy to develop, they do not generalize well as the rules have to be fine-tuned for each dataset. Also, rule-based systems do not take the context of the words into account. The deficiencies of the rule-based systems are mostly mitigated by the use of supervised machine learning approaches. These machine learning systems are trained as a classifier model, where each word is labeled as PHI or not, often identifying the different PHI types as well. With all their efficacies and conveniences, these systems still have one common problem: the dependency on handcrafted features, which are challenging and time-consuming.

Recent approaches in Natural Language Processing (NLP) tasks such as Named-entity Recognition (NER) and Parts of Speech (POS) tagging [9] using non-linear neural networks have shown promising results without any handcrafted features or rules. The features in these systems are learned automatically with other parameters of the network during training on a labeled dataset. However, all of these approaches used the variants of Recurrent Neural Networks (RNNs). More specifically, using bidirectional Long Short Term Memory (LSTMs) for determining the correlation between the words in unstructured textual EHRs. Since 2017, several new models for neural language modeling have been proposed [10, 11], which are built entirely with attention layers, without convolution or recurrence. These papers report new state-of-

the-art F1 scores for various language modeling task, including NER. Theoretically, the new self-attention based approaches should also improve the de-identification results. As our first task, we evaluate the performance of self-attention model on the de-identification task. To the best of our knowledge, we are the first to employ self-attention mechanism to the de-identification problem. We also propose an architecture employing multiple neural networks for de-identification and analyze them in different settings.

While these solutions report accuracy over 98% for the de-identification problem, these deep learning models themselves have an innate privacy concern. Although the deep learning models for de-identification employed regularization techniques to avoid overfitting and unintentionally obfuscate the details of the training data, the vector representations used in the hidden layers of deep learning models are hardly explainable. The superiority of deep learning models implies that these representations may potentially memorize finer details of at least some of the training data [12]. The work of Coavoux *et al.* [13] in 2018, supports this claim as they showed that private variable could be learned from the hidden representations of models trained for text classification and sentiment analysis. Recently, Pan *et al.* [14] showed two novel attacks to steal sensitive information from the sentence embeddings being used in general purpose language models(e.g. BERT [11]). Before that, Fredrikson *et al.* demonstrated a model-inversion attack that recovers images from a facial recognition system [15]. The existing de-identification models are very similar in construction to the Natural Language Processing (NLP) models used by Coavoux *et al.* [13] and are therefore exposed to similar attacks. Thus, the opportunity of reproducibility and transfer

learning from the current de-identification models is lost.

One possible solution to this problem of de-identification model publishing is to reduce the sensitivity of these models towards the training data. In other words, training the models in a way that when any individual record is removed from the training set, the change on the result is negligible. Thus, reducing the dependency on any particular training data. As mentioned earlier, regularization techniques have the same effect to some extent; however, these techniques do not provide any quantifiable rigorous privacy guarantees. differential privacy [16] has established itself as a rigorous standard for providing privacy guarantees. Consequently, a considerable amount of attempts have been made to provide machine learning models with differential privacy. For instance, Chaudhuri *et al.* [17] added a noise vector to a logistic regression classifier to achieve differential privacy. Pathak *et al.* [18] proposed a multi-party aggregation classifier protocol where individual classifiers are averaged and added with a stochastic noise component. Both, Song *et al.* [19] and Shorki *et al.* [20] proposed a differentially private Stochastic Gradient Descent (SGD) [21] algorithm. However, the privacy bounds given by Shorki *et al.* [20] were for each parameter in the model, which made the privacy guarantees for the larger model virtually non-existent. Abadi *et al.* [12] proposed a differentially private SGD that uses stricter bounds on the privacy loss. Hamm *et al.* [22] proposed the use of knowledge transfer between a collection of models trained on individual devices into a single model that guaranteed differential privacy. Their work further adopted for more generalized applications by Papernot *et al.* [23] in 2016. Both Abadi *et al.* [12] and Papernot *et al.* [23] proposed such algorithms that reduce the footprint of individual training data by

sub-dividing tasks into smaller batches. For our second task, we utilize the differentially private [16] training techniques to publish the de-identification models. We introduced a framework that allows medical research entities (both individuals and institutions) to publish trained de-identification model for future public usage.

1.1 Contribution

Our contributions in this research work could be described in the two following subsections.

1.1.1 Deep Learning Based De-identification Model

In our first task we proposed an architecture to de-identify EHRs using different deep neural networks: i) bi-directional Gated Recurrent Units (GRU) [24] ii) Stacked Recurrent Neural Network (RNN) [25] structure combining GRUs and/or Long Short Term Memory (LSTM) [26] units and iii) finally, replacing the recurrent units with only a self-attention based mechanism. Notably, these approaches employ two different embedding schemes (fixed and dynamic) that are not considered by earlier attempts. We also conducted multiple experiments to compare the results with the existing state-of-the-art approach [27] on three standard de-identification datasets. The main contributions of this work are summarized as follow:

Utility Metric

We introduced two general purpose utility metrics and one specific application to judge the quality of the de-identified data. To the best of our knowledge, this

is the first article to introduce utility metrics for a de-identified textual dataset. Experimental results showed that our de-identified documents have comparable utility in different settings compared to the state-of-the-art.

Scalability

We analyzed the performance of self-attention mechanism and compared the results to other RNN based models. Experimental results show that the proposed model requires significantly less time to train and predict compared to the existing models. This is significant since deep learning-based algorithms are computationally expensive and often do not scale well for larger datasets. Experiments on three different datasets show that our model can perform well in both smaller (i2b2 2014) and larger (Nursing Note, MIMIC-III) datasets.

Accuracy

Our proposed methods achieve 85.9% F1-score (4% improvement) compared to the state-of-the-art method [27] for the Nursing Notes. Furthermore, our methods outperform the predecessors by attaining 99.97% F1-score (0.03% improvement) and 98.22% F1-score (0.4% improvement) on the MIMIC-III and i2b2 dataset, respectively.

1.1.2 Differentially Private De-identification Framework

We propose a private ensemble network for de-identification. Whereas before the trained models were unsafe to publish, by using a differentially private training

method, any number of the trained model could be added to this ensemble. We showed that by using a minimality aggregation algorithm, the recall value for the de-identification task could be improved. We also outlined two different scenarios: hospitals that (i) do and ii) do not have any labeled dataset where the PHIs are marked in the EHR texts and demonstrate the efficacy of private ensemble. We achieved over 97% F1-score on datasets i2b2 2014 [28] and MIMIC-III [29], and over 79% F1-score on the Nursing Note [30] dataset. Below, we are summarizing our contributions.

Differentially Private Model

To the best of our knowledge, this is the first work to use differentially private deep learning models for the de-identification of unstructured EHR.

Ensemble Network

We demonstrate the feasibility of combining multiple differentially private de-identification models in an ensemble network to achieve a competitive performance to overcome the limited data challenge.

Accuracy

We utilize our private ensemble network in two different scenarios that achieved a 97.9% and 75.6% recall in the i2b2 2014 [28] and Nursing Note [30] dataset, respectively, which are an incremental improvement over the state-of-the-art approaches.

1.2 Thesis Organization

This Thesis is organized as follows:

- **Chapter 2** provides the necessary machine learning and neural network background needed to understand the techniques adopted to privately publish unstructured data.
- **Chapter 3** presents a brief discussion on the related literature.
- **Chapter 4** describes the deep learning models used for the de-identification process. The following two publications resulted from this chapter.
  - **Ahmed, T.**, Aziz, M.M.A., and Mohammed, N. De-identification of electronic health record using neural network. NATURE Sci Rep 10, 18600 (2020). https://doi.org/10.1038/s41598-020-75544-1
  - **Ahmed, T.**, and Mohammed, N. Privacy-Preserving Medical Text Data Publishing with Machine Learning. Encyclopedia of Machine Learning and Data Science, to appear.
- **Chapter 5** describes the differentially private de-identification framework. The following publication resulted from the chapter.
  - **Tanbir Ahmed**, Md Momin Al Aziz,, Noman Mohammed and Xiaoqian Jiang. 2021. Privacy Preserving Neural Networks for Electronic Health RecordsDe-Identification. In 12th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (BCB '21), Au-

gust 1–4, 2021,Gainesville, FL, USA.ACM, New York, NY, USA, 6 Pages, https://doi.org/10.1145/3459930.3469555. To appear.

- **Chapter 6** concludes the Thesis providing some future research directions.

Chapter 2

Background

In this chapter we discuss some of the necessary background to understand the Thesis. We discuss the structure of the medical data targeted here in Section 2.1. In Section 2.2, we discuss the relevant machine learning techniques utilized for the proposed methods. Section 2.3 outlines the background for the differential privacy.

2.1 Unstructured Medical Data Processing

As the Thesis deals with unstructured textual medical data, we discuss the details about them in the upcoming sections:

2.1.1 Electronic Health Record (EHR)

Since the ratification of the HITECH Act in February 2009, there has been a significant increase in healthcare facilities that adopted Electronic Health Records (EHR). Consequently, as of 2017, 96% hospitals in the United States now store EHRs

using software-based systems [4]. The adoption of EHR is stipulated to make the patient-centered information accessible instantly and securely to authorized users. Although EHRs primarily contain patients' medical and treatment histories collected from healthcare providers' offices, EHR systems are built to cover an expansive range of patients' healthcare. EHRs can contain a patient's previous medical record like their immunization dates, history of illness, medications used, allergies to be aware of in future treatment, radiology images captured during illness, laboratory test results, etc. It provides the healthcare providers with access to evidence-based tools that can be used while deciding on a new patient's care with similar symptoms. Thus a well-designed EHR system can automate and streamline provider workflow. EHR creates a more seamless flow of healthcare information within the digital medical infrastructure. The mandatory adoption of EHR has the ability to leverage the ongoing electronic and computational revolution to transform how healthcare is delivered to the people in need.

EHRs include a wide variety of data forms. A few of these allows system designers to store them as a structured data form. However, about 80% of all the EHR stored does not allow a structured/relational data structure—the patient's clinical notes, discharge summaries, nursing notes, etc., are examples of such unstructured textual medical data. These contain a significant amount of information about a patient's medical history. This Thesis focuses on these unstructured data due to their considerable importance in almost all medical research branches. We contrived an example of patient note of Mr. Xander Horvath in Figure 2.1a. Due to the privacy restrictions we could not add an example directly from the original dataset. However we tried

to maintain similarity to an original patient note. Although this example may give an impression of an EHR's structure, not all documents follow any of this predefined construction based on stylometry. The primary objective of our research work is to preserve the privacy of the patients and medical practitioners from this unstructured format.

Table 2.1: HIPPA defined PHI Category and Sub-Categories.

PHI Instances	
Category	**Sub-Category**
DATE	NA
NAME	DOCTOR, PATIENT,USERNAME
AGE	NA
CONTACT	PHONE, FAX,EMAIL
ID	MEDICALRECORD, IDNUM, DEVICE, BIOID
LOCATION	HOSPITAL, CITY, STATE, STREET, ZIP, COUNTRY, ORGANIZATION, LOCATION-OTHER
PROFESSION	NA

2.1.2 Protected Health Information (PHI)

EHRs opens up the opportunity for providing access to large dataset of patient healthcare. Medical and healthcare researchers from all schools of study could be benefited from this access. However, the EHRs usually contain sensitive information about the patients, such as name, date of birth, address, date of admission, diagnosed diseases, name of caregivers, etc. These are private information and termed as Protected Health Information (PHI). As mentioned before, it is unlawful to share these notes publicly as they are confidential to the patient and the healthcare institution. To share these EHRs, the data holders need to remove these identifying information,

(a)

PATIENT NOTE

Name: Mr. Horvath, Xander
Record Date: 2019-05-07
MRN: 9345612
Date of Birth: 1921-03-17
Sex: M
History of Present Illness: Mr. Horvath, 49 years old, male, is seen today. He came from Hallstatt, Hungery, complaining of abdominal pain and indigestion. The dull, constant pain is located in the upper right quadrants of his abdomen. He has stopped his Prednisone on his own because he was gaining weight. The pain started on three days ago on 4^th May, 2019 and is occasionally also in his shoulder blade. Rolaids used to work on the pain. Although did not help in the last incident. I have not seen him since 29^{th} December, 2018.

PMH: no significant PMH

Family HX: His sister, Alice 63 yo woman had her gall bladder removed 2 years ago. Father died of alzheimers and mother had a heart attack last year.

O: Vital Signs: **BP** 125/85 **P** 70bpm **R** 16 breaths **T** 99.1^o F

Isla Kazlauskas, MD, EHMS, pager 09876

(b)

PATIENT NOTE

Name: [**Name**]
Record Date: [**Date**]
MRN: [**Medical Record**]
Date of Birth: [**Date**]
Sex: M
History of Present Illness: Mr. Horvath, [**Age**] years old, male, is seen today. He came from [**Location Other**], complaining of abdominal pain and indigestion. The dull, constant pain is located in the upper right quadrants of his abdomen. He has stopped his Prednisone on his own because he was gaining weight. The pain started on three days ago on [**Date**] and is occasionally also in his shoulder blade. Rolaids used to work on the pain. Although did not help in the last incident. I have not seen him since [**Date**]

PMH: no significant PMH

Family HX: His sister, [**Name**] [**Age**] woman had her gall bladder removed 2 years ago. Father died of alzheimers and mother had a heart attack last year.
O: Vital Signs: **BP** 125/85 **P** 70bpm **R** 16 breaths **T** 99.1^o F

[**Doctor*], MD, [**Hospital**], pager [**Phone**]

Figure 2.1: (a) Example of a Patient Note and (b) De-identified Patient Note

as stipulated by the HIPAA safe harbor method [31], which specified 18 categories of information, as mentioned in Table 2.1. A de-identification mechanism will annotate

the EHRs with these categories and remove them before dissemination.

Figure 2.1a shows an example of a patient note containing PHI instances (underlined) that could be used to identify the patient if released publicly. Furthermore, the document also contains the caregiver's name and contact, which are also deemed sensitive for the healthcare institution. On the contrary, Figure 2.1b shows how this sensitive information should be de-identified as we removed the sensitive words. Here, the categories `DATE, PATIENT, MEDICALRECORD, PHONE, HOSPITAL` are defined by HIPAA [5]. We show the total list of PHI categories and sub-categories defined by HIPAA in Table 2.1.

2.1.3 Privacy Model for Textual EHR

In this Thesis, we study the privacy of unstructured textual medical record. Our research involves identifying (and removing) the PHI instances outlined by HIPAA [31] from these records. These medical records are reported by a physician, nurse or lab technician, and contain sensitive information of patients such as patients' family and social history, medical encounters, demographics, etc. Notably, as this is a free-form text without any predefined structure, it solely depends on the practitioner (i.e., doctor, nurse) how s/he writes the patient or nursing notes. Therefore, we follow the 'HIPAA Safe Harbour Method' [32] as our sole privacy model. Please note that existing privacy models (e.g., differential privacy, k-anonymity) cannot be used to release textual data. By definition, the release of raw textual data will violate the definition of differential privacy. Indeed, if the goal is to release partial information (e.g., frequent keywords), then it is possible to satisfy privacy definitions like differential

privacy. The goal is to find techniques to release EHRs without tokens (or words) of categories outlined by HIPAA.

2.2 Machine Learning Background

2.2.1 Recurrent Neural Network (RNN)

RNN is more suitable for processing sequential data like sentence or documents. RNNs preserve the sequential information in hidden states, which cascades many time steps forward effecting the output at each time step. Thus, RNNs create long term dependencies between events separated by multiple time steps. Let's assume that at time step t an RNN cell takes token x_t and information about the previous time step h_{t-1} as input. It preserves the information learned in time step t in hidden state h_t and moves forward this information to the next time step. At time step t, output y_t is as follows:

$$h_t = \tanh(W_h \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix})$$

$$y_t = W_y h_t$$

Here, W_h and W_y are the weight matrices for the RNN unit. These weight matrices are updated by a learning rule. The change in the values represents the change in the error of the network and is determined during backpropagation by calculating the partial derivative of the error with regard to the weight matrices. This partial derivative is called *gradient*.

Long Short Term Memory

RNN correlates information between two sequential time steps through multiplication. Here, information passes through many stages of multiplications while RNN tries to establish a correlation between the final output and remote input. This makes gradients susceptible to vanishing or exploding. Although in theory RNNs are capable of capturing long term dependencies, due to this gradient vanishing/exploding problem they fail to do so.

LSTMs are a variant of RNN, which solve this issue by introducing a memory unit in the single RNN cell in addition to the hidden states [26]. An LSTM cell also has four differentiable gates implemented with element-wise multiplication by the sigmoid function. These gates determine the information flow in the memory unit and to the next time step. Let's consider a single LSTM cell that takes x_t, h_{t-1} and c_{t-1}, and generates hidden state h_t, memory unit c_t at t step as follows:

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Here, i, f, o, g represent the gates used in a LSTM cell. i and f determine whether to write or erase information from the memory cell, respectively. On the other hand, the g gate determines how much information should be written in the memory cell,

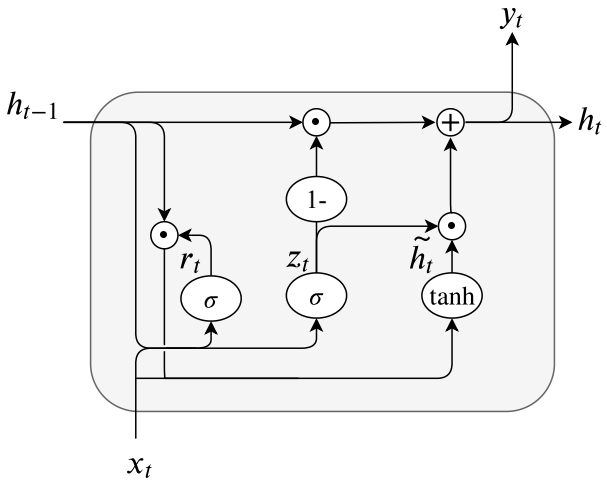


Figure 2.2: Example of a GRU Cell with $\sigma, tanh$ activation functions and x_t, y_t as the input and output, respectively. h_{t-1} is the hidden state from previous time step and the output is forwarded to the next time step as h_t.

and o decides the amount of saved information sent as output. These decisions are regulated by the weight matrices W and learned during training.

Symbols σ and tanh refer to element-wise sigmoid and hyperbolic tangent functions, and $\odot$ is an element-wise multiplication. The sigmoid function keeps the values within a certain range forcing the gates to behave like a logic gate. This gating mechanism eliminates the matrix multiplication during backpropagation. As evident from the equation, backpropagation from c_t to c_{t-1} only requires element-wise multiplication.

Gated Recurrent Unit

As shown in Figure 2.2, the GRU controls the flow of information like the LSTM, but without having to use a memory cell [24]. It just exposes the full hidden content without any control. GRU cells are computationally more efficient as they possess a less complex structure. For capturing both the context information from both sides,

we are using a bidirectional GRU structure. Let's assume a GRU unit takes x_t and h_{t-1} and produces the hidden state h_t using the following formulas:

$$\begin{pmatrix} r_t \\ z_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \end{pmatrix} W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$p_t = r \odot h_{t-1}$$

$$\tilde{h_t} = \tanh(W \begin{pmatrix} x_t \\ p_t \end{pmatrix})$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h_t}$$

Although GRU does not have any memory unit or output gate, it has a reset and update gate, which are represented with r and z, respectively. Symbols $\sigma, \tanh$ and $\odot$ refer to the same functions as previously mentioned. We utilize GRU and LSTM units in our context modeling layer on different configurations. There are many other variants of RNNs available, but they do not add any meaningful improvement over our performance; hence, we do not report those results.

2.2.2 Word Embedding

The Embedding Layer takes the sequence of tokens from the data layer and generates a unique numeric encoding for each token. Earlier, NLP systems encoded each token as a discrete atomic symbol. These encodings were random and perceived no useful information pertaining to the relationship between two different tokens. Furthermore, it did not consider the token's context in which it was used. Consequently, the neural network trained on these random values could leverage very little infor-

mation while processing each token sequentially. The unique and arbitrary encoding for each token led to data sparsity, which demanded more data for training neural networks and therefore we do not consider them in our proposed architecture.

Shortcomings of unique encoding are mostly resolved in the Vector Space Models (VSMs) [33], where each token is represented in a continuous vector space. In VSMs, semantically similar words or tokens are mapped to nearby points in a fixed dimensional geometric space. This mapping is called word embedding and are popular in NLP pipelines, where a fixed-sized vector represents one word. These numeric word embeddings showed that the operations among 'King' - 'Man' + 'Queen' is 'Woman'. In our proposed models, we used two different types of embeddings: Fixed Embedding Dynamic Embedding.

Fixed Embedding

In a fixed valued word embedding, each token will be represented with one numeric vector. This vector will remain the same regardless of the context or where the tokens occur in a sentence. For example, the word "right" will always have one embedding even if we had a sentence like "You were right when we made the right turn", where the word right has different meanings depending on the usage. Fixed embeddings are trained via neural networks on larger text corpus like Wikipedia or newspapers, and we used this embedding in the RNN models.

Dynamic Embedding

In the dynamic embedding, tokens have different embeddings depending on their meanings and the order of occurrence in a sentence. For instance, for the same

example above, "You were right when we made the right turn"; this embedding will generate different vectors for the two occurrences of "right". The first "right", for instance, would be closer to a word like correct, whereas the second one would denote direction. The dynamic embedding of words could be generated using a bidirectional LSTM or self-attention mechanisms. For the simplicity of the model architecture, we used the latter one in our attention model.

2.2.3 Precision, Recall and F1-Score

To assess the accuracy/performance of the model, we computed the precision (P), recall (R), and F1-score of our architecture which are defined as follows:

$$P = \frac{TP}{(TP + FP)} \tag{2.1}$$

$$R = \frac{TP}{(TP + FN)} \tag{2.2}$$

$$F1 = \frac{2 \times precision \times recall}{(precision + recall)} \tag{2.3}$$

Here,

- TP = The number of PHI instances our model correctly labeled as a PHI
- FP = The number of non-sensitive tokens our model labeled as a PHI
- FN = The number of sensitive tokens (PHI) our model labeled as not a PHI.

Intuitively, precision is the proportion of predicted named entities that are ground truth labels, recall is the proportion of ground truth named entities that are correctly predicted, and F1-score is the harmonic mean of precision and recall.

2.3 Differential Privacy

In this Thesis, we used differential privacy [16] to protect patients' privacy from the published de-identification models, which is a method of publicly releasing the aggregate information of a dataset without any privacy breach of individual data participants. This mechanism that works for arbitrary datasets, which guarantees quantifiable privacy over the data or any outputs generated from the released data. Therefore, any method theoretically proven to be differentially private will always hold the privacy guarantee irrespective of the dataset used. Differential privacy could be formally defined as follows:

Definition 2.3.1 *(Differential Privacy) Let $\mathcal{M} : \mathcal{X} \rightarrow \mathcal{Z}$ be a randomized mechanism, where $\mathcal{X}$ is the set of input values and $\mathcal{Z}$ is set of values determined by $\mathcal{M}$ for each $x \in \mathcal{X}$. $\mathcal{M}$will be (ε, δ)-differentially private if for any two adjacent inputs $x_1, x_2 \in \mathcal{X}$ and for every measurable subset of outputs $\mathcal{S} \subseteq \mathcal{Z}$ we have,*

$$Pr[\mathcal{M}(x_1) \in \mathcal{S}] \leq e^{\varepsilon} Pr[\mathcal{M}(x_2) \in \mathcal{S}] + \delta$$

Here, $Pr[\mathcal{M}(x) \in \mathcal{S}]$ is the probability that for input x the output would be $\mathcal{M}(x)$. This additive parameter δ was later introduced [34] to further relax the original ε-differential privacy definition.

2.3.1 Definition of the Parameters

The parameters in a differentially private mechanism are used to quantify the privacy ensured by employing it. The parameter ε in Definition 2.3.1 is referred to as the privacy budget, used to regulate the noise added to the original dataset before

public release. For example, when the privacy budget $\varepsilon = 0$ and $\delta = 0$, the general definition of differential privacy will result in $\mathcal{P}[\mathcal{M}(x_1) \in \mathcal{S}] \leq 1 * \mathcal{P}[\mathcal{M}(x_2) \in \mathcal{S}]$. The resulting equation infers that the randomized mechanism, $\mathcal{M}$ will answer any query with the same probability for both x_1 and x_2. Consequently, any output of $\mathcal{M}$ or further analysis on the results of $\mathcal{M}$ does not depend on individual data participant, eventually providing the highest level of privacy for the dataset. In a more generalized term, the lower the values of ε, the higher the privacy and vice versa. However, the higher privacy comes at a price-the quality or utility of the released data as it will add higher noise to the original data. Therefore, the challenge while using DP is finding the precise balance between the highest utility achievable with the lowest loss of privacy.

The parameter δ also bears similar importance to ε. As mentioned before, ε regulates the privacy budget, whereas δ refers to the privacy guarantee. If any algorithm supports $\varepsilon > 0, \delta = 0$, it is called pure differential privacy. On the other hand it is referred as approximate differential privacy when $\varepsilon > 0, \delta > 0$. For an Approximate differential privacy where $\delta > 0$ denotes that the mechanism $\mathcal{M}$ will hold the ε privacy guarantee with probability $1 - \delta$, and fail for δ times. Consequently, like ε, the lower values of δ ensures a better privacy guarantee. In summary, for an algorithm that supports (ε, δ) differential privacy, ε determines an upper bound on the impact of individual data participant on the final result and δ offers a bound on how often the algorithm will fail to ensure the intended privacy.

2.3.2 Sensitivity of a Function

One of the major components of a differentially private mechanism is the functions we want to execute on the data we intend to protect. Let f be such a function which could be defined as $f : \mathcal{S} \rightarrow \mathbb{R}^n$, where $\mathcal{S}$ can consider a count query represented with f. To execute f, every record $n_i \in \mathcal{S}$ is checked against the query conditions of f. Hence, $f_{count}(\mathcal{S}) = \sum_1^n cond(n_i)$ where $cond(n_i)$ can have only hold a boolean value. The sensitivity of f could be defined as,

Definition 2.3.2 *(Sensitivity) For $\mathcal{S}_1, \mathcal{S}_2 \in \mathcal{S}$ Sensitivity of a function, $f : \mathcal{S} \rightarrow \mathbb{R}^n$:*

$$\Delta f = \max_{\mathcal{S}_1, \mathcal{S}_2 \in \mathcal{S}} ||f(\mathcal{S}_1) - f(\mathcal{S}_2)||_1$$

The definition above thus shows the sensitivity of function for a single record. In other words, the sensitivity of a function is the largest possible change that one row can make on the result of that function for any dataset.

2.3.3 Gaussian Mechanism

The resulting dataset after a differentially private algorithm is achieved through adding noise to the original data. Adding a fixed noise would defeat the purpose. Hence different mechanisms are used to generate randomized noise. In this Thesis, we used a Gaussian mechanism. A Gaussian mechanism adds a random noise from the independent and identically distributed Gaussian distributions according to:

$$f(\mathcal{S})' = f(\mathcal{S}) + \mathcal{N}(\sigma^2), \tag{2.4}$$

where, $\sigma = \sqrt{2\Delta \log(1.25/\delta)}/\varepsilon$ and centered around 0. ε, δ are defined as above, and Δ is the sensitivity of the function executed on the dataset.

Chapter 3

Literature Review

We discuss our related works in two subsections, where in the first one, we discuss some of the previous works done in the de-identification approach for data publishing. In the second subsection, we discuss a few of the recent works in privacy-preserving machine learning and model publishing.

3.1 De-identification of EHR

Several systems have been proposed to de-identify text data in the healthcare domain. Here, we characterized these systems into three major categories: rule-based, machine learning-based, and hybrid systems. While reviewing the literature in this field, we found a common metric used for the evaluation of a de-identification system. Some familiarity with these metrics will help a better understanding of the comparison of different approaches proposed over the years. These approaches are evaluated of three major metrics- Precision (P), Recall (R), and their harmonic mean

F1-score.

3.1.1 Rule-based Approach

Early machine-assisted de-identification systems were mostly rule-based for text data. These systems used patterns, regular expressions, and dictionary lookups to de-identify different EHRs such as patient notes, discharge summaries, and nursing notes. In one of the earliest attempts in 2005, Douglass *et al.* [35] developed a semi-automated method to allow clinicians to highlight PHI instances on the screen of a tablet PC. Their method could compare and combine the selections of different experts reading the same notes. Another approach from Douglass *et al.* [6] used lexical look-up tables, regular expressions, and simple heuristics to locate PHI instances. It is noteworthy that these rules or fixed dictionary entries were done manually and did not scale for large text corpus containing unstructured healthcare information.

3.1.2 Machine Learning Approach

In recent years, a few machine learning approaches have been proposed as well, which automatically learns the sensitivity of a given text or word sequence. These machine learning algorithms for de-identification mostly include decision trees, support vector machines, hidden markov models, and conditional random fields (CRFs). Szarvas *et al.* [36] proposed an iterative anonymization method on semi-structured documents. Their proposed method labeled all PHI instances inferred from the structure of the document and then utilizes this information to find further PHI instances. In 2014, this area of research received much-deserved attention, courtesy to

the UTHealth i2b2 de-identification challenge. It was this competition that produced several machine learning based automated tools on the i2b2 2014 dataset.

During this challenge, Chen *et al.* [8] introduced a non-parametric Bayesian hidden markov model using a Dirichlet process (HMM-DP). They managed to score a 91% F1 score (with 87.9% recall) in the competition. Later they improved the HMM-DP model and aligned it with a CRF model, and obtained a 93.7% F1 score (Recall: 91%). On the same competition, He *et al.* [37] proposed a CRF based de-identification system with a significant focus on preprocessing and feature generation from medical records, and achieved 95.71% precision and 90.51% recall, respectively.

3.1.3 Hybrid Approach

The UTHeath i2b2 de-identification challenge has also produced some hybrid systems where different de-identification pipeline worked individually, and their results were combined with an ensembling algorithm to produce the eventual results. Liu *et al.* [38] proposed a system that uses a token-level CRF, a character-level CRF, and a rule-based subsystem as their de-identification pipelines. They achieved 91.85% F1 score (Precision: 92.82%, Recall: 90.91%) for their ensemble model. Dehghan *et al.* [39] combined the results of a dictionary and rule-based subsystem with a CRF based subsystem. They trained a separate CRF model for each of the PHI types and post-processed the results with hand-crafted rules. On their second submission in the competition, they scored a precision of 96.55% and recall of 93.16%. Yang *et al.* [40] proposed a similar approach. However, they used the CRF model only for the PHI categories with sufficient amount of available training data and keyword spotting and

rule-based approach for the PHI categories for fewer sample instances. With a 93.6% F1 score Yang *et al.* [40] was the winner of the de-identification challenge.

In the early stages of deep learning, the hybrid approaches dominated this domain. In 2016, Dernoncourt *et al.* [27] proposed the first neural network for de-identifying patient notes. They used both character and word-level embedding vectors. For character-level embedding, they used a bidirectional LSTM and combined the result with word-level one. Another bidirectional LSTM was used for the context encoding, and finally, they used a CRF for optimizing the label prediction. Until now, this model had the best result in the i2b2 2014 dataset. Therefore, we will use this model as our benchmark. Yadav [41] *et al.* compared two different variants of RNN (Jordan-type and Elman-type Networks) for the context analysis. They achieved 89.63% and 90.18% F1 for Elman-type and Jordan-type networks respectively. As they did not use the total i2b2 2014 dataset, their results cannot be directly compared with other systems. In 2017, Liu *et al.* [42] proposed another hybrid system with four subsystems based on bidirectional LSTM, CRF and rule-based. Khin *et al.* [43] used a deep contextualized word-level embedding and bidirectional LSTM character-level embedding together. For other components of the network, they used a bidirectional LSTM and CRF identical to Dernoncourt *et al.* [27]. Table 3.1 gives a comparison of precision and recall for these hybrid approaches.

3.2 Privacy-preserving Machine Learning (PPML)

Over the years, there have been many techniques centered around training machine learning models in collaboration with multiple data owners without the risk

Table 3.1: Precision (%) and Recall (%) comparison for the i2b2 2014 dataset [28].

Methods	**Year**	**Approach**	**i2b2 dataset [28]**	
			Precision	**Recall**
Liu *et al.* [38]	2015	Token CRF	85.89	82.55
		Character CRF	93.88	88.85
		Rule	28.3	1.742
		Ensemble	92.82	90.91
Dernoncourt *et al.* [27]	2016	ANN+CRF	97.56	92.86
Liu *et al.* [42]	2017	CRF	95.16	91.12
		LSTM	95.26	93.34
		LSTM-Fea	95.43	93.61
		Ensemble	96.46	93.80
		Overall	96.46	93.80

of privacy violation of the data participants. We can characterize these approaches into two primary categories-utilizing cryptographic methods or differentially private data release. Even though cryptographic methods based on Fully Homomorphic Encryption and Secured Multi-party Computation provide a well established security guarantee, differential privacy based machine learning approaches are more pertinent to our work. Hence, we deliberately omit the discussion of cryptoghraphic PPML.

3.2.1 Cryptographic Approaches

In this category of work, when machine learning applications require sensitive data from multiple owners, the data is encrypted and contributed to the computation server. The models are then trained, and predictions are generated on the encrypted data. This method effectively reduces the problem to a multi-party secure computation setting. Techniques like *homomorphic encryption* [44], *garbled circuits* [45], *secret sharing* [46], and *secure processors* [47] widely used in the multi-party secure computation are also prevalent in achieving PPML.

Fully Homomorphic Encryption (FHE) is the form of encryption that allows arbitrary operations (addition, multiplication, etc.) on the ciphertext. FHE generates an encrypted result, and the decrypted result matches the result of the operations as if they had been performed on plaintext. Such a scheme enables the construction of sophisticated machine learning algorithms. FHE schemes have been used in image classification [48, 49, 50], statistical analysis [51, 52], and Genome-Wide Association Studies [53, 54]. However, textual data analysis on encrypted data is a very recent application of FHE and most pertinent to our work. Al Badawi *et al.* [55] is proposing one of the first models which analyze sentiments on encrypted textual data using FHE. Before, Costantino *et al.* [56] proposed an FHE based text analysis framework, where they developed bag-of-words (bow) classification. However, the classification model is trained on decrypted frequency vectors after these are received in encrypted form from a trusted server, which computes the frequencies using FHE.

Another cryptographic approach to text analysis or Natural Language Processing is Multi-Party Computation (MPC). In 2019 Reich *et al.* [57] proposed the first privacy-preserving text classification based on Secure Multiparty Computation. They proposed a new privacy-preserving protocol for secured feature extraction and subsequent classification with logistic regression and tree ensembles. The protocol proposed by Reich *et al.* ensures that the application does not learn anything about the author from the query, and the author does not learn anything about the classification model. Cock *et al.* [58] showed an application of the protocol outlined by Reich *et al.* in hate-speech detection. One of the main drawbacks of FHE is the high computational overhead. Multi-Party Computation (MPC) offers an alternative cryptographic solu-

tion to FHE with a low computational cost. However, MPC solutions require high interaction between the users and cloud; hence they are bandwidth-bound.

3.2.2 Differentially Private Approaches

FHE schemes use a noise term in the ciphertext, which grows proportionally with every arithmetic operation. The decryption of these ciphertext yields the correct result only when the noise term is relatively small. As a result, even if in simple deep learning models, FHE is applicable, complex NLP models based on RNNs and stacked Transformer models would result in a multiplicative depth that would corrupt the result of the computation once decrypted. Most of the NLP models are currently based on either variation of RNNs or Transformer, using FHE schemes for implementing these models seems far fetched. This restriction effectively motivated us to use differentially private optimization algorithms to train our multi-headed self-attention based de-identification model while preserving the privacy of the participants in the training data.

Differential privacy has established itself as a rigorous standard for providing privacy guarantees. Consequently, a considerable amount of attempts have been made to provide machine learning models with differential privacy. These includes work by Bassily *et al.* [59]; Chaudhuri *et al.* [17]; Pathak *et al.* [18], Song *et al.* [19], and Duchi *et al.* [60]. Furthermore, Shorki *et al.* [20] proposed a privacy-preserving distributed SGD algorithm in 2015. However, the privacy bounds were given for each parameters in the model, which made the privacy guarantees for larger model virtually non-existent. The differentially private SGD that we are using in our work provides a

stricter bounds on the privacy loss (details discussed in sec 5.1.2). Hamm *et al.* [22] proposed the use of knowledge transfer between a collection of models trained on individual devices into a single model that guaranteed differential privacy. Their work further adopted for more generalized applications by Papernot *et al.* [23] in 2016.

Chapter 4

Deep Learning Based De-identification Model

In this chapter we discuss our proposed de-identification architecture. In Section 4.1 we describe the system architecture in detail. Section 4.2 describes how we setup the model. In Section 4.3 we present the results and Section 4.4 present an analysis of the result. Finally, in Section 4.5 we discuss the limitations of our proposed method.

4.1 Methodology

In this section, we discuss our proposed method in detail, which has four major layers with varying components in each layer (except Data Layer). In the following subsections, we describe these layers sequentially as appear in the processing pipeline. Furthermore, our implementation of the models is available at [61].

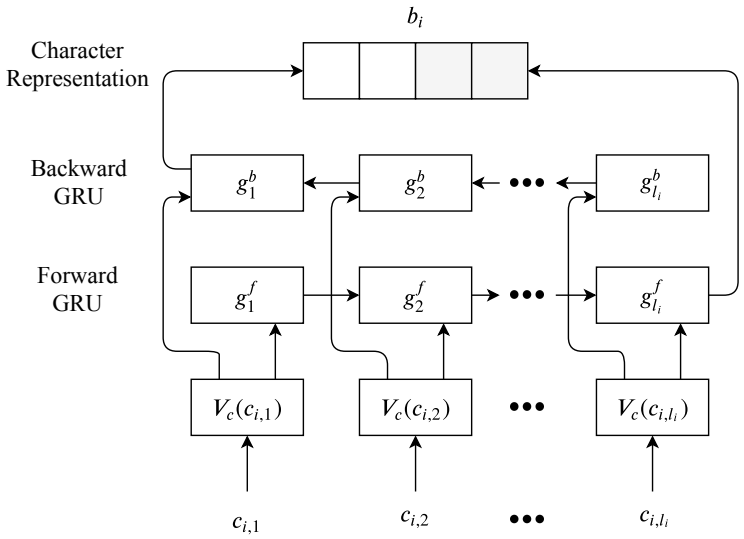


Figure 4.1: The structure of the Embedding Layer.

4.1.1 Data Layer

The Data Layer preprocesses the input EHRs considered for the network. The data layer splits every sentence of the EHR documents into sequences of tokens (words). Then, each token is assigned to a unique numeric value according to their occurrence sequence in the datasets. In other words, the tokens in the datasets were serialized and each token had a fixed serial number as their identifier. These numbers are then used to convert the sentences into sequences of numeric values or vector. In other words, one numeric vector will represent a sentence of the EHR. The labels (PHI subcategories) in the training set are also indexed using unique numeric values using similar mechanism to the tokens. For attention model, these numeric values are the ids found from the vocabulary list. The numeric representations of the sentences and their associated label sequences are fed into the embedding layer.

4.1.2 Embedding Layer

The embedding layer captures the semantic meaning and features by embedding each token to a vector. Token embeddings improve the semantic information captured by the neural networks in the context modeling layer. However, there are still unperceived information by the network in case of out-of-vocabulary or misspelled infrequent words, a different noun or verb endings. One solution is lemmatizing each token before training the network, but it will also cause the loss of information (e.g. the distinction between noun and verb). To solve this problem, we used the character level embedding for each token in addition to word level embedding.

Character level embedding. We used a bidirectional GRU network to determine the character level embedding. In our models, we depart from other benchmark methods, which predominantly used a bidirectional LSTM for computing the character level embedding. A bidirectional GRU consists of a forward and a backward GRU. The hidden states of the forward GRU capture information from the past positions whereas backward units capture information from future positions as it is fed in reverse order. The final hidden state of a backward or forward GRU summarizes the entire character sequence. Figure 4.1 shows the network structure for such character embedding. Let $s = \{s_1, ..., s_n\}$ be the input sentence with n is the length of the sentence. Also let $c_{i,1}, ..., c_{i,l_i}$ be the sequence of characters that comprise the i^{th} token s_i, where l_i is the number of characters in s_i. The character-level token encoder generates the character based token embedding of s_i by first mapping each character $c_{i,j}$ to a vector $V_c(c_{i,j})$, called a character embedding. Then the sequence $V_c(c_{i,1}), ..., V_c(c_{i,l_i})$ is passed to a bidirectional GRU, where $g_1^b, ..., g_{l_i}^b$ and $g_1^f, ..., g_{l_i}^f$ are

the hidden states for backward GRU and forward GRU layer respectively. The character level representation b_i is calculated by concatenation of the outputs of hidden state g_1^b and $g_{l_i}^f$.

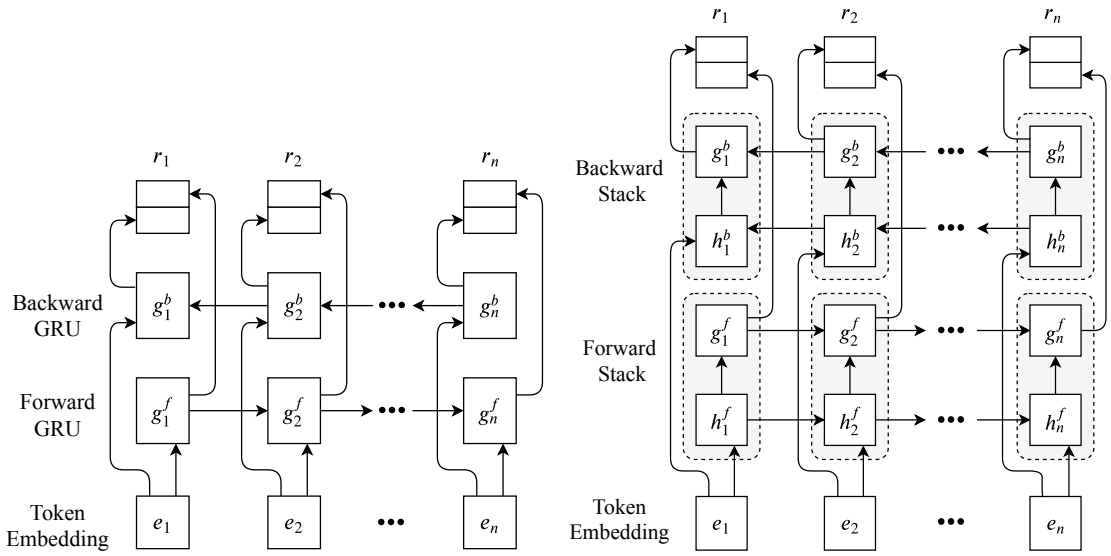
Word level embedding. A general purpose pre-trained word-embedding like Glove [62] or Word2Vec [63] is not available from clinical notes. Hence, for the fixed word level embedding, we used the the general purpose GloVe [62] pretrained model. The publicly available GloVe model contains token embedding for 6 billion unique tokens. We retrieved the vectors corresponding to the EHR tokens with 100 dimensions from GloVe and used them hereafter. The final output e_i from the token embedding layer for i^{th} token s_i is the concatenation of the word embedding $V_T(s_i)$ and the character based token embeddings b_i. In summary, when the character embedding layer receives a sequence of tokens $s_{1:n}$ as input, it will output the sequence of token embedding $e_{1:n}$. It is noteworthy that we also used a dynamic embedding scheme where the usage or placement of a token in a sentence determines its numeric values on a fixed size vector.

4.1.3 Context Modeling Layer

RNN Models

The proposed RNN models use both LSTM and GRU units. Each model employs bidirectional RNN sublayer as it helps to capture both future and past information Previous state-of-the-art approach [27] utilized single bidirectional LSTMs, whereas we use bidirectional GRUs and stacked GRU+LSTM models.

GRU Model. As mentioned in the before, GRU units offer fewer parameters com-



(a) Bidirectional GRU Model (b) Stacked RNN Model

Figure 4.2: Structure of different context modeling layer with different components

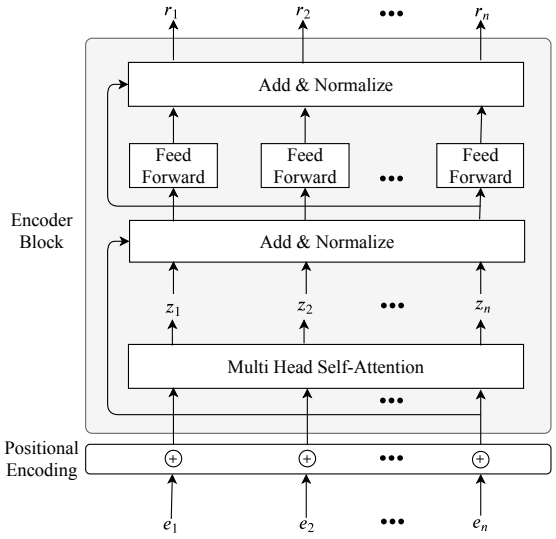


Figure 4.3: The simplified structure of the encoder stack for a sentence of n tokens.

pared to LSTMs. In our first variant of the context modeling layer, we used a bidirectional GRU layer with one forward GRU and one backward GRU. In contrast to the construction of character-level embedding, the final output is calculated by

concatenating the output of hidden states at each time step. Figure 4.2a depicts the construction of the context modeling layer using such bidirectional GRU. Here, $g^f = \{g_1^f, g_2^f, ..., g_n^f\}$ and $g^b = \{g_1^b, g_2^b, ..., g_n^b\}$ be the hidden states of the forward and backward GRU, respectively. The final output vector r_i for the i^{th} token of a sentence (length n) is formed by concatenating the forward and backward hidden states g_i^f and g_i^b respectively (i.e., $r_i = (g_i^f.g_i^b)$). These output vectors r_i will be used by the label decoding layer.

LSTM-GRU Model. Then, we analyzed the effect of deeper models by stacking RNN models. For the second variant, we used two bidirectional GRU layers. Let g^f and g^b be the set of hidden states for forward and backward GRU respectively. g_i^f and g_i^b are the hidden states of forward and backward GRU at i^{th} time step. Also, let h^f and h^b be the set of hidden states for forward and backward LSTM, and the hidden states of forward and backward GRU at i^{th} time step be g_i^f and g_i^b. For the forward RNN layer, the output of hidden state h_i^f is relayed to g_i^f. The backward RNN layer followed the same mechanism. The final output r_i at i^{th} time step for the i^{th} token in a n length sequence is formed by concatenating the output of hidden state g_i^f and g_i^f i.e., $r_i = (g_i^f.g_i^b)$. Figure 4.2b illustrates the construction of this model.

GRU-GRU Model. The final variant of the stacked RNN model follows the same structure as the previous model, but the forward and backward LSTM sublayer was replaced with a siamese forward, backward GRU sublayer respectively.

Attention Model

The RNN models described above are sequential in nature. The hidden states of these models for a time step is a function of the previous step. During training, this sequentiality becomes pivotal while processing longer sentences as memory constraint curtail the opportunity of batch processing. To overcome such constraint, we are using a multi-headed self-attention mechanism for generating a dynamic word embedding instead of the fixed word embedding techniques. The dynamic context-aware embedding eliminated the necessity of an additional RNN layer for context analysis. The input for this model is a vector of numeric values incoming from the data layer. The output values generated from this model are forwarded to the Label Decoding Layer.

Multi-headed self-attention. The multi-headed self-attention mechanism used here was first introduced by Vaswani *et al.* [10] as a replacement for firmly established RNNs for sequence modeling and transduction problems such as language modeling and machine translation. The model introduced in [10] namely, Transformer follows the encoder-decoder architecture of traditional transduction models. Instead of using an RNN or CNN, Transformer uses only attention mechanisms. However, we used a bidirectional Transformer, which is often referred to as Transformer Encoder. This is similar to the BERT [11], which is conditioned on both the left and right context.

Self-attention sublayers employ h attention heads. The sublayer output is retrieved from concatenating the results from each head and a parameterized linear transformation. Each attention head operates on an input sequence passed from the data layer. This input vector is added with positional encoding and forwarded to

the self-attention sublayer. Figure 4.3 shows a simplified structure of the mechanism where the self-attention sublayer receives input sequence $e = e_1, ..., e_n$ of n tokens after positional encoding and computes a new sequence $z = z_1, ..., z_n$ of the same length. Each output element, z_i, is computed as the weighted sum of linearly transformed input elements:

$$z_i = \sum_{j=1}^{n} \gamma_{ij}(e_{ij}W^V) \tag{4.1}$$

Each weight matrix, γ_{ij}, is computed using softmax function:

$$\gamma_{ij} = \frac{\exp a_{ij}}{\sum_{k=1}^{n} \exp a_{ik}} \tag{4.2}$$

A compatibility function is used to compute

$$a_{ij} = \frac{(e_i W^Q)(e_j W^K)^T}{\sqrt{d_z}} \tag{4.3}$$

which compare two input elements. This scaled dot product was chosen as the compatibility function due to its computational efficiency. d_z used for computing a_{ij}, is the dimension of the matrix $e_j W^K$ and W^Q, W^K, W^V are parameter matrices. These parameter matrices are different for each layer and attention head and learned during training.

4.1.4 Label Decoding Layer

The label decoding layer takes vectors containing the context information as input and predicts the PHI instances (sub-categories like `DOCTOR, PATIENT, HOSTPITAL,` etc. shown in Table 2.1). As PHI instances could have multiple tokens/names we

used the BIOES (which stands for `Begin, Inside, Outside, End` and `,Single`, indicating the position of the token in the PHI instance) tagging scheme [64], which distinguishes between the end of a multi-token PHI instance and single token PHI instance. Now, we are using two different methods–softmax and CRF model for predicting labels from the vectors calculated in the context modeling layer. For the RNN models, we used an Fully Connected Neural Network (FNN) before the softmax/CRF layer. However, as the attention model uses an FNN within the encoder block, we directly used the output of the attention model as an input to the softmax/CRF layer.

Softmax Layer. The softmax layer normalizes the values received from the FNN to probabilities. Let's assume, this layer receives $r = \{r_1, ..., r_n\}$, as the input sequence where r_i is the i^{th} token's score. Then, the probability for the j^{th} label (PHI) for i^{th} token would be,

$$q_j(r_i) = \frac{\exp r_{ij}}{\sum_k \exp r_{ik}} \tag{4.4}$$

Here k is the total number of labels. The network is trained to minimize the cross-entropy loss, $\mathcal{L}(p, q)$ where p is the original probability for j^{th} label.

CRF Layer. CRF models consider the correlations between labels in neighboring positions and jointly decode the best chain of labels for a given input sentence. The use of CRF algorithm to predict the labels from the features extracted in the context modeling layer is a standard practice in a NER task. Our experimental results further emphasized the effectiveness of the CRF algorithm while predicting interdependent labels. Let, $y = \{y_1, ..., y_n\}$ represent a generic sequence of labels for r. $\theta(r)$ denotes

the set of possible label sequences for r. The probabilistic model of CRF determines a conditional probability, $p(y|r; W, b)$ over all possible label sequences y with respect to r, where W and b are the weight vector and bias. During training, we used the maximum conditional likelihood estimation. For a training set, $\{(r_i, y_i)\}$, the logarithm of the likelihood is given by:

$$\mathcal{L}(W, b) = \sum_i \log p(y|r; W, b) \tag{4.5}$$

Maximum likelihood training chooses parameters such that the log-likelihood $\mathcal{L}(W, b)$ is maximized. Decoding is to search for the label sequence $y*$ with the highest conditional probability.

$$y* = \underset{y \epsilon \theta(r)}{\operatorname{argmax}}\, p(y|r; W, b) \tag{4.6}$$

4.1.5 Loss Function

In addition to the log-likelihood and cross-entropy loss calculation, we modified the loss function to favor recall over precision. We introduced a new hyperparameter, δ_p, which dictates the maximum weight of the penalizing factor, ρ. This penalizing factor (ρ) depends on the false negative rate and δ_p. The false negative rate, FNR, is defined as

$$FNR = FN/(1 + FN + TP) \tag{4.7}$$

where FN and TP are the count of false negatives and true positives, respectively. Notably, ρ is calculated using the equation,

$$\rho = FNR \times [(\delta_p - 1) + 1] \tag{4.8}$$

We formulated the eq. of FNR and ρ in a way which ensures that the FNR is never ∞ and ρ always has a value within $[0, \delta_p]$. Finally, the recall heavy loss is the original loss $L_{regular}$ weighted by ρ

$$L_{recall} = L_{regular} \times \rho \tag{4.9}$$

4.2 Model Setup

4.2.1 Parameter Initialization

- **Word Embeddings.** For RNN models, we used Stanford's publicly available GloVe [62] 100-dimensional embeddings, which are trained on 6 billion words from Wikipedia. For the attention model, we used the WordPiece [65] embeddings with a vocabulary size of 30,000.

- **Character Embeddings.** We considered only the lower case characters, and the dimensions are set to 50 as we randomly initialize them.

- **Weight Matrices and Bias Vectors.** The other trainable matrices and vectors are initialized using Xavier Initialization [66]. Bias vectors are initialized to zero except the forget gate bias in the bidirectional RNN layers (initialized to 1.0).

4.2.2 Optimization Algorithm

All of the models were trained using Adam [67] optimizer without any decay rate tuning, and updating all the parameters at each gradient step. We choose an initial learning rate of $1e-5$ for all of the models. We used a gradient clipping of 5.0 to reduce the effect of 'gradient exploding' as well. We explored with stochastic gradient descent, but it did not add any meaningful improvement upon the Adam optimizer in our primary experiments. For regularization, in RNN models, a dropout was applied to the embedding and context modeling layer before forwarding to the label decoding layer. In the attention model, a dropout was used only before the label decoding layer. For 100 epochs, the RNN model takes around 33 hours to train on Titan V (12GB) graphics processing unit for the i2b2 dataset. For the same configuration, these model takes around 24 hours to train on the MIMIC-III dataset [68]. The attention model takes around 15 hours and 12 hours on the i2b2 2014 and MIMIC-III datasets, respectively.

Early Stopping. We used early stopping based on validation set performance. The model automatically stops training if the accuracy did not improve for 10 consecutive epochs.

Fine Tuning. For each of the embeddings, we fine-tuned initial embeddings, modifying them during gradient updates of the neural network model by back-propagating gradients. The effectiveness of this method has been previously explored in sequential and structured prediction problems [69].

Dropout training. We applied different dropouts to reduce overfitting and regularize the underlying models. The dropouts are applied succeeding the embedding layer

Table 4.1: Training Hyperparameters.

Layer	**Hyper-parameter**	**Value**
Embedding	GRU hidden state dimension	25
	Word embedding dimension	100
Context Modeling	GRU hidden state dimension	150
	LSTM hidden state dimension	150
Attention	Attention Head Number	12
	Attention Block	12
	Hidden Size	768
Dropout	Dropout rate	0.5
	Initial learning rate	1e-5
	Gradient clipping	5

and the attention layer, and before the CRF layer.

4.2.3 Hyperparameter Tuning

Table 4.1 gives an overview of the hyperparameters we analyzed. We tuned these hyperparameters on the validation sets by random search. The most important character embedding hidden state dimension, attention head, and encoder block were set as 25, 12, and 12, respectively, whereas the hidden size for the attention model was 768.

4.3 Results

We evaluated our architecture on three different datasets: i2b2 [28], Nursing Note [30] (aka MIMIC-II) and MIMIC-III [68]. The i2b2 is the benchmarking dataset due to its primary usage over a de-identification competition held in 2014. The dataset has categorized the EHRs into train, validation, and test sets where the annotations were done manually. On the other hand, the Nursing Note or the MIMIC-III both

went through some form of computer-assisted de-identification. The i2b2 dataset was much smaller compared to the other two. MIMIC-III is much larger (60k patient admission data) corpus compared to i2b2 (1,304 documents). We considered 4,441 discharge summaries from MIMIC-III averaging around 1,387,990 (1.3 M) words. 20% (891 EHRs) of the dataset was selected for testing, whereas the training set was split 80 : 20 for training and validation, respectively. We used the BIOES tagging scheme instead of standard BIO2, as previous studies have reported meaningful improvements with this scheme. We present the corpora statistics regarding the number of PHI instances in Section A.2. Furthermore, the Nursing Note dataset has 2,434 documents with 1,826 PHI instances. We split the dataset as the MIMIC-III dataset in 80:20 ratio to create the training (1948 documents) and test (486 documents) sets. The experimental results are presented into three categories: accuracy in predicting PHI tokens, utility analysis of the de-identified data, and the execution time of the proposed models.

4.3.1 Accuracy

To assess the accuracy/performance of the model, we computed the precision (P), recall (R), and F1-score of our architecture which are defined in Equation 2.1, Equation 2.2 and Equation 2.3, respectively. Table 4.2 shows the best results we have found for each of our models on the three datasets. The results reported in Table 4.2 are evaluated based on the detection of PHI tokens vs non-PHI tokens (i.e., binary HIPAA token-based evaluation). We used Dernoncourt *et al.* 's [27] model as our performance benchmark as this work has a better performance than any existing

Table 4.2: Precision (P%), Recall (R%) and F1 (%) comparison for each dataset.

	i2b2 2014 [28]			**MIMIC-III** [68]			**Nursing Notes** [30]		
Model	**P(%)**	**R(%)**	**F1(%)**	**P(%)**	**R(%)**	**F1(%)**	**P(%)**	**R(%)**	**F1(%)**
Dernoncourt *et al.* [27]	97.920	97.835	97.877	99.91	99.97	99.94	86.10	69.70	77.00
Khin *et al.* [43]	98.300	97.370	97.830	-	-	-	**91.40**	74.30	81.20
GRU (Proposed)	98.746	95.859	97.281	99.946	**100.00**	**99.973**	81.82	66.23	73.21
GRU-GRU (Proposed)	**99.011**	95.124	97.028	99.939	99.994	99.967	80.00	71.00	75.23
LSTM-GRU (Proposed)	98.749	95.278	96.982	99.940	99.998	99.969	85.14	68.18	75.72
Self-attention (Proposed)	98.031	**98.410**	**98.220**	**99.957**	98.788	99.369	89.20	**82.90**	**85.90**

models for the i2b2 dataset. For the i2b2 dataset, we reported the results directly from their paper. However, their preprocessed MIMIC-III dataset is not publicly available, and it was not possible to reproduce the dataset solely from their paper. Hence, for a fair comparison, we had to reproduce the results using the publicly available implementation of their architecture. We used the pre-trained models with `binary` evaluation method outlined in their implementation and reported the best result we found. Dernoncourt *et al.* [27] did not report their results on Nursing Note. We used their best performing pretrained model and ran the model until convergence as we did for the MIMIC-III dataset, and reported the results. From Table 4.2, we can see that Khin *et al.* [43] has better performance (81.2% F1 Score) than Dernoncourt [27] model (77.0% F1 Score). Hence, we regard Khin *et al.* [43] as our benchmark for the Nursing Note dataset. However, Khin *et al.* did not report their results on MIMIC-III dataset and their implementation was publicly available at the time this Thesis is written. Thus, the columns for the MIMIC-III dataset of this model are left blank in Table 4.2.

The results in Table 4.2 show that our attention model has better F1 score (98.220%) than all our RNN based models on the i2b2 dataset, and it is 0.343%

more than the reported value by the state-of-the-art [27]. The attention model also has a 0.111% higher precision and 0.575% recall value than [27]. Surprisingly, the GRU model has the best F1 score (97.281%) than the other RNN models, although it falls behind of the attention model by 0.939%. Among the stacked RNN models, the GRU-GRU model has a 0.046% F1 score gain over the LSTM-GRU model. The GRU-GRU model has a higher precision value than the self-attention model, however, a very low recall value (95.124%) leads to a lower (97.028%) F1 score. Furthermore, the GRU, GRU-GRU, and LSTM-GRU model have 98.746%, 99.011%, and 98.749% precision score, respectively. All these precision values are higher than the benchmark approach.

Results on the MIMIC-III dataset are slightly different. The prior model [27] performed better on this dataset compared to the i2b2 dataset. Their F1 score is 99.94% that leaves a limited margin for improvement. However, our proposed models still managed to incur improvements over the state-of-the-art model [27]. Unlike the i2b2 dataset, the GRU model has the best F1 score (99.973%). Despite having the best precision score, (99.957%) a 1.212% lower recall value resulted in 99.369% F1 score for the attention model. In addition to the GRU model, the GRU-GRU and the LSTM-GRU model both have better F1 score (99.967% and 99.969% respectively) than [27]. Please be noted that Khin [43] did not have their implementation publicly available. Therefore, we are unable to report the results for their model on the MIMIC-III dataset. In summary, our GRU, GRU-GRU, and LSTM-GRU model have 0.033%, 0.027% and 0.029% higher score than the baseline model [27].

Now, our self-attention model outperforms both models by at least 4.5% in the

Nursing Note dataset. However, our proposed RNN and stacked RNN models performed underwhelmingly in this dataset compared to other datasets. Even though all of the three models (GRU, GRU-GRU & LSTM-GRU) have precision values over 80%, the RNN based models have lower recall values compared to Khin *et al.* [43] and our self-attention model.

The i2b2 2014 dataset is the standard dataset for evaluating any de-identification system. For further analysis of our model we thus experimented on this dataset. The results presented in Table 4.3, & Table 4.4 show results found only in the i2b2 dataset. In Table 4.3, we study the effect of character embedding and different dropout values on our GRU and self-attention model. First, we evaluated our GRU model without any character embedding (CE) at a 50% dropout value. Then we added the character embedding at the same dropout rate. Then we incrementally decreased the dropout value to 25% and 0%. F1-scores for these two dropout rates were 95.791% and 96.817%, respectively, which is lower than the F1-score at 50% dropout rate (97.281%).

The self-attention model uses dynamic embedding. Therefore, there was no requirement to use character embedding with self-attention. Now, the highest F1-score was found for a 10% dropout rate. Although, at higher dropout rates (25% & 50%), the recall value increases, the precision values continue to drop. We report the results at 25% dropout in Table 4.2 as at this rate, the model has fairly higher values for both recall (98.41%) and precision (98.03%). Table 4.3 also shows the impact of different mechanisms in the label decoding layer, for the GRU and the self-attention model. For both of these models, when a CRF algorithm is used to optimize the detection of

Table 4.3: Effect of Dropout Rate, character embedding (CE), CRF and Softmax layer on GRU and Self-attention(SAT) model i2b2 dataset, where P (%), R (%), F1 (%) denote precision, recall, F1 score, respectively.

Model	**CE**	**CRF**	**Softmax**	**Dropout**	**P(%)**	**R(%)**	**F1(%)**
GRU	×	✓	×	0.50	97.859	84.560	90.725
GRU	✓	✓	×	0.00	98.544	93.187	95.791
GRU	✓	✓	×	0.25	98.677	95.026	96.817
GRU	✓	✓	×	0.50	**98.746**	**95.859**	**97.281**
GRU	✓	×	✓	0.50	97.768	95.206	96.470
SAT	-	✓	×	0.10	**98.705**	97.923	**98.313**
SAT	-	✓	×	0.25	98.031	98.410	98.220
SAT	-	×	✓	0.25	98.854	97.400	98.122
SAT	-	✓	×	0.50	93.782	**98.559**	96.111

PHI instances, the F1-scores are higher (0.81% and 0.098%, respectively) compared to when a softmax function (detailed in Section 4.1.4) was used.

To improve the recall value of the self-attention model, during training we modified the loss function. We introduced a new hyperparameter, δ_p to control the penalty for low recall value. (Section 4.1.5 gives details about the formulation of this hyperparameter.) Table 4.4 shows the results we found for different values of δ_p for the i2b2 dataset. For $\delta_p = 10$, the recall value improves 0.375% over the values for $\delta_p = 1$. This improvement indicates that by penalizing the network with respect to the false negative prediction rate, we can improve the recall value. Although the precision value drops by 0.247%, the recall bears more significance than the precision for the de-identification.

4.3.2 Utility Analysis

To measure the utility of the de-identified EHRs, we propose two general-purpose utility metric: the Bilingual Evaluation Understudy (BLEU) [70] and Topic Modeling

Table 4.4: Effect of Penalty δ_p on the Self-attention model for i2b2 2014 dataset.

δ_p	**P(%)**	**R(%)**	**F1(%)**
1	99.046	97.131	98.079
3	98.283	97.437	97.858
5	**99.060**	97.269	**98.156**
10	98.799	97.506	98.148
15	99.056	95.720	97.360

and one classification application using the de-identified data as a specific purpose utility metric. We discussed the evaluation method of the classification application in Section 4.3.2. Now, to analyze the results for BLEU score and Topic modeling evaluation, we calculated two reference scores:

- *Baseline*: To the best of our knowledge, Dernoncourt *et al.* [27] reports the state-of-the-art results for de-identification. We calculated the BLEU scores and estimated the topics from the de-identified documents by [27], which serves as the baseline for our utility evaluation.

- *Ground-truth (GT)*: For our second reference score, the BLEU scores and Topic Modeling estimations were done for the ground truth de-identified documents. These documents contained the original labels of the PHI instances. Hence, if we calculate the precision or recall for these documents, we will always get 100%.

Before discussing our results on different utility metrics, we need to examine the relationship of utility with respect to precision and recall (especially, FP and FN). Notably, in a de-identification problem, privacy relies only on recall (or TP and FN) as higher recall denotes that the model was more successful in identifying PHI instances

Table 4.5: BLEU score comparison in the i2b2 dataset for different n-grams. Column BLEU-1, BLEU-2, BLEU-3 list the scores for $n = 1, 2$ and 3 where P (%), R (%), F1 (%) denote precision, recall, F1 score, respectively

Method	**P(%)**	**R(%)**	**F1(%)**	**BLEU-1**	**BLEU-2**	**BLEU-3**
Ours	98.03	98.41	98.22	0.923	0.912	0.786
Baseline	97.92	97.84	97.88	0.900	0.887	0.759
GT	100.0	100.0	100.0	0.898	0.885	0.757

Table 4.6: Percentage of phrase and word matches for 5 topics each with 30 words for i2b2 dataset

Method	**P(%)**	**R(%)**	**F1(%)**	**Topic1**	**Topic2**	**Topic3**	**Topic4**	**Topic5**	**Overall**
Ours	98.031	98.410	98.22	86.6	43.3	60.0	36.7	60.0	84
Baseline	97.920	97.835	97.88	70.0	60	46.7	16.7	46.7	84
GT	100.00	100.00	100.0	86.7	60.0	63.3	60	70.0	83.3

offering better privacy. However, the utility is not as straightforward as it takes FP into account as well. When a model wrongfully labels a token as PHI (=FP), it is sanitized and removed in the de-identified document. Similarly, if the same model classifies a PHI token as non-sensitive (=FN), it gets to stay in the de-identified document (privacy breach). Now, higher values of FPs (lower precision) will dictate more tokens to be missing from the document whereas, higher FNs (lower recall) will increase the token count. Since the utility of the de-identified document relies on its tokens, we need to consider both precision and recall while analyzing it.

BLEU Scores

BLEU used different n-grams to output scores, which essentially reveals how similar two sentences are in a document. In Table 4.5, we show the BLEU scores where the ground-truth (GT) documents contained all tokens except the PHI identifiers; hence, the 100% precision and recall value. As the maximum number of removable

tokens are sanitized from the GT documents, its utility should be the lowest while compared with the original document (with all tokens). Therefore, the BLEU scores presented for GT is the maximum achievable utility for a fully private de-identified data.

Topic Modeling

We used Latent Dirichlet Allocation (LDA) [71] as the probabilistic model for our second utility metric. LDA is an unsupervised ML algorithm that clusters relevant words to a topic. The number of topics was defined according to the coherence score as we experimentally set it as 5. Hence, our LDA model generated 5 different topics, each with 30 words (total 150 words) from the original (with PHI), GT, Baseline, and our de-identified corpus. Table 4.6 shows the ratio of word frequency in the de-identified documents compared to the original document for the individual 5 topics. We utilized the 30 words from each topic to get the percentage of matches between the original and our de-identified dataset. This matching percentage for each topic was also calculated for our baseline and the GT documents as well.

Classification Application

Task Description. We also present an practical application of the de-identified data in a machine learning task. We used a supervised deep learning model for EHR classification task to measure the utility of the de-identified data. Each EHR in the MIMIC-III dataset [68] is associated with multiple International Classification of Diseases (ICD9) [72] code. We propose a classifier model on the MIMIC-III dataset (both original, and de-identified) to predict these ICD9 disease code from the underlying

Table 4.7: Selected Diseases and their ICD9 codes for EHR classification task

#	**Disease Description**	**ICD9 Codes**	**EHR No.**
1	Subendocardial infarction, initial episode of care	410.71	307
2	Acute respiratory failure	518.81	471
3	Congestive heart failure, unspecified	428.0	387
4	Coronary atherosclerosis of native coronary artery	414.01	545
5	Pneumonitis due to inhalation of food or vomitus	507.0	287
6	Pneumonia, organism unspecified	486	274
7	Intracerebral hemorrhage	431	126

Table 4.8: ICD9 codes considered in 3, 5 and 7 disease classifiers where the codes are described in Table 4.7

Classifier	410.71	518.81	428.0	414.01	507.0	486	431
3	✓	✓	✓	×	×	×	×
5	✓	✓	✓	✓	✓	×	×
7	✓	✓	✓	✓	✓	✓	✓

texts. Like the analysis of the BLEU score and topic modeling, we have three models to compare:

- *Non-sanitized*. Refers to the model trained on the original non-sanitized/raw (with PHI) MIMIC-III data.

- *Baseline*. Our first benchmark is the classification model trained on the data de-identified by the model proposed Dernoncourt *et al.* [27] as this is the state-of-the-art de-identification model.

- *Ours*. We also train a model with the data de-identified by our Self-attention model. We then compare its prediction accuracy with the aforementioned benchmark model.

Table 4.9: Comparison of EHR classifications on the de-identified MIMIC-III with 3, 5 and 7 different ICD9 codes (or diseases). Complete list of ICD9 codes can be found in (Table 4.8)

	P(%)	**R(%)**	**F1(%)**	**Accuracy**		
				3	**5**	**7**
Non-sanitized	-	-	-	91.5	84.1	80.9
Baseline	99.91	99.97	99.94	91.5	85.8	83.4
Ours	99.98	98.79	99.37	91.8	88.1	85.8

Data Selection. As previously mentioned, each document in the MIMIC-III dataset is associated with multiple ICD9 codes. Now, for our de-identification task, we randomly selected 4,441 documents. Now, we labeled this dataset with their associated ICD9 codes. Table 4.7 shows the ICD9 codes we selected and the number of the documents related to these codes. We selected seven ICD9 codes that have the highest frequency in the original dataset. Thus, the classifier's training dataset is a subset of the original MIMIC-III dataset that we used for the de-identification task. Furthermore, each document is associated with multiple ICD9 codes. So, there are overlaps of documents between the EHR numbers as shown in Table 4.7.

Classification Model. We used the same model architecture for the classification task as the model used in our de-identification task with two major distinctions. First, we modified the 'Data Layer' (Sec 4.1.1) to consider each document as one sequence. Thus, the maximum sequence length used is the highest word/token count for any document in the dataset. Second, we changed the 'Label Decoding Layer' (Sec. 4.1.4) to predict the ICD9 codes of the documents, instead of predicting the label (PHI categories) for each token of the documents. We used only a softmax layer after receiving the scores from the FNN layer. The softmax layer predicted the probabilities

Table 4.10: Execution time in seconds for RNN and Self-attention based model on i2b2 dataset

Work	**i2b2**		#Parameters
	Train	Test	
Baseline [27]	794.36	266.43	3,085,440
RNN (GRU)	802.02	268.18	3,084,940
Self-attention	**693.38**	**124.01**	**110,000,000**

for each of the ICD9 codes for the input documents. In keeping similarities with our previous discussion of the softmax layer, let's assume, this layer receives $r = \{r_1, \ldots, r_n\}$, as the scores for each of the ICD9 codes. Then, the probability for the j^{th} ICD9 code for the input document would be,

$$q_j(r_i) = \frac{\exp r_{ij}}{\sum_k \exp r_{ik}} \tag{4.10}$$

Here k is the total number of ICD9 codes. We considered $k = \{3, 5, 7\}$ for our experiments. For each dataset, we trained three models to distinguish between three, five, and seven categories (ICD9 Codes) of documents.

4.3.3 Execution Time

Table 4.10 shows the training and testing time for the i2b2 dataset. We calculated the computation time for the Baseline Dernoncourt *et al.* [27] model and our GRU based and the self-attention model. These computations were conducted on the same machine having NVIDIA Titan V GPU (12 GB memory). Note, here the testing time refers to the prediction time for an unknown corpus replicating the real-world use-case and the training time denotes a single epoch. Now, from Table 4.10, we can see that the testing time for the self-attention model is significantly lower than the

RNN based model. Moreover, even though the attention model has almost 35 times more parameters to train, it was able to train the model in less time than both the RNN based models. An RNN based model looks at the tokens sequentially, whereas the self-attention model looks at the whole sentence at the same time and process it in conjunction. Therefore, the self-attention model can parallelize the token inputs and learn the overall concept of a sentence at the same time.

4.4 Discussion

In this work, we examined GRUs, stacked GRUs, and different permutations of these units to increase the recall for the de-identification task. However, we observed that we need to change the core technique to self-attention to achieve faster execution time and better recall. In the following subsections, we discuss our observations and the implications from the results we presented in the previous section.

4.4.1 Effect of RNN models

In the i2b2 dataset, all three RNN models have better precision than [27]. Although all of these models have over 95% recall value, they fall behind the benchmark model resulting in the lower F1 score. Our GRU-GRU model has better precision than the single GRU model. However, the GRU model has a better F1 score than both GRU-GRU and LSTM-GRU models as it has much higher recall value than both of these models. As mentioned earlier, this is surprising as the additional RNN should have produced a better result. We speculate that this extra layer led to memorizing more training data, which consequently resulted a poor performance in the test set.

For the MIMIC-III dataset, the results are almost similar to the i2b2 dataset where the GRU model has the best F1 score. However, the performance differences between the proposed RNN models are nominal. The better performance of the GRU model signifies that the extra RNN layer does not help improve the performance. Moreover, the stacked RNN models take significant time to converge.

4.4.2 Effect of Attention Models

In the i2b2 dataset, our attention model outperforms all the RNN based models. This model is finetuned on a model pretrained on Wikipedia. The large volume of the corpus makes the model better equipped to handle Out-of-Vocabulary words. Also, our results suggest that the context-aware word embedding used for the pretrained model gave an edge during processing different uses of the same word as noun and verb. On the MIMIC-III dataset, the attention model has the highest precision. However, the low recall value indicates the high number of false-negative values. Our analysis of the result shows that most of the errors accrued due to tokenizations errors. For example, a "[","*" or"]" was counted as PHI token in the dataset which our self-attention model predicted as not-PHI as the model is not custom-tailored to detect these patterns.

4.4.3 Effect of CRF

From Table 4.3, we observe that for the GRU model, there is a substantial gain in the F1 score when we use CRF instead of the softmax layer. The CRF layer helps the model to consider the correlations between neighboring labels and jointly decode

the labels. A softmax function individually predicts the label based on the vectors received from the context modeling layer. It does not take into account the prediction of the neighboring tokens, in contrast to the CRF algorithm. This argument is also applicable to the results found for the self-attention model. The attention model has a 0.823% rise in precision when using the softmax layer. However, the recall value for the attention model increases by 1.01%, resulting in a 0.098% F1 improvement.

4.4.4 Effect of Character Embedding and Dropouts

From Table 4.3, we observed that using the character embedding gave an 8.637% gain in the recall value. This improvement is expected as without the character embedding the model did not have any way of learning the similarity between words. This result motivated us to use the character embedding for all of our other RNN based models. Furthermore, we avoid the overfitting issues by regularization and dropout, employed in all layers of the deep neural network. Our results in Table 4.3 denote that a 50% dropout resulted in 1.49% improvement compared to the 0% dropout with GRUs. Higher dropout values ($> 50\%$) reduced the probability of overfitting but also reduced the model's ability to learn. Therefore, we used a maximum of 50% dropout while training the models for results in the *test dataset*. For the self-attention model, the F1 score (for test set) follows a similar pattern. Lower dropout values converge quickly (< 20 epochs) and overfit the training dataset. However, it performs poorly on the test sets as it only achieves 97.923% recall, which improves for 25% dropouts. Note, for 50% dropout the performance of the self-attention model drops in terms of precision (93.78%) which is essential for utility (details in Section 4.3.2).

4.4.5 Effect of Precision and Recall on the utility

Our model and earlier baseline work from Dernoncourt *et al.* [27] did not achieve 100% precision or recall and resulted in higher utility scores. This is entirely due to the number of tokens available or removed in the de-identified documents determined by the FNs or FPs, respectively. For example, our 0.58% recall improvement reduced the number of PHI instances compared to the baseline approach. It is important to note that we have higher precision (+0.11%), which means we did not remove more tokens wrongfully in comparison to the baseline eventually, increasing the BLEU score slightly (0.023). These observations are consistent for $n = \{2, 3\}$ grams as we have a better utility score compared to the baseline in all cases. It is critical to understand that BLEU scores do not necessarily represent the absolute utility of the de-identified data. As mentioned earlier, we treated de-identification as a machine translation problem where any non-sanitized (with PHI) document is translated into sanitized (without PHI) documents. The ratio of the matched word in both sanitized and non-sanitized documents thus directly reflects the utility of the sanitized documents when considering BLEU as our utility metrics, as the privacy of the sanitized document is guaranteed with the recall percentage of the de-identification model.

Table 4.6 shows that our model offered similar (overall) word frequency as the baseline and GT. This result is consistent with our previous utility results in Section 4.3.2, where the GT has a 100% precision and recall value. Similar to our prior argument, our model has a precision value of 98.031% better than the current baseline [27]. Therefore, more non-sensitive words were left in the de-identified document compared to the baseline. However, we do not perform better than GT in the topic-

wise word frequency for all topics. From Table 4.6, we can see that our model has a better score than the baseline for all the topics except Topic-2. Nevertheless, we compensate for that performance cumulatively on other topics.

The results in Table 4.9 show that all accuracy values decreases with the increment in the disease numbers. For example, when we consider the first three diseases, the accuracy is 91.8%, which decreases to 88.1% for the first fives in our sanitized dataset. This decreasing accuracy is prevailing across both de-identified datasets and different disease numbers.

In an automated de-identification task, a machine learning model can often simplify it by identifying the less-frequent words. This is possible as PHI tokens (*i.e.*, Names, Location, etc.) may appear only a few times in the whole corpus. Furthermore, they usually follow the same pattern (*i.e.*, Dates). It is noteworthy that these words do not affect the diagnosed disease as they are unimportant while making an ICD9 prediction. As both de-identification tasks essentially reduce the number of tokens from the original MIMIC-III corpus, it inherently simplifies the learning complexity imposed by inconsequential tokens (for disease prediction). Hence, the two models trained with the de-identified datasets from Dernoncourt *et al.* (Baseline) and ours provide better accuracy than the non-sanitized model.

The model trained with the de-identified dataset from our proposed model has better accuracy than the baseline. A closer look at the predicted results from both de-identification models showed that our self-attention model de-identified 7.91% of the total words, whereas Dernoncourt *et al.* de-identified 8.19% from the whole corpus. This higher percentage of the de-identified words may suggest better privacy

(including FPs); although, our self-attention model achieves a higher precision that establishes the fact that our model de-identified a lower number of non-sensitive words (lower FPs). These non-sensitive words have attributed to the incremental accuracy improvement compared to the other approach.

4.5 Limitations

In this article, we conducted experiments on three different real-life datasets varying various neural network parameters. Below, we discuss some of our observations and limitations of the proposed approach.

- **Class Imbalance:** We measure performance based on precision and recall (relies on TP, FP, and FN). We did not discuss anything about the true negatives or negative classes in general. One of the interesting sides of the problem is the size of the negative classes (number of non-sensitive tokens) compared to the positive ones. For example, in i2b2 we had only 11,243 positive (PHI) tokens, whereas negative class was 35 times bigger with 394,790 tokens. Currently, we do not incorporate this imbalance in our data processing or architecture model. In the future, we plan to investigate further this issue to increase performance.

- **Preprocessing MIMIC-III:** As mentioned earlier, we manually annotated the MIMIC-III dataset with existing data. This process may contain some edge cases which we could not manually intervene due to the large size of the corpus. Nevertheless, we used the Nursing Note dataset to verify our performance over larger corpus, including the original annotations. Due to the unavailability of a

larger corpus with original PHI labels, we think our experiments on the MIMIC-III dataset will provide insights on how the proposed models will perform head-to-head for a large dataset.

- **Quantifiable Privacy and Utility Model:** As discussed in Section 4.3.2, the privacy loss can be quantified by the recall metric. Since a single false negative token can re-identify an entire EHR at the worst case (i.e., patient's name), a maximum recall of 100% should ensure its privacy. However, increasing recall might adversely affect the precision (via increasing false positives), which will consequently reduce the utility of the underlying data. Measuring the privacy-utility tradeoff is an interesting future research direction.

- **Transfer Learning.** Although we experimented with different real-life datasets, we did not check the transferability of the model. Theoretically, the model should perform reasonably well in case of transfer learning. However, we could not experiment due to the lack of compatible datasets. This is a possible future work for our research.

- **Inference Attacks:** Like all existing works for this problem, our goal is to satisfy the HIPAA privacy requirements. We do not evaluate the re-identification risk through linkage or inference attacks [31]. Please note that existing privacy models (e.g., differential privacy, k-anonymity) cannot be used to release unstructured textual data. By definition, the release of raw textual data will violate the definition of differential privacy. Indeed, if the goal is to release partial information (e.g., frequent keywords), then it is possible to satisfy pri-

vacy definitions like differential privacy. Our goal is to release the raw textual data without PHI instances. We reiterate that this is a significant problem for data owners. Currently, data owners use costly manual approach. An accurate automated approach will significantly reduce the cost of de-identification. Developing an efficient anonymization algorithm for textual data that can provide a provable privacy guarantee remains an open research challenge.

Chapter 5

Differentially Private De-identification Framework

In this chapter we discuss our proposed differentially private de-identification framework. In Section 5.1 we discuss the architecture of the framework. Section 5.2 describes the two usage scenario of this framework. Section 5.3 presents the results and in Section 5.4 we analyze the experimental results.

5.1 Ensemble Framework Architecture

In this section, we describe the ensemble framework and the entities involved in the framework. We also give a brief description of the differentially private deep learning model used for our experiments. Later, we show two experimental scenarios in which the framework could be useful.

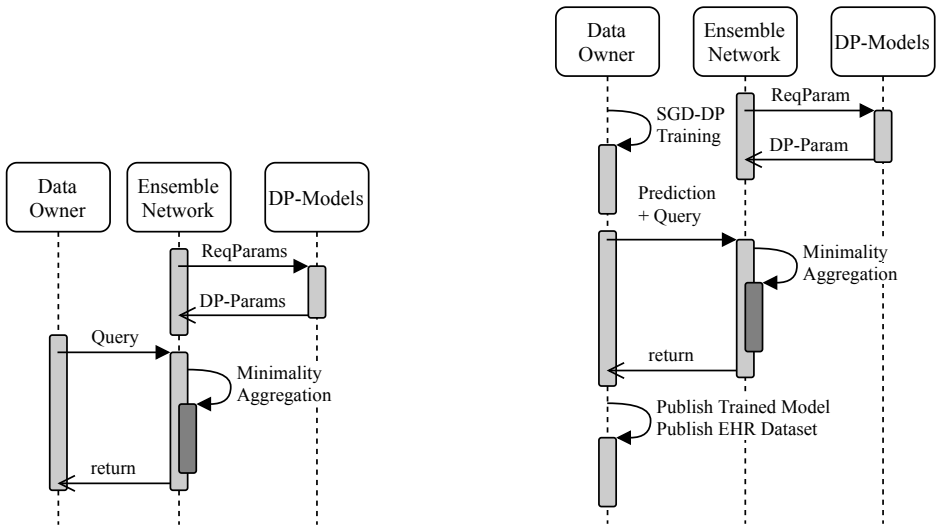
5.1.1 Framework & Entities

There are three main entities involved in our framework as shown in Figure 5.1.

- *Data Owners* are usually hospitals, research organizations, individual researchers, or any organization involved in collecting EHR and are willing to share the data with the research community. Data owners are responsible for the de-identification of the EHRs before public dissemination.

- *DP-Models* are the publicly available differentially private EHR de-identification models. Data owners, having a labeled dataset, can train their deep-learning model using the optimization algorithm (in Section 5.1.2) and publish the model for public use. These models are trained to predict the PHI association of each token in medical texts. In other words, these models predict whether a token is sensitive (PHI) or non-sensitive (Not-PHI) as for multiple models trained on different label sets, the *Minimality Aggregation* (def 5.1.1) algorithm may not reach a consensus.

- *Ensemble Network* is responsible for detecting the PHI tokens in the EHR dataset. It accumulates all the publicly available DP-Models. Upon receiving a request to de-identify, it tests each DP-Models to predict the token labels. Once it has the predictions, it runs the *Minimality Aggregation* algorithm and returns the final prediction. If the data owner has its trained model, it requests including its prediction as well. The ensemble network considers this prediction along with the other DP-Model outputs.



(a) Scenario A (b) Scenario B

Figure 5.1: Differentially private ensemble network usage scenarios. (a) Shows the workflow of the framework when the data owners do not have a labeled dataset and (b) Shows the workflow when the data owners have a labeled dataset.

Definition 5.1.1 *(Minimality Aggregation) For n number of DP-models in the ensemble, the prediction count for each token in the EHR dataset, $j \in [0, n]$. The aggregation algorithm sends queries to each of the models and predicts any token as sensitive if the $j \geq 1$.*

5.1.2 Differentially Private De-identification Model

Model Architecture

The model used in this work has almost the same architecture as the one we described in Chapter 4. However, here instead of using a separate context modeling layer, we opted to use only the self-attention mechanism for determining the dynamic embedding of the tokens. For the convenience of discussion, we describe the components we used precisely for this work.

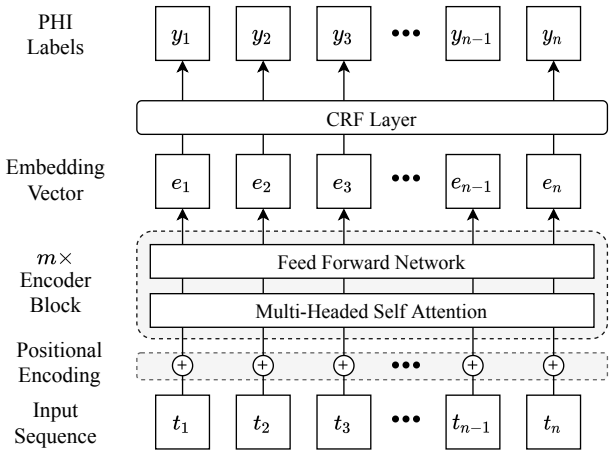


Figure 5.2: Differentially private de-identification model where n is the maximum sequence number and m is the number of encoder block used.

Data Layer The data layer is primarily responsible for the preprocessing of the dataset. In our experiments, we used three different EHR datasets–i2b2 [28], Nursing Note [30] (aka MIMIC-II), and MIMIC-III [68] containing patient notes and discharge summaries of patients. The data layer breaks down the documents into sequences/examples where we used a maximum sequence length of 128. These examples are then converted to a numeric vector where every word is assigned a unique numerical value. The labels for the tokens are also indexed. Now, these numeric sequences and their associated label sequences are forwarded for embedding. In Figure 5.2, $t_1, t_2, \ldots, t_n$ represents the tokens of a input sequence, where n is the maximum sequence length.

Embedding Layer The embedding layer maps each word of the incoming sequences into high dimensional vector space. (See Section 2.2.2 for more details on word embedding). Now in our de-identification model, we used a context-aware dynamic embedding, which assigns different vectors to a single word depending on its

context in the sentence. We used a multi-headed self-attention mechanism for calculating the dynamic embedding. As mentioned before, the self-attention mechanism was first introduced in 2017 by Vaswani *et al.* [10] where the authors introduced the Transformer model, which is constructed as a traditional language modeling system with encoder and decoder stacks. However, instead of using a variant of RNN, Transformer uses self-attention in its encoder and decoder stacks. Here, we employed only the encoder from the Transformer model similar to the BERT [11] model. The transformer-encoder stack uses both the left and right context while calculating its embedding.

Figure 5.2 shows a simplified structure of an encoder block with two sublayers–a multi-headed self-attention layer and a simple fully connected feed-forward layer. m represents the number of identical encoder blocks in the stack. A positional encoding is added to each token before fed to the first encoder block. The output vector of the m^{th} encoder block contains the contextualized embedding of each token. In the figure $e_1, e_2, \ldots, e_n$ represents the embedding vector for tokens $t_1, t_2, \ldots, t_n$, respectively. These embeddings are then passed to the label decoding layer.

Label Decoding Layer The label decoding layer receives the contextualized vector representation of the tokens and predicts the PHI type of the token. We used the Conditional Random Field (CRF) [73] algorithm for detecting whether a token is PHI or not. In a sequence labeling task, CRF considers the adjoining labels as these tokens are related to each other. This gives CRF an advantage over naive softmax based prediction. In Figure 5.2, $y_1, y_2, \ldots, y_n$ represents the labels for the input tokens.

Model Training

The architecture described above has over $110M$ hidden parameters, and these parameters are optimized during training. In this work, we are using differentially private training algorithms to ensure that these parameters do not memorize any sensitive information about the patients from the training set. Differential privacy was initially proposed by Dwork *et al.* [16] for protecting the privacy of any individual as it promises a quantifiable guarantee that their identity or presence in the dataset can not be revealed, regardless of any background information. For more details on the discussion of differential privacy see Section 2.3.

Algorithm 1 Differentially Private Stochastic Gradient Descent [12]

Input : Examples $\{x_1, ..., x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$

Parameters: *lots size* L, *clipping threshold* C, *learning rate* α_t, *noise scale* σ

Initialization: *random initialization of parameters,* θ_0

for $t \in [T]$ **do**

  Select L_t examples with sampling probability L/N

  foreach $i \in L_t$ **do**

    Calculate $G_t(x_i) \leftarrow \Delta_{\theta_t}\mathcal{L}(\theta_t, x_i)$ `//calculate gradient`

  end

  $\bar{G}_t(x_i) \leftarrow G_t(x_i)/\max(1, \frac{\|G_t(x_i)\|_2}{C})$ `//clip` l_2 `norm from gradient`

  $\tilde{G}_t \leftarrow \frac{1}{L}(\sum_i \bar{G}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 I))$ `//gaussian mechanism for output perturbation`

  $\theta_{t+1} \leftarrow \theta_t - \alpha_t \tilde{G}_t$ `//descent`

end

Output: θ_T, privacy cost (ε, δ)

Algorithm 1 outlines the DP learning algorithm we used for training our de-identification model. The algorithm was first proposed by Abadi *et al.* [12] and is similar to the minibatch Stochastic Gradient Descent (SGD) with a modification on the gradient update where additional noise was introduced. Suppose θ represents

the $110M$ parameters of our model. We train these parameters by minimizing the loss function $\mathcal{L}(\theta)$ using Algorithm 1. At each step of the algorithm, the gradient $\Delta_\theta \mathcal{L}(\theta, x_i)$ is calculated for a random subset of examples. Then the l_2 norm is clipped of each gradient and the gradient, G is replaced by $G/max(1, \frac{||G||_2}{C})$. This ensures that if $||G||_2 \leq C$, then G is preserved otherwise G is scaled down by C. The average of these gradients is then calculated. The calculation and clipping of gradients is a deterministic function where the Gaussian noise mechanism was employed to obtain the output perturbation according to definition 2.3.1.

The final step of this algorithm is a step towards the opposite direction of the average noisy gradient. The original proposal introduced a concept of 'Lots,' which is represented with L [12]. During training, multiple batches are grouped together into these lots to add noise. This helps to reduce large memory consumption. Each lot is constructed by selecting each example independently with a probability $q = L/N$ where N is the size of the input dataset (in our case, the total number of examples). The output of this algorithm is the converged model, and the privacy loss calculated using a newly introduced 'Moments' accountant. This ensures that at every step the algorithm is $(\mathcal{O}(q\varepsilon\sqrt{T}), \delta)$-differentially private where δ is small and $T \geq 1/q$. Throughout this Thesis, we will refer to Algorithm 1 as *DP-SGD*.

5.2 Usage Scenario

Let there be n DP-Models for de-identification already publicly available. If a data owner wants to publish a dataset that is not yet publicly available, but of significance, one of two scenarios will arrive. In scenario A, the data owner might

have enough resources to label the dataset. In that case, it will be able to train its model. In scenario B, the data owner might not have enough resources to create a labeled dataset. Figure 5.1a and Figure 5.1b show sequence diagrams on how the ensemble framework works for scenario A and scenario B, respectively.

Scenario A–Data owners do not have labeled (PHIs) dataset

Step 1: The ensemble network accumulates the publicly available *DP-Models*. As these models are trained using the DP-SGD algorithm, the parameters do not reveal anything new information about the underlying participants. Now, this step is essential as the ensemble network is completely dependent on the DP-Models.

Step 2: The data owner decides to publish an EHR dataset that is not yet publicly available. It breaks down the EHR documents into *queries*, where each query corresponds to a word sequence and sends the query sequence to the ensemble network for de-identification. We assume the communication protocol between the data owner and the ensemble network is secured.

Step 3: After receiving the query sequence, the ensemble network collects the predictions from each of the models in the ensemble. It then runs the *Minimality Aggregation* (def. 5.1.1) algorithm on the predictions and calculates the final prediction. Now, Step 2 and Step 3 iterates until the whole dataset is de-identified.

Step 4: The final de-identified dataset is then returned to the data owner to publish the dataset.

Scenario B–Data owners have labeled dataset

Step 1: The data owner has a labeled dataset, which allows it to train its DP-Model. We used the model architecture and training mechanism described in Section 5.1.2. In parallel to the model training, the ensemble network downloads the publicly available DP-Models similarly to Step 1 described in Scenario A.

Step 2: The data owner follows Step 2 of Scenario A with one exception. Along with the query sequence, it sends its predictions for the sequence.

Step 3: The ensemble network aggregates the result from the predictions collected from the member models and the prediction received from the data owner. It follows the method outlined in Step 3 of Scenario A.

Step 4: The final de-identified dataset is then returned to the data owner, and the data owner may choose to publish the dataset along with the trained model.

5.3 Results

We evaluated the performance of individual DP-Models and the ensemble network with Precision (P), Recall (R), and F1-score from the predictions which are defined in Equation 2.1, Equation 2.2 and Equation 2.3, respectively. As previously mentioned, we used three different datasets for the experiments. Each of these datasets is split into a train, test, and a valid set. We trained and validated our DP-Model on the train and validation sets, respectively. The results we discuss in this section are evaluated on the test set. Table 5.1 shows the results for experiments in scenario A where data

Table 5.1: Results for Scenario-A where data owners **do not** have a `labeled dataset` and using dp-private pre-trained model. Here M, N and I represents the DP-Models trained on MIMIC-III, Nursing Note and i2b2 2014 Dataset, respectively. M-N, I-M and N-I represents the ensemble networks built with the DP-Models in different combination.

	i2b2 2014 [28]		
	P	**R**	**F1**
M	49.304	13.191	20.814
N	**91.359**	66.725	**77.123**
M-N	77.482	**68.057**	72.464

(a) Test set: i2b2

	Nursing Note [30]		
	P	**R**	**F1**
I	**58.844**	**64.000**	**61.314**
M	13.462	2.667	4.452
I-M	51.064	**64.190**	56.805

(b) Test set: Nursing Note

	MIMIC-III [68]		
	P	**R**	**F1**
N	41.047	0.127	0.254
I	**96.814**	20.766	34.197
N-I	96.652	**20.767**	34.188

(c) Test set: MIMIC-III

owners do not have the resources to label the dataset. To simulate scenario A, we first train three DP-Models separately on i2b2 2014 [28], MIMIC-III [68], and Nursing Note [30] dataset and refer them to as $\mathbb{I}, \mathbb{M}$, and $\mathbb{N}$, respectively. Then, we select two DP-Models trained on two datasets and evaluate their results on the test set of the remaining dataset. Now, Table 5.1a-5.1c shows the results where the models never seen the testing dataset. For example, in Table 5.1a, DP-models $\mathbb{M}$ and $\mathbb{N}$ are trained on MIMIC-III and Nursing Note dataset, respectively. While testing on the i2b2 dataset, model $\mathbb{N}$ has far better precision or recall score than model $\mathbb{M}$. However,

when combined, $\mathbb{M} - \mathbb{N}$ ensemble has the better recall (68.057%) value than any other individual models (13.191% and 66.725% for $\mathbb{M}$ and $\mathbb{N}$, respectively). The precision value of the ensemble, however, goes down 13.877% along with the 4.659% F1 score. Table 5.1b shows the results for DP-Models $\mathbb{I}$ and $\mathbb{M}$ tested on the Nursing Note dataset. Here model $\mathbb{M}$ again has far worse performance than the other model (in this case, $\mathbb{I}$). We discuss on the possible reason for the low performance of model $\mathbb{M}$ in section 5.4. Nevertheless, $\mathbb{I} - \mathbb{M}$ ensemble has the better recall value (64.190%) than either of the model $\mathbb{I}$ or $\mathbb{M}$. The precision value again is lower than the highest precision value (58.844%) found for model $\mathbb{I}$. Table 5.1c shows the results for the final combination of the experiment where the DP-Models $\mathbb{I}$ and $\mathbb{N}$ is tested on the MIMIC-III dataset. The highest precision here is found for the model $\mathbb{I}$ (96.814%), which is again dropped by 0.162% in the ensemble $\mathbb{N} - \mathbb{I}$. The recall value of this ensemble is 20.767%, which is higher than any individual DP-model.

Table 5.2 shows the results for the experiments in scenario B, where the data owner has a labeled dataset. Now, as the data owner has the labeled dataset, in these experimental settings, we can compare our work with the existing de-identification model. For the i2b2 dataset, Dernoncourt [27] has state-of-the-art results, and for the Nursing Note, dataset Khin [43] has the state-of-the-art result. As mentioned earlier, we selected the randomly 4441 documents for MIMIC-III dataset. So, we had to calculate the results for Dernoncourt [27] on MIMIC-III dataset for ourselves using their publicly available model. We initialized their model will all the pre-trained models provided and reported the best result for Nursing Note and MIMIC-III dataset. Khin [43], however, did not provide any public implementation of their

Table 5.2: Results for Scenario B where data owners **do have** a `labeled dataset` and the DP-Models are trained separately.

Model	**i2b2 2014** [28]			**MIMIC-III** [68]			**Nursing Note** [30]		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Dernoncourt [27]	97.920	97.835	97.877	99.91	99.97	99.94	86.1	69.7	77.0
Khin [43]	98.300	97.370	97.830	-	-	-	91.40	74.30	81.20
Self-attention-DP	97.918	97.149	97.532	99.955	98.987	99.469	87.21	73.180	79.583
Ensemble-DP	82.568	**97.883**	89.576	99.216	99.081	99.149	53.360	**75.619**	62.569

model. Hence, the fields are left blank.

Now, Table 5.2 shows the results for scenario B, where each of the DP-models is trained individually, and the final result is calculated on the target dataset. It is noteworthy that each model is trained on their respective training set. And there was no overlap among these datasets during training. At first, we show the results for our self-attention model for each of the datasets. In both i2b2 2014 and MIMIC-III database the DP-model managed a competitive F1-score (97.532% and 99.469%) to the Dernoncourt *et al.* [27] model (97.877% and 99.94%, respectively). In the Nursing Note dataset, although we outperformed Dernoncourt*et al.* [27] by almost 2.5% F1-score, our model falls short of 1.617% F1-score of the model proposed by Khin *et al.* [43]. The final row in Table 5.2 shows the results for the ensemble networks. In each of the three datasets, the ensemble network's recall value is always significantly better than the individually trained DP-models, which demonstrate the effectiveness of the ensemble network. The precision value goes down as other models influence the decision. However in the context of the de-identification problem the recall values bears more importance than the precision. Furthermore, for dataset i2b2 and Nursing Note dataset, the Ensemble-DP has 0.048% and 1.319% improvement in the recall value over the benchmark models.

Table 5.3: Change of ε for small to medium noise where for all the dataset `batch size`=16, `epoch`=100 and `delta`=$10^{-}5$. Here 'Noise' refers to the multiplier used during the gradient update.

Dataset	**# of Docs.**	**Noise**	ε	**P(%)**	**R(%)**	**F1(%)**
i2b2 2014 [28]	790	0.25	356	98.683	97.669	98.173
		0.45	21.7	98.009	97.228	97.617
		0.55	9.15	97.918	97.149	97.532
MIMIC-III [68]	3550	0.25	316	99.934	99.079	99.505
		0.55	6.95	99.955	98.987	99.469
Nursing Note [30]	1948	0.25	343	90.692	72.381	80.508
		0.55	8.41	87.215	73.180	79.583

5.4 Discussion

The performance of DP-Models, shown in Table 5.1 are not as good as we expected. One of the main reasons we speculate is the heterogeneity of the underlying datasets. In other words, training data in these datasets were very different, and the models trained on these datasets could not generalize well to a completely different test set. Nevertheless, the ensembles' recall values always improved as the algorithm prioritizes a low false-negative rate over a low false-positive rate. This prioritizing mechanism, however, is responsible for the lower precision of the ensembles.

The results of ensemble networks in Table 5.2 are also consistent with the observations of Table 5.1a-5.1c. The Ensemble-DP has a higher recall value as it predicts a token as private (PHI) if only one of the underlying DP-models in the ensemble predicts it as such. Alternatively, we could classify according to a consensus of predictions that could improve the precision; however, it would open up the possibility of missing more sensitive information that we wanted to avoid. As we have seen from the results of Table 5.1, models trained on a different dataset often do not generalize well enough to newer test cases.

Table 5.3 shows the relationship between ε and the performance of the DP-Models. We calibrated the ε value to the noise multiplier, while the delta (δ), training epoch, and batch size were always fixed. The value of epsilon is inversely proportional to the noise. And for larger values of epsilon (ε), the performance was better for DP-Models. We found that for a larger dataset, the privacy budget was smaller. This is expected as with an increasing number of documents, the probability of choosing an example reduced, which in turn reduced the value of epsilon. For the i2b2 2014 [28], MIMIC-III [68], and Nursing Note [30] dataset, we chose epsilon $9.15, 6.95$, and 8.41 as they gave a reasonable privacy guarantee with competitive accuracy.

Chapter 6

Conclusion

6.1 Summary

In Chapter 4 we proposed an architecture employing new deep learning methods to de-identify textual data and analyzed their performance with existing methods. Experimental results showed that our self-attention based approach is computationally efficient and performs better than the state-of-the-art models. We also introduced novel approaches to measure the utility of the de-identified documents and analyze the relationship between utility, precision and recall value of neural network-based models. Finally, our proposed loss function improves the recall value compared to regular loss value, although there is a trade-off between the recall and precision values.

In Chapter 5 we explored privacy-preserving deep learning techniques for de-identification. We showed that using a differentially private optimization algorithm, we can achieve a competitive score in three different datasets. Then, we proposed an ensemble framework for de-identification, where these publicly released differentially

private models could be used. We showed that the ensemble framework improves the recall value over the individual models. After closer evaluation of our experimental results in two different settings, we conclude that our framework is most useful when the data owner has a labeled dataset, and the models are separately trained without any overlapping data.

6.2 Future Works

One of the major directions for future exploration of this work is comprehensive utility analysis for the differentially private models. In Chapter 4 we introduced the utility analysis in de-identification. In the future, these experiments could be applied to the differentially private de-identification model. In Chapter 5 we presented the performance of the model for different values of epsilon(ε). Further experiments could be designed to extrapolate the co-relation between the value of ε and the utility of the de-identified data. In other words, a quantitative analysis of the relationship between privacy and utility for unstructured text data.

Another possible future direction for this is a quantifiable analysis of the re-identification risk for both the de-identified data and the differentially private models. There have been several works on identifying the privacy vulnerability of the trained model in the field of Natural Language Processing. However, no direct work of these sorts could be found for the de-identification problem. Hence, an exciting future work could be analyzing the co-relation between the de-identification model's performance and the re-identification risk of the de-identified data.

Appendix A

Additional Results

A.1 Category-wise Precision and Recall

Table A.1: HIPAA defined PHI category wise Precision and recall comparison for the self-attention model for the i2b2 2014 and MIMIC-III dataset.

HIPAA PHI	i2b2 2014			MIMIC-III		
	P	R	F1	P	R	F1
ID	99.185	96.358	97.751	100.00	99.955	99.978
Contact	99.113	95.717	97.386	100.00	100.00	100.00
Name	98.929	98.456	98.692	99.959	99.997	99.978
Location	94.832	96.122	95.473	99.876	100.00	99.938
Date	98.840	100.00	99.416	99.968	100.00	99.984
Age	98.206	99.176	98.689	-	-	-
Profession	91.554	98.905	95.088	-	-	-

Table A.1 presents the category-wise precision, recall and F1 scores listed in Table 2.1. For the i2b2 dataset, the `DATE` and `AGE` category have the highest recall value, although both have lower precision than the `ID` category, which has the highest precision value. The lowest F1 score is for the `CONTACT` type. For MIMIC-III dataset, the

Table A.2: PHI token count in each category for i2b2 2014 and MIMIC-III dataset.

PHI Instances		**i2b2 2014** [28]		**MIMIC-III** [68]	
Category	**Sub-Category**	**Train**	**Test**	**Train**	**Test**
DATE	NA	7450	4931	30843	6625
NAME	DOCTOR, PATIENT, USERNAME	4425	2833	54084	14525
AGE	NA	1254	774	36	0
CONTACT	PHONE, FAX, EMAIL	381	229	379	75
ID	MEDICALRECORD, IDNUM, DEVICE, BIOID	910	639	12665	3257
LOCATION	HOSPITAL, CITY, STATE, STREET, ZIP, COUNTRY, ORGANIZATION, LOCATION-OTHER	2306	1680	7423	1819
PROFESSION	NA	263	157	-	-
Total	**NA**	**16962**	**11243**	**105430**	**26301**

lowest F1 score was found for the `LOCATION` category, which is similar to the result for the i2b2 dataset as the location names were sparse. The MIMIC-III dataset did not have any `AGE` or `PROFESSION` type of PHI is the test set. Hence, the respective columns are left blank.

A.2 Dataset Breakdown

Table A.2 shows the details for the i2b2 2014 [28] and MIMIC-III [68] datasets.

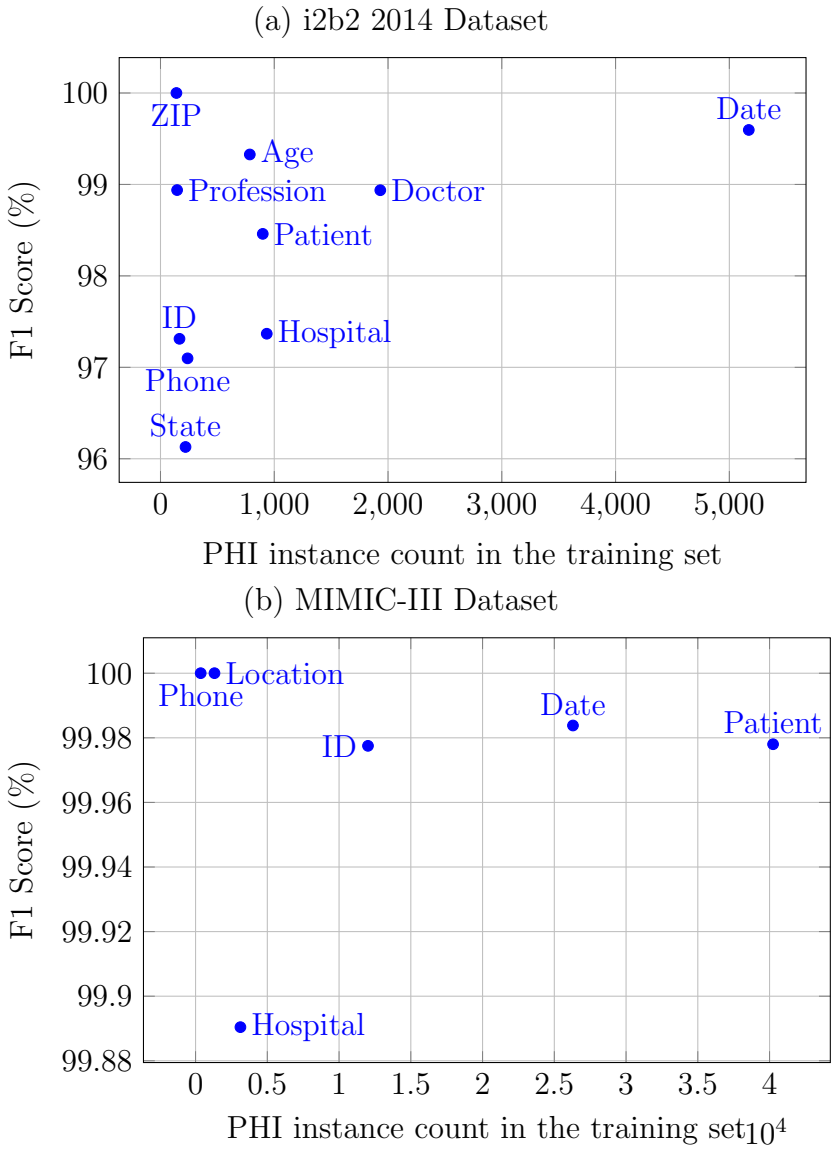


Figure A.1: Impact of the number of PHI instances in the training set on the attention model's performance for each PHI category in the i2b2 2014 and MIMIC-III dataset.

A.3 Effect of training set size

Figure A.1 depicts the impact of the number of PHI instances in the training set on the attention model's performance. The horizontal axis of Figure A.1 represents the total number for each category of PHI instances in the training set. The vertical axis

shows the F1 score for each of these categories in the test set. As expected, entities with a large number of labeled instances in the training dataset tend to be detected more accurately. However, the correlation is far from conclusive. Some instances with a lower number of labeled instances are detected more accurately than some types with a higher number of labeled instances. This indicates that some instances are harder to detect than others. For example in the i2b2 dataset (Figure A.1a), although `ZIP` type PHI has fewer labeled data than `DOCTOR` or `HOSPITAL`, it is more accurately detected than the latter ones. This result is expected since tokens containing a `ZIP` number are typically similar, while names of hospitals and doctors can appear in differently.

A.4 Variation of Scenario B

Table A.3 shows results for the variation where each model is fine-tuned on a pre-trained model. For example, in Table A.3a Init-i2b2 refers to model fine-tuned over a DP-Model pre-trained on i2b2. Init-i2b2-MIMIC refers to a DP-Model fine-tuned on a DP-Model pre-trained sequentially on the i2b2 [28] and MIMIC-III [68] datasets. Table A.3b-A.3c shows the results for other setup variations for scenario B. In this variation, the model is fine-tuned over the DP-models with the target dataset. Table A.3a shows the results for Nursing Note dataset. This table shows the results for models first fine-tuned on the DP-model trained on the i2b2 dataset. Later, it shows the model's results, which is first fine-tuned on the i2b2 and then the MIMIC-III dataset. All of the training in these experiments is differentially private. Now, Table A.3, where pre-trained DP-Models are fine-tuned, does not show

Table A.3: Results for Scenario B where data owners **do have** a `labeled dataset` and using pre-trained DP-Models.

Model	**Nursing Note** [30]		
	P	**R**	**F1**
Dernoncourt [74]	86.1	69.7	77.0
Khin [43]	91.4	74.3	81.2
Self-attention-DP	87.215	73.180	79.583
Init–i2b2	82.420	68.762	74.974
Init–i2b2-MIMIC	82.833	73.524	77.901

(a) Test set : Nursing Note

Model	**i2b2 2014** [28]		
	P	**R**	**F1**
Dernoncourt [74]	97.920	97.835	97.877
Self-attention-DP	97.918	97.149	97.532
Init–MIMIC	98.024	96.867	97.442
Init–NN-MIMIC	97.469	96.506	96.985

(b) Test set : i2b2 2014

Model	**MIMIC-III** [68]		
	P	**R**	**F1**
Dernoncourt [74]	99.91	99.97	99.94
Self-attention-DP	99.955	98.987	99.469
Init–i2b2	99.946	99.058	99.500
Init–NN-i2b2	99.944	98.955	99.447

(c) Test set: MIMIC-III

much improvement over Self-attention-DP models, which did not use pre-trained DP-Models. One probable reason could be the DP-Models are already trained on a noisy gradient and using DP-SGD further added noise to the parameter, resulting in low performance.

Bibliography

[1] C. Pedersen, P. Schneider, and D. Scheckelhoff, "ASHP national survey of pharmacy practice in hospital settings: Prescribing and transcribing-2016," *In AJHP*, 2017. ii

[2] Y. Wang, Wang *et al.*, "Clinical information extraction applications: a literature review," *In JBI*, 2018. ii

[3] J. L. Fernández-Alemán, Señor *et al.*, "Security and privacy in electronic health records: A systematic literature review," *In JBI*, 2013. 1

[4] F. Toscano, E. O'Donnell *et al.*, "Electronic health records implementation: can the european union learn from the us?" *EJPH*, 2018. 1, 12

[5] H. Office for Civil Right, "Standards for privacy of individually identifiable health information. final rule." *Federal Register*, 2002. 2, 15

[6] M. J. Douglass, G. D. Cliffford, A. Reisner *et al.*, "De-identification algorithm for free-text nursing notes," *In CinC*, 2005. 2, 3, 27

[7] I. Neamatullah, M. M. Douglass, L.-w. H. Lehman *et al.*, "Automated de-

identification of free-text medical records," *BMC Medical Informatics and Decision Making*, 2008. 2

[8] T. Chen, R. M. Cullen, and M. Godwin, "Hidden markov model using dirichlet process for de-identification," *In JBI*, 2015. 3, 28

[9] X. Ma and E. H. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," *CoRR*, 2016. 3

[10] A. Vaswani, Shazeer *et al.*, "Attention is All you Need," in *NIPS*, 2017. 3, 40, 70

[11] J. Devlin, Chang *et al.*, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv:1810.04805*, 2018. 3, 4, 40, 70

[12] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318. 4, 5, 71, 72

[13] M. Coavoux, S. Narayan, and S. B. Cohen, "Privacy-preserving neural representations of text," in *Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 1–10. 4

[14] X. Pan, M. Zhang, S. Ji, and M. Yang, "Privacy risks of general-purpose language models," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, may 2020. 4

[15] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1322–1333. 4

[16] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014. 5, 6, 22, 71

[17] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *Advances in neural information processing systems*, 2009, pp. 289–296. 5, 32

[18] M. Pathak, S. Rane, and B. Raj, "Multiparty differential privacy via aggregation of locally trained classifiers," in *Advances in Neural Information Processing Systems*, 2010, pp. 1876–1884. 5, 32

[19] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, 2013, pp. 245–248. 5, 32

[20] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1310–1321. 5, 32

[21] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016. 5

[22] J. Hamm, Y. Cao, and M. Belkin, "Learning privately from multiparty data," in *International Conference on Machine Learning*, 2016, pp. 555–563. 5, 33

[23] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *arXiv preprint arXiv:1610.05755*, 2016. 5, 33

[24] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv:1409.0473*, 2014. 6, 18

[25] C. Goller and A. Kuchler, "Learning task-dependent distributed representations by backpropagation through structure," in *ICNN*, 1996. 6

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997. 6, 17

[27] F. Dernoncourt, J. Y. Lee, O. Uzuner, and P. Szolovits, "De-identification of patient notes with recurrent neural networks," *In JAMIA*, 2016. 6, 7, 29, 30, 37, 47, 48, 49, 52, 55, 57, 58, 61, 76, 77

[28] A. Stubbs, Kotfila *et al.*, "Automated systems for de-identification of longitudinal clinical narratives," *JBI'15*. 8, 30, 46, 48, 69, 75, 77, 78, 79, 83, 85, 86

[29] 8

[30] J. Lee, Scott *et al.*, "Open-access mimic-ii database for intensive care research," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE. 8, 46, 48, 69, 75, 77, 78, 79, 86

[31] S. L. Garfinkel, "De-identification of personal info." *National institute of standards and technology*, 2015. 14, 15, 64

[32] "Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule," https://www.hhs.gov/hipaa/index.html. 15

[33] A. Nematzadeh, S. C. Meylan, and T. L. Griffiths, "Evaluating vector-space models of word representation, or, the unreasonable effectiveness of counting words near other words." in *CogSci*, 2017. 20

[34] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2006, pp. 486–503. 22

[35] M. Douglass, G. D. Clifford *et al.*, "Computer-assisted de-identification of free text in the mimic ii database," in *CinC*, 2004. 27

[36] G. Szarvas, Farkas *et al.*, "State-of-the-art anonymization of medical records using an iterative machine learning framework," *In JAMIA*, 2007. 27

[37] B. He, Y. Guan, J. Cheng *et al.*, "CRFs based de-identification of medical records," *In JBI*, 2015. 28

[38] Z. Liu, Chen *et al.*, "Automatic de-identification of electronic medical records using token-level and character-level conditional random fields," *In JBI*, 2015. 28, 30

[39] A. Dehghan, Kovacevic *et al.*, "Combining knowledge-and data-driven methods for de-identification of clinical narratives," *In JBI*, 2015. 28

[40] H. Yang and J. M. Garibaldi, "Automatic detection of protected health information from clinic narratives," *Journal of biomedical informatics*, vol. 58, pp. S30–S38, 2015. 28, 29

[41] S. Yadav, Ekbal *et al.*, "Deep learning architecture for patient data de-identification in clinical records," in *Proceedings of the clinical natural language processing workshop (ClinicalNLP)*, 2016, pp. 32–41. 29

[42] Z. Liu, B. Tang, and X. Wang, "De-identification of clinical notes via recurrent neural network and conditional random field," *In JBI*, 2017. 29, 30

[43] K. Khin, Burckhardt *et al.*, "A deep learning architecture for de-identification of patient notes: Implementation and evaluation," *arXiv preprint arXiv:1810.01570*, 2018. 29, 48, 49, 50, 76, 77, 86

[44] C. Gentry and D. Boneh, *A fully homomorphic encryption scheme*. Stanford university Stanford, 2009, vol. 20, no. 9. 30

[45] A. C.-C. Yao, "How to generate and exchange secrets," in *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*. IEEE, 1986, pp. 162–167. 30

[46] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979. 30

[47] F. McKeen, I. Alexandrovich, I. Anati, D. Caspi, S. Johnson, R. Leslie-Hurd, and C. Rozas, "Intel® software guard extensions (intel® sgx) support for dynamic memory management inside an enclave," in *Proceedings of the Hardware and Architectural Support for Security and Privacy 2016*, 2016, pp. 1–9. 30

[48] A. A. Badawi, J. Chao, J. Lin, C. F. Mun, J. J. Sim, B. H. M. Tan, X. Nan, K. M. M. Aung, and V. R. Chandrasekhar, "The alexnet moment for homomorphic encryption: Hcnn, the first homomorphic cnn on encrypted data with gpus," *arXiv preprint arXiv:1811.00778*, 2018. 31

[49] E. Chou, J. Beal, D. Levy, S. Yeung, A. Haque, and L. Fei-Fei, "Faster cryptonets: Leveraging sparsity for real-world encrypted inference," *arXiv preprint arXiv:1811.09953*, 2018. 31

[50] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International Conference on Machine Learning*, 2016, pp. 201–210. 31

[51] L. J. Aslett, P. M. Esperança, and C. C. Holmes, "Encrypted statistical machine learning: new privacy preserving methods," *arXiv preprint arXiv:1508.06845*, 2015. 31

[52] W. Lu, S. Kawasaki, and J. Sakuma, "Using fully homomorphic encryption for statistical analysis of categorical, ordinal and numerical data." *IACR Cryptol. ePrint Arch.*, vol. 2016, p. 1163, 2016. 31

[53] W.-J. Lu, Y. Yamada, and J. Sakuma, "Privacy-preserving genome-wide association studies on cloud environment using fully homomorphic encryption," in *BMC medical informatics and decision making*, vol. 15, no. S5. Springer, 2015, p. S1. 31

[54] S. Wang, Y. Zhang, W. Dai, K. Lauter, M. Kim, Y. Tang, H. Xiong, and X. Jiang, "Healer: homomorphic computation of exact logistic regression for secure rare disease variants analysis in gwas," *Bioinformatics*, vol. 32, no. 2, pp. 211–218, 2016. 31

[55] A. Al Badawi, L. Hoang, C. Fook Mun, K. Laine, and K. M. M. Aung, "Privft: Private and fast text classification with homomorphic encryption," *arXiv*, pp. arXiv–1908, 2019. 31

[56] G. Costantino, A. La Marra, F. Martinelli, A. Saracino, and M. Sheikhalishahi, "Privacy-preserving text mining as a service," in *2017 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2017, pp. 890–897. 31

[57] D. Reich, A. Todoki, R. Dowsley, M. De Cock *et al.*, "Privacy-preserving classification of personal text messages with secure multi-party computation," in *Advances in Neural Information Processing Systems*, 2019, pp. 3757–3769. 31

[58] M. De Cock, R. Dowsley, A. C. Nascimento, D. Reich, and A. Todoki, "Privacy-preserving classification of personal text messages with secure multi-party computation: An application to hate-speech detection," *arXiv preprint arXiv:1906.02325*, 2019. 31

[59] R. Bassily, A. Smith, and A. Thakurta, "Private empirical risk minimization: Efficient algorithms and tight error bounds," in *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE, 2014, pp. 464–473. 32

[60] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Privacy aware learning," *Journal of the ACM (JACM)*, vol. 61, no. 6, pp. 1–57, 2014. 32

[61] "Implementation of the rnn and attention models, evaluation scripts, and api," https://github.com/UofM-DSP/WebDI.git. 34

[62] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014. 37, 44

[63] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013. 37

[64] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition." Association for Computational Linguistics, 2009. 42

[65] Y. Wu, M. Schuster, Chen *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016. 44

[66] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AISTATS*, 2010. 44

[67] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014. 45

[68] S. Gehrmann, F. Dernoncourt, Y. Li, E. T. Carlson *et al.*, "Comparing deep learning and concept extraction based methods for patient phenotyping from clinical narratives," *PLOS ONE*, 2018. 45, 46, 48, 54, 69, 75, 77, 78, 79, 83, 85, 86

[69] R. Collobert, J. Weston, L. Bottou, M. Karlen, and Kavukcuoglu, "Natural language processing (almost) from scratch," *In JMLR*, 2011. 45

[70] K. Papineni, Roukos *et al.*, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002. 51

[71] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003. 54

[72] K. J. O'malley, K. F. Cook, M. D. Price, K. R. Wildes, J. F. Hurdle, and C. M. Ashton, "Measuring diagnoses: Icd code accuracy," *Health services research*, vol. 40, no. 5p2, pp. 1620–1639, 2005. 54

[73] J. Lafferty, McCallum *et al.*, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001. 70

[74] F. Dernoncourt, J. Y. Lee, O. Uzuner, and P. Szolovits, "De-identification of patient notes with recurrent neural networks," *Journal of the American Medical Informatics Association*, vol. 24, no. 3, pp. 596–606, 2017. 86