Information Extraction with Neural Networks

by

Ji Young Lee

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2017

Author .....
Department of Electrical Engineering and Computer Science
May 18, 2017

Certified by .....
Peter Szolovits
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Information Extraction with Neural Networks

by

Ji Young Lee

Submitted to the Department of Electrical Engineering and Computer Science
on May 18, 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science

Abstract

Electronic health records (EHRs) have been widely adopted, and are a gold mine for clinical research. However, EHRs, especially their text components, remain largely unexplored due to the fact that they must be de-identified prior to any medical investigation. Existing systems for de-identification rely on manual rules or features, which are time-consuming to develop and fine-tune for new datasets. In this thesis, we propose the first de-identification system based on artificial neural networks (ANNs), which achieves state-of-the-art results without any human-engineered features. The ANN architecture is extended to incorporate features, further improving the de-identification performance. Under practical considerations, we explore transfer learning to take advantage of large annotated dataset to improve the performance on datasets with limited number of annotations. The ANN-based system is publicly released as an easy-to-use software package for general purpose named-entity recognition as well as de-identification. Finally, we present an ANN architecture for relation extraction, which ranked first in the SemEval-2017 task 10 (ScienceIE) for relation extraction in scientific articles (subtask C).

Thesis Supervisor:

Peter Szolovits
Professor of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology.

Thesis Committee:

Roger Mark
Distinguished Professor of Health Sciences and Technology and of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology.

Hung H. Bui
Senior machine learning research scientist at Adobe Research, San Jose, CA.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my advisor Peter Szolovits for the continuous support of my research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D. study.

Besides my advisor, I would like to thank the rest of my thesis committee, Roger Mark and Hung Bui, for their insightful comments and encouragement. Roger Mark and his group, the MIT Laboratory for Computational Physiology, have generously shared the MIMIC dataset for my research on de-identification. I would like to especially thank Alistair Johnson, Li-wei Lehman, and Tom Pollard for their helpful suggestions and technical assistance. I also had a great pleasure working with Hung Bui and Trung Bui at Adobe Research during the summer.

Throughout the course of my Ph.D. study, I was very glad to be surrounded by extremely bright and warm-hearted colleagues in the CSAIL Clinical Decision Making Group: William Boag, Franck Dernoncourt, Michele Filannino, Marzyeh Ghassemi, Owen Hsu, Nathan Hunt, Mohamed Kane, Lydia Letham, Matthew McDermott, Tristan Naumann, Divya Pillai, Harini Suresh, and Ziyu Wang. All the work presented in this thesis was performed jointly with Franck Dernoncourt.

I am also grateful for the financial support provided by Philips Research, which funded part of my Ph. D. study. I would like to especially thank Oladimeji Farri for organizing the joint research efforts between Philips Research and MIT. Through the MIT-Philips alliance, I have also had great opportunities to work with Ozlem Uzuner and Anna Rumshisky.

The other part of the funding for my Ph.D. study was provided by the scholarship from Kwanjeong Educational Foundation. I would like to thank Chonghwan Lee, the founder of the educational foundation, for supporting me throughout my undergraduate as well as graduate education.

Lastly, I would like to thank my family and friends for supporting me spiritually throughout this thesis work and my life in general.

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

We investigate in this thesis several natural language processing methods to de-identify and extract information from patient notes. In this chapter, we outline the motivations of our work.

1.1 Background and Motivation

In many countries such as the United States, medical professionals are strongly encouraged to adopt electronic health records (EHRs) and may face financial penalties if they fail to do so [32, 128]. The Centers for Medicare & Medicaid Services have paid out more than $30 billion in EHR incentive payments to hospitals and providers who have attested to meaningful use as of March 2015. As a result, EHR datasets are increasingly widely adopted, as shown in Figure 1-1, and medical investigations may greatly benefit from them. One of the key components of EHRs is patient notes: the information they contain can be critical for a medical investigation because much information present in texts cannot be found in the other elements of the EHR.

However, before patient notes can be shared with medical investigators, some types of information, referred to as protected health information (PHI), must be removed in order to preserve patient confidentiality. In this work, we present a new method to de-identify clinical texts.

We also present a new method for information extraction. Once we have text data such

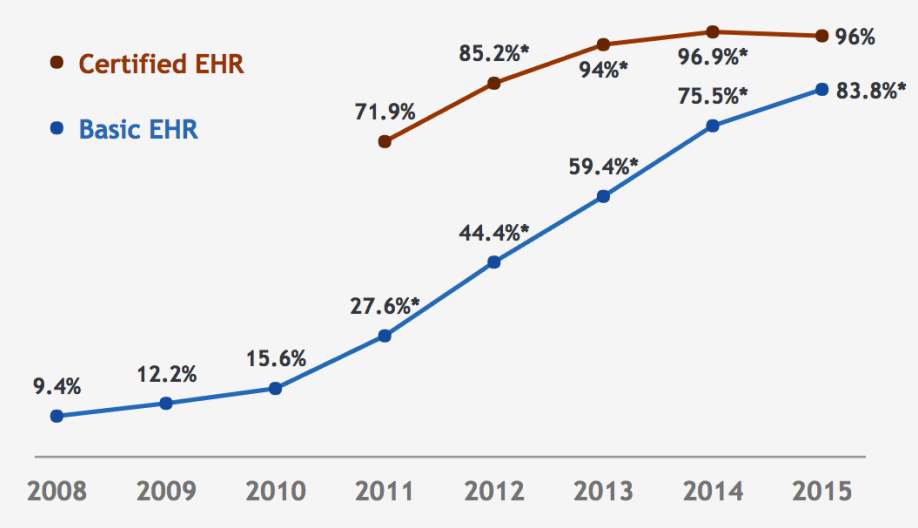


Figure 1-1: Percent of non-federal acute care hospitals with adoption of at least a basic EHR with notes system and possession of a certified EHR. Source: [54]

as de-identified patient notes to use for research purposes, we often need to extract relevant information from text. One of the most fundamental steps in information extraction is named-entity recognition (NER). This can be tackled using the same model as the de-identification, as de-identification is an instance of NER. After extracting named entities, the next possible step in information extraction is to extract relations among the named entities. For example, the patient notes not only contain description of the medical conditions, test, and treatments, but also what the authors, i.e., medical practitioners think how these relate to each other. This may include whether a medical treatment improved or worsened a medical condition, or which test indicates a medical condition. Therefore, it would be useful to extract relations among the entities of interest, which is the focus of the latter part of this thesis.

1.1.1 De-identification

The task of removing PHI from a patient note is referred to as *de-identification*, since the patient cannot be identified once PHI is removed. In the United States, the Health Insurance Portability and Accountability Act (HIPAA) [89] defines 18 different types of PHI, ranging from patient names to phone numbers.

De-identification can be either manual or automated. Manual de-identification means

that the PHI is labeled by human annotators. There are three main shortcomings of this approach. First, only a restricted set of individuals is allowed to access the identified patient notes, thus the task cannot be crowdsourced. Second, humans are prone to mistakes. Neamatullah et al. [87] asked 14 clinicians to detect PHI in approximately 130 patient notes: the results of the manual de-identification varied from clinician to clinician, with recall ranging from 0.63 to 0.94. Third, human annotation is costly. Douglass et al. [35, 33] reported that annotators were paid US$50 per hour and read 20,000 words per hour at best.

As a matter of comparison, the MIMIC dataset [45, 99, 57], which contains data from 50,000 intensive care unit (ICU) stays, consists of 100 million words. This would require 5,000 hours of annotation, which would cost US$250,000 at the same pay rate. Given the annotators' spotty performance, each patient note would have to be annotated by at least two different annotators: it would therefore cost at least US$500,000 to de-identify the notes in the MIMIC dataset. Consequently, the first and crucial step to explore patient notes is automated de-identification.

1.1.2 Information extraction

Once physician notes can be made available thanks to the existence of efficient de-identification programs, one can then extract medical information, and classify relations between the extracted content.

Concept extraction refers to the task of identifying useful concepts from natural language text. For clinical texts, the concepts in consideration could be medical conditions, treatments, and tests.

Relation extraction can be defined as the task of classifying the relationship between given pairs of concepts. For example, relationships can exist between medical condition and treatment (e.g., "Treatment X is administered for medical condition Y", "Treatment X is not administered because of medical condition Y", "Treatment X improves medical condition Y"), between medical condition and test (e.g., "Test X reveals medical condition Y", "Test X was conducted to investigate medical condition Y"), and between medical conditions (e.g., "Medical condition X indicates the presence of medical condition Y").

As mentioned in the previous section, much information is present in physician notes. Leveraging concept and relation extraction techniques gives medical investigators access to knowledge not present in the structured fields of EHRs. We give two specific examples in the rest of this section. Concept extraction can be used for patient cohort identification [121], which can, among other downstream applications, help investigators identify individuals who may be eligible for clinical trials. It can also help conduct retrospective studies to identify clinically significant patterns at much lower cost than prospective trials [62], although retrospective studies do not have the same force of certitude as carefully planned Randomized Controlled trials (RCTs). Extracted relations can also be used for a variety of tasks such as the development of medical ontologies, question-answering systems, and clinical trials [43].

1.2 Contributions

Existing de-identification systems are either rule-based systems or machine learning systems based on manual features. The performance of these methods largely depends on the quality of rules or features developed by human experts. However, quality rules or features are challenging and time-consuming to develop, and do not scale well for new datasets.

Meanwhile, recent approaches to natural language processing (NLP) based on artificial neural networks (ANNs) do not require handcrafted rules or features, as they can automatically learn effective features. ANNs have shown promising results for various tasks in NLP, such as language modeling [79], text classification [106, 59, 15, 67], question answering [127, 125], and machine translation [6, 117, 113].

Inspired by the performance of ANNs for various other NLP tasks, we introduce a de-identification and information extraction system based on ANNs that do not require manual features, and also explore various extensions. The contributions of this thesis are as follows.

- We propose the first ANN architecture for de-identification, which achieves state-of-the-art results without using any manual features. This work was published at the Journal of American Medical Informatics Association (JAMIA) [31].
- We extend the ANN architecture for de-identification by adding the capability to

utilize features. We show that incorporating high-quality features derived from electronic health records such as dictionaries of patient and doctor names further improves de-identification performance. This work was published at the ClinicalNLP workshop at COLING 2016 [70].

- We explore transfer learning of the ANN architecture for de-identification, by transferring the parameters learned from a large annotated dataset to smaller datasets with a limited number of annotations. This work is under review [69].
- We publicly release the ANN de-identification program as an easy-to-use software package for general purpose named-entity recognition as well as de-identification. This work is under review [30].
- We develop an ANN architecture for relation extraction, which ranked first in the SemEval-2017 task 10 (ScienceIE) [5] for relation extraction in scientific articles (subtask C). This work was published at SemEval 2017 [68].

1.3 Organization

The rest of this thesis is organized as follows:

- Chapter 2 presents our work on de-identification of patient notes with ANNs.
- Chapter 3 presents our work on feature-augmented ANNs for de-identification.
- Chapter 4 presents our work on transfer learning.
- Chapter 5 presents our publicly-released software package for named-entity recognition and de-identification.
- Chapter 6 presents our work on relation extraction.
- Chapter 7 presents our conclusions and outlines fruitful problems for future investigation.

Chapter 2

De-identification of patient notes with recurrent neural networks

As mentioned in the previous chapter, de-identification is a prerequisite for medical research using patient notes. In this chapter, we propose the first ANN architecture for de-identification. Our model achieves state-of-the-art results on two different datasets without using any manual features. This work was published at JAMIA [31].

2.1 Introduction and Related work

The task of removing protected health information (PHI) from a patient note is referred to as *de-identification*, since the patient cannot be identified once PHI is removed. In the United States, the Health Insurance Portability and Accountability Act (HIPAA) [89] defines 18 different types of PHI, ranging from patient names to phone numbers. Table 2.1 presents the exhaustive list of PHI types as defined by HIPAA.

Automated de-identification systems can be classified into two categories: rule-based systems and machine-learning-based systems. Rule-based systems typically rely on patterns, expressed as regular expressions and gazetteers, defined and tuned by humans. They do not require any labeled data (aside from labels required for evaluating the system), and are easy to implement, interpret, maintain, and improve, which explains their large presence in the industry [22]. However, they need to be meticulously fine-tuned for each new

PHI categories	PHI types	HIPAA	i2b2	MIMIC
AGE	Ages ≥ 90	x	x	x
	Ages < 90		x	
CONTACT	Telephone and fax numbers	x	x	x
	Electronic mail addresses	x	x	x
	URLs or IP addresses*	x	x	x
DATE	Dates (month and day parts)	x	x	x
	Year		x	x
	Holidays		x	x
	Day of the week		x	
ID	Social security numbers	x	x	x
	Medical record numbers	x	x	x
	Account numbers	x	x	x
	Certificate or license numbers	x	x	x
	Vehicle or device identifiers	x	x	x
	Biometric identifiers or full face photographic images*	x	x	x
LOCATION	Addresses and their components smaller than a state	x	x	x
	State		x	x
	Country		x	x
	Employers	x	x	x
	Hospital name		x	x
	Ward name			x
NAME	Names of patients and family members	x	x	x
	Provider name		x	x
PROFESSION	Profession		x	

Table 2.1: PHI types as defined by HIPAA, i2b2, and MIMIC. PHI categories are defined in the i2b2 dataset. The PHI types marked with * do not appear in either dataset.

dataset, are not robust to language changes (e.g., variations in word forms, typographical errors, or infrequently used abbreviations), and cannot easily take into account the context (e.g., "Mr. Parkinson" is PHI, while "Parkinson's disease" is not PHI). Rule-based systems are described in [11, 7, 37, 42, 50, 84, 87, 98, 115, 118].

To alleviate some downsides of the rule-based systems, there have been many attempts to use supervised machine learning algorithms to de-identify patient notes by training a classifier to label each word as PHI or not PHI, sometimes distinguishing between different PHI types. Common statistical methods include decision trees [116], log-linear models, support vector machines [48, 123, 51], and conditional random fields [2], the latter being employed in most of the state-of-the-art systems. For a thorough review of existing sys-

tems, see [77, 110]. All these methods share two downsides: they require a decent sized labeled dataset and much feature engineering. As with rules, quality features are challenging and time-consuming to develop.

Recent approaches to natural language processing based on artificial neural networks (ANNs) do not require handcrafted rules or features, as they can automatically learn effective features by performing composition over tokens which are represented as vectors, often called token embeddings. The token embeddings are jointly learned with the other parameters of the ANN. They can be initialized randomly, but can be pre-trained using large unlabeled datasets typically based on token co-occurrences [80, 24, 95]. The latter often performs better, since the pre-trained token embeddings explicitly encode many linguistic regularities and patterns. As a result, methods based on ANNs have shown promising results for various tasks in natural language processing, such as language modeling [79], text classification [106, 59, 15, 67], question answering [127, 125], machine translation [6, 117, 113], as well as named-entity recognition [24, 64, 63]. A few methods also use vector representations of characters as inputs in order to either replace or augment token embeddings [60, 64, 63].

Inspired by the performance of ANNs for various other NLP tasks, this article introduces the first de-identification system based on ANNs. Unlike other machine learning based systems, ANNs do not require manually-curated features, such as those based on regular expressions and gazetteers. We show that ANNs achieve state-of-the-art results on de-identification of two different datasets for patient notes, the i2b2 2014 challenge dataset and the MIMIC dataset.

2.2 Methods and materials

We first present a de-identifier we developed based on a conditional random field (CRF) model in Section 2.2.1. This de-identifier yields state-of-the-art results on the i2b2 2014 dataset, which is the reference dataset for comparing de-identification systems. This system will be used as a challenging baseline for the ANN model that we will present in Section 2.2.2. The ANN model outperforms the CRF model, as outlined in Section 2.3.

2.2.1 CRF model

In the CRF model, each patient note is tokenized and features are extracted for each token. During the training phase, the CRF's parameters are optimized to maximize the likelihood of the gold standard labels. During the test phase, the CRF predicts the labels. The performance of a CRF model depends mostly on the quality of its features. We used a combination of n-gram, morphological, orthographic, and gazetteer features. These are similar to features used in the best-performing CRF-based competitors in the i2b2 challenge [130, 73]. In order to effectively incorporate context when predicting a label, the features for a given token are computed based on that token and on the four surrounding tokens.

2.2.2 ANN model

The main components of the ANN model are recurrent neural networks (RNNs). In particular, we use a type of RNN called Long Short Term Memory (LSTM) [55], as discussed in Section 2.2.2.

The system is composed of three layers:

- Character-enhanced token embedding layer (Section 2.2.2),
- Label prediction layer (Section 2.2.2),
- Label sequence optimization layer (Section 2.2.2).

The character-enhanced token embedding layer maps each token into a vector representation. The sequence of vector representations corresponding to a sequence of tokens are input to the label prediction layer, which outputs the sequence of vectors containing the probability of each label for each corresponding token. Lastly, the sequence optimization layer outputs the most likely sequence of predicted labels based on the sequence of probability vectors from the previous layer. All layers are learned jointly. Figure 2-1 shows the ANN architecture.

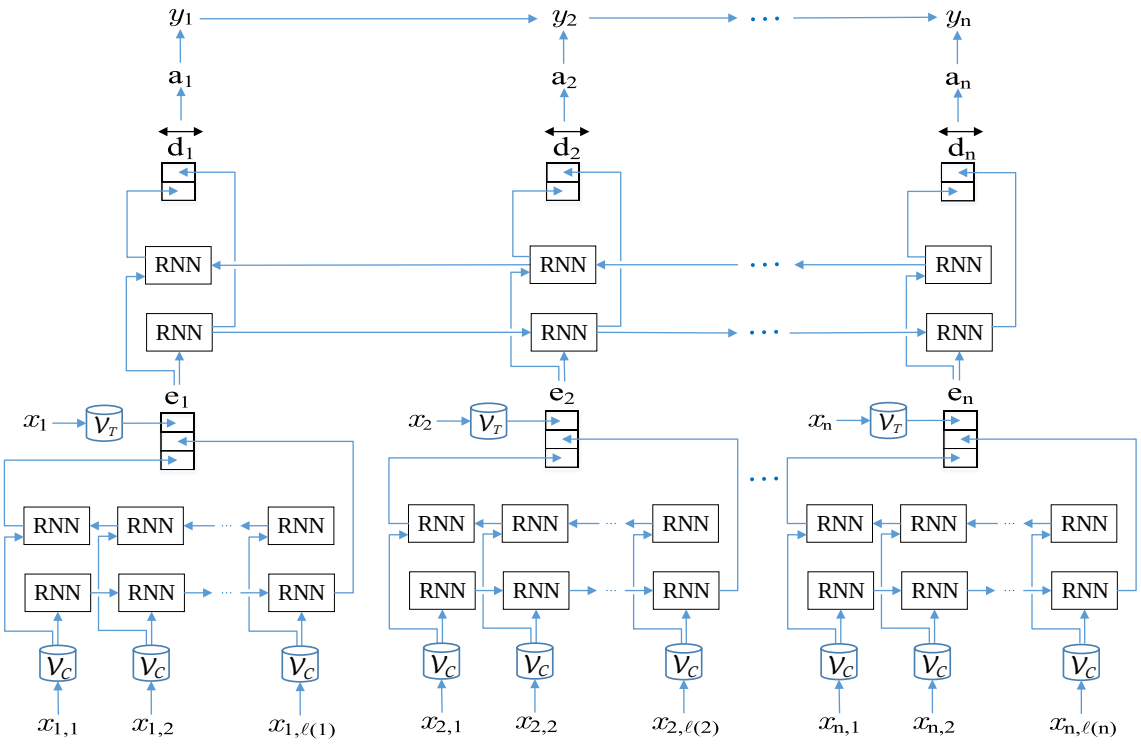


Figure 2-1: Architecture of the artificial neural network (ANN) model. RNN stands for recurrent neural network. The type of RNN used in this model is Long Short Term Memory (LSTM). n is the number of tokens, and x_i is the i^{th} token. $\mathcal{V}_T$ is the mapping from tokens to token embeddings. $\ell(i)$ is the number of characters and $x_{i,j}$ is the j^{th} character in the i^{th} token. $\mathcal{V}_C$ is the mapping from characters to character embeddings. $\mathbf{e}_i$ is the character-enhanced token embeddings of the i^{th} token. $\overleftrightarrow{\mathbf{d}}_i$ is the output of the LSTM of label prediction layer, $\mathbf{a}_i$ is the probability vector over labels, y_i is the predicted label of the i^{th} token.

In the following, we denote scalars in italic lowercase (e.g., k, b_f), vectors in bold lowercase (e.g., $\mathbf{s}$, $\mathbf{x}_i$), and matrices in italic uppercase (e.g., W_f) symbols. We use the colon notations $x_{i:j}$ and $\mathbf{v}_{i:j}$ to denote the sequence of scalars $(x_i, \ldots, x_j)$, and vectors $(\mathbf{v}_i, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_j)$, respectively.

Bidirectional LSTM

An RNN is a neural network architecture designed to handle input sequences of variable sizes, but it fails to model long term dependencies. An LSTM is a type of RNN that mitigates this issue by keeping a memory cell that serves as a summary of the preceding elements of an input sequence. More specifically, given a sequence of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, at each step $t = 1, \ldots, n$, an LSTM takes as input $\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}$ and produces the hidden state $\mathbf{h}_t$ and the memory cell $\mathbf{c}_t$ based on the following formulas:

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(W_i\,[\mathbf{x}_t;\ \mathbf{h}_{t-1};\ \mathbf{c}_{t-1}] + \mathbf{b}_i) \\
\tilde{\mathbf{c}}_t &= \tanh(W_c\,[\mathbf{x}_t;\ \mathbf{h}_{t-1}] + \mathbf{b}_c)] \\
\mathbf{c}_t &= (1 - \mathbf{i}_t) \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \\
\mathbf{o}_t &= \sigma(W_o\,[\mathbf{x}_t;\ \mathbf{h}_{t-1};\ \mathbf{c}_t] + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}
$$

where W_i, W_c, W_o are weight matrices and $\mathbf{b}_i, \mathbf{b}_c, \mathbf{b}_o$ are bias vectors used in the input gate, memory cell, and output gate calculations, respectively. The symbols $\sigma(\cdot)$ and $\tanh(\cdot)$ refer to the element-wise sigmoid and hyperbolic tangent functions, and $\odot$ is the element-wise multiplication. $\mathbf{h}_0 = \mathbf{c}_0 = \mathbf{0}$.

A bidirectional LSTM consists of a forward LSTM and a backward LSTM, where the forward LSTM calculates the forward hidden states $(\overrightarrow{\mathbf{h}}_1, \overrightarrow{\mathbf{h}}_2, \ldots, \overrightarrow{\mathbf{h}}_n)$, and the backward LSTM calculates the backward hidden states $(\overleftarrow{\mathbf{h}}_1, \overleftarrow{\mathbf{h}}_2, \ldots, \overleftarrow{\mathbf{h}}_n)$ by feeding the input sequence in the backward order, from $\mathbf{x}_n$ to $\mathbf{x}_1$.

Depending on the application of the LSTM, one might need an output sequence corresponding to each element in the sequence, or a single output that summarizes the whole sequence. In the former case, the output sequence $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n$ of the LSTM is obtained by concatenating the hidden states of the forward and the backward LSTMs for each element i.e., $\overleftrightarrow{\mathbf{h}}_t = (\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t)$ for $t = 1, \ldots, n$. In the latter case, the output is obtained by concatenating the last hidden states of the forward and the backward LSTMs i.e., $\overleftrightarrow{\mathbf{h}} = (\overrightarrow{\mathbf{h}}_n; \overleftarrow{\mathbf{h}}_n)$.

Character-enhanced token embedding layer

The character-enhanced token embedding layer takes a token as input and outputs its vector representation. The latter results from the concatenation of two different types of embeddings: the first one directly maps a token to a vector, while the second one comes from the output of a character-level token encoder.

The direct mapping $\mathcal{V}_T(\cdot)$ from token to vector, often called a token (or word) embedding, can be pre-trained on large unlabeled datasets using programs such as word2vec [80,

78, 81] or GloVe [95], and can be learned jointly with the rest of the model. Token embeddings, often learned by sampling token co-occurrence distributions, have desirable properties such as locating semantically similar words closely in the vector space, hence leading to state-of-the-art performance for various tasks.

While the token embeddings capture the semantics of tokens to some degree, they may still suffer from data sparsity. For example, they cannot account for out-of-vocabulary tokens, misspellings, and different noun forms or verb endings. One solution to remediate some of these issues would be to lemmatize tokens before training, but this approach may fail to retain some useful information such as the distinction between some verb and noun forms.

We address this issue by using character-based token embeddings, which incorporate each individual character of a token to generate its vector representation. This approach enables the model to learn sub-token patterns such as morphemes (e.g., suffix or prefix) and roots, thereby capturing out-of-vocabulary tokens, different surface forms, and other information not contained in the token embeddings.

Let $x_{i,1}, \ldots, x_{i,\ell(i)}$ be the sequence of characters that comprise the i^{th} token x_i, where $\ell(i)$ is the number of characters in x_i. The character-level token encoder generates the character-based token embedding of x_i by first mapping each character $x_{i,j}$ to a vector $\mathcal{V}_C(x_{i,j})$, called a character embedding, via the mapping $\mathcal{V}_C(\cdot)$. Then the sequence of vectors $\mathcal{V}_C(x_{i,1}), \ldots, \mathcal{V}_C(x_{i,\ell(i)})$ is passed to a bidirectional LSTM, which outputs the character-based token embedding $\overleftrightarrow{\mathbf{b}}_i$

As a result, the final output $\mathbf{e}_i$ of the character-enhanced token embedding layer for i^{th} token x_i is the concatenation of the token embedding $\mathcal{V}_T(x_i)$ and the character-based token embedding $\overleftrightarrow{\mathbf{b}}_i$. In summary, when the character-enhanced token embedding layer receives a sequence of tokens $x_{1:n}$ as input, it will output the sequence of token embeddings $\mathbf{e}_{1:n}$.

Label prediction layer

The label prediction layer takes as input the sequence of vectors $\mathbf{e}_{1:n}$, i.e., the outputs of the character-enhanced token embedding layer, and outputs $\mathbf{a}_{1:n}$, where the t^{th} element of $\mathbf{a}_n$ is the probability that the n^{th} token has the label t. The labels are either one of the PHI

types or non-PHI. For example, if one aims to predict all 18 HIPAA-defined PHI types, there would be 19 different labels.

The label prediction layer contains a bidirectional LSTM that takes the input sequence $\mathbf{e}_{1:n}$ and generates the corresponding output sequence $\overleftrightarrow{\mathbf{d}}_{1:n}$. Each output $\overleftrightarrow{\mathbf{d}}_i$ of the LSTM is given to a feed-forward neural network with one hidden layer, which outputs the corresponding probability vector $\mathbf{a}_i$.

Label sequence optimization layer

The label sequence optimization layer takes the sequence of probability vectors $\mathbf{a}_{1:n}$ from the label prediction layer as input, and outputs a sequence of labels $y_{1:n}$, where y_i is the label assigned to the token x_i.

The simplest strategy to select the label y_i would be to choose the label that has the highest probability in $\mathbf{a}_i$, i.e. $y_i = \arg\max_k \mathbf{a}_i[k]$. However, this greedy approach fails to take into account the dependencies between subsequent labels. For example, it may be more likely to have a token with the PHI type STATE followed by a token with the PHI type ZIP than any other PHI type. Even though the label prediction layer has the capacity to capture such dependencies to a certain degree, it may be preferable to allow the model to directly learn these dependencies in the last layer of the model.

One way to model such dependencies is to incorporate a matrix T that contains the transition probabilities between two subsequent labels. $T[i, j]$ is the probability that a token with label i is followed by a token with the label j. The score of a label sequence $y_{1:n}$ is defined as the sum of the probabilities of individual labels and the transition probabilities:

$$s(y_{1:n}) = \sum_{i=1}^{n} \mathbf{a}_i[y_i] + \sum_{i=2}^{n} T[y_{i-1}, y_i].$$

These scores can be turned into probabilities of the label sequences by taking a softmax function over all possible label sequences. During the training phase, the objective is to maximize the log probability of the gold label sequence. In the testing phase, given an input sequence of tokens, the corresponding sequence of predicted labels is chosen as the one that maximizes the score.

2.3 Experiments and results

2.3.1 Datasets

We evaluate our two models on two datasets: the i2b2 2014 and MIMIC de-identification datasets. The i2b2 2014 dataset was released as part of the 2014 i2b2/UTHealth shared task Track 1 [110]. It is the largest publicly available dataset for de-identification. Ten teams participated in this shared task, and 22 systems were submitted. As a result, we used the i2b2 2014 dataset to compare our models against state-of-the-art systems.

The MIMIC de-identification dataset was created for this work as follows. The MIMIC-III dataset [57, 45, 99] contains data for 61,532 ICU stays over 58,976 hospital admissions for 46,520 patients, including 2 million patient notes. In order to make the notes publicly available, a rule-based de-identification system [34, 35, 33] was written for the specific purpose of de-identifying patient notes in MIMIC, leveraging dataset-specific information such as the list of patient names or addresses. The system favors recall over precision: there are virtually no false negatives, while there are numerous false positives. To create the gold standard MIMIC de-identification dataset, we selected 1,635 discharge summaries, each belonging to a different patient, containing a total of 60.7k PHI instances. We then annotated the PHI instances detected by the rule-based system as true positives or false positives. We found that 15% of the PHI instances detected by the rule-based system were false positives.

Table 2.1 introduces the PHI types and Table 2.2 presents the datasets' sizes. For the test set, we used the official test set for the i2b2 dataset, which is 40% of the dataset; we randomly selected 20% of the MIMIC dataset as the test set for this dataset.

	i2b2	**MIMIC**
Vocabulary size	46,803	69,525
Number of notes	1,304	1,635
Number of tokens	984,723	2,945,228
Number of PHIs	28,867	60,725
Number of PHI tokens	41,355	78,633

Table 2.2: Overview of the i2b2 and MIMIC datasets.

2.3.2 Evaluation metrics

To assess the performance of the two models, we computed the precision, recall, and F1-score. Let TP be the number of true positives, FP the number of false positives, and FN the number of false negatives. Precision, recall, and F1-score are defined as follows: precision $= \frac{TP}{TP+FP}$, recall $= \frac{TP}{TP+FN}$, and F1-score $= \frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$. Intuitively, precision is the proportion of the predicted PHI labels that are gold labels, recall is the proportion of the gold PHI labels that are correctly predicted, and F1-score is the harmonic mean of precision and recall.

2.3.3 Training and hyperparameters

The model is trained using stochastic gradient descent, updating all parameters, i.e., token embeddings, character embeddings, parameters of bidirectional LSTMs, and transition probabilities, at each gradient step. For regularization, dropout is applied to the character-enhanced token embeddings before the label prediction layer. Below are the choices of hyperparameters and token embeddings, optimized using a subset of the training set:

- character embedding dimension: 25
- character-based token embedding LSTM dimension: 25
- token embedding dimension: 100
- label prediction LSTM dimension: 100
- dropout probability: 0.5

We selected the hyperparameters manually, though we could have used some hyperparameter optimization techniques [10, 27, 28]. We tried pre-training token embeddings on the i2b2 2014 dataset and the MIMIC dataset[1] using word2vec and GloVe. Both word2vec and GloVe were trained using a window size of 10, a minimum vocabulary count of 5, and 15 iterations. Additional parameters of word2vec were the negative sampling and the model type, which were set to 10 and skip-gram, respectively. We also experimented with the publicly available[2] token embeddings such as GloVe trained on Wikipedia and Giga-

[1] For MIMIC, we used the entire dataset containing 2 million notes and 800 million tokens.
[2] http://nlp.stanford.edu/projects/glove/

word 5 [92]. The results were quite robust to the choice of the pre-trained token embeddings. The GloVe embeddings trained on Wikipedia articles yielded slightly better results, and we chose them for the rest of this work.

2.3.4 Results

All results were computed using the official evaluation script from the i2b2 2014 de-identification challenge. Table 2.3 presents the main results, based on binary token-based precision, recall, and F1-score for HIPAA-defined PHI only. These PHI types are the most important since only those are required to be removed by law. On the i2b2 dataset, our ANN model has a higher F1-score and recall than our CRF model as well as the best system from the i2b2 2014 de-identification challenge, which was the Nottingham system [130]. The only freely available, off-the-shelf program for de-identification, called the MITRE Identification Scrubber Toolkit (MIST) [2], performed poorly. Combining the outputs of our ANN and CRF models, by considering a token to be PHI if it is identified as such by either model, further increases the performance in terms of F1-score and recall.

It should be noted that the Nottingham system was specifically fine-tuned for the i2b2 dataset as well as the i2b2 evaluation script. For example, the Nottingham system post-processes the detected PHI terms in order to match the offset of the gold PHI tokens, such as modifying “MR:6746781" to “6746782" and “MWFS" to “M", “W", “F", “S".

On the MIMIC dataset, our ANN model also has a higher F1-score and recall than our CRF model. Interestingly, combining the outputs of our ANN and CRF models did not increase the F1-score, because precision was negatively impacted. However, the recall did benefit from combining the two models. MIST was much more competitive on this dataset.

We calculated the statistical significance of the differences in precision, recall, and F1-score between the CRF and ANN models using approximate randomization with 9999 shuffles. The significance levels of the differences in precision, recall, and F1-score are 0.37, 0.02, 0.22 for the i2b2 dataset, and 0.08, 0.00, 0.00 for the MIMIC dataset, respectively.

Model	i2b2			MIMIC		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Nottingham	**99.000**	96.680	97.680	-	-	-
MIST	95.288	75.691	84.367	97.739	97.164	97.450
CRF	98.560	96.528	97.533	99.060	98.987	99.023
ANN	98.320	97.380	97.848	**99.208**	99.251	**99.229**
CRF + ANN	97.920	**97.835**	**97.877**	98.820	**99.398**	99.108

Table 2.3: Performance (%) on the PHI as defined in the HIPAA. We evaluated the systems based on the detection of PHI tokens versus non-PHI tokens (i.e., binary HIPAA token-based evaluation). The best performance for each metric on each dataset is highlighted in bold. Nottingham is the best performing system from the 2014 i2b2/UTHealth shared task Track 1. MIST, the MITRE Identification Scrubber Toolkit, is a freely available de-identification program. CRF is the model based on Conditional Random Field, ANN is the model based on Artificial Neural Network, and CRF+ANN is the result obtained by combining the outputs of the CRF model and the ANN model. The Nottingham system could not be run on the MIMIC dataset, as it is not publicly available.

2.3.5 Error analysis

Figure 2-2 shows the binary token-based F1-scores for each PHI category. The ANN model outperforms the CRF model on all categories for both datasets, with the exception of the ID category (which mostly contains medical record numbers) in the i2b2 dataset. This is due to the fact that the CRF model uses sophisticated regular expression features that are tailored to detect ID patterns such as "38:Z8912708G".

Another interesting difference between the ANN and the CRF results is the PROFESSION category: the ANN significantly outperforms the CRF. The reason behind this result is that the embeddings of the tokens that represent a profession tend to be close in the token embedding space, which allows the ANN model to generalize well. We tried assembling various gazetteers for the PROFESSION category, but all of them were performing significantly worse than the ANN model.

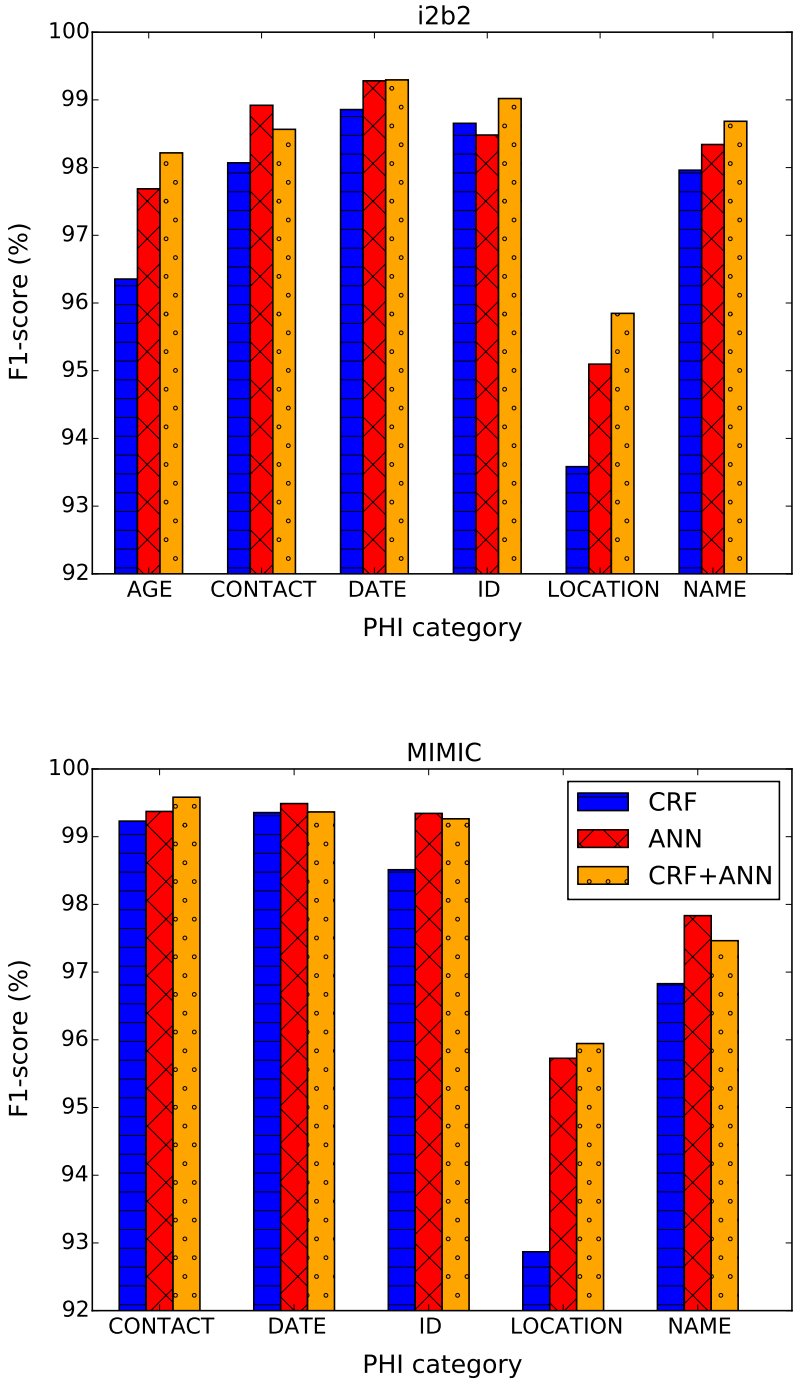


Figure 2-2: Binary token-based F1-scores for each PHI category. The evaluation is based on PHI types that are defined by HIPAA as well as additional PHI types specific to each dataset. Each PHI category and the corresponding PHI types are defined in Table 1. The PROFESSION category exists only in the i2b2 dataset, and was removed from the graph to avoid distorting the y-axis: the F1-scores are 72.014, 82.035, and 81.664 with the CRF, ANN, and CRF+ANN, respectively. For the same reason, the AGE category in MIMIC was removed: the F1-scores are 80.851, 81.481, and 92.308 with the CRF, ANN, and CRF+ANN, respectively.

Table 2.4 presents some examples of gold PHI instances correctly predicted by the ANN model that the CRF model failed to predict, and conversely. This illustrates that the ANN model efficiently copes with the diversity of the contexts in which tokens appear, whereas the CRF model can only address the contexts that are manually encoded as features. In other words, the ANN model's intrinsic flexibility allows it to better capture the variance in human languages than the CRF model. For example, it would be challenging and time-consuming to engineer features for all possible contexts such as "had a stroke at 80", "quit smoking in 08", "on the 29th of this month", and "his friend Epstein". The ANN model is also very robust to variations in surface forms, such as misspellings (e.g., "in teh late 60s", "Khazakhstani", "01/19/:0"), tokenizations (e.g., "Results02/20/2087", "MC # 0937884Date"), and different phrases referring to the same semantic meaning (e.g., "San Rafael Mount Hospital", "Rafael Mount", "Rafael Hospital"). Furthermore, the ANN model is able to detect many PHI instances despite not having explicit gazetteers, as examples in the LOCATION and PROFESSION categories illustrate. We conjecture that the character-enhanced token embeddings contain rich enough information to effectively function as gazetteers, as tokens with similar semantics are closely located in the vector representation [80, 24, 60].

On the other hand, CRF is good at rarely occurring patterns that are written in highly specialized regular expression patterns (e.g., "38:Z8912708G", "53RHM") or tokens that are included in the gazetteers (e.g., "Christmas", "WPH", "rosenberg", "Motor Vehicle Body Repairer"). For example, the PHI token "Christmas" only occurs in the test set, and unless the context gives a strong indication, the ANN model cannot detect it, whereas the CRF model could, as long as it is included in the gazetteers.

PHI category	ANN	CRF
AGE	had a stroke at **80** and died of ?another stroke at age 83. PERSONAL DATA AND OVERALL HEALTH: Now **63**, FH: Father: Died @ **52** from EtOH abuse Tobacco: smoked from age 7 to **15**, has not smoked since 15. History of Present Illness **86**F reports worsening b/l leg pain.	HPI: **53**RHM who going to bed Wednesday was in usoh, but Tobacco: Quit at **38** y/o; ETOH: 1-2 beers/week; Caffeine:
CONTACT	by phone, Dr. Ivan Guy. Call w/ questions **86383**. H/O paroxysmal afib VNA **171-311-7974**	
DATE	During his **May** hospitalization he had dysphagia Social history: divorced, quit smoking in **08**, sober x 10 yrs, She is to see him on the **29th** of this month at 1:00 p.m. He did have a renal biopsy in teh late **60s** adn thus will Results**02/20/2087** NA 135, K 3.2 (L), CL 96 (L), CO2 30.6, DD: 01/18/20 DT: **01/19/:0** DV: 01/18/20	She is looking forward to a good **Christmas**. She is here today
ID	3/23 for bradycardia. P/G model # **5435**, serial # 4712198, NotePt: Ulysses Ogrady MC # **0937884**Date: 10/07/69	DD:05/05/2095 DT:05/05/2095 **WK:65255 :4653** NO GROWTH TO DATE Specimen: **38:Z8912708G** Collected
LOCATION	Works in programming at **Audiovox**. Formerly at BrightPoint. He has remote travel hx to the **Rockefeller Centre**, History of Present Illness: Pt is a 59 yo **Khazakhstani** male, who was admitted to **San Rafael Mount Hospital** following nauseas and was brought to **Rafael Mount** ED. Anemia: On admission to **Rafael Hospital**,	2nd set biomarkers (**WPH**): Creatine Kinase Isoenzymes Hospitalized 2115 **TCH** for ROMI 2120 TCH new onset
NAME	ATCH: 655-75-45 Dear Harry and **Yair**: My thanks for Patient lives in Flint with his friend **Epstein**. Health care proxy-Yes, son (**West**) Allergies DUTASTERIDE	Lab Tests **Amador**: the lab results show good levels of 10MG PO qd : 05/10/2066 - 04/15/2068 ACT : **rosenberg** 128 Williams Ct M **OSCAR, JOHNNY** Hyderabad, WI 62297
PROFESSION	Social history: Married, **glazier**, 3 grown adult children Has VNA. Former civil engineer, **supervisor**, consultant. He was formerly self-employed as a **CPA** and would often Communications senior manager, **marketing**, worked for and Concrete Finisher (25yrs). He is a **veteran**. Former tobacco user, works part time in **securities**.	Social history: He is retried **Motor Vehicle Body Repairer**.

Table 2.4: Examples of correctly detected PHI instances (in bold) by the ANN and CRF models for the i2b2 dataset. The examples in the ANN column are only predicted by the ANN model and not predicted by the CRF model, and conversely. Typographical errors are from the original text.

2.3.6 Effect of training set size

Figure 2-3 shows the impact of the training set size on the performance of the models on the MIMIC dataset. When the training set size is very limited, the CRF performs slightly better than the ANN model, since the CRF model can leverage handcrafted features without much training data. As the training set size increases, the ANN model starts to significantly outperform the CRF model, since the parameters including the embeddings are automatically fine-tuned with more data, and therefore the features learned by the ANN model become increasingly more refined than the manually handcrafted features. As a result, combining the outputs of the CRF and ANN models increases the F1-score over the ANN model only for small training set size and yields a less competitive F1-score than the ANN model for bigger training set size.

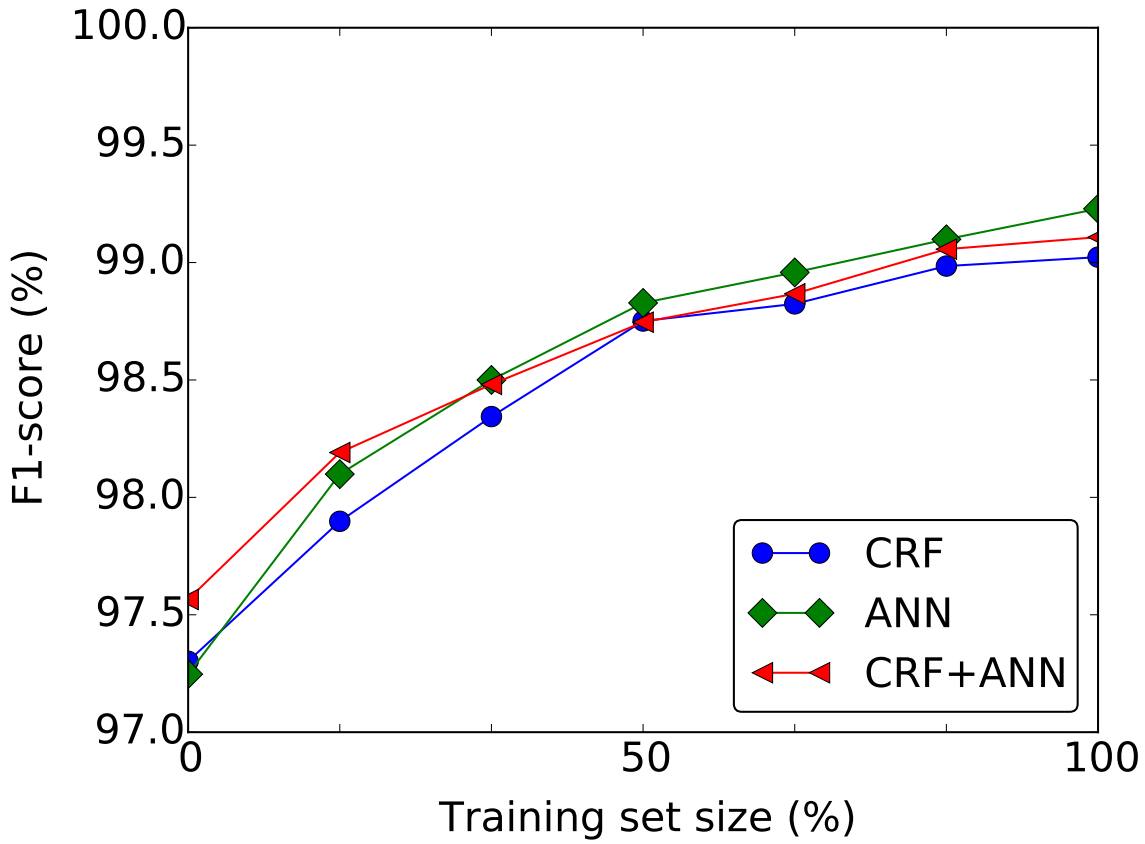


Figure 2-3: Impact of the training set size on the binary HIPAA token-based F1-scores on the MIMIC dataset. 100% training set size refers to using all of the dataset minus the test set.

2.3.7 Ablation analysis

In order to quantify the importance of various elements of the ANN model, we tried 4 variations of the model, eliminating different elements one at a time. Figure 2-4 presents the results of the ablation tests. Removing either the label sequence optimization layer, pre-trained token embeddings, or token embeddings slightly decreased the performance. Surprisingly, the ANN performed pretty well with only character embeddings and without the token embeddings, and eliminating the character embeddings was more detrimental than eliminating the token embeddings. This suggests that the character-based token embeddings may be capturing not only the sub-token level features, but also the semantics of the tokens themselves.

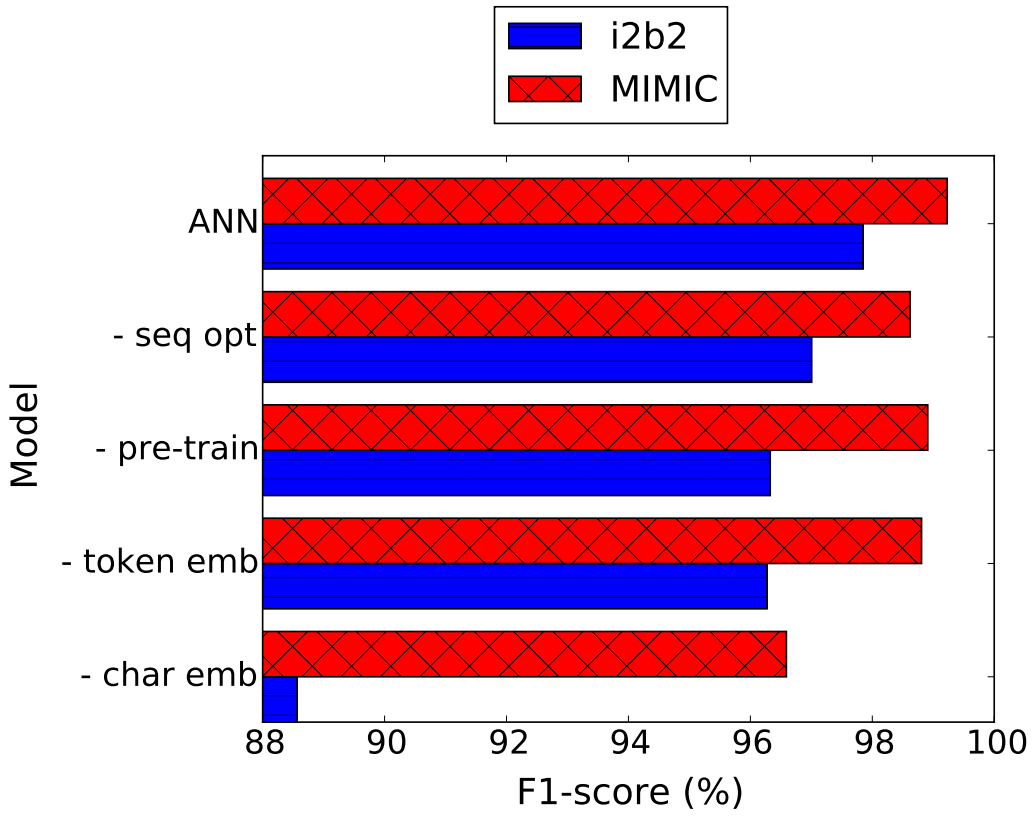


Figure 2-4: Ablation test performance based on binary HIPAA token-based evaluation. ANN is the model based on Artificial Neural Network. "- seq opt" is the ANN model without the label sequence optimization layer. "- pre-train" is the ANN model where token embeddings are initialized with random values instead of pre-trained embeddings. "- token emb" is the ANN model using only character-based token embeddings, without token embeddings. "- character emb" is the ANN model using only token embeddings, without character-based token embeddings.

2.4 Conclusions

We proposed the first system based on ANN for patient note de-identification. It outperforms state-of-the-art systems based on CRF on two datasets, while requiring no hand-crafted features. Utilizing both the token and character embeddings, the system can automatically learn effective features from data by fine-tuning the parameters. It jointly learns the parameters for the embeddings, the bidirectional LSTMs as well as the label sequence optimization, and can make use of token embeddings pre-trained on large unlabeled datasets. Quantitative and qualitative analysis of the ANN and CRF models indicates that the ANN model better incorporates context and is more flexible to variations inherent in human languages than the CRF model.

From the viewpoint of deploying an off-the-shelf de-identification system, our results in Table 2.3 demonstrate recall on the MIMIC discharge summaries over 99%, which is quite encouraging. Figure 2-2, however, shows that the F1-score on the NAME category, probably the most sensitive PHI type, falls just below 98% for the ANN model. We anticipate that adding gazetteer features based on the local institution's patient and staff census should improve this result, which will be explored in the next chapter.

Chapter 3

Feature-augmented neural networks for patient note de-identification

In the previous chapter, the first ANN-based de-identification system has been proposed, yielding state-of-the-art results. Unlike other systems, it does not rely on human-engineered features, which allows it to be quickly deployed, but does not leverage knowledge from human experts or from EHRs. In this chapter, we explore a method to incorporate human-engineered features as well as features derived from EHRs to the ANN-based de-identification system. Our results show that the addition of features, especially the EHR-derived features, further improves the state-of-the-art in patient note de-identification, including some of the most sensitive PHI types such as patient names. This work was published at COLING 2016 [70].

3.1 Introduction and related work

A naive approach to de-identification is to manually identify PHI. However, this is costly [35, 33] and unreliable [87]. Consequently, there has been much work developing automated de-identification systems. These systems are either based on rules or machine-learning models. Rule-based systems typically rely on patterns, expressed as regular expressions and gazetteers, defined and tuned by humans [11, 7, 37, 42, 50, 84, 87, 98, 115, 118].

Machine-learning-based systems train a classifier to label each token as PHI or not PHI.

Some systems are more fine-grained by detecting which PHI type a token belongs to. Different statistical methods have been explored for patient note de-identification, including decision trees [116], log-linear models, support vector machines (SVMs) [48, 123, 51], and conditional random field (CRFs) [2]. A thorough review of existing systems can be found in [77, 110].

A more recent system discussed in Chapter 2 has introduced the use of artificial neural networks (ANNs) for de-identification, and obtained state-of-the-art results [31]. The system does not use any manually-curated features. Instead, it solely relies on character and token embeddings. While this allows the system to be developed and deployed faster, it fails to give users the possibility to add features engineered by human experts. Additionally, in practical settings of de-identification, patient notes typically come from a hospital EHR database, which contains metadata such as which patient each note pertains to, and other information such as the names of all doctors who work at the hospital where the patient was treated. The features derived from EHR databases may be useful for boosting the performance of de-identification systems. In this chapter, we present a method to incorporate features to this ANN-based system, and show that it further improves the state-of-the-art.

3.2 Method

The first model based on ANNs for patient note de-identification was introduced in Chapter 2. We build upon this model, which utilizes both token and character embeddings to learn effective features from data by fine-tuning the parameters. The main components of the ANN model are Long Short Term Memories (LSTMs) [55], which are a type of recurrent neural network (RNN).

The model is composed of three layers: a character-enhanced token embedding layer, a label prediction layer, and a label sequence optimization layer. The character-enhanced token embedding layer maps each token into a vector representation. The sequence of vector representations corresponding to a sequence of tokens are input to the label prediction layer, which outputs the sequence of vectors containing the probability of each label for each corresponding token. Lastly, the sequence optimization layer outputs the most likely

sequence of predicted labels based on the sequence of probability vectors from the previous layer. All layers are learned jointly. For more details on the basic ANN model, see [31].

We augment this ANN model by adding features that are human-engineered or derived from our EHR database, as presented in Table 3.1. The majority of human-engineered features are taken from [39], a few more features come from [130], and additional gazetteers are collected using online resources. All features are binary and computed for each token. The binary feature vector comprising all features for a given token is fed into a feedforward neural network, the output vector of which is concatenated to the corresponding token embeddings, at the output of the character-enhanced token embedding layer, as Figure 3-1 illustrates.

Feature types	**Features**	
Note metadata	Patient's first name, patient's last name	EHR features
Hospital data	Doctor's first names, doctor's last names	
Morphological	Ends with s, is the first letter capitalized, contains a digit, is numeric, is alphabetic, is alphanumeric, is title case, is all lower case, is all upper case, is a stop word	
Semantic/Wordnet	Hypernyms, senses, lemma names	
Temporal	Seasons, months, weekdays, times of the day, years, years followed by apostrophe, festivity dates, holidays, cardinal numbers, decades, fuzzy quantifier (e.g., "approximately", "few"), future trigger (e.g., "next", "tomorrow")	
Gazetteers	Honorifics for doctors, honorifics, medical specialists, medical specialties, first names, last names, last name prefixes, street suffixes, US cities, US states (including abbreviations), countries, nationalities, organizations, professions	
Regular expressions	Email, age, date, phone, zip code, id number, medical record number	

Table 3.1: Feature list. Note metadata and hospital data are derived from the EHR database. Morphological, semantic/wordnet, and temporal features are commonly used features for NLP tasks. Gazetteers and regular expressions are specifically engineered for the task.

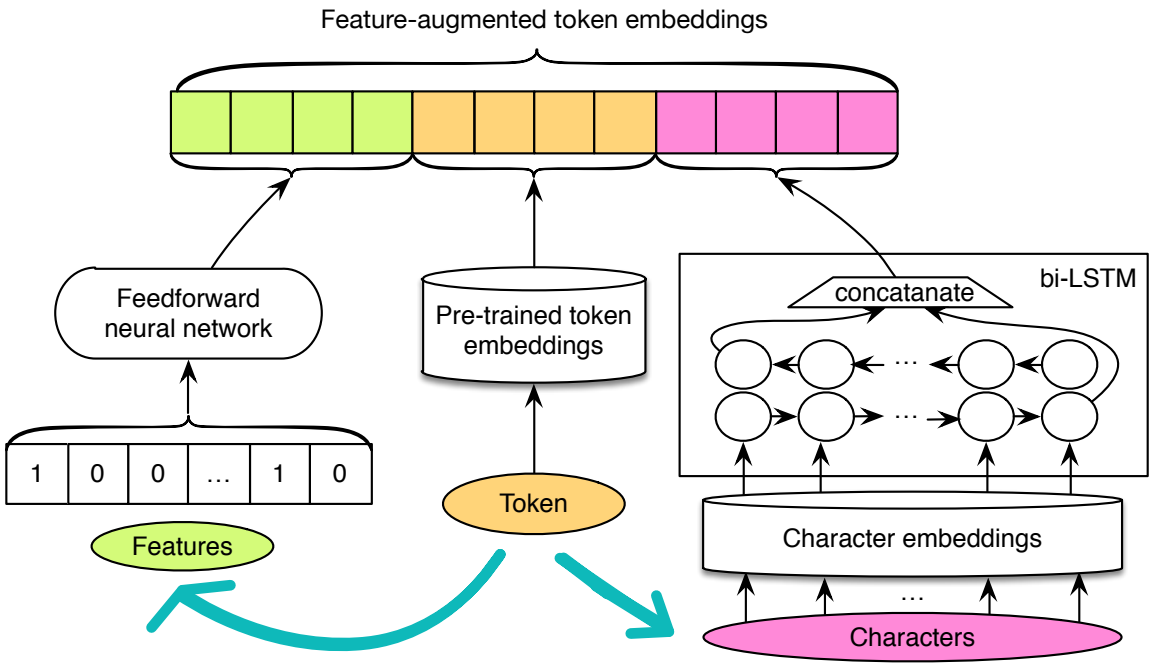


Figure 3-1: Feature-augmented token embeddings. Each token is mapped to a token embedding that is the concatenation of three elements: the output of a feedforward neural network that takes the features as input, a pre-trained token embedding, and the output of a bidirectional-LSTM (bi-LSTM) that takes the character embeddings as input.

3.3 Experiments

We evaluate our model on the de-identification dataset introduced in Chapter 2, which is a subset of the MIMIC-III dataset [45, 99, 57], using the same train/validation/test split (70%/10%/20%). We chose this dataset as each note comes with metadata, such as the patient's name, and it is the largest de-identification dataset available to us. It contains 1,635 discharge summaries, 2,945,228 tokens, 69,525 unique tokens, and 78,633 PHI tokens.

The model is trained using stochastic gradient descent, updating all parameters, i.e., token embeddings, character embeddings, parameters of bidirectional LSTMs as well as feedforward neural networks, and transition probabilities, at each gradient step. For regularization, dropout is applied to the feature-augmented token embeddings before the label prediction layer. We set the character embedding dimension to 25, the character-based token embedding LSTM dimension to 25, the token embedding dimension to 100, the label prediction LSTM dimension to 100, the dropout probability to 0.5, and we use GloVe embeddings [95] trained on Wikipedia and Gigaword 5 [92] articles as pre-trained token

embeddings. For the feedforward network that takes binary features as input, its output dimension is set to equal the input dimension, i.e. the number of binary features used. The hyperparameters were optimized based on the performance on the validation set.

3.4 Results

	Optimized by F1-score			Optimized by recall		
	Precision	Recall	F1-score	Precision	Recall	F1-score
No feature	99.103	99.197	99.150	98.557	99.376	98.965
EHR features	99.100	99.304	99.202	98.771	**99.441**	99.105
All features	**99.213**	**99.306**	**99.259**	**98.880**	99.420	**99.149**

Table 3.2: Binary HIPAA token-based results (%) for the ANN model, averaged over 5 runs. The metric refers to the detection of PHI tokens versus non-PHI tokens, amongst PHI types that are defined by HIPAA. "No feature" is the model utilizing only character and word embeddings, without any feature. "EHR features" uses only 4 features derived from EHR database: patient first name, patient last name, doctor first name, and doctor last name. "All features" makes use of all features, including the EHR features as well as other engineered features listed in Table 3.1. "Optimized by F1-score" and "optimized by recall" means that the epochs for which the results are reported are optimized based on the highest F1-score or the highest recall on the validation set, respectively.

Table 3.2 presents the main results. The epochs for which the results are reported are optimized based on either the highest F1-score or the highest recall on the validation set. As expected, choosing the epoch based on the recall improves the recall on the test set, while lowering the precision. Overall, adding features consistently improves the results.

Table 3.3 details the results for each PHI type. The system using only the EHR features yields the highest recall for 6 out of 12 PHI types. Most importantly, the recall for patient and doctor names are higher when using features than when using no feature: this is expected as the patient name of the note and the doctor names are used as features. In fact, the two remaining false negatives for patient names are annotation errors. For example, in the sentence "The patient responded well to *Natrecor* in the past, but the improvement disappeared soon", the drug name *Natrecor* was incorrectly marked as a patient name by the human annotator. This result is highly remarkable as patient names are the most sensitive information in a patient note [107].

Adding all features often lowers the recall compared to using EHR features only, although the F1-score remains virtually unchanged. This is somewhat surprising, as we

	No feature			EHR features			All features			
	P	R	F1	P	R	F1	P	R	F1	Support
Zip	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	24
Date	98.90	99.77	99.33	98.95	**99.79**	**99.36**	**98.99**	99.69	99.34	20627
Phone	98.31	**99.58**	98.94	**98.98**	99.46	99.22	99.42	99.32	**99.37**	1438
Patient	96.89	98.34	97.61	98.62	99.14	98.88	**99.21**	**99.27**	**99.24**	302
ID	99.57	98.24	98.90	99.31	**98.82**	**99.07**	**99.77**	97.97	98.86	612
Doctor[1]	97.47	98.17	97.82	97.27	**98.48**	97.87	**97.56**	98.20	**97.88**	3676
Location	96.02	95.71	95.86	96.41	**96.49**	96.45	**96.65**	96.32	**96.46**	462
Age ≥ 90	75.12	94.29	83.60	77.04	**95.72**	**85.35**	**78.93**	93.57	84.80	28
Hospital[1]	94.78	95.39	95.08	94.77	**95.52**	95.14	**95.53**	95.50	**95.51**	1259
State[1]	99.36	**94.33**	**96.76**	**99.68**	94.03	96.73	99.39	91.94	95.49	67
Street	96.77	85.25	90.54	**97.63**	85.25	**90.96**	93.91	**86.56**	89.81	61
Country[1]	87.51	85.00	86.11	**89.29**	82.50	85.67	86.87	**95.00**	**90.56**	16
Binary	98.41	99.19	98.80	98.48	**99.27**	98.87	**98.61**	99.15	**98.88**	28572

Table 3.3: Binary token-based results (%). The reported results are optimized by recall, and averaged over 5 runs. The symbol [1] indicates that the PHI type is not required by HIPAA. The PHI type "location" designates any location that is not a street name, zip code, state or country. P stands for precision, R for recall, and F1 for F1-score.

had expected that the features would help since using the same feature set with a CRF to perform de-identification yields results comparable to the state-of-the-art ANN models as discussed in Chapter 2. This could be explained as follows. Human-engineered features tend to have higher precision than recall, as it is often hard to design regular expressions or gazetteers that can detect all possible instances or variations of the desired entities. We conjecture that as the ANN model learns to rely more on such features, it might lose the ability to learn to pick up tokens that deviate from engineered features, resulting in a lower recall. For example, we notice that the phone PHI tokens that are not detected by the model using all features, but are detected by the other two models, are ill-formed phone numbers such as "617-554-|2395", or phone extensions such as "617-690-4031 ext 6599". Since the phone regular expressions do not capture these two examples, they are more likely to be false negatives in the model that uses the phone regular expression features.

3.5 Conclusion

In this chapter, we presented an extension of the ANN-based model for patient note de-identification that can incorporate features. We showed that adding features results in an increase of the recall, in particular features leveraging information from the associated EHRs, namely patient names and doctor names. Our results suggest that constructing patient note de-identification systems should be performed using structured information from the EHRs, the latter being available in a typical, real-life setting. We restricted our EHR-derived features to patient and doctor names, but it could be extended to the many other structured fields that EHR contain, such as patients' addresses, phone numbers, email addresses, professions, and ages.

Chapter 4

Transfer learning for de-identification with neural networks

In the previous chapters, we introduced an architecture based on ANNs that shows promising results for de-identification. In order to achieve high performance, ANNs need to be trained on a labeled dataset of decent size, but labels might be difficult to obtain for the dataset on which the user wishes to perform de-identification. In this chapter, we analyze to what extent transfer learning may address this issue. In particular, we demonstrate that transferring ANN parameters trained on a large labeled dataset to another dataset with a limited number of labels improves upon state-of-the-art results on two different datasets for patient note de-identification. This work is under review [69].

4.1 Introduction

As introduced in the previous chapters, the task of removing PHI from a patient note is referred to as *de-identification*. Existing de-identification systems are often rule-based approaches or feature-based machine learning approaches. However, these techniques require additional lead time for developing and fine-tuning the rules or features specific to each new dataset. Meanwhile, recent efforts using ANNs have yielded state-of-the-art performance without using any manual features [31]. Compared to the previous systems, ANNs have a competitive advantage that the model can be fine-tuned on a new dataset without the over-

head of manual feature development, as long as some labels for the dataset are available.

However, it may still be inefficient to mass deploy ANN-based de-identification systems in practical settings, since creating annotations for patient notes is especially difficult. This is due to the fact that only a restricted set of individuals is authorized to access original patient notes; the annotation task cannot be crowd-sourced, making it slow and expensive to obtain a large annotated corpus. Medical professionals are therefore wary to explore patient notes because of this de-identification barrier, which considerably hampers medical research.

In this chapter, we analyze to what extent transfer learning may improve de-identification performance on datasets with a limited number of labels. By training an ANN model on a large dataset (MIMIC) and transferring it to smaller datasets (i2b2 2014 and i2b2 2016), we demonstrate that transfer learning allows us to outperform the state-of-the-art results.

4.2 Related Work

Transfer learning has been studied for a long time. There is no standard definition of transfer learning in the literature [71]. We follow the definition from [91]: transfer learning aims at performing a task on a target dataset using some knowledge learned from a source dataset. The idea has been applied to many fields such as speech recognition [126] and finance [108].

The successes of ANNs for many applications over the last few years have escalated the interest in studying transfer learning for ANNs. In particular, much work has been done for computer vision [132, 90, 134]. In these studies, some of the parameters learned on the source dataset are used to initialize the corresponding parameters of the ANNs for the target dataset.

Fewer studies have been performed on transfer learning for ANN-based models in the field of natural language processing. For example, Mou et al. [85] focused on transfer learning with convolutional neural networks for sentence classification. To the best of our knowledge, no study has analyzed transfer learning for ANN-based models in the context of NER.

4.3 Model

For the transfer learning experiments, we use the model introduced in Chapter 2 and Figure 2-1. The model is illustrated again as a component-centric view in Figure 4-1, in order to clarify which of the six components are transferred for the experiments:

1. **Token embedding layer** maps each token to a token embedding.
2. **Character embedding layer** maps each character to a character embedding.
3. **Character LSTM layer** takes as input character embeddings and outputs a single vector that summarizes the information from the sequence of characters in the corresponding token.
4. **Token LSTM layer** takes as input a sequence of token vectors, which are formed by concatenating the outputs of the token embedding layer and the character LSTM layer, and outputs a sequence of vectors.
5. **Fully connected layer** takes the output of the token LSTM layer as input, and outputs vectors containing the scores of each label for the corresponding tokens.
6. **Sequence optimization layer** takes the sequence of vectors from the output of the fully connected layer and outputs the most likely sequence of predicted labels, by optimizing the sum of unigram label scores as well as bigram label transition scores.

Figure 4-1 shows how these six components are interconnected to form the model.

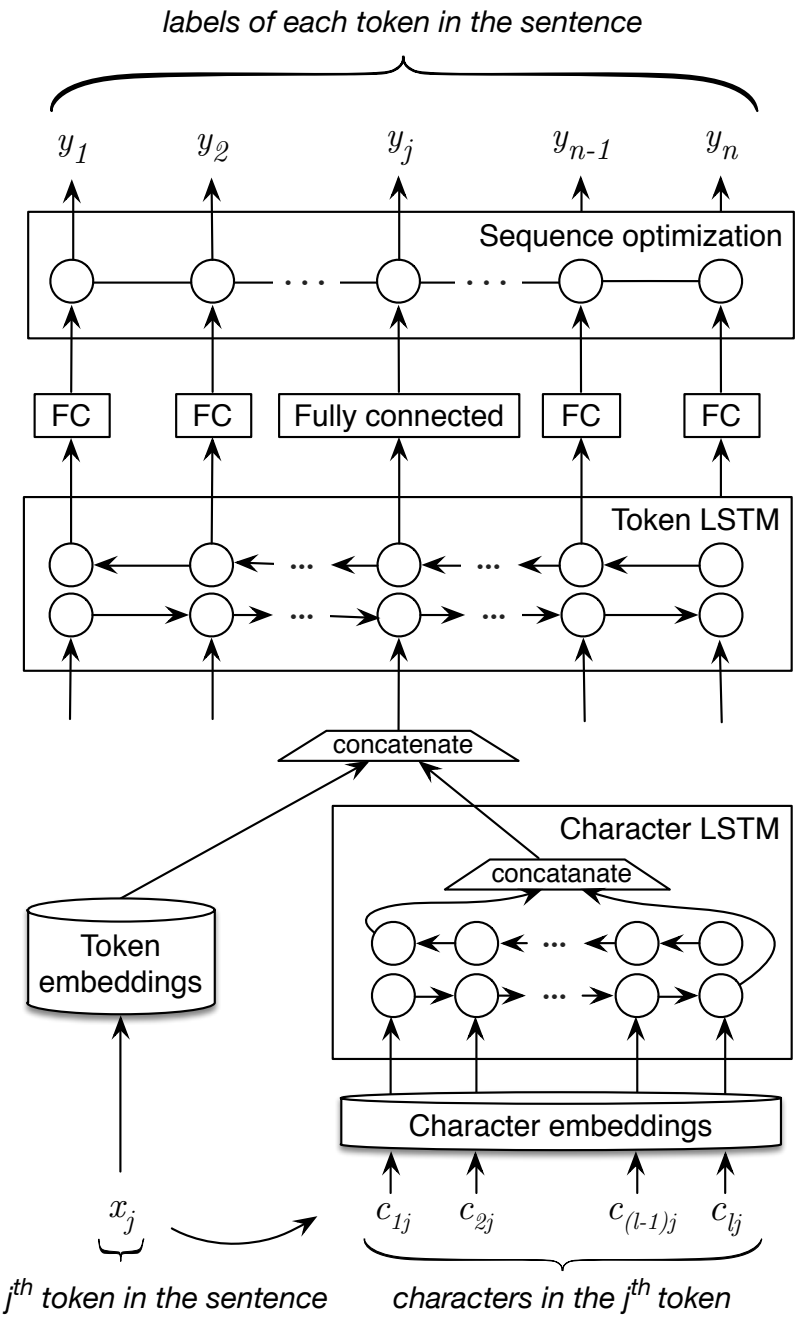


Figure 4-1: ANN model for NER. For transfer learning experiments, we train the parameters of the model on a source dataset, and transfer all (Experiment 1) or some (Experiment 2) of the parameters to initialize the model for training on a target dataset. For Experiment 2, we transfer the parameters of up to the six components in the following order: token embedding layer, character embedding layer, character LSTM layer, token LSTM layer, fully connected layer, and sequence optimization layer. Note that this figure illustrates the same model as Figure 2-1, except that it indicates the clear boundaries and names of the six components.

4.4 Experiments

4.4.1 Datasets

We use three de-identification datasets for the transfer learning experiments: MIMIC, i2b2 2014, and i2b2 2016. The MIMIC de-identification dataset was introduced in [31] and also described in Chapter 2, and is a subset of the MIMIC-III dataset [57, 45, 99]. The i2b2 2014 and 2016 datasets were released as part of the 2014 i2b2/UTHealth shared task Track 1 [110] and the 2016 i2b2 CEGS N-GRID shared task, respectively. Table 4.1 presents the datasets' sizes.

	MIMIC	**i2b2 2014**	**i2b2 2016**
Vocabulary size	69,525	46,803	61,503
Number of notes	1,635	1,304	1,000
Number of tokens	2,945,228	984,723	2,689,196
Number of PHI instances	60,725	28,867	41,142
Number of PHI tokens	78,633	41,355	54,420

Table 4.1: Overview of the MIMIC and i2b2 datasets. PHI stands for protected health information. Note that this table is the same as Table 2.2, except with the addition of the third data set, i2b2 2016.

4.4.2 Transfer learning

The goal of transfer learning is to leverage the information present in a source dataset to improve the performance of an algorithm on a target dataset. In our setting, we apply transfer learning by training the parameters of the ANN model on the source dataset (MIMIC), and using the same ANN to retrain on the target dataset (i2b2 2014 or 2016) for fine-tuning. We use MIMIC as the source dataset since it is the dataset with the most labels. We perform two sets of experiments to gain insights on how effective transfer learning is and which parameters of the ANN are the most important to transfer.

Experiment 1 Quantifying the impact of transfer learning for various training set sizes of the target dataset. The primary purpose of this experiment is to assess to what extent transfer learning improves the performance on the target dataset. We experiment with

different training set sizes to understand how many labels are needed for the target dataset to achieve reasonable performance with and without transfer learning.

Experiment 2 Analyzing the importance of each parameter of the ANN in the transfer learning. Instead of transferring all the parameters, we experiment with transferring different combinations of parameters. The goal is to understand which components of the ANN are the most important to transfer. The lowest layers of the ANN tend to represent task-independent features, whereas the topmost layers are more task-specific. As a result, we try transferring the parameters starting from the bottommost layer up to the topmost layer, adding one layer at a time.

4.5 Results

Experiment 1 Figure 4-2 compares the F1-scores of the ANN trained only on the target dataset against the ANN trained on the source dataset followed by the target dataset. Transfer learning improves the F1-scores over training only with the target dataset, though the improvement diminishes as the number of training samples used for the target dataset increases. This implies that the representations learned from the source dataset are efficiently transferred and exploited for the target dataset.

Therefore, when transfer learning is adopted, fewer annotations are needed to achieve the same level of performance as when the source dataset is unused. For example, on the i2b2 2014 dataset, performing transfer learning and using 16% of the i2b2 training set leads to similar performance as not using transfer learning and using 34% of the i2b2 training set. Transfer learning thus allows us to cut by half the number of labels needed on the target dataset in this case.

For both the i2b2 2014 and 2016 datasets, the performance gains from transfer learning are greater when the training set size of the target dataset is small. The largest improvement can be observed for i2b2 2014 when using 5% of the dataset as the training set (consisting of around 2k PHI tokens out of 50k tokens), where transfer learning increases the F1-score by around 3.1 percent point, from 90.12 to 93.21. Even when all of the training set is used,

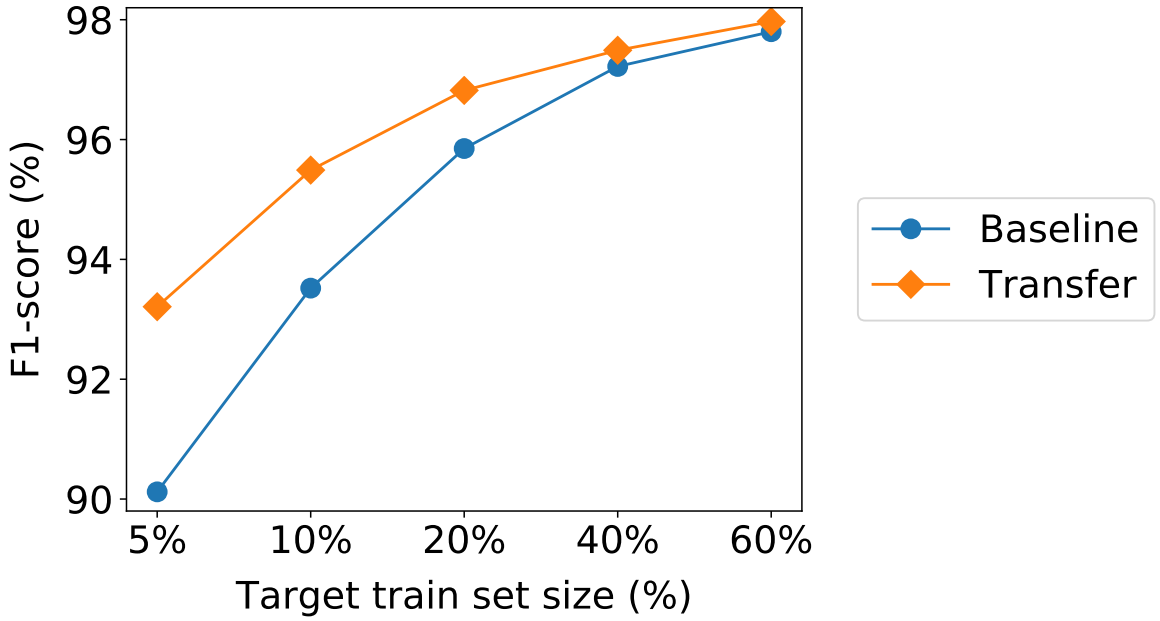
the F1-score improves when using transfer learning, albeit by just 0.17 percent point, from 97.80 to 97.97.

Experiment 2 Figure 4-3 shows the importance of each layer of the ANN in transfer learning. We observe that transferring a few lower layers is almost as efficient as transferring all layers. For i2b2 2014, transferring up to the token LSTM shows great improvements for each layer, but there is less improvement for each added layer beyond that. For i2b2 2016, larger improvements can be observed up to the character LSTM and less so beyond that layer.

The parameters in the lower layers therefore seems to contain most information that is relevant to the de-identification task in general, which supports the common hypothesis that higher layers of ANN architectures contain the parameters that are more specific to the task as well as the dataset used for training.

Despite the observation that transferring a few lower layers may be sufficient for efficient transfer learning, it is interesting to see that adding the topmost layers to the transfer learning does not hurt the performance. When retraining the model on the target dataset, the ANN is able to adapt to the target dataset quite well despite some the higher layers being initialized to parameters that are likely to be more specific to the source dataset.

(a) i2b2 2014



(b) i2b2 2016

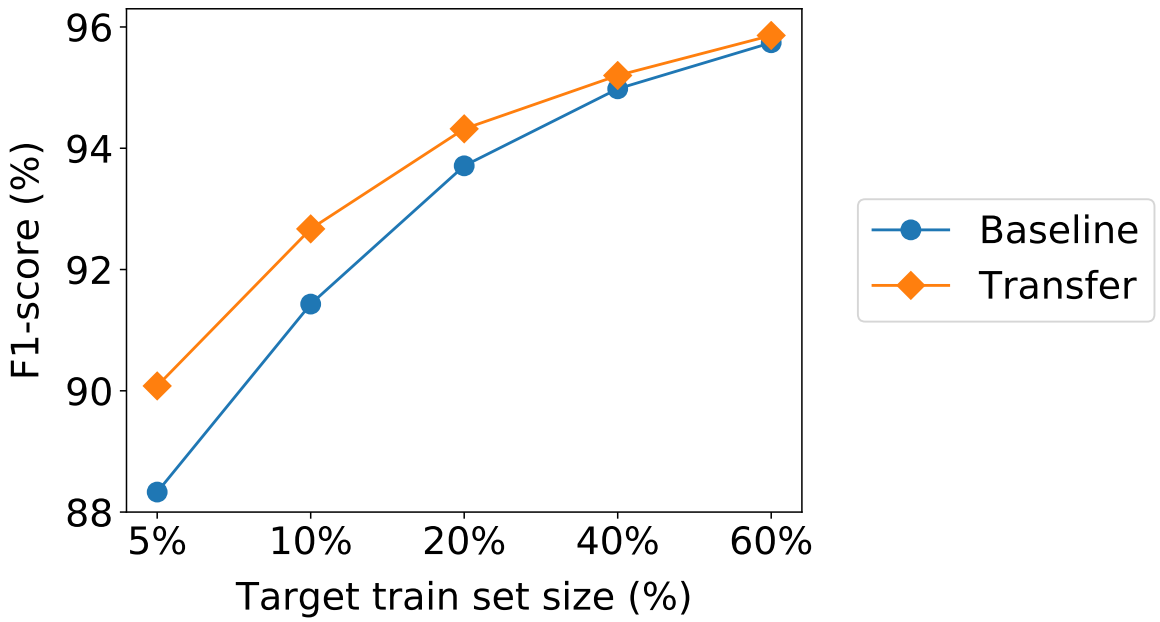
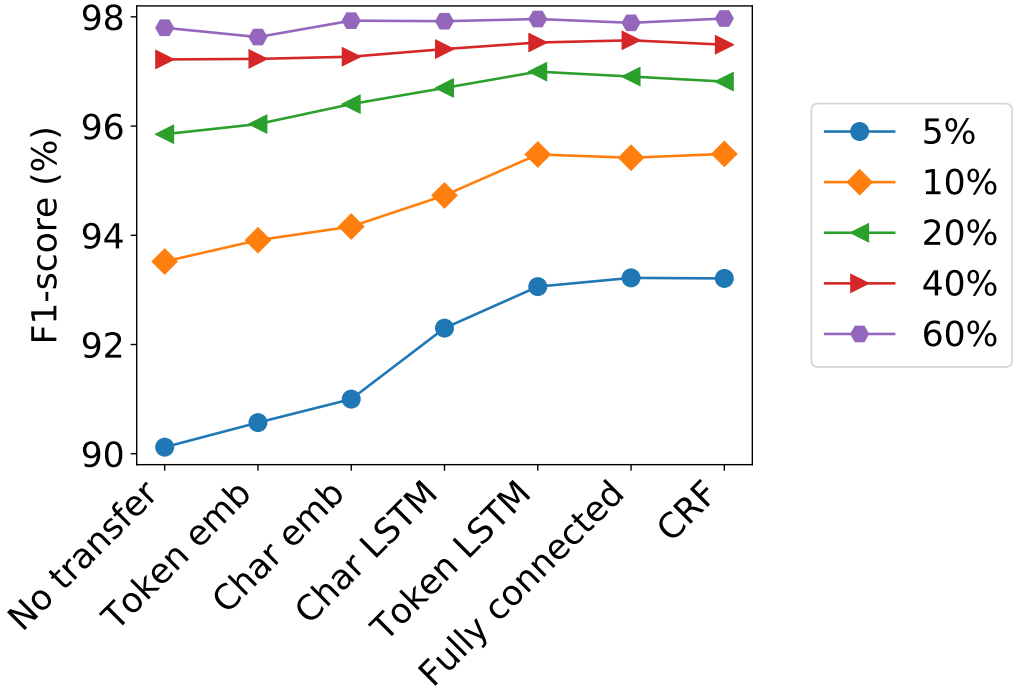


Figure 4-2: Impact of transfer learning on the F1-scores. Baseline corresponds to training the ANN model only with the target dataset, and transfer learning corresponds to training on the source dataset followed by training on the target dataset. The target training set size is the percentage of training set in the whole dataset, and 60% corresponds to the full official training set.

(a) i2b2 2014



(b) i2b2 2016

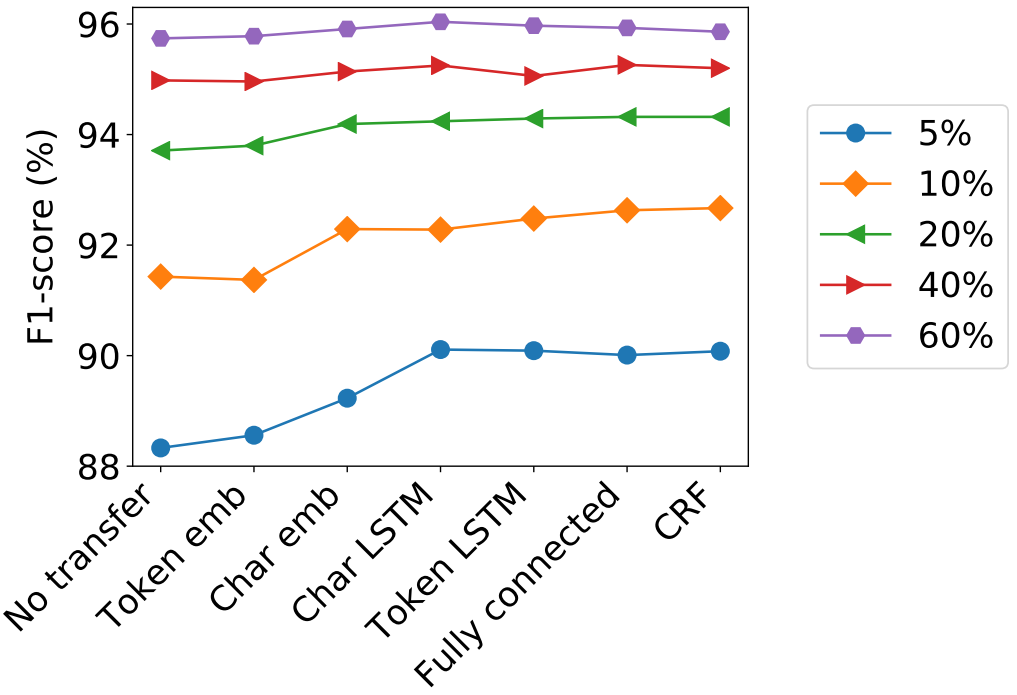


Figure 4-3: Impact of transferring the parameters up to each layer of the ANN model using various training set sizes on the target dataset: 5%, 10%, 20%, 40%, and 60% (official training set).

4.6 Conclusion

In this chapter, we have studied transfer learning with ANNs for NER, specifically patient note de-identification, by transferring ANN parameters trained on a large labeled dataset to another dataset with limited human annotations. We demonstrated that transfer learning improves the performance over the state-of-the-art results on two datasets. Transfer learning may be especially beneficial for a target dataset with small number of labels. Our experiments show that transferring some of the lower layers can be as efficient as transferring all the layers, but one can simply transfer all the layers and retrain on the target dataset without hurting the performance if an appropriate threshold is unknown.

Chapter 5

NeuroNER: an Easy-to-Use Named-Entity Recognition Tool based on ANN

In the previous chapters, we introduced the ANN architecture for the de-identification of patient notes that outperforms existing systems. The de-identification task is a specialized form of named-entity recognition (NER), which aims at identifying entities of interest in a text. Therefore, the same ANN architecture can be applied more broadly to any kind of NER task since it learns effective features from the data instead of relying on manually engineered features. For example, it can be used to identify entities from newspaper articles, or medical concepts from clinical text. However, ANNs may be challenging to use for the non-expert user. In this chapter, we present NeuroNER[1], an easy-to-use named-entity recognition tool based on ANNs. The user can annotate entities using a graphical web-based user interface (BRAT): the annotations are then used to train an ANN, which in turn predicts entities' locations and categories in new texts. NeuroNER aims at making this annotation-training-prediction flow smooth and accessible to anyone. This work is under review [30].

[1]NeuroNER can be found online at: https://github.com/Franck-Dernoncourt/NeuroNER

5.1 Introduction

Named-entity recognition (NER) aims at identifying entities of interest in the text, such as location, organization and temporal expression. Identified entities can be used in various downstream applications such as patient note de-identification and information extraction systems. They can also be used as features for machine learning systems for other natural language processing tasks.

Early systems for NER relied on rules defined by humans. Rule-based systems are time-consuming to develop, and cannot be easily transferred to new types of texts or entities. To address these issues, researchers have developed machine-learning-based algorithms for NER, using a variety of learning approaches, such as fully supervised learning, semi-supervised learning, unsupervised learning, and active learning. NeuroNER is based on a fully supervised learning algorithm, which is the most studied approach [86].

Fully supervised approaches to NER include support vector machines (SVM) [4], maximum entropy models [17], decision trees [102] as well as sequential tagging methods such as hidden Markov models [14], Markov maximum entropy models [61], and conditional random fields (CRFs) [76, 120, 8, 38]. Similar to rule-based systems, these approaches rely on handcrafted features, which are challenging and time-consuming to develop and may not generalize well to new datasets.

More recently, artificial neural networks (ANNs) have been shown to outperform other supervised algorithms for NER [24, 64, 70, 63]. The effectiveness of ANNs can be attributed to their ability to learn effective features jointly with model parameters directly from the training dataset, instead of relying on handcrafted features developed from a specific dataset. However, ANNs remain challenging to use for non-expert users.

Contributions NeuroNER aims at making ANN-based named-entity recognition available to anyone, by focusing on usability. To enable the users to create or modify annotations for an new or existing corpus, NeuroNER interfaces with the web-based annotation program BRAT [109]. NeuroNER makes the annotation-training-prediction flow smooth and accessible to anyone, while leveraging the state-of-the-art prediction capabilities of

ANNs. NeuroNER is open source and freely available online[2].

5.2 Related Work

Existing publicly available NER systems geared toward non-experts do not use ANNs. For example, Stanford NER [41], ABNER [103], the MITRE Identification Scrubber Toolkit (MIST) [2], [16], BANNER [66] and NERsuite [23] rely on CRFs. GAPSCORE uses SVMs [21]. Apache cTAKES [100] and Gate's ANNIE [26, 75] use mostly rules. NeuroNER, the first ANN-based NER system for non-experts, is more generalizable to new corpora due to the ANNs' capability to learn effective features jointly with model parameters.

Furthermore, in many cases, the NER systems assume that the user already has an annotated corpus formatted in a specific data format. As a result, the user often has to connect their annotation tool with the NER systems by reformatting annotated data, which can be time-consuming and error-prone. Moreover, if the user wants to manually improve the annotations predicted by the NER system (e.g., if they use the NER system to accelerate the human annotations), they have to perform additional data conversion. NeuroNER aims to streamline the process by incorporating BRAT, a widely-used and easy-to-use annotation tool.

5.3 System Description

NeuroNER comprises two main components: an NER engine and an interface with BRAT. NeuroNER also comes with real-time monitoring tools for training, and pre-trained models that can be loaded to the NER engine in case the user does not have access to labelled training data. Figure 5-1 presents an overview of the system.

[2]NeuroNER can be found online at https://github.com/Franck-Dernoncourt/NeuroNER

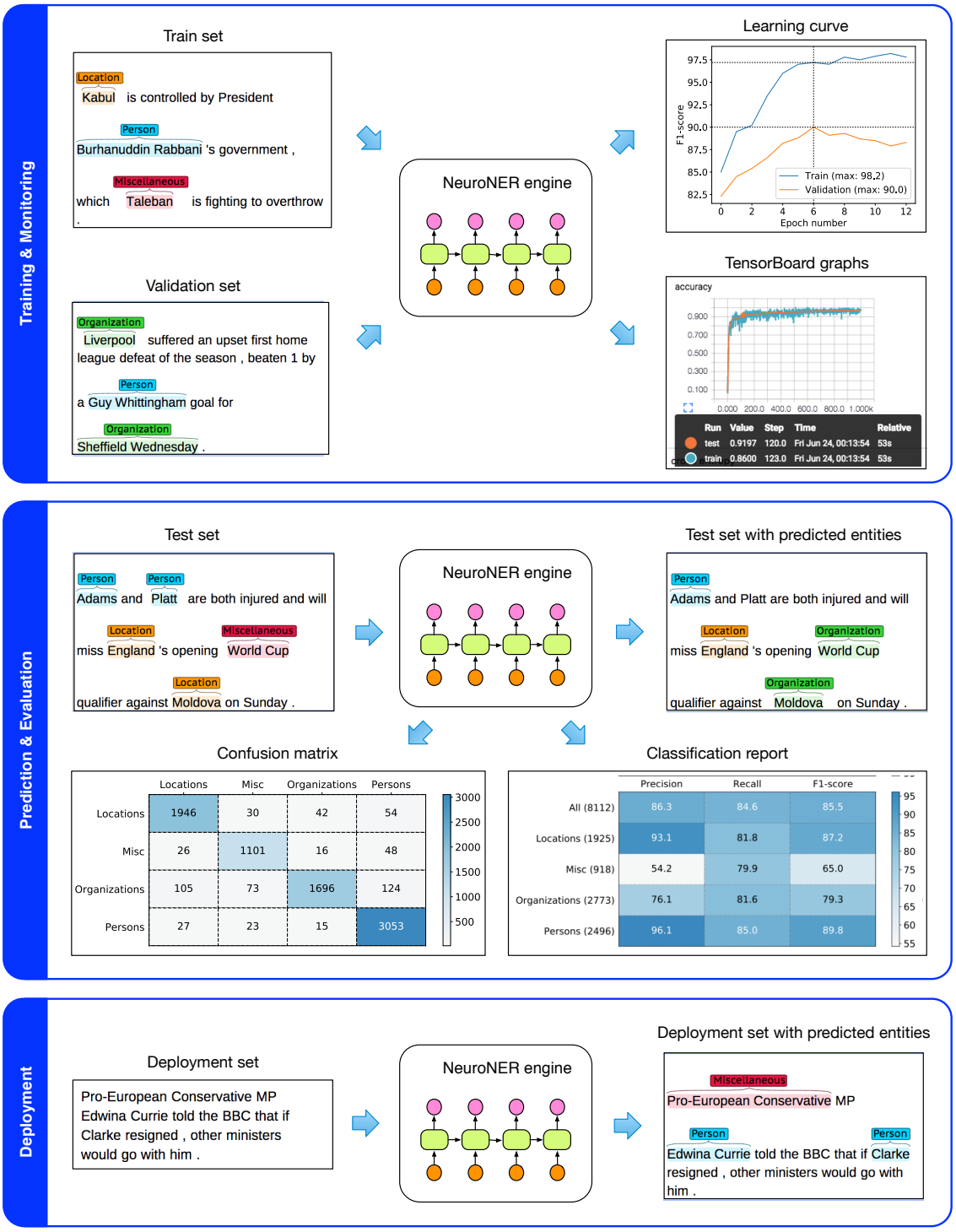


Figure 5-1: NeuroNER system overview. In the NeuroNER engine, the training set is used to train the parameters of the ANN, and the validation set is used to determine when to stop training. The user can monitor the training process in real time via the learning curve and TensorBoard. To evaluate the trained ANN, the labels are predicted for the test set: the performance metrics can be calculated and plotted by comparing the predicted labels with the gold labels. The evaluation can be done at the same time as the training if the test set is provided along with the training and validation sets, or separately after the training or using a pre-trained model. Lastly, the NeuroNER engine can label the deployment set, i.e., any new text without gold labels.

5.3.1 NER engine

The NER engine takes as input three sets of data with gold labels: the training set, the validation set, and the test set. Additionally, it can also take as input the deployment set, which refers to any new text without gold labels that the user wishes to label. The files that comprise each set of data should be in the same format as used for the annotation tool BRAT and organized in the corresponding folder.

The NER engine's ANN contains three layers:

- Character-enhanced token-embedding layer,
- Label prediction layer,
- Label sequence optimization layer.

The character-enhanced token-embedding layer maps each token to a vector representation. The sequence of vector representations corresponding to a sequence of tokens is then input to label prediction layer, which outputs the sequence of vectors containing the probability of each label for each corresponding token. Lastly, the label sequence optimization layer outputs the most likely sequence of predicted labels based on the sequence of probability vectors from the previous layer. All layers are learned jointly.

The ANN as well as the training process have several hyperparameters such as character embedding dimension, character-based token-embedding LSTM dimension, token embedding dimension, and dropout probability. All hyperparameters may be specified in a configuration file that is human-readable, so that the user does not have to dive into any code. Listing 5.1 presents an excerpt of the configuration file.

```
[dataset]
dataset_folder               = dat/conll

[character_lstm]
using_character_lstm         = True
char_embedding_dimension     = 25
char_lstm_dimension          = 50

[token_lstm]
token_emb_pretrained_file    = glove.txt
token_embedding_dimension    = 200
token_lstm_dimension         = 300

[crf]
using_crf                    = True
random_initial_transitions   = True

[training]
dropout                      = 0.5
patience                     = 10
maximum_number_of_epochs     = 100
maximum_training_time        = 10
number_of_cpu_threads        = 8
number_of_gpus               = 0
```

Listing 5.1: Excerpt of the configuration file used to define the ANN as well as the training process. Only the `dataset_folder` parameter needs to be changed by the user: the other parameters have reasonable default values, which the user may optionally tune.

5.3.2 Real-time monitoring for training

As training an ANN may take many hours, or even a few days on very large datasets, NeuroNER provides the user with real-time feedback during the training for monitoring purposes. Feedback is given through two different means: plots generated by NeuroNER, and TensorBoard.

Plots NeuroNER generates several plots showing the training progress and outcome at each epoch. Plots include the evolution of the overall F1-score over time, confusion matrix visualizing the number of correct versus wrong predictions for each class, and classification report showing the F1-score, precision and recall for each class.

TensorBoard As NeuroNER is based on TensorFlow, it leverages the functionalities of TensorBoard. TensorBoard is a suite of web applications for inspecting and understanding TensorFlow runs and graphs. It allows the user to view in real time the performance achieved by the ANN being trained. Moreover, since it is web-based, these performances can be conveniently shared with anyone remotely. Lastly, since graphs generated by TensorBoard are interactive, the user may gain further insights on the ANN performances.

5.3.3 Pre-trained models

Some users may prefer not to train any ANN model, either due to time constraints or unavailable gold labels. For example, if the user wants to tag protected health information, they might not be able to have access to a labeled identifiable dataset. To address this need, NeuroNER provides a set of pre-trained models. Users are encouraged to contribute by uploading their own trained models. NeuroNER also comes with several pre-trained token embeddings, either with word2vec [78, 80, 81] or GloVe [95], which the NeuroNER's engine can load easily once specified in the configuration file.

5.3.4 Annotations

NeuroNER is designed to smoothly integrate with the web-based annotation tool BRAT, so that non-expert users may create or improve annotations. Specifically, NeuroNER addresses two main use cases:

- creating new annotations from scratch, e.g. if the goal is to annotate a dataset for which no gold label is available,
- improving the annotations of an already labeled dataset: the annotations may have been done by another human or by a previous run of NeuroNER.

In the latter case, the user may use NeuroNER interactively, by iterating between manually improving the annotations and running the NeuroNER's engine with the new annotations to obtain more accurate annotations.

NeuroNER takes as input datasets in BRAT format, and outputs BRAT-formatted predictions, which makes it easy to start training right from the annotations as well as visualize and analyze the predictions. BRAT was chosen for two main reasons: it is easy to use, and it can be deployed as a web application, which allows crowdsourcing. As a result, the user may quickly gather a vast amount of annotations by using crowdsourcing marketplaces in the Internet such as Amazon Mechanical Turk [19] and CrowdFlower [40].

5.3.5 System requirements

NeuroNER runs on Linux (tested with Ubuntu), Mac OS X, and Microsoft Windows. It requires Python 3.5, TensorFlow 1.0 [1], and scikit-learn [94]. A setup script is provided to make the installation straightforward. It can use the GPU if available, and the number of CPU threads and GPUs to use can be specified in the configuration file.

5.3.6 Performance

To assess the quality of NeuroNER's predictions, we use two publicly and freely available datasets for named-entity recognition: CoNLL-2003 and i2b2 2014. CoNLL-2003 [119] is a widely studied dataset with 4 usual types of entity: persons, organizations, locations and miscellaneous names. We use the English version.

The i2b2 2014 dataset [110] was released as part of the 2014 i2b2/UTHealth shared task Track 1. It is the largest publicly available dataset for de-identification, which is a form of named-entity recognition where the entities are protected health information such as patients' names and patients' phone numbers. Ten teams participated in this shared task, and 22 systems were submitted.

Table 5.1 compares NeuroNER with state-of-the-art systems on CoNLL-2003 and i2b2 2014. The performances of NeuroNER are on par with the state-of-the-art systems.

Model	CoNLL-2003	i2b2 2014
Best published	90.9	97.9
Our model	90.5	97.6

Table 5.1: F1-scores (%) on the test set comparing NeuroNER with the best published methods in the literature, viz. Passos et al. [93] for CoNLL-2003 and Dernoncourt et al. [31] for i2b2 2014. Passos et al. [93] employs a CRF model with features comprising gazetteers and token embeddings trained with a skip-gram model modified to use external lexicons. The result for Dernoncourt et al. [31] is also introduced as "CRF+ANN" model in Table 2.3.

5.4 Conclusions

In this chapter, we have presented NeuroNER, an ANN-based NER tool that is accessible to non-expert users and yields state-of-the-art results. Addressing the need of many users who want to create or improve annotations, NeuroNER smoothly integrates with the web-based annotation tool BRAT.

Chapter 6

Relation extraction

In the previous chapter, we introduced the ANN architecture for extracting named entities from natural language text. The next step toward more advanced information extraction is to identify the relations among the extracted entities. In this chapter, we present a system based on a convolutional neural network to extract relations. Our model ranked first in the SemEval-2017 task 10 (ScienceIE[1] [5]) for relation extraction in scientific articles (subtask C). This work will be published at ACL 2017 [68].

6.1 Introduction and related work

The number of articles published every year keeps increasing [36, 65], with well over 50 million scholarly articles published so far [56]. While this repository of human knowledge contains invaluable information, it has become increasingly difficult to take advantage of all available information due to its sheer amount.

One challenge is that the knowledge present in scholarly articles is mostly unstructured. One approach to organize this knowledge is to classify each sentence [58, 3, 53, 29]. Another approach is to extract entities and relations between them, which is the focus of the ScienceIE shared task at SemEval-2017 [5].

Relation extraction can be seen as a process comprising two steps that can be done jointly [72] or separately: first, entities of interest need to be identified, and second, the

[1] http://scienceie.github.io

relation among the entities has to be determined. In this work, we concentrate on the second step (often referred to as relation extraction or classification) and on binary relations, i.e. relations between two entities. Extracted relations can be used for a variety of tasks such as question-answering systems [96], ontology extension [101], and clinical trials [43].

In this chapter, we describe the system that we submitted for the ScienceIE shared task. Our system is based on convolutional neural networks and ranked first for relation extraction (subtask C).

Existing systems for relation extraction can be classified into five categories: systems based on hand-built patterns [131], bootstrapping methods [18], unsupervised methods [46], distant supervision [104], and supervised methods. We focus on supervised methods, as the ScienceIE shared task provides a labeled training set.

Supervised methods for relation extraction commonly employ support vector machines [122, 124, 82, 49], naïve Bayes [133], maximum entropy [111], or conditional random fields [114]. These methods require the practitioner to handcraft features, such as surface, lexical, syntactic features [47] or features derived from existing ontologies [97]. The use of kernels based on dependency trees has also been explored [20, 25, 137].

More recently, a few studies have investigated the use of artificial neural networks for relation extraction [105, 88, 52]. Our approach follows this line of work.

6.2 Model

Our model for relation extraction comprises three parts: preprocessing, convolutional neural network (CNN), and postprocessing.

6.2.1 Preprocessing

The preprocessing step takes as input each raw text (i.e., a paragraph of a scientific article in ScienceIE) as well as the location of all entities present in the text, and outputs several examples. Each example is represented as a list of tokens, each with four features: the relative positions of the two entity mentions, their entity types and part-of-speech (POS) tags. Figure 6-1 shows an example from the ScienceIE corpus in the table on the left.

Sentence and token boundaries as well as POS tags are detected using the Stanford CoreNLP toolkit [74], and every pair of entity mentions of the same type within each sentence boundary is considered to have a potential relation. We also remove any references (e.g. [1, 2]), which are irrelevant to the task, and ensure that the sentences are not too long by eliminating the tokens before the beginning of the first entity mention and after the end of the second entity mention.

6.2.2 CNN architecture

The CNN takes each preprocessed sentence as input and predicts the relation between the two entities. The CNN architecture, illustrated in Figure 6-1, consists of four main layers, similar to the one used in text classification [24, 59, 67, 44].

1. the **embedding layer** converts each feature (word, relative positions 1 and 2, type of entity, and POS tag) into an embedding vector via a lookup table and concatenates them.
2. the **convolutional layer** with ReLU activation transforms the embeddings into feature maps by sliding filters over the tokens
3. the **max pooling layer** selects the highest feature value in each feature map by applying the max operator.
4. the **fully connected layer** with softmax activation outputs the probability of each relation.

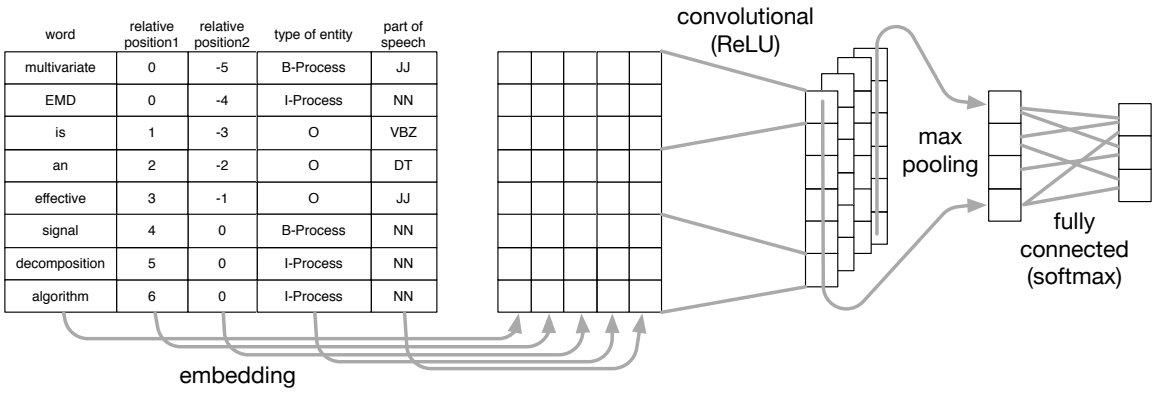


Figure 6-1: CNN architecture for relation extraction. The left table shows an example of input to the model.

6.2.3 Rule-based postprocessing

The postprocessing step uses the rules in Table 6.1 to correct the relations detected by the CNN, or to detect additional relations. These rules were developed from the examples in the training set, to be consistent with common sense.

Examples	Rule format	Relations detected
transmission electron microscopy (TEM)	A (B)	If B is an abbreviation of A, then A and B are synonyms.
high purity standard metals (Sn, Pb, Zn, Al, Ag, Ni)	A (B, ... , Z)	If any of B, ... , Z is a hyponym of A, then all of them are hyponyms of A.
(TEMs), scanning electron microscopes	(A) B	A and B have no relation.
DOTMA/DOPE	A/B	A and B have no relation.

Table 6.1: Rules used for postprocessing. We considered B to be an abbreviation of A if the first letters of each token in A form B. The examples are from the training and development sets

6.2.4 Implementation

During training, the objective is to maximize the log probability of the correct relation type. The model is trained using stochastic gradient descent with minibatch of size 16, updating all parameters, i.e., token embeddings, feature embeddings, CNN filter weights, and fully connected layer weights, at each gradient descent step. For regularization, dropout is applied before the fully connected layer, and early stop with a patience of 10 epochs is used based on the development set.

The token embeddings are initialized using publicly available[2] pre-trained token embeddings, namely GloVe [95] trained on Wikipedia and Gigaword 5 [92]. The feature embeddings and the other parameters of the neural network are initialized randomly.

To deal with class imbalance, we upsampled the synonym and hyponym classes by duplicating the examples in the positive classes so that the *upsampling ratio*, i.e., the ratio of the number of positive examples in each class to that of the negative examples, is at least 0.5. Without the upsampling, the trained model would have poor performance.

[2] http://nlp.stanford.edu/projects/glove/

6.3 Experiments

6.3.1 Dataset

We evaluate our model on the ScienceIE dataset [5], which consists of 500 journal articles evenly distributed among the domains of computer science, material sciences and physics. Three types of entities are annotated: process, task, and material. The relation between each pair of entities of the same type within a sentence is annotated as either "Synonym-of", "Hyponym-of", or "None". Table 6.2 shows the number of examples for each relation class.

Relation	**Train**	**Dev**	**Test**
Hyponym-of	420	123	95
Synonym-of	253	45	112
None	5355	1240	1503
Total	6028	1408	1710

Table 6.2: Number of examples for each relation class in ScienceIE. "Dev": Development.

6.3.2 Hyperparameters

Table 6.3 details the experiment ranges and choices of hyperparameters. The results were quite robust to the choice of hyperparameters within the specified ranges.

Hyperparameter	**Choice**	**Experiment range**
Token embedding dim.	100	50 – 300
Feature embedding dim.	10	5 – 50
CNN filter height	5	3 – 15
Number of CNN filters	200	50 – 500
Dropout probability	0.5	0 – 1
Upsampling ratio	3	0.5 – 5

Table 6.3: Experiment ranges and choices of hyperparameters. For these experiments, we used the official training set as the training/development set with a $75\%/25\%$ split, and the official development set as the test set.

6.3.3 Argument ordering strategies

One of the main challenges in relation extraction is the ordering of arguments in relations, as many relations are *order-sensitive*. For example, consider the sentences “A dog is an animal" and "This animal is a dog". If we set “dog” to be the first argument and “animal” the second, then the corresponding relation is “Hyponym-of”; however, if we reverse the argument order, then the “Hyponym-of” relation does not hold any more. However, since the words “animal” and “dog” may appear in text in either of the two possible orderings as shown in the two examples, an algorithm has no way of knowing which entity is the first argument and which is second if only the raw text is provided. Put differently, if we do not specify the first and second arguments, the relation between them cannot be determined, at least in case of an asymmetric relation such as “Hyponym-of”.

Therefore, it is crucial to ensure that 1) the CNN is provided with the information about the argument order, and 2) it is able to utilize the given information efficiently. In our work, the former point is addressed by providing the CNN with the two relative position features compared to the first and the second argument of the relation respectively. In order to verify the latter point, we experimented with four strategies for argument ordering, outlined in Table 6.4.

annotation	A (arg1) is a Hyponym of (rel) B (arg2)					
order in text	... A ... B ...			... B ... A ...		
strategy	**rel**	**arg1**	**arg2**	**rel**	**arg1**	**arg2**
correct order	Hypo	A	B	Hypo	A	B
correct order	Hypo	A	B	Hypo	A	B
w/ neg. smpl.	None	B	A	None	B	A
fixed order	Hypo	A	B	Hyper	B	A
any order	Hypo	A	B	Hyper	A	B
	Hyper	B	A	Hypo	B	A

annotation	A (arg1) is a Synonym of (rel) B (arg2)					
order in text	... A ... B ...			... B ... A ...		
strategy	**rel**	**arg1**	**arg2**	**rel**	**arg1**	**arg2**
correct order	Synon	A	B	Synon	A	B
correct order	Synon	A	B	Synon	A	B
w/ neg. smpl.	Synon	B	A	Synon	B	A
fixed order	Synon	A	B	Synon	B	A
any order	Synon	A	B	Synon	A	B
	Synon	B	A	Synon	B	A

Table 6.4: Argument ordering strategies. "w/ neg. smpl.": with negative sampling [129], "rel": relation, "arg": argument. "Synon", "Hypo", "Hyper", and "None" refers to the "Synonym-of", "Hyponym-of", "Hypernym-of", and "None' relations. Note that the "Hypernym-of" relation is the reverse of the "Hyponym-of" relation, introduced in addition to the relations annotated for the dataset.

6.4 Results and Discussion

Table 6.5 shows the results from experimenting with various argument ordering strategies. The correct order strategy performed the worst, but the negative sampling improved over it slightly, while the fixed order and any order strategies performed the best. The latter two strategies performed almost equally well in terms of micro-averaged F1-score. This implies that for relation extraction it may be advantageous to use both the original relation classes as well as their "reverse" relation classes for training, instead of using only the original relation classes with the "correct" argument ordering (with or without the negative sampling). Moreover, ordering the argument as the order of appearance in the text and training once per relation (i.e., fixed order) is as efficient as training each relation as two examples in two possible argument ordering, one with the original relation class and the

Labels used	Training strategy	Evaluation strategy	Hyponym-of			Synonym-of			Micro-averaged		
			P	R	F1	P	R	F1	P	R	F1
All	correct order	any order	0.193	0.101	0.132	0.782	0.640	0.703	0.409	0.245	0.306
	corr. w/ n. s.	any order	0.431	0.127	0.196	0.826	0.756	0.788	**0.638**	0.295	0.404
	any order	any order	0.482	0.197	0.279	0.784	0.756	0.769	0.621	0.346	0.444
	any order	fixed order	**0.486**	0.195	0.278	0.773	0.753	0.763	0.621	0.345	0.443
	fixed order	any order	0.372	0.218	0.274	0.743	0.756	0.749	0.516	0.362	0.425
	fixed order	fixed order	0.425	0.213	0.283	0.803	0.753	0.777	0.578	0.358	0.441
Hyponym	correct order	any order	0.108	0.069	0.084	-	-	-	-	-	-
	corr. w/ n. s.	any order	0.215	0.115	0.148	-	-	-	-	-	-
	any order	any order	0.384	0.246	0.299	-	-	-	-	-	-
	any order	fixed order	0.410	0.235	0.298	-	-	-	-	-	-
	fixed order	any order	0.385	**0.249**	**0.301**	-	-	-	-	-	-
	fixed order	fixed order	0.409	0.237	0.297	-	-	-	-	-	-
Synonym	any order	any order	-	-	-	0.855	0.771	0.811	-	-	-
	any order	fixed order	-	-	-	0.852	**0.776**	**0.812**	-	-	-
Hyp+Syn	any + any	any + fixed	0.385	0.228	0.285	**0.857**	0.771	**0.812**	0.553	**0.373**	**0.445**

Table 6.5: Results for various ordering strategies on the development set of the ScienceIE dataset, averaged over 10 runs each. "corr. w/ n. s.": correct order with negative sampling. Hyp+Syn is obtained by merging the output of the best hyponym classifier and that of the best synonym classifier. For these experiments, we used the official training set as the training/development set with a $75\%/25\%$ split, and the official development set as the test set.

other with the reverse relation class (i.e., any order), despite the small size of the dataset.

The difference in performance between the correct order versus the fixed or any order strategies is more prominent for the "Hyponym-of" relation than for the "Synonym-of" relation. This is expected, since the argument ordering strategy is different only for the order-sensitive "Hyponym-of" relation. It is somewhat surprising though, that the correct order strategy performs worse than the other strategies even for order-insensitive "Synonym-of" relation. This may be due to the fact that the model does not see any training examples with the reversed argument ordering for the "Synonym-of" relation. In comparison, the negative sampling strategy, which learns from both the original and reversed argument ordering for the "Synonym-of" relation, the performance is comparable to the two best performing strategies.

We have also experimented with different evaluation strategies for the models trained with the any order and fixed order strategies. When the model is trained with the any order strategy, the choice of the evaluation strategy does not impact the performance. In

contrast, when the model is trained with the fixed order strategy, it performs better if the same strategy is used for evaluation. This may be the reason why the model trained with the correct order strategy does not perform as well, since it has to be evaluated with a different strategy from training, namely the any order strategy, as we do not know the correct ordering of arguments for examples in the test set.

We have also tried training binary classifiers for the "Hyponym-of" and the "Synonym-of" relations separately and then merging the outputs of the best classifiers for each relations. While the binary classifiers individually performed better than the multi-way classifier for each corresponding relation class, the overall performance based on the micro-averaged F1-score did not improve over the multi-way classifier after merging the outputs of the hyponym and the synonym classifiers.

Based on the results from the argument ordering strategy experiments, we submitted the model trained using the fixed order strategy, which ranked number one in the challenge. The result is shown in Table 6.6.

Relation	**Precision**	**Recall**	**F1-score**
Synonym-of	0.820	0.813	0.816
Hyponym-of	0.455	0.421	0.437
Micro-averaged	0.658	0.633	0.645

Table 6.6: Result on the test set of the ScienceIE dataset, using the official train/dev/test split.

To quantify the importance of various features of our model, we trained the model by gradually adding more features one by one, from word embeddings, relative positions, and entity types to POS tags in order. The results on the importance of the features as well as postprocessing are shown in Figure 6-2. Adding the relative position features improved the performance the most, while adding the entity type improved it the least. Note that even without the postprocessing, the F1-score is 0.63, which still outperforms the second-best system with the F1-score of 0.54.

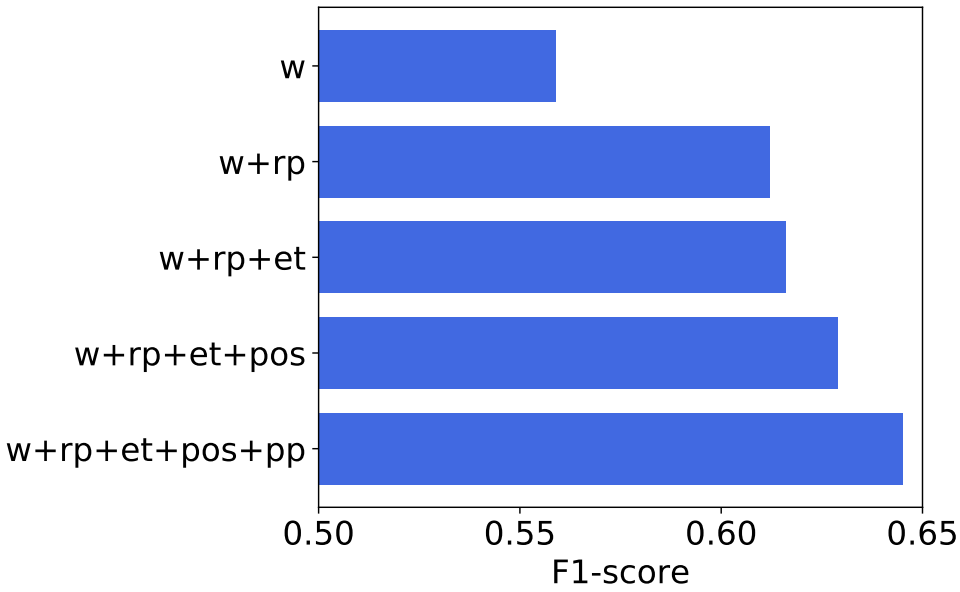


Figure 6-2: Importance of features of CNN and postprocessing rules. w: word embeddings, rp: relative positions to the first and the second arguments, et: entity types, pos: POS tags.

Figure 6-3 quantifies the impact of the two preprocessing steps, deleting brackets and cutting sentences, introduced to compensate for the small dataset size. Cutting the sentence before the first entity and after the second entity resulted in a dramatic impact on the performance, while deleting brackets (i.e., removing the reference marks) improve the performance modestly. This implies that the text between the two entities contains most of the information about the relation between them.

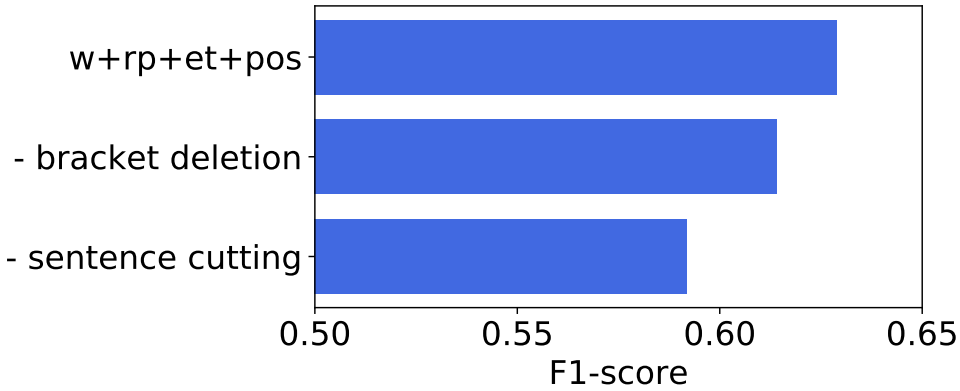


Figure 6-3: Impact of bracket deletion and sentence cutting. "w+rp+et+pos" represents the CNN model trained using all features with both bracket deletion and sentence cutting during preprocessing. "-bracket deletion" is the same model trained only without bracket deletion, and "-sentence cutting" only without sentence cutting.

6.5 Conclusion

In this chapter, we have presented an ANN-based approach to relation extraction, which ranked first in the SemEval-2017 task 10 (ScienceIE) for relation extraction in scientific articles (subtask C). We have experimented with various strategies to incorporate argument ordering for ordering-sensitive relations, showing that an efficient strategy is to fix the arguments ordering as appears on the text by introducing reverse relations. We have also demonstrated that cutting the sentence before the first entity and after the second entity is effective for small datasets.

Chapter 7

Conclusions

7.1 Contributions

This thesis introduced several ANN architectures for information extraction. Most importantly, we have presented the first ANN-based de-identification system, which achieves state-of-the-art results without any manual features. We have also explored several extensions of the system.

Users of the de-identification system may have access to other information in EHRs, such as metadata of patient notes or directories of doctors in the hospital, which can be high-quality features for de-identification. We have extended the ANN architecture to incorporate features, further improving the de-identification performance when high-quality features are utilized.

Another practical issue when performing de-identification may arise from the difficulty of creating labels for the dataset to be de-identified. We have explored transfer learning to take advantage of a large annotated dataset to improve the performance on datasets with a limited number of annotations.

As the ANN automatically learns effective features instead of depending on hand-crafted features, it can also be used for named-entity recognition or other tasks involving sequential tagging such as part-of-speech tagging or chunking. The ANN-based system has been publicly released as an easy-to-use software package for general purpose named-entity recognition as well as de-identification.

Finally, we have presented an ANN architecture for relation extraction, which is the next step toward more advanced information extraction. Our system has ranked first in the SemEval-2017 task 10 (ScienceIE) for relation extraction in scientific articles (subtask C).

7.2 Future work

Though we have shown promising results for de-identification using ANN-based systems, our work leaves several research directions to be explored before it can be mass deployed under practical settings. The main roadblock to the mass deployment is the cost of obtaining enough annotations to ensure satisfactory performance. Our transfer learning experiments have shown that leveraging large annotated datasets may lead to some performance improvement on a target dataset with few annotations. However, even the improved results are not yet on par with the case when there are plenty of annotations on the target dataset.

In order to meet the high standards for de-identification for preserving patients' privacy without spending too many resources for creating annotations, the following research directions may be worth exploring.

Multi-task learning Closely related to transfer learning, multi-task learning attempts to improve the performance on a dataset by training an ANN model on multiple datasets simultaneously with mostly shared parameters and specialized parameters only for the last layers. In a few NLP tasks, multi-task learning for ANNs has been shown to reduce generalization errors, thus resulting in improved performance [9, 24].

Active learning Instead of annotating many training examples from scratch, active learning attempts to annotate as few examples as possible by predicting the most important examples to annotate using some algorithm, based on a small set of existing annotations. Often the annotation step and the prediction step are alternated until desired scores are achieved. In a notable recent work, Zhang et al. [136] presents a novel active learning approach for sentence classification where they select instances that contain words likely to most affect the embeddings.

Data augmentation One of the most naive solutions to the small size of training set for machine learning algorithm is to boost the training data by creating synthetic examples from the existing ones. Previous research has demonstrated that appropriate data augmentation techniques are useful for controlling generalization error for deep learning models. Although this technique is commonly used in image or speech recognition, it has not been explored much for NLP tasks as modifying text cannot be done easily by using a straightforward signal transformation, unlike for image or speech data [135]. Zhang et al. [135] have explored a naive technique for data augmentation in text classification by randomly replacing words in the text by one of their synonyms, showing some improvement.

There are a few interesting research directions towards more advanced information extraction in the medical domain.

Joint extraction of named entities and relations In our ANN models for NER and relation extraction, the NER and relation extraction are done in two separate steps. More specifically, our model for relation extraction assumes that the location of named entities of interest are provided with the text for relation extraction. Many works without this assumption often report significantly decreased performance due to cascading errors from the entity recognition step [97], when relations are extracted from the output of NER. Miwa et al. [83] have explored joint extraction of named entities and relations to improve the performance over the two-step approach.

Clinical timeline extraction Once we have a system for extracting entities and relations jointly, we can apply it to patient notes to extract a clinical timeline. There has been a lot of interest in this domain, propelled by SemEval and i2b2 challenges [13, 12, 112], where the goal is to extract temporal expressions and event expressions, as well as temporal relations between them in patient notes. This task would be useful to support medical practitioners to keep track of patients' clinical history more efficiently, or to perform medical research with information from text in addition to other components of patient records such as lab results.

The above are only few examples of the possible information extraction methods to apply to clinical text, but there are plenty of opportunities where NLP techniques for information extraction could help advance medical research. Once these techniques for information extraction become more reliable, they would be extremely useful for helping medical practitioners or researchers locate and utilize relevant information more efficiently. Furthermore, more advanced information extraction techniques would lead to more accurate predictive modeling for the clinical domain, by providing a way to incorporate information from unstructured elements of EHRs to complement structured elements.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv:1603.04467*, 2016.

[2] John Aberdeen, Samuel Bayer, Reyyan Yeniterzi, Ben Wellner, Cheryl Clark, David Hanauer, Bradley Malin, and Lynette Hirschman. The MITRE Identification Scrubber Toolkit: design, training, and assessment. *International journal of medical informatics*, 79(12):849–859, 2010.

[3] Iman Amini, David Martinez, and Diego Molla. Overview of the ALTA 2012 Shared Task. In *Australasian Language Technology Association Workshop 2012*, volume 7, page 124, 2012.

[4] Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 8–15. Association for Computational Linguistics, 2003.

[5] Isabelle Augenstein, Mrinal Kanti Das, Sebastian Riedel, Lakshmi Nair Vikraman, and Andrew McCallum. SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications. In *Proceedings of the International Workshop on Semantic Evaluation*, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[7] Bruce A Beckwith, Rajeshwarri Mahaadevan, Ulysses J Balis, and Frank Kuo. Development and evaluation of an open source software tool for deidentification of pathology reports. *BMC medical informatics and decision making*, 6(1):1, 2006.

[8] Yassine Benajiba and Paolo Rosso. Arabic named entity recognition using conditional random fields. In *Proc. of Workshop on HLT & NLP within the Arabic World, LREC*, volume 8, pages 143–153, 2008.

[9] Yoshua Bengio et al. Learning deep architectures for AI. *Foundations and Trends® in Machine Learning*, 2(1):1–127, 2009.

[10] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc., 2011.

[11] Jules J Berman. Concept-match medical data scrubbing: how pathology text can be used in research. *Archives of pathology & laboratory medicine*, 127(6):680–686, 2003.

[12] Steven Bethard, Leon Derczynski, Guergana Savova, James Pustejovsky, and Marc Verhagen. SemEval-2015 Task 6: Clinical TempEval. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 806–814, Denver, Colorado, June 2015. Association for Computational Linguistics.

[13] Steven Bethard, Guergana Savova, Wei-Te Chen, Leon Derczynski, James Pustejovsky, and Marc Verhagen. SemEval-2016 Task 12: Clinical TempEval. *Proceedings of SemEval*, pages 1052–1062, 2016.

[14] Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201. Association for Computational Linguistics, 1997.

[15] Phil Blunsom, Edward Grefenstette, Nal Kalchbrenner, et al. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014.

[16] William Boag, Kevin Wacome, Tristan Naumann, and Anna Rumshisky. CliNER: A lightweight tool for clinical named entity recognition. *AMIA Joint Summits on Clinical Research Informatics (poster)*, 2015.

[17] Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. NYU: Description of the MENE named entity system as used in MUC-7. In *In Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998.

[18] Sergey Brin. Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*, pages 172–183. Springer, 1998.

[19] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon's mechanical turk a new source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1):3–5, 2011.

[20] Razvan C Bunescu and Raymond J Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 724–731. Association for Computational Linguistics, 2005.

[21] Jeffrey T Chang, Hinrich Schütze, and Russ B Altman. GAPSCORE: finding gene and protein names one word at a time. *Bioinformatics*, 20(2):216–225, 2004.

[22] Laura Chiticariu, Yunyao Li, and Frederick R Reiss. Rule-based information extraction is dead! long live rule-based information extraction systems! In *EMNLP*, pages 827–832, October 2013.

[23] HC Cho, N Okazaki, M Miwa, and J Tsujii. NERsuite: a named entity recognition toolkit. *Tsujii Laboratory, Department of Information Science, University of Tokyo, Tokyo, Japan*, 2010.

[24] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[25] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics, 2004.

[26] Hamish Cunningham, Yorick Wilks, and Robert J Gaizauskas. GATE: a general architecture for text engineering. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 1057–1060. Association for Computational Linguistics, 1996.

[27] Franck Dernoncourt, Elias Baedorf Kassis, and Mohammad Mahdi Ghassemi. Hyperparameter selection. In *Secondary Analysis of Electronic Health Records*, pages 419–427. Springer International Publishing, 2016.

[28] Franck Dernoncourt and Ji Young Lee. Optimizing neural network hyperparameters with gaussian processes for dialog act classification. *IEEE Spoken Lanuage Technology*, 2016.

[29] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. Neural networks for joint sentence classification in medical paper abstracts. *EACL 2017*, 2016.

[30] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. NeuroNER: an easy-to-use program for named-entity recognition based on neural networks. *arXiv:1705.05487*, 2017.

[31] Franck Dernoncourt, Ji Young Lee, Özlem Uzuner, and Peter Szolovits. De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 2016.

[32] Catherine M DesRoches, Chantal Worzala, and Scott Bates. Some hospitals are falling behind in meeting 'meaningful use' criteria and could be vulnerable to penalties in 2015. *Health Affairs*, 32(8):1355–1360, 2013.

[33] Margaret Douglas, Gari Clifford, Andrew Reisner, George Moody, and Roger Mark. Computer-assisted de-identification of free text in the MIMIC II database. In *Computers in Cardiology, 2004*, pages 341–344. IEEE, 2004.

[34] Margaret Douglass. Computer-assisted de-identification of free-text nursing notes. Master's thesis, Massachusetts Institute of Technology, 2005.

[35] Margaret Douglass, Gari Cliffford, Andrew Reisner, William Long, George Moody, and Roger Mark. De-identification algorithm for free-text nursing notes. In *Computers in Cardiology, 2005*, pages 331–334. IEEE, 2005.

[36] Benjamin G Druss and Steven C Marcus. Growth and decentralization of the medical literature: implications for evidence-based medicine. *Journal of the Medical Library Association*, 93(4):499, 2005.

[37] Elliot M. Fielstein, Steven H. Brown, and Theodore Speroff. Algorithmic de-identification of VA medical exam text for HIPAA privacy compliance: Preliminary findings. *Medinfo*, 1590, 2004.

[38] Michele Filannino, Gavin Brown, and Goran Nenadic. ManTIME: Temporal expression identification and normalization in the TempEval-3 challenge. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 53–57, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics.

[39] Michele Filannino and Goran Nenadic. Temporal expression extraction with extensive feature type selection and a posteriori label adjustment. *Data & Knowledge Engineering*, 100:19–33, 2015.

[40] Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. Annotating named entities in twitter data with crowdsourcing. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88. Association for Computational Linguistics, 2010.

[41] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[42] Jeff Friedlin and Clement J McDonald. A software tool for removing patient identifying information from clinical documents. *Journal of the American Medical Informatics Association*, 15(5):601–610, 2008.

[43] Oana Frunza and Diana Inkpen. Extracting relations between diseases, treatments, and tests from clinical data. In *Canadian Conference on Artificial Intelligence*, pages 140–145. Springer, 2011.

[44] Sebastian Gehrmann, Yeran Li, Eric T. Carlson, Franck Dernoncourt, Joy T. Wu, Jonathan Welt, David W. Grant, Patrick D Tyler, and Leo A. Celi. Comparing rule-based and deep learning approaches to patient phenotyping using clinicians' notes. *arXiv:1703*, 2017.

[45] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[46] Edgar Gonzalez and Jordi Turmo. Unsupervised relation extraction by massive clustering. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 782–787. IEEE, 2009.

[47] Cyril Grouin, Asma Ben Abacha, Delphine Bernhard, Bruno Cartoni, Louise Deleger, Brigitte Grau, Anne-Laure Ligozat, Anne-Lyse Minard, Sophie Rosset, and Pierre Zweigenbaum. CARAMBA: concept, assertion, and relation annotation using machine-learning based approaches. In *i2b2 Medication Extraction Challenge Workshop*, 2010.

[48] Yikun Guo, Robert Gaizauskas, Ian Roberts, George Demetriou, and Mark Hepple. Identifying personal health information using support vector machines. In *i2b2 workshop on challenges in natural language processing for clinical data*, pages 10–11. American Medical Informatics Association, 2006.

[49] Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics, 2005.

[50] Dilip Gupta, Melissa Saul, and John Gilbertson. Evaluation of a deidentification (De-Id) software engine to share pathology reports and clinical documents for research. *American journal of clinical pathology*, 121(2):176–186, 2004.

[51] Kazuo Hara. Applying a SVM based chunker and a text classifier to the deid challenge. In *i2b2 Workshop on challenges in natural language processing for clinical data*, pages 10–11. American Medical Informatics Association, 2006.

[52] Kazuma Hashimoto, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. Simple customization of recursive neural networks for semantic relation classification. In *EMNLP*, pages 1372–1376, 2013.

[53] Hamed Hassanzadeh, Tudor Groza, and Jane Hunter. Identifying scientific artefacts in biomedical literature: The evidence based medicine use case. *Journal of biomedical informatics*, 49:159–170, 2014.

[54] JaWanna Henry, Yuriy Pylypchuk, Talisha Searcy, and Vaishali Patel. Adoption of electronic health record systems among us non-federal acute care hospitals: 2008-2015. *ONC Data Brief*, 35, 2016.

[55] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[56] Arif E Jinha. Article 50 million: an estimate of the number of scholarly articles in existence. *Learned Publishing*, 23(3):258–263, 2010.

[57] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3, 2016.

[58] Su Nam Kim, David Martinez, Lawrence Cavedon, and Lars Yencken. Automatic classification of sentences to support evidence based medicine. *BioMed Central (BMC) Bioinformatics*, 12(2):1, 2011.

[59] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751. Association for Computational Linguistics, 2014.

[60] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[61] N Kumar and Pushpak Bhattacharyya. Named entity recognition in Hindi using MEMM. *Techbical Report, IIT Mumbai*, 2006.

[62] Fabrício SP Kury, Vojtech Huser, and James J Cimino. Reproducing a prospective clinical study as a computational retrospective study in MIMIC-II. In *AMIA Annual Symposium Proceedings*, volume 2015, page 804. American Medical Informatics Association, 2015.

[63] Matthieu Labeau, Kevin Löser, and Alexandre Allauzen. Non-lexical neural architecture for fine-grained POS tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 232–237, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[64] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270, 2016.

[65] Peder Olesen Larsen and Markus Von Ins. The rate of growth in scientific publication and the decline in coverage provided by science citation index. *Scientometrics*, 84(3):575–603, 2010.

[66] Robert Leaman, Graciela Gonzalez, et al. BANNER: an executable survey of advances in biomedical named entity recognition. In *Pacific symposium on biocomputing*, volume 13, pages 652–663, 2008.

[67] Ji Young Lee and Franck Dernoncourt. Sequential short-text classification with recurrent and convolutional neural networks. In *Human Language Technologies 2016: The Conference of the North American Chapter of the Association for Computational Linguistics, NAACL HLT 2016*, 2016.

[68] Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. MIT at SemEval-2017 Task 10: Relation Extraction with Convolutional Neural Networks. In *Proceedings of the International Workshop on Semantic Evaluation*, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[69] Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. Transfer learning for named-entity recognition with neural networks. *arXiv:1705.06273*, 2017.

[70] Ji Young Lee, Franck Dernoncourt, Özlem Uzuner, and Peter Szolovits. Feature-augmented neural networks for patient note de-identification. *COLING Clinical NLP*, 2016.

[71] Qi Li. Literature survey: domain adaptation algorithms for natural language processing. *Department of Computer Science The Graduate Center, The City University of New York*, pages 8–10, 2012.

[72] Qi Li and Heng Ji. Incremental joint extraction of entity mentions and relations. In *ACL (1)*, pages 402–412, 2014.

[73] Zengjian Liu, Yangxin Chen, Buzhou Tang, Xiaolong Wang, Qingcai Chen, Haodi Li, Jingfeng Wang, Qiwen Deng, and Suisong Zhu. Automatic de-identification of electronic medical records using token-level and character-level conditional random fields. *Journal of Biomedical Informatics*, 58:S47–S52, 2015.

[74] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014.

[75] Diana Maynard and Hamish Cunningham. Multilingual adaptations of annie, a reusable information extraction tool. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 219–222. Association for Computational Linguistics, 2003.

[76] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 188–191. Association for Computational Linguistics, 2003.

[77] Stephane M Meystre, F Jeffrey Friedlin, Brett R South, Shuying Shen, and Matthew H Samore. Automatic de-identification of textual documents in the electronic health record: a review of recent research. *BMC medical research methodology*, 10(1):1, 2010.

[78] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013.

[79] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, volume 2, page 3, 2010.

[80] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[81] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751, 2013.

[82] Anne-Lyse Minard, Anne-Laure Ligozat, and Brigitte Grau. Multi-class SVM for relation extraction from clinical reports. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 604–609, Hissar, Bulgaria, September 2011.

[83] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany, August 2016. Association for Computational Linguistics.

[84] Frances P Morrison, Li Li, Albert M Lai, and George Hripcsak. Repurposing the clinical record: can an existing natural language processing system de-identify clinical notes? *Journal of the American Medical Informatics Association*, 16(1):37–39, 2009.

[85] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. How transferable are neural networks in nlp applications? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 479–489, Austin, Texas, November 2016. Association for Computational Linguistics.

[86] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[87] Ishna Neamatullah, Margaret Douglass, H Lehman Li-wei, Andrew Reisner, Mauricio Villarroel, William J Long, Peter Szolovits, George B Moody, Roger G Mark, and Gari D Clifford. Automated de-identification of free-text medical records. *BMC medical informatics and decision making*, 8(1):1, 2008.

[88] Thien Huu Nguyen and Ralph Grishman. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of NAACL-HLT*, pages 39–48, 2015.

[89] HHS Office for Civil Rights. Standards for privacy of individually identifiable health information. final rule. *Federal Register*, 67(157):53181, 2002.

[90] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.

[91] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[92] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword fifth edition, linguistic data consortium. Technical report, Linguistic Data Consortium, Philadelphia, 2011.

[93] Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 78–86, Ann Arbor, Michigan, June 2014. Association for Computational Linguistics.

[94] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[95] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12:1532–1543, 2014.

[96] Deepak Ravichandran and Eduard Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 41–47. Association for Computational Linguistics, 2002.

[97] Bryan Rink, Sanda Harabagiu, and Kirk Roberts. Automatic extraction of relations between medical concepts in clinical texts. *Journal of the American Medical Informatics Association*, 18(5):594–600, 2011.

[98] Patrick Ruch, Robert H Baud, Anne-Marie Rassinoux, Pierrette Bouillon, and Gilbert Robert. Medical document anonymization with a semantic lexicon. In *Proceedings of the AMIA Symposium*, page 729. American Medical Informatics Association, 2000.

[99] Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, Gari Clifford, Li-Wei Lehman, George Moody, Thomas Heldt, Tin H Kyaw, Benjamin Moody, and Roger G Mark. Multiparameter intelligent monitoring in intensive care II (MIMIC-II): a public-access intensive care unit database. *Critical care medicine*, 39(5):952, 2011.

[100] Guergana K Savova, James J Masanz, Philip V Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C Kipper-Schuler, and Christopher G Chute. Mayo clinical text analysis and knowledge extraction system (cTAKES): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

[101] Alexander Schutz and Paul Buitelaar. RelExt: A tool for relation extraction from text in ontology extension. In *International semantic web conference*, volume 2005, pages 593–606. Springer, 2005.

[102] Satoshi Sekine et al. NYU: description of the Japanese NE system used for MET-2. In *Proc. Message Understanding Conference*, 1998.

[103] Burr Settles. ABNER: An open source tool for automatically tagging genes, proteins, and other entity names in text. *Bioinformatics*, 21(14):3191–3192, 2005.

[104] Rion Snow, Daniel Jurafsky, Andrew Y Ng, et al. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*, volume 17, pages 1297–1304, 2004.

[105] Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics, 2012.

[106] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642, 2013.

[107] Brett R South, Danielle Mowery, Ying Suo, Jianwei Leng, Óscar Ferrández, Stephane M Meystre, and Wendy W Chapman. Evaluating the effects of machine pre-annotation and an interactive annotation interface on manual de-identification of clinical text. *Journal of biomedical informatics*, 50:162–172, 2014.

[108] Cosmin Stamate, George D Magoulas, and Michael SC Thomas. Transfer learning approach for financial applications. *arXiv:1509.02807*, 2015.

[109] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107. Association for Computational Linguistics, 2012.

[110] Amber Stubbs, Christopher Kotfila, and Özlem Uzuner. Automated systems for the de-identification of longitudinal clinical narratives: Overview of 2014 i2b2/UTHealth shared task track 1. *Journal of biomedical informatics*, 58:S11–S19, 2015.

[111] Ang Sun and Ralph Grishman. Active learning for relation type extension with local and global data views. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1105–1112. ACM, 2012.

[112] Weiyi Sun, Anna Rumshisky, and Özlem Uzuner. Evaluating temporal relations in clinical text: 2012 i2b2 challenge. *Journal of the American Medical Informatics Association*, 20(5):806–813, 2013.

[113] Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. Translation modeling with bidirectional recurrent neural networks. In *EMNLP*, pages 14–25, 2014.

[114] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128, 2006.

[115] Latanya Sweeney. Replacing personally-identifying information in medical records, the Scrub system. In *Proceedings of the AMIA annual fall symposium*, page 333. American Medical Informatics Association, 1996.

[116] György Szarvas, Richárd Farkas, and András Kocsor. A multilingual named entity recognition system using boosting and c4.5 decision tree learning algorithms. In *Discovery Science*, pages 267–278. Springer, 2006.

[117] Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. Recurrent neural networks for word alignment model. In *ACL (1)*, pages 1470–1480, 2014.

[118] Sean M Thomas, Burke Mamlin, Gunther Schadow, and Clement McDonald. A successful technique for removing names in pathology reports using an augmented search and replace method. In *Proceedings of the AMIA Symposium*, page 777. American Medical Informatics Association, 2002.

[119] Erik F Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics, 2003.

[120] Richard Tzong-Han Tsai, Cheng-Lung Sung, Hong-Jie Dai, Hsieh-Chuan Hung, Ting-Yi Sung, and Wen-Lian Hsu. NERBio: using selected word conjunctions, term normalization, and global patterns to improve biomedical named entity recognition. *BMC bioinformatics*, 7(5):S11, 2006.

[121] Alexander Turchin, Merri L Pendergrass, and Isaac S Kohane. DITTO—a tool for identification of patient cohorts from the text of physician notes in the electronic medical record. In *AMIA*, 2005.

[122] Özlem Uzuner, Jonathan Mailoa, Russell Ryan, and Tawanda Sibanda. Semantic relations for problem-oriented medical records. *Artificial intelligence in medicine*, 50(2):63–73, 2010.

[123] Özlem Uzuner, Tawanda C Sibanda, Yuan Luo, and Peter Szolovits. A de-identifier for medical discharge summaries. *Artificial intelligence in medicine*, 42(1):13–35, 2008.

[124] Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

[125] Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 707–712, Beijing, China, July 2015. Association for Computational Linguistics.

[126] Dong Wang and Thomas Fang Zheng. Transfer learning for speech and language processing. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2015 Asia-Pacific*, pages 1225–1237. IEEE, 2015.

[127] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards AI-complete question answering: A set of prerequisite toy tasks. *ICLR*, 2016.

[128] Adam Wright, Stanislav Henkin, Joshua Feblowitz, Allison B McCoy, David W Bates, and Dean F Sittig. Early results of the meaningful use program for electronic health records. *New England Journal of Medicine*, 368(8):779–780, 2013.

[129] Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. Semantic relation classification via convolutional neural networks with simple negative sampling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 536–540, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[130] Hui Yang and Jonathan M Garibaldi. Automatic detection of protected health information from clinic narratives. *Journal of biomedical informatics*, 58:S30–S38, 2015.

[131] Roman Yangarber and Ralph Grishman. NYU: Description of the Proteus/PET system as used for MUC-7. In *In Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998.

[132] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[133] Godandapani Zayaraz and Suresh Kumara. Concept relation extraction using naïve bayes classifier for ontology-based question answering systems. *Journal of King Saud University-Computer and Information Sciences*, 27(1):13–24, 2015.

[134] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[135] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[136] Ye Zhang, Matthew Lease, and Byron C Wallace. Active discriminative text representation learning. *Thirty-first AAAI Conference on Artificial Intelligence*, 2017.

[137] Guodong Zhou, Min Zhang, Dong-Hong Ji, and Qiaoming Zhu. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *EMNLP-CoNLL*, volume 2007, pages 728–736, 2007.