Sequential Short-Text Classification with Neural Networks

by

Franck Dernoncourt

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

Sequential Short-Text Classification with Neural Networks

by

Franck Dernoncourt

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

Medical practice too often fails to incorporate recent medical advances. The two main reasons are that over 25 million scholarly medical articles have been published, and medical practitioners do not have the time to perform literature reviews. Systematic reviews aim at summarizing published medical evidence, but writing them requires tremendous human efforts. In this thesis, we propose several natural language processing methods based on artificial neural networks to facilitate the completion of systematic reviews. In particular, we focus on short-text classification, to help authors of systematic reviews locate the desired information. We introduce several algorithms to perform sequential short-text classification, which outperform state-of-the-art algorithms. To facilitate the choice of hyperparameters, we present a method based on Gaussian processes. Lastly, we release PubMed 20k RCT, a new dataset for sequential sentence classification in randomized control trial abstracts.

Thesis Supervisor:

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

Thesis Committee:

> Patrick Henry Winston
> Ford Professor of Artificial Intelligence and Computer Science at the Massachusetts Institute of Technology.
>
> Trung H. Bui
> Senior Research Scientist at Adobe Research, San Jose.

Acknowledgments

This thesis would not have been possible without the guidance, encouragement and funding from my advisor, Peter Szolovits. I am especially grateful for the latitude in his research group to explore a wide range of topics and ideas, which we gradually narrowed down to form this work. I have also had the pleasure to work with Anna Rumshisky and Ozlem Uzuner [34, 75]. More generally, my research laboratory, CSAIL Clinical Decision Making Group, was a fruitful environment and I feel lucky to have been surrounded by such great and diverse colleagues: William Boag, Michele Filannino, Marzyeh Ghassemi, Owen Hsu, Nathan Hunt, Mohamed Kane, Lydia Letham, Yuan Luo, Matthew McDermott, Tristan Naumann, Divya Pillai, Harini Suresh, Ziyu Wang. All the work presented in this thesis was performed jointly with Ji Young Lee.

I also thank the MIT Laboratory for Computational Physiology for all the interesting collaborations we have had [27, 92, 82, 99]. In particular, I had a great time working with Leo Celi, Christina Chen, Mohammad Ghassemi, Alistair Johnson, Roger Mark, Tom Pollard, and Felipe Torres. I deeply appreciate the motivations underlying the creation of the MIMIC database, and it is my conviction that research would progress much faster if all laboratories had such a mindset toward open science. MIMIC is an example to take inspiration from.

Outside these two research laboratories, I would like to thank my thesis committee, Trung Bui and Patrick Winston, for their insightful comments and encouragement. I also had some great research collaborations with Samuel Finlayson, Sebastian Gehrmann, Yeran Li, Elias Baedorf Kassis, Edward Moseley, Raymond Sarmiento, Patrick Tyler, Adrian Velasquez, Jonathan Welt, Joy Wu [43, 120]. I spent a summer at Adobe Research, where I had the pleasure to work with Trung Bui, Hung Bui, and Walter Chang [29, 30, 13]. I also had some fruitful collaborations with Una-May O'Reilly and Kalyan Veeramachaneni, who supervised my research during my first two years at MIT [115, 24, 49, 50, 114, 37, 36, 35]. This intellectually simulating research environment helped me for my own side projects [22, 23, 26, 25] as well.

My projects relied heavily on a 1,500-core OpenStack computer cluster as well as 4

GPU servers. I thank the patience of MIT CSAIL technical members Jonathan Proulx and Stephen Jahl for answering all my bug reports and other miscellaneous issues, and the generosity of Quanta Computer and Philips Research, who funded a large part of the hardware. I also warmly thank Steve Ruggiero, Jay Sekora, and Garrett Wollman for their Unix expertise.

I am very grateful for the financial support provided by Philips Research, who funded most of my PhD. In addition to the financial support, I have had the opportunity to work with Philips Research scientists Eric Carlson and Oladimeji Farri on two very interesting projects.

Outside research, I thank Yongwook Bryce Kim for mentoring me throughout the PhD. His advice was always very accurate, and helped me a lot. I also greatly appreciate the support from Osvaldo Jimenez, Alain Mille, and Michael Zock.

Lastly, I thank my family for their support.

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

We investigate in this thesis several natural language processing (NLP) methods to mine information from medical research articles. In this introduction, we outline the motivations of our work.

1.1 Background and Motivation

Over 50 million scholarly articles have been published [57], and the number of articles published every year keeps increasing [39, 71]. Approximately half of them are medical papers indexed by MEDLINE, managed by the U.S. National Library of Medicine. While the abstracts are typically available, the vast majority of papers are not freely accessible as full-text, as shown in Figure 1-1. However, the abstracts still represent an immensely rich dataset.

We focus on medical papers in this thesis, and particularly on randomized controlled trials (RCTs), as they are commonly considered to be the best source of medical evidence (see Figure 1-2). An RCT is a study in which subjects sharing the same medical condition are randomly allocated to one of several clinical interventions. At least one of these clinical interventions is regarded as a control intervention, i.e., an intervention against which it makes sense to compare another intervention. A common control intervention is the treatment that is normally given to patients with the same medical condition.

The number of RCTs published every year is steadily increasing, as shown in Figure 1-

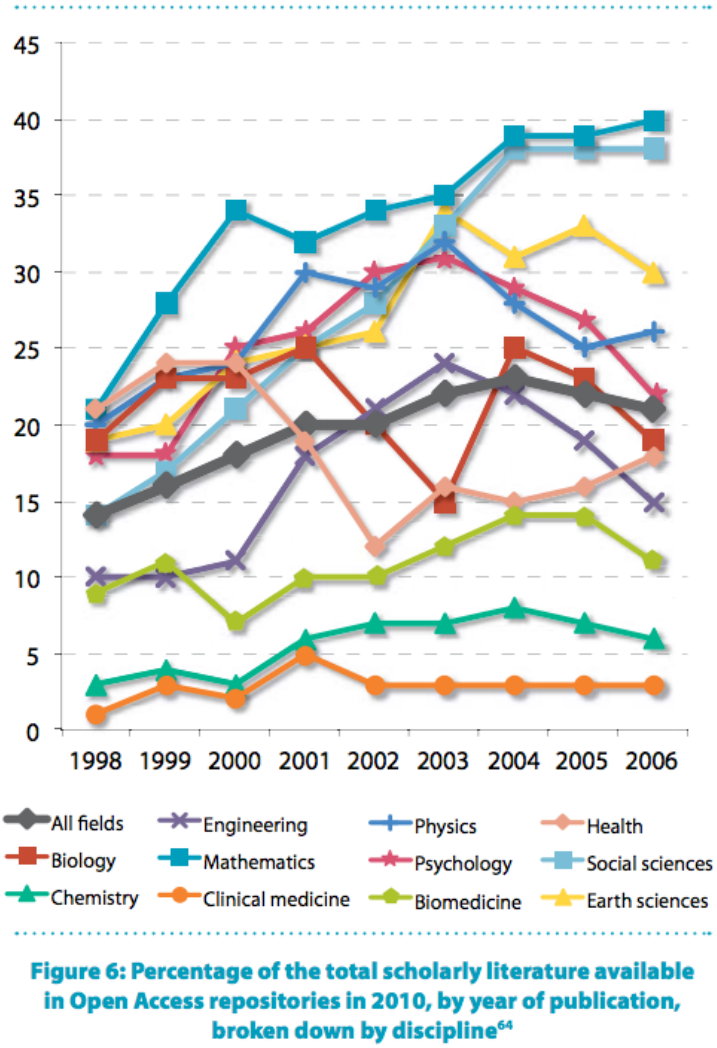


Figure 1-1: Percentage of total scholarly literature available in open access repositories by year of publication broken down by discipline. Source: [110]

3. Despite this growing amount of evidence, evidence-based medicine (EBM) is still very far from having reached its goals. For instance, fewer than half of all the medical treatments delivered today are supported by evidence [97], and there is a lack—at least in the United States—of a clear prioritization of the gaps in medical evidence and an allocation of clinical research resources to efficiently and effectively fill these evidence gaps [42].

One of the main difficulties in finding evidence amongst published RCTs is the sheer number of RCTs: over one million. The average physician does not have time to perform literature reviews: Ely et al. [41] report that the average time that a physician spends searching for a topic is two minutes. Systematic reviews (SRs) aim at summarizing pub-

Figure 1-2: Level of evidence. The higher in the pyramid, the more reliable the evidence is. There exist other sources of medical evidence such as cohort studies, observational studies, case studies, but they are regarded of lower quality evidence than RCTs and SRs. Source: [113]

lished medical evidence, often concentrating on one medical condition and one or several medical interventions, so that physicians can rapidly access a succinct overview for the topics of interest to them. SRs are the only superior type of evidence compared to RCTs. However, SRs are extremely time-consuming to complete [58]: for example, Allen and Olkin [1] report that the median time spent to conduct a clinical SR is 1,139 hours. It typically takes 2.5 to 6.5 years for a primary study publication to be included and published in a new SR [40]. Further, within 2 years of the publication of SRs, 23% are out of date because they have not incorporated new evidence that might change the SR's primary results [101]

We believe that NLP can help organize EBM by facilitating the completion of SRs, and subsequently help physicians provide more adequate care to their patients. When researchers search for previous literature, for example, they often skim through abstracts in order to quickly check whether the papers match the criteria of interest. This process is easier when abstracts are *structured*, i.e., the text in an abstract is divided into semantic headings such as objective, method, result, and conclusion. However, a significant portion of published paper abstracts is *unstructured*, which makes it more difficult to quickly access the information of interest. Consequently, classifying each sentence of an abstract to

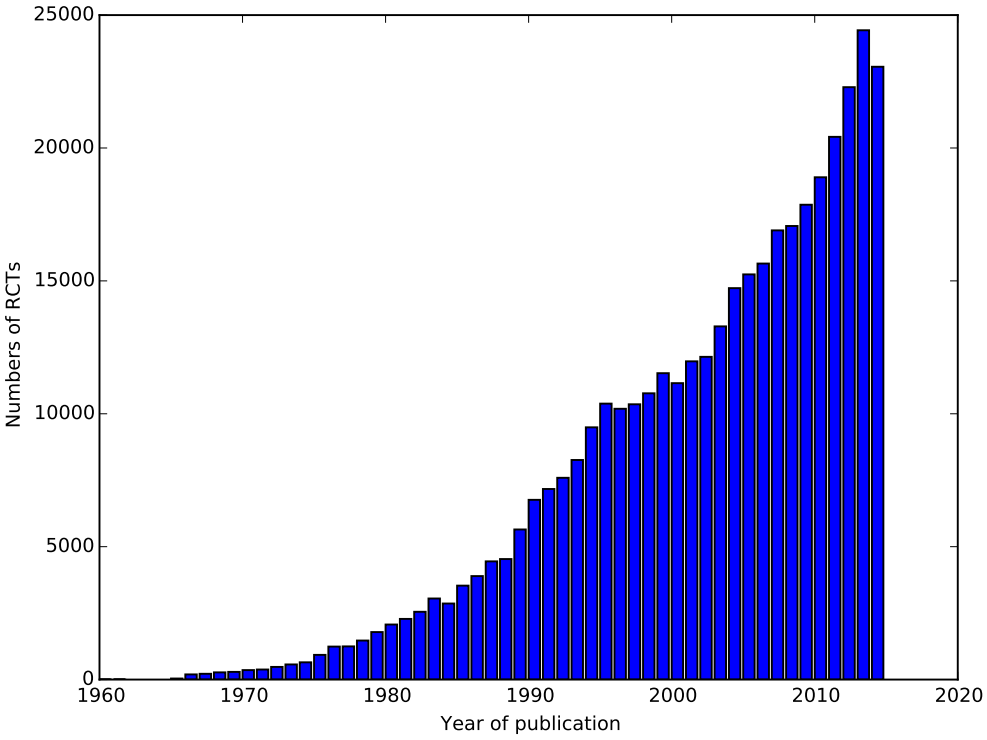


Figure 1-3: Number of RCTs published per year. It is estimated that around 50% of published RCTs are not in MEDLINE [80].

an appropriate heading can significantly reduce time to locate the desired information, as illustrated in Table 1-4. Besides assisting humans, this task may also be useful for a variety of downstream applications such as automatic text summarization, information extraction, and information retrieval.

1.2 Contributions

Inspired by the performance of artificial neural network-based (ANN) systems for non-sequential short-text classification, we introduce several models based on recurrent neural networks (RNNs) and convolutional neural networks (CNNs) for short-text classification. We evaluate them on the sentence classification task in medical research articles, which we introduced in the previous section, and on the dialog act classification task, which we will introduce in Chapter 2.

Short-text classification is an important task in many areas of NLP, including sentiment

Achilles tendinopathy (AT) is a common and difficult to treat musculoskeletal disorder. The purpose of this study is to examine whether 1 injection of platelet-rich plasma (PRP) would improve outcomes more effectively than placebo (saline) after 3 months when used to treat AT. A total of 24 male patients with chronic AT (median disease duration, 33 months) were randomized (1:1) to receive either a blinded injection of PRP (n = 12) or saline (n = 12). Patients were informed that they could drop out after 3 months if they were dissatisfied with the treatment. After 3 months, all patients were reassessed (no dropouts). No difference between the PRP and the saline group could be observed with regard to the primary outcome (VISA-A score: mean difference [MD], -1.3; 95% CI, -17.8 to 15.2; P = .868). Secondary outcomes were pain at rest (MD, 1.6; 95% CI, -0.5 to 3.7; P = .137), pain while walking (MD, 0.8; 95% CI, -1.8 to 3.3; P = .544), pain when tendon was squeezed (MD, 0.3; 95% CI, -0.2 to 0.9; P = .208). PRP injection did not result in an improved VISA-A score over a 3-month period compared with placebo. The only secondary outcome demonstrating a statistically significant difference between the groups was change in tendon thickness; this difference indicates that a PRP injection could increase tendon thickness compared with saline injection. The conclusions are limited to the 3 months after treatment owing to the large dropout rate.

Figure 1-4: Example of abstract with the method section highlighted. Abstracts in the medical field can be long. This abstract was taken from [68] and several sentences have been removed for the sake of conciseness. Providing clinical researchers and practitioners a tool that would allow them to highlight the section(s) that they are interested in would help them to explore the literature more efficiently.

analysis, question answering, or dialog management. Many different approaches have been developed for short-text classification, based on machine learning methods such as logistic regression [44], support Vector machines (SVMs) [103], naive Bayes [117], or random forests [11]. Several recent studies using ANNs have shown promising results, including convolutional neural networks [66, 10, 61] and recursive neural networks [105].

Existing ANN systems classify short texts in isolation, i.e., without considering preceding or succeeding short texts. However, short texts often appear in sequence (e.g., sentences in a document or utterances in a dialog), and therefore using information from preceding or succeeding short texts may improve the classification accuracy. Previous works on *sequential* short-text classification are based on non-ANN approaches, such as Hidden Markov Models (HMMs) [94, 107], maximum entropy [3], naive Bayes [76], and CRF.

The contributions of this thesis are threefold:

- We propose two ANN architectures for sequential short-text classification, which we evaluate on medical and conversational datasets. The first architecture performs for-

ward sequential short-text classification, i.e., it classifies the current short-text based on the current short-text as well as the preceding short-texts. The second architecture performs bidirectional sequential short-text classification, i.e., it classifies the current short-text based on the current short-text as well as the preceding short texts *and* the succeeding short texts. This work was published at NAACL 2016 [72] and EACL 2017 [31].

- We explore a hyperparameter optimization strategy for ANN based on Gaussian Processes. This work was published at IEEE SLT 2016 [28].
- We introduce PubMed 20k RCT, a new dataset for sequential sentence classification in medical abstracts. This work is under submission [33].

1.3 Organization

The rest of this thesis is organized as follows:

- Chapter 2 presents our work on forward sequential short-text classification.
- Chapter 3 presents our work on bidirectional sequential short-text classification and introduces a new data set, PubMed 20k RCT.
- Chapter 4 presents our work on ANN hyperparameter optimization.
- Chapter 5 draws conclusions from the work and discusses fruitful avenues for further research.

Chapter 2

Forward Sequential Short-Text Classification

As we mentioned in the introduction, recent approaches based on artificial neural networks (ANNs) have shown promising results for short-text classification. However, many short texts occur in sequences (e.g., sentences in a document or utterances in a dialog), and most existing ANN-based systems do not leverage the preceding short texts when classifying a subsequent one. In this chapter, we present a model based on recurrent neural networks and convolutional neural networks that incorporates the preceding short texts. We call this task forward sequential short-text classification. Our model outperforms state-of-the-art results on three datasets for dialog act prediction.

A dialog act characterizes an utterance in a dialog based on a combination of pragmatic, semantic, and syntactic criteria. Its accurate detection is useful for a range of applications, from speech recognition to automatic summarization [107]. Previous works on sequential short-text classification are mostly based on non-ANN approaches, such as Hidden Markov Models (HMMs) [94, 107, 109], maximum entropy [3], naive Bayes [76], and conditional random fields (CRFs) [64, 93].

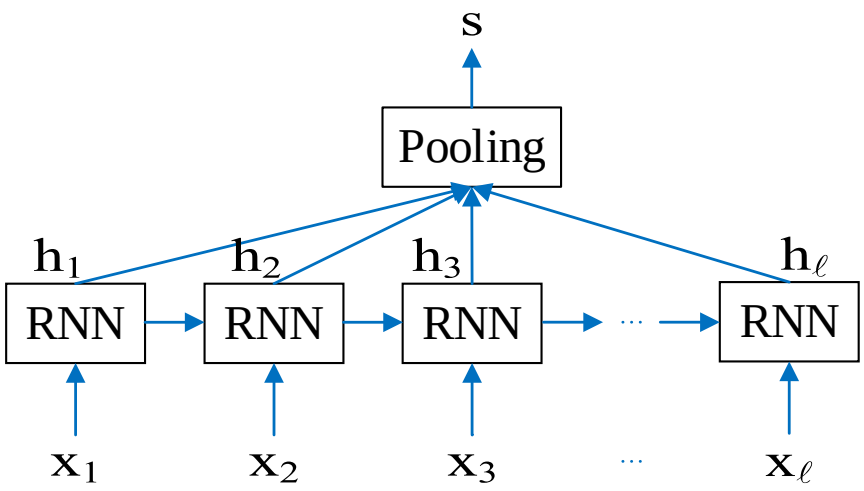


Figure 2-1: RNN architectures for generating the vector representation $\mathbf{s}$ of a short text $\mathbf{x}_{1:\ell}$.

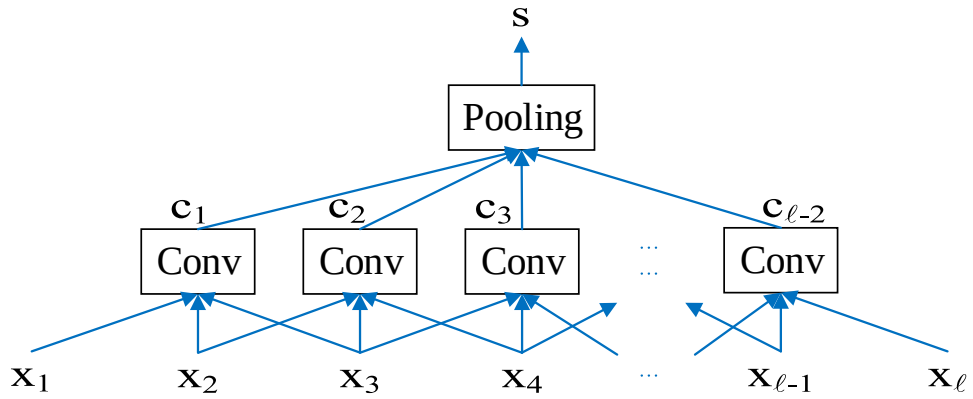


Figure 2-2: CNN architectures for generating the vector representation $\mathbf{s}$ of a short text $\mathbf{x}_{1:\ell}$. Conv refers to convolution operations, and the filter height $h = 3$ is used in this figure.

2.1 Model

Our model comprises two parts. The first part generates a vector representation for each short text using either the RNN or CNN architecture, as discussed in Section 2.1.1 and Figure 2-1 as well as Figure 2-2. The second part classifies the current short text based on the vector representations of the current as well as a few preceding short texts, as presented in Section 4.2.1 and Figure 2-3.

We denote scalars with italic lowercase symbols (e.g., k, b_f), vectors with bold lowercase symbols (e.g., $\mathbf{s}$, $\mathbf{x}_i$), and matrices with italic uppercase symbols (e.g., W_f). We use the colon notation $\mathbf{v}_{i:j}$ to denote the sequence of vectors $(\mathbf{v}_i, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_j)$. We will use

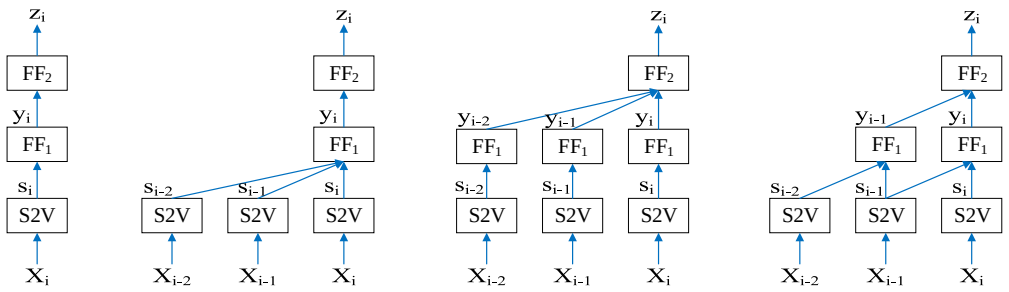


Figure 2-3: Four instances of the two-layer feedforward ANN used for predicting the probability distribution over the classes $\mathbf{z}_i$ for the i^{th} short-text $\mathbf{X}_i$. S2V stands for short text to vector, which is the RNN/CNN architecture that generates $\mathbf{s}_i$ from $\mathbf{X}_i$. From left to right, the history sizes (d_1, d_2) are $(0,0), (2,0), (0,2)$ and $(1,1)$. $(0,0)$ corresponds to the non-sequential classification case.

these notations in the rest of the thesis.

2.1.1 Short-text representation

A given short text of length ℓ is represented as the sequence of m-dimensional word vectors $\mathbf{x}_{1:\ell}$, which is used by the RNN or CNN model to produce the n-dimensional *short-text representation* $\mathbf{s}$.

RNN-based short-text representation

We use a variant of RNN called Long Short Term Memory (LSTM) [52]. For the t^{th} word in the short-text, an LSTM takes as input $\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}$ and produces $\mathbf{h}_t, \mathbf{c}_t$ based on the following formulas:

$$\mathbf{i}_t = \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i)$$

$$\mathbf{f}_t = \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f)$$

$$\tilde{\mathbf{c}}_t = \tanh(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t$$

$$\mathbf{o}_t = \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t)$$

where $W_j \in \mathbb{R}^{n \times m}$, $U_j \in \mathbb{R}^{n \times n}$ are weight matrices and $\mathbf{b}_j \in \mathbb{R}^n$ are bias vectors, for $j \in \{i, f, c, o\}$. The symbols $\sigma(\cdot)$ and $\tanh(\cdot)$ refer to the element-wise sigmoid and

hyperbolic tangent functions, and $\odot$ is the element-wise multiplication. $\mathbf{h}_0 = \mathbf{c}_0 = \mathbf{0}$.

In the pooling layer, the sequence of vectors $\mathbf{h}_{1:\ell}$ output from the RNN layer are combined into a single vector $\mathbf{s} \in \mathbb{R}^n$ that represents the short-text, using one of the following mechanisms: last, mean, and max pooling. Last pooling takes the last vector, i.e., $\mathbf{s} = \mathbf{h}_\ell$, mean pooling averages all vectors, i.e., $\mathbf{s} = \frac{1}{\ell}\sum_{t=1}^{\ell} \mathbf{h}_t$, and max pooling takes the element-wise maximum of $\mathbf{h}_{1:\ell}$.

CNN-based short-text representation

Using a *filter* $W_f \in \mathbb{R}^{h \times m}$ of height h, a convolution operation on h consecutive word vectors starting from t^{th} word outputs the scalar *feature*

$$c_t = \text{ReLU}(W_f \bullet X_{t:t+h-1} + b_f)$$

where $X_{t:t+h-1} \in \mathbb{R}^{h \times m}$ is the matrix whose i^{th} row is $\mathbf{x}_i \in \mathbb{R}^m$, and $b_f \in \mathbb{R}$ is a bias. The symbol $\bullet$ refers to the dot product and $\text{ReLU}(\cdot)$ is the element-wise rectified linear unit function.

We perform convolution operations with n different filters, and denote the resulting features as $\mathbf{c}_t \in \mathbb{R}^n$, each of whose dimensions comes from a distinct filter. Repeating the convolution operations for each window of h consecutive words in the short-text, we obtain $\mathbf{c}_{1:\ell-h+1}$. The short-text representation $\mathbf{s} \in \mathbb{R}^n$ is computed in the max pooling layer, as the element-wise maximum of $\mathbf{c}_{1:\ell-h+1}$. (We tried using mean-pooling instead of max-pooling, but it yielded lower performance).

2.1.2 Sequential short-text classification

Let $\mathbf{s}_i$ be the n-dimensional short-text representation given by the RNN or CNN architecture for the i^{th} short text in the sequence. The sequence $\mathbf{s}_{i-d_1-d_2:i}$ is fed into a two-layer feedforward ANN that predicts the class for the i^{th} short text. The hyperparameters d_1, d_2 are the history sizes used in the first and second layers, respectively.

The first layer takes as input $\mathbf{s}_{i-d_1-d_2:i}$ and outputs the sequence $\mathbf{y}_{i-d_2:i}$ defined as

$$\mathbf{y}_j = \tanh\left(\sum_{d=0}^{d_1} W_{-d}\,\mathbf{s}_{j-d} + \mathbf{b}_1\right),\ \forall j \in [i-d_2, i]$$

where $W_0, W_{-1}, W_{-d_1} \in \mathbb{R}^{k\times n}$ are the weight matrices, $\mathbf{b}_1 \in \mathbb{R}^k$ is the bias vector, $\mathbf{y}_j \in \mathbb{R}^k$ is the *class representation*, and k is the number of classes for the classification task.

Similarly, the second layer takes as input the sequence of class representations $\mathbf{y}_{i-d_2:i}$ and outputs $\mathbf{z}_i \in \mathbb{R}^k$:

$$\mathbf{z}_i = \text{softmax}\left(\sum_{j=0}^{d_2} U_{-j}\,\mathbf{y}_{i-j} + \mathbf{b}_2\right)$$

where $U_0, U_{-1}, U_{-d_2} \in \mathbb{R}^{k\times k}$ and $\mathbf{b}_2 \in \mathbb{R}^k$ are the weight matrices and bias vector.

The final output $\mathbf{z}_i$ represents the probability distribution over the set of k classes for the i^{th} short-text: the j^{th} element of $\mathbf{z}_i$ corresponds to the probability that the i^{th} short-text belongs to the j^{th} class.

2.2 Datasets and Experimental Setup

2.2.1 Datasets

We evaluate our model on the dialog act classification task using the following datasets:

- DSTC 4: Dialog State Tracking Challenge 4 [62, 63].
- MRDA: ICSI Meeting Recorder Dialog Act Corpus [55, 102]. The 5 classes are introduced in [3].
- SwDA: Switchboard Dialog Act Corpus [60].

For MRDA, we use the train/validation/test splits provided with the datasets. For DSTC 4 and SwDA, only the train/test splits are provided.[1] Table 2.1 presents statistics on the datasets.

[1] All train/validation/test splits can be found at https://github.com/Franck-Dernoncourt/naacl2016

Dataset	$\|C\|$	$\|V\|$	Train	Validation	Test
DSTC 4	89	6k	24 (21k)	5 (5k)	6 (6k)
MRDA	5	12k	51 (78k)	11 (16k)	11 (15k)
SwDA	43	20k	1003 (193k)	112 (23k)	19 (5k)

Table 2.1: Overview of the datasets for dialogue act classification. $|C|$ is the number of classes, $|V|$ the vocabulary size. For the train, validation and test sets, we indicate the number of dialogs (i.e., sequences) followed by the number of utterances (i.e., short texts) in parenthesis.

2.2.2 Training

The model is trained to minimize the negative log-likelihood of predicting the correct dialog acts of the utterances in the train set, using stochastic gradient descent with the Adadelta update rule [123]. At each gradient descent step, weight matrices, bias vectors, and word vectors are updated. For regularization, dropout is applied after the pooling layer, and early stopping is used on the validation set with a patience of 10 epochs.

2.3 Results and Discussion

To find effective hyperparameters, we varied one hyperparameter at a time while keeping the other ones fixed. Table 2.2 presents our hyperparameter choices.

Hyperparameter	Choice	Experiment Range
LSTM output dim. (n)	100	50 – 1000
LSTM pooling	max	max, mean, last
LSTM direction	unidir.	unidir., bidir.
CNN num. of filters (n)	500	50 – 1000
CNN filter height (h)	3	1 – 10
Dropout rate	0.5	0 – 1
Word vector dim. (m)	200, 300	25 – 300

Table 2.2: Experiment ranges and choices of hyperparameters. Unidir refers to the regular RNNs presented in Section 2.1.1, and bidir refers to bidirectional RNNs introduced in [100].

	d_2	LSTM		
d_1		0	1	2
DSTC4	0	63.1 (62.4, 63.6)	65.7 (65.6, 65.7)	64.7 (63.9, 65.3)
	1	**65.8** (65.5, 66.1)	65.7 (65.3, 66.1)	64.8 (64.6, 65.1)
	2	65.7 (65.0, 66.2)	65.5 (64.4, 66.1)	64.9 (64.6, 65.2)
MRDA	0	82.8 (82.4, 83.1)	83.2 (82.9, 83.4)	82.9 (82.4, 83.4)
	1	83.2 (82.6, 83.7)	83.8 (83.5, 84.4)	83.6 (83.2, 83.8)
	2	**84.1** (83.5, 84.4)	83.9 (83.4, 84.7)	83.3 (82.6, 84.2)
SwDA	0	66.3 (65.1, 68.0)	67.9 (66.3, 68.6)	67.8 (66.7, 69.0)
	1	68.4 (67.8, 68.8)	67.8 (65.5, 68.9)	67.3 (65.5, 69.5)
	2	**69.5** (68.9, 70.2)	67.9 (66.5, 69.4)	67.7 (66.9, 68.9)

Table 2.3: Accuracy (%) on different architectures and history sizes d_1, d_2. For each setting, we report average (minimum, maximum) computed on 5 runs. Sequential classification ($d_1 + d_2 > 0$) outperforms non-sequential classification ($d_1 = d_2 = 0$). We also tried gated recurrent units (GRUs) [15] and the basic RNN, but the results were generally lower than LSTM. The numbers reported in bold correspond to the largest values for each dataset.

We initialized the word vectors with the 300-dimensional word vectors pretrained with word2vec on Google News [83, 86] for DSTC 4, and the 200-dimensional word vectors pretrained with GloVe on Twitter [91] for MRDA and SwDA, as these choices yielded the best results among all publicly available word2vec, GloVe, SENNA [17, 18] and RNNLM [85] word vectors.

The effects of the history sizes d_1 and d_2 for the short-text and the class representations, respectively, are presented in Tables 2.3 and 2.4 for both the LSTM and CNN models. In both models, increasing d_1 while keeping $d_2 = 0$ improved their performance by 1.3-4.2 percentage points. Conversely, increasing d_2 while keeping $d_1 = 0$ yielded better results, but the performance increase was less pronounced: incorporating sequential information at the short-text representation level was more effective than at the class representation level.

Using sequential information at both the short-text representation level and the class representation level does not help in most cases and may even lower performance. We hypothesize that short-text representations contain richer and more general information than class representations due to their larger dimension. Class representations may not convey any additional information over short-text representations, and are more likely to propagate errors from previous misclassifications.

Table 2.5 compares our results with the state-of-the-art. Overall, our model shows competitive results, while requiring no human-engineered features. Rigorous comparisons

d_1 \ d_2		CNN		
		0	1	2
DSTC4	0	64.1 (63.5, 65.2)	65.4 (64.7, 66.6)	65.1 (63.2, 65.9)
	1	65.3 (64.1, 65.9)	65.1 (62.1, 66.2)	64.9 (64.4, 65.6)
	2	65.7 (64.9, 66.3)	**65.8** (65.2, 66.1)	65.4 (64.5, 66.0)
MRDA	0	83.2 (83.0, 83.4)	83.5 (82.9, 84.0)	83.8 (83.4, 84.2)
	1	**84.6** (84.5, 84.9)	**84.6** (84.4, 84.8)	84.1 (83.8, 84.4)
	2	84.4 (84.1, 84.8)	**84.6** (84.5, 84.7)	84.4 (84.2, 84.7)
SwDA	0	67.0 (65.3, 68.7)	69.1 (68.5, 70.0)	69.7 (69.2, 70.9)
	1	69.9 (69.1, 70.9)	69.8 (69.3, 70.6)	69.9 (68.8, 70.6)
	2	**71.4** (70.4, 73.1)	71.1 (70.2, 72.1)	70.9 (69.7, 71.7)

Table 2.4: Accuracy (%) on different architectures and history sizes d_1, d_2. For each setting, we report average (minimum, maximum) computed on 5 runs. Sequential classification ($d_1 + d_2 > 0$) outperforms non-sequential classification ($d_1 = d_2 = 0$). The numbers reported in bold correspond to the largest values for each dataset.

are challenging to draw, as many important details such as text preprocessing and train/-valid/test split may vary, and many studies fail to perform several runs despite the randomness in some parts of the training process, such as weight initialization.

Model	DSTC 4	MRDA	SwDA
CNN	65.5	**84.6**	**73.1**
LSTM	**66.2**	84.3	69.6
Majority class	25.8	59.1	33.7
SVM	57.0	–	–
Graphical model	–	81.3	–
Naive Bayes	–	82.0	–
HMM	–	–	71.0
Memory-based Learning	–	–	72.3

Table 2.5: Accuracy (%) of our models and other methods from the literature. The majority class model predicts the most frequent class. SVM: [29]. Graphical model: [56]. Naive Bayes: [76]. HMM: [107]. Memory-based Learning: [96]. All five models use features derived from transcribed words, as well as previous predicted dialog acts except for Naive Bayes. For SwDA, the Cohen's kappa coefficient for the interlabeler agreement is 0.84. The interlabeler agreement could not be obtained for MRDA, and DSTC 4 was labeled by a single annotator. For the CNN and LSTM models, the presented results are the test set accuracy of the run with the highest accuracy on the validation set. The numbers reported in bold correspond to the largest values for each dataset.

2.4 Conclusion

In this chapter we have presented an ANN-based approach to sequential short-text classification. We demonstrate that adding sequential information improves the quality of the predictions, and the performance depends on what sequential information is used in the model. Our model achieves state-of-the-art results on three different datasets for dialog act prediction.

Chapter 3

Bidirectional Sequential Short-Text Classification

In the previous chapter, we have explored new ANN-based models for forward sequential short-text classification, i.e., ANN classifiers that use the preceding short texts. They have two downsides:

- They purposedly do not use the succeeding short texts. This is convenient if the application is real-time, such as a live dialogue, but in many cases the entire sequence of short-texts can already be accessed when the classification starts.
- They classify one short texts at the time, in contrast to classifying all short texts of the sequence at once, i.e., performing structured prediction.

In this chapter, we propose a model that remediates these two issues. Our model outperforms the state-of-the-art results on two different datasets for the task of sequential sentence classification in medical abstracts, which we have introduced and motivated in the introduction of this thesis.

Our model makes use of both token and character embeddings for classifying sentences, and has a sequence optimization layer that is learned jointly with other components of the model. We evaluate our model on the NICTA-PIBOSO dataset as well as a new dataset we compiled based on the PubMed database.

3.1 Related Work

Existing systems for sequential sentence classification are mostly based on naive Bayes [98, 53], support vector machine [81, 122, 51, 122], Hidden Markov models [77], and conditional random fields (CRFs) [65, 48, 51]. They often require numerous hand-engineered features based on lexical (bag-of-words, n-grams, dictionaries, cue words), semantic (synonyms, hyponyms), structural (part-of-speech tags, headings), and sequential (sentence position, surrounding features) information.

On the other hand, recent approaches to natural language processing (NLP) based on artificial neural networks (ANNs) do not require manual features, as they are trained to automatically learn features based on word as well as character embeddings. Moreover, ANN-based models have achieved state-of-the-art results on various NLP tasks, including the most relevant task of text classification [106, 66, 61, 124, 19, 121, 38]. For text classification, many ANN models use word embeddings [106, 66, 61, 43], and most recent works are based on character embeddings [124, 19, 121]. Approaches combining word and character embeddings have also been explored [38, 34].

However, most existing works using ANNs for short-text classification do not use any context. This is in contrast with sequential sentence classification, where each sentence in a text is classified taking into account its context, i.e., the surrounding sentences and possibly the whole text. One exception is our work on dialog act classification that we introduced in the previous chapter, where each utterance in a dialog is classified into its dialog act. However, only the preceding utterances were used, as the system was designed with real-time applications in mind.

3.2 Model

3.2.1 ANN model

Our ANN model consists of three components: a hybrid token embedding layer, a sentence label prediction layer, and a label sequence optimization layer. Figure 3-1 presents a graphical overview of the model.

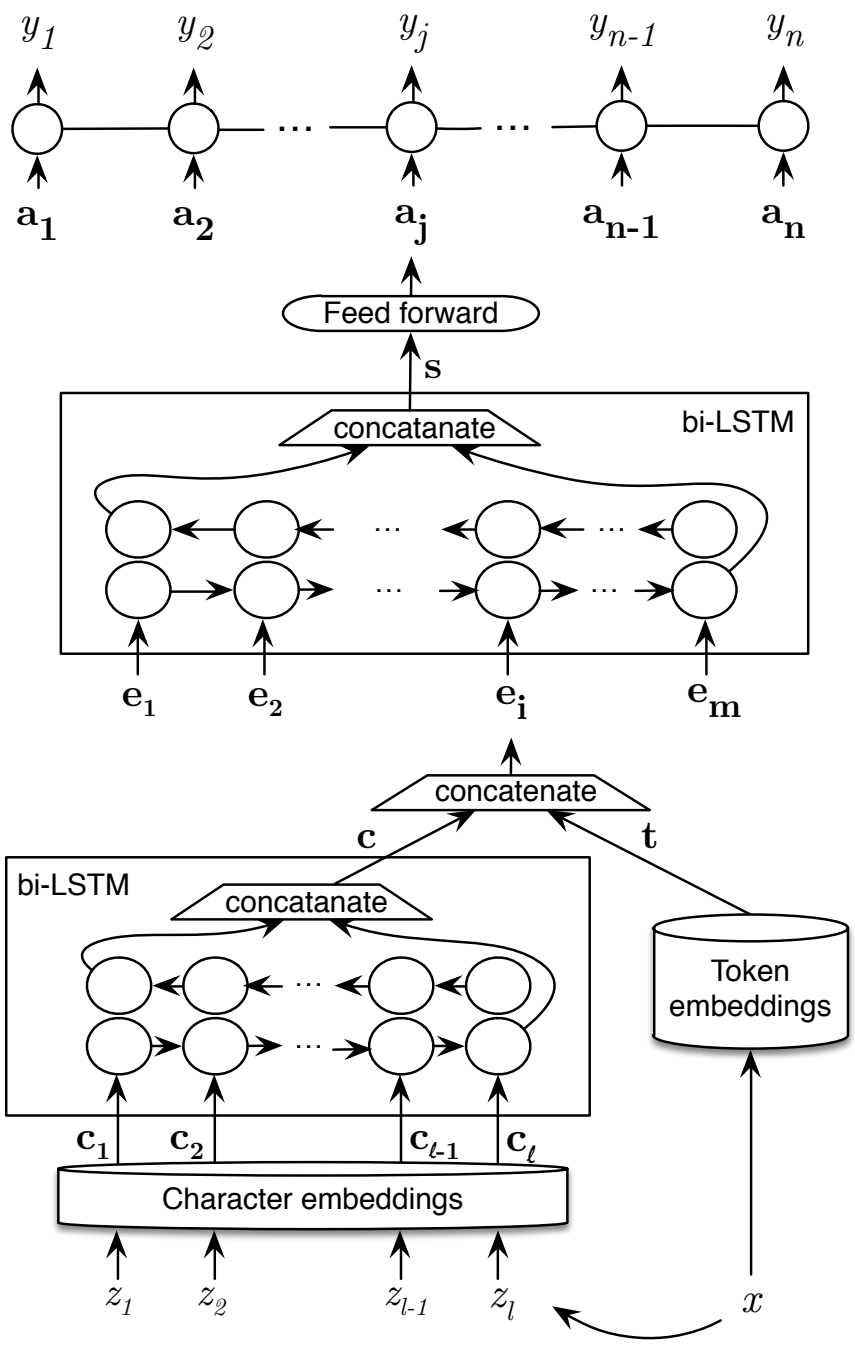


Figure 3-1: ANN model for sequential sentence classification. x: token, $\mathbf{t}$: token embeddings (300), z_i: i^{th} character of x, $\mathbf{c}_i$: character embeddings (25), $\mathbf{c}$: character-based token embeddings (50), $\mathbf{e}_i$: hybrid token embeddings (350), $\mathbf{s}$: sentence vector (200), $\mathbf{a}_j$: sentence label vector (number of classes), y_j: sentence label. The numbers in parenthesis indicate the dimension of the vectors. Token embeddings are initialized with GloVe [91] embeddings pretrained on Wikipedia and Gigaword 5 [89]. Replacing LSTMs with convolutional neural networks did not improve the results: we therefore use LSTMs.

Hybrid token embedding layer

The hybrid token embedding layer takes a token as an input and outputs its vector representation utilizing both the token embeddings and as well as the character embeddings.

Token embeddings are a direct mapping $\mathcal{V}_T(\cdot)$ from token to vector, which can be pre-trained on large unlabeled datasets using programs such as word2vec [86, 83, 87] or GloVe [91]. Character embeddings are also defined in an analogous manner, as a direct mapping $\mathcal{V}_C(\cdot)$ from character to vector.

Let $z_{1:\ell}$ be the sequence of characters that comprise a token x. Each character z_i is first mapped to its embedding $\mathbf{c}_i = \mathcal{V}_C(z_i)$, and the resulting sequence $\mathbf{c}_{1:\ell}$ is input to a bidirectional LSTM, which outputs the character-based token embedding $\mathbf{c}$.

The output $\mathbf{e}$ of the hybrid token embedding layer for the token x is the concatenation of the character-based token embedding $\mathbf{c}$ and the token embedding $\mathbf{t} = \mathcal{V}_T(x)$. Using characters as input to the ANN has been explored by other works such as [67].

Sentence label prediction layer

Let $x_{1:m}$ be the sequence of tokens in a given sentence, and $\mathbf{e}_{1:m}$ be the corresponding embedding output from the hybrid token embedding layer. The sentence label prediction layer takes as input the sequence of vectors $\mathbf{e}_{1:m}$, and outputs $\mathbf{a}$, where the k^{th} element of $\mathbf{a}$, denoted $\mathbf{a}[k]$, reflects the probability that the given sentence has label k.

To achieve this, the sequence $\mathbf{e}_{1:m}$ is first input to a bidirectional LSTM, which outputs the vector representation $\mathbf{s}$ of the given sentence. The vector $\mathbf{s}$ is subsequently input to a feedforward neural network with one hidden layer, which outputs the corresponding probability vector $\mathbf{a}$.

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.

In order to model dependencies between subsequent labels, we incorporate a matrix T

that contains the transition probabilities between two subsequent labels; we define $T[i, j]$ as 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:

$$p(\hat{y}_{1:n}) = \frac{e^{s(\hat{y}_{1:n})}}{\sum_{y_{1:n} \in Y^n} e^{s(y_{1:n})}}$$

with Y being the set of all possible labels. 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.

Computing the denominator $\sum_{y \in Y^n} e^{s(y_{1:n})}$ can be done in $O(n|C|^2)$ time using dynamic programming (where $|C|$ denotes the number of classes), as demonstrated below. Let $A_{(n,y_n)}$ be the log of the sum of the scores of all the sequence of length n the last label of which is y_n. Then:

$$\begin{aligned}
A_{(n,y_n)} &\stackrel{\text{def.}}{=} \log \left(\sum_{y_{1:(n-1)} \in Y^{n-1}} e^{s(y_{1:n})} \right) \\
&= \log \left(\sum_{y_{1:(n-1)} \in Y^{n-1}} e^{s\left(y_{1:(n-1)}\right) + T(y_{n-1}, y_n) + a_n(y_n)} \right) \\
&= \log \left(\sum_{y_{n-1} \in Y} \left(\sum_{y_{1:(n-2)} \in Y^{n-2}} e^{s\left(y_{1:(n-1)}\right)} \right) e^{T(y_{n-1}, y_n) + a_n(y_n)} \right) \\
&= \log \left(\sum_{y_{n-1} \in Y} e^{A_{(n-1,y_{n-1})}} e^{T(y_{n-1}, y_n) + a_n(y_n)} \right)
\end{aligned}$$

Since $A_{(n,y_n)}$ can be computed in $\Theta(|C|)$ time given $\left\{ A_{(n-1,y_{n-1})} | y_{n-1} \in Y \right\}$, comput-

Dataset	Size	Labels	Manual	RCT	Available
[46]	200	PI	y	y	email
[16]	327	IComp	y	y	no
[12]	28631	POIcomp	n	n	no
[65]	1000	PIBOSObj	y	n	email
[54]	23472	PIO	n	n	no
[95]	1356	Poe	n	y	no
[125]	19893	PIOSObj	y	n	no
[20]	194	PICO	n	y	public[1]
[53]	19854	PIO	n	y	no
PubMed 20k RCT	20000	BObjSOC	n	y	no

Table 3.1: Overview of existing datasets for sentence classification in medical abstracts. The size is expressed in terms of number of abstracts. The "labels" column uses the following abbreviations: B: background; C: conclusion; Comp: comparison; I: intervention; Icomp: intervention and comparison; O: outcome; Obj: objective; P: population; Poe: patient-oriented outcome; S: study design (a.k.a. method) In the "manual" column, "y" means that the dataset was manually annotated, "n" otherwise. In the RCT column, "y" means that the dataset only contains RCTs, "n" otherwise. For [12], the size was inferred as follows: the paper indicates that the training set contains 28631 abstracts and that it represents 90% of the data, which means there are $25768/0.9 = 28631$ abstracts in total.

ing $\left\{A_{(n,y_n)}|y_n \in Y\right\}$ takes $\Theta(|C|^2)$ time given $\left\{A_{(n-1,y_{n-1})}|y_{n-1} \in Y\right\}$. Consequently, computing $\left\{A_{(n,y_n)}|y_n \in Y\right\}$ takes $O(n|C|^2)$ time.

3.3 PubMed 20k RCT

In this section, we present PubMed 20k RCT, a new dataset based on PubMed for sequential sentence classification. The dataset consists of 20,000 abstracts of randomized controlled trials (RCTs). Each sentence of each abstract is annotated with their role in the abstract using one of the following classes: background, objectives, methods, results, or conclusion. The dataset is freely available at https://github.com/Franck-Dernoncourt/pubmed-rct.

The purpose of releasing this dataset is two-fold. First, the majority of datasets for sequential short-text classification are small: we hope that releasing a new large dataset will help develop more accurate algorithms for that task. Second, from an application perspective as we have mentioned before, clinical researchers and practitioners need better tools to efficiently skim through the medical literature.

3.3.1 Existing Datasets

Existing datasets for classifying sentences in medical abstracts are either small, not publicly available, or do not focus on RCTs. Table 3.2 presents an overview of existing datasets.

The most studied dataset to our knowledge is the NICTA-PIBOSO corpus published in [65]. This dataset was the basis of the ALTA 2012 Shared Task [2], in which 8 competing research teams participated to build the most accurate classifier.

Only the dataset published in [20] is publicly available: half of the datasets can only be obtained via email inquiries, and the other half are not accessible (unanswered email requests or negative replies). The only public dataset is also the smallest one.

3.3.2 Dataset Construction

Abstract Selection

Our dataset is constructed upon the MEDLINE/PubMed Baseline Database published in 2016[2], which we will refer to as PubMed in this paper. PubMed is managed by the United States National Library of Medicine (NLM) at the National Institutes of Health. It can be accessed online by anyone, free of charge and without having to go through any registration. It contains 24,358,442 records. A record typically consists of metadata on one article, as well as the article's title and in many cases its abstract. Metadata information may include the authors' names, the authors' affiliations, and more when available.

We use the following information from each PubMed record to build our dataset: the article's PubMed ID (PMID), the article's abstract along with the abstract's structure if available, and the article's Medical Subject Headings (MeSH) terms. MeSH is the NLM controlled vocabulary thesaurus used for indexing articles for PubMed.

We select abstracts from PubMed based on the two following criteria:

- the abstract has to belong to an RCT. We rely on the article's MeSH terms only to select RCTs. Specifically, the MeSH term "D016449" corresponds to an RCT: if an article does not have the MeSH term D016449, then its abstract is not included in our

[2] nlm.nih.gov/databases/download/pubmed_medline.html (mirror)

dataset. 399,254 abstracts fit this criterion.

- the abstract has to be structured. In order to qualify as structured, it has to contain between 3 and 9 sections (inclusive), and it should not contain any section labeled as "None", "Unassigned", or "" (empty string). Only 0.5% of abstracts have fewer than 3 sections or more than 9 sections: we chose to discard these outliers. The label of each section was originally given by the authors of the articles, typically following the guidelines given by journals: as many labels exist, PubMed maps them into a smaller set of standardized labels: background, objective, methods, results, conclusions, "None", "Unassigned", or "" (empty string).

195,654 abstracts fit these two criteria, i.e., are both structured and belong to an RCT. We choose 20k abstracts from them by taking the abstracts with the highest PMIDs, which is a proxy for the publication date (in most cases, the higher the PMID, the more recently published the article is).

Dataset Split

The dataset contains 20k abstracts and is randomly split into three sets: a training set containing 15k abstracts, a validation set containing 2500 abstracts, a test set containing 2500 abstracts. We name this dataset PubMed 20k RCT, the prefix k meaning 1000, as defined by the International System of Units [112].

Dataset Format

The dataset is provided as three text files: one file for the training set, one file for the validation set, and one file for the test set. Each file has the same format: each line corresponds to either a PMID or a sentence with its capitalized label at the beginning. Each token is separated by a space. Listing 3.1 shows an excerpt from these files.

For each abstract, sentence and token boundaries are detected using the Stanford CoreNLP toolkit [79]. Digits were replaced by the character @ (at sign).

OBJECTIVE: This study evaluated an eating disorder intervention multimedia program modeled after self-help eating disorder treatment programs. It was hypothesized that women who completed the program would increase their body satisfaction and decrease their preoccupation with weight and frequency of disordered eating behaviors.
METHOD: Participants were 57 undergraduate females randomly assigned to either the intervention or control group. Psychological functioning was assessed at baseline, at 3 months postintervention, and at 3 months follow-up.
RESULTS: Intervention group subjects significantly improved their scores on all psychological measures over time. When compared to the control group, however, only the intervention group's improvements on the Body Shape Questionnaire were statistically significant.
DISCUSSION: This study has demonstrated that minimally effective eating disorder intervention programs can be delivered. A revised program that eliminates interface problems and increases the structure of the intervention is likely to be even better received and more effective.

Figure 3-2: Example of structured RCT abstract, obtained from PubMed. This abstract was taken from [68]

```
###9813759
OBJECTIVE This study evaluated an eating disorder intervention     [...]
OBJECTIVE It was hypothesized that women who completed the program [...]
METHODS Participants were @ undergraduate females randomly         [...]
METHODS Psychological functioning was assessed at baseline , at    [...]
RESULTS Intervention group subjects significantly improved their   [...]
RESULTS When compared to the control group , however , only the    [...]
CONCLUSIONS This study has demonstrated that minimally effective   [...]
CONCLUSIONS A revised program that eliminates interface problems   [...]
```

Listing 3.1: Example of one abstract as formatted in the PubMed 20k RCT dataset set. The PMID of the corresponding article is 9813759; the article can be found that https://www.ncbi.nlm.nih.gov/pubmed/9813759. Figure 3-2 presents the abstract as shown in PubMed.

3.3.3 Dataset Analysis

Figure 3-3 counts the number of sentences per label: the least common label (objective) is approximately four times less frequent than the most common label (results), which indicates that the dataset is not excessively unbalanced. Figure 3-4 shows the distribution of the number of sentences per abstract. Figure 3-5 shows the distribution of the number of

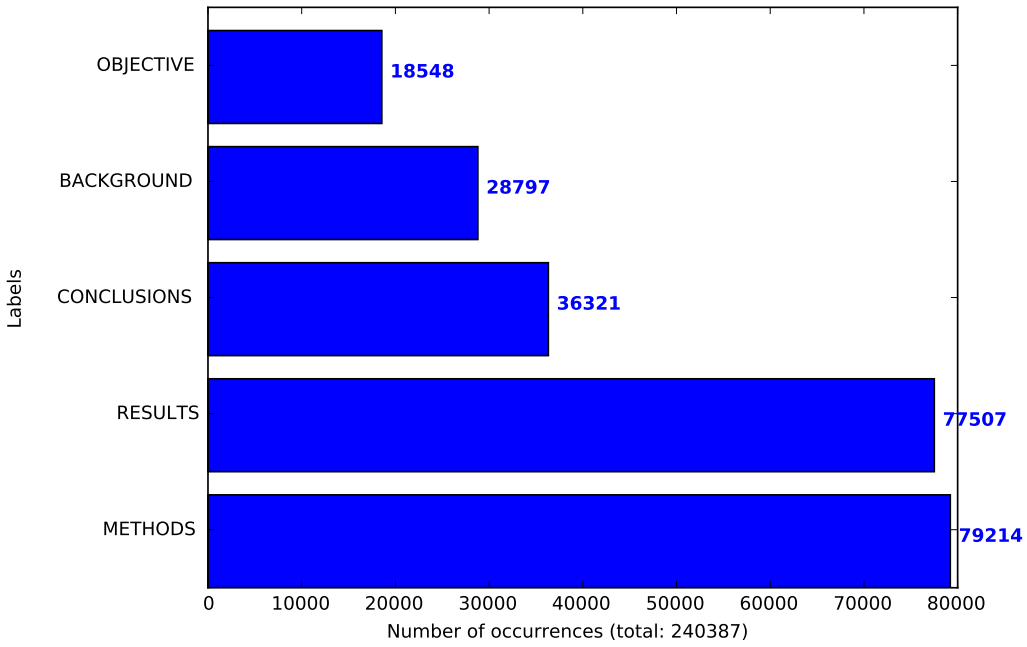


Figure 3-3: Number of sentences per label in PubMed 20k RCT

tokens the sentence.

3.4 Experiments

3.4.1 Datasets

We evaluate our model on the sentence classification task using the following two medical abstract datasets, where each sentence of the abstract is annotated with one label. Table 3.2 presents statistics on each dataset.

NICTA-PIBOSO This dataset was introduced in [65] and was the basis of the ALTA 2012 Shared Task [2].

PubMed 20k RCT This corpus was introduced in the previous section. It is based on the PubMed database of biomedical literature and uses 5 sentence labels: objectives, background, methods, results and conclusions

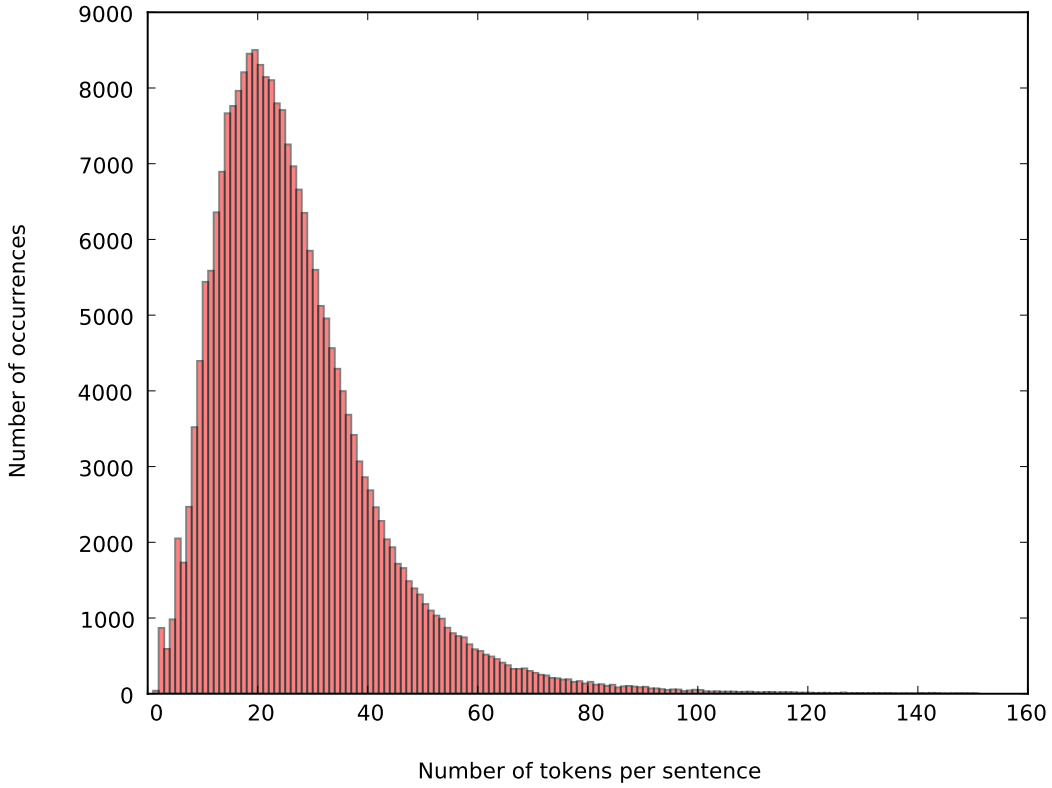


Figure 3-4: Distribution of the number of tokens the sentence in PubMed 20k RCT. Minimum: 1; mean: 26.2; maximum: 338; variance: 227.6; skewness: 2.0; kurtosis: 8.7.

Dataset	$\|C\|$	$\|V\|$	Train	Validation	Test
NICTA-PIBOSO	6	17k	722 (8k)	77 (0.9k)	200 (2k)
PubMed 20k RCT	5	68k	15k (195k)	2.5k (33k)	2.5k (33k)

Table 3.2: Overview of the PubMed and the NICTA datasets for sentence classification. $|C|$ denotes the number of classes, $|V|$ the vocabulary size. For the train, validation and test sets, we indicate the number of abstracts followed by the number of sentences in parentheses.

3.4.2 Training

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. We selected the hyperparameters manually, though we could have used some hyperparameter optimization tech-

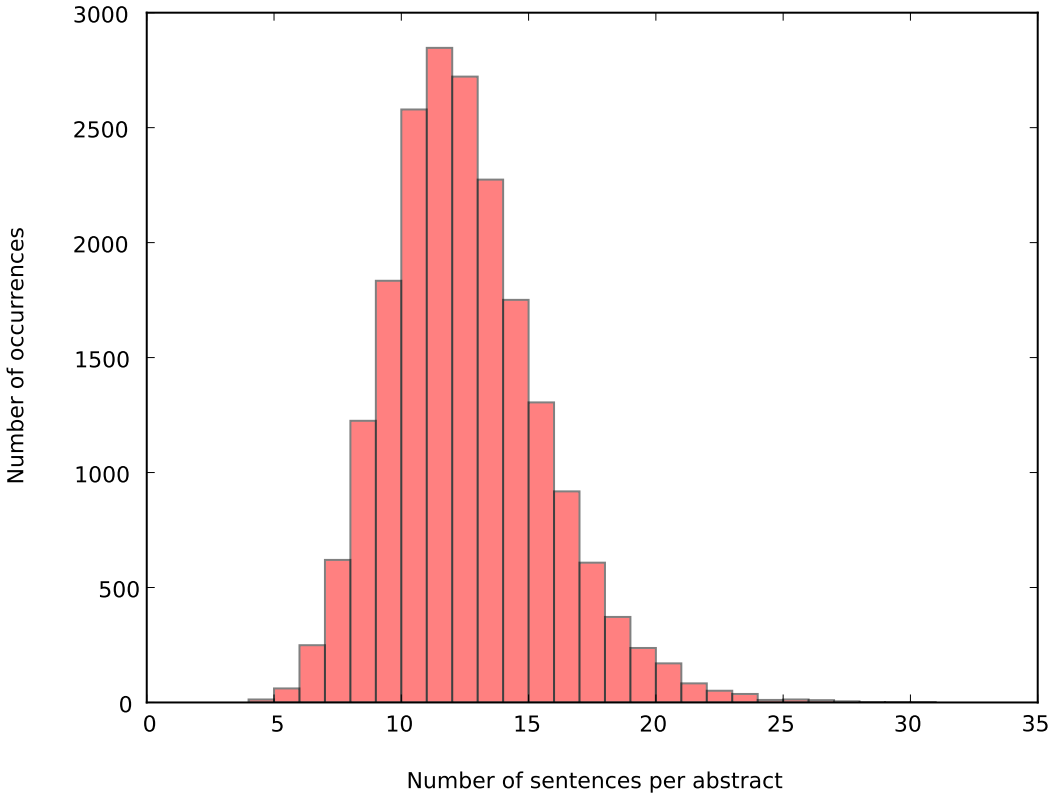


Figure 3-5: Distribution of the number of sentences per abstract in PubMed 20k RCT. Minimum: 3; mean: 11.6; maximum: 51; variance: 9.5; skewness: 0.9; kurtosis: 2.6.

niques [8, 28].

3.5 Results and Discussion

The first baseline (LR) is a classifier based on logistic regression using n-gram features extracted from the current sentence: it does not use any information from the surrounding sentences. The baseline was implemented with scikit-learn [90].

The second baseline (Forward ANN) uses the model presented in [72]: it computes sentence embeddings for each sentence, then classifies the current sentence given a few preceding sentence embeddings as well as the current sentence embedding.

The third baseline (CRF) is a CRF that uses n-grams as features: each output variable of the CRF corresponds to a label for a sentence, and the sequence the CRF considers is the entire abstract. The CRF baseline therefore uses both preceding and succeeding sentences

Model	PubMed 20k	NICTA
LR	83.1	71.6
Forward ANN	86.1	75.1
CRF	89.5	81.2
Best published	–	82.0
Our model	**90.0**	**82.7**

Table 3.3: F1-scores on the test set with several baselines, the best published method [78] from the literature, and our model. Since PubMed 20k RCT was introduced in this work, there is no previously published method for this dataset. The presented results for the ANN-based models are the F1-scores on the test set of the run with the highest F1-score on the validation set.

Model	PubMed 20k	NICTA
Full model	89.9	82.7
- character emb	89.7	82.7
- pre-train	88.7	78.0
- token emb	88.9	77.0
- seq opt	85.0	72.8

Table 3.4: Ablation analysis. F1-scores are reported. "- character emb" is our model using only token embeddings, without character-based token embeddings. "- pre-train" is our model where token embeddings are initialized with random values instead of pre-trained embeddings. "- token emb" is our model using only character-based token embeddings, without token embeddings. "- seq opt" is our model without the label sequence optimization layer. These numbers were averaged over 10 runs.

when classifying the current sentence. Lastly, the model presented in [78] developed a new approach called feature stacking, which is a metalearner that combines multiple feature sets, and is the best performing system on NICTA-PIBOSO published in the literature. The baseline was implemented with CRFsuite [88].

Table 3.3 compares our model against several baselines as well as the best performing model [78] in the ALTA 2012 Shared Task, in which 8 competing research teams participated to build the most accurate classifier for the NICTA-PIBOSO corpus.

The LR system performs honorably on PubMed 20k RCT (F1-score: 83.1), but quite poorly on NICTA-PIBOSO (F1-score: 71.6): this suggests that using the surrounding sentences may be more important in NICTA-PIBOSO than in PubMed 20k RCT.

The Forward ANN system performs better than the LR system, and worse than the CRF: this is expected, as the Forward ANN system only uses the information from the preceding sentences but do not use any information from the succeeding sentences, unlike the CRF.

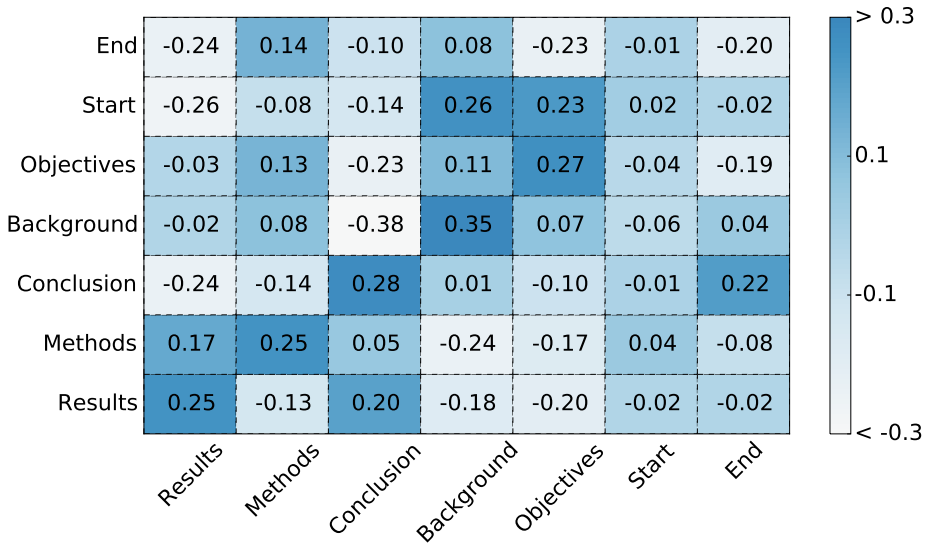


Figure 3-6: Transition matrix learned on PubMed 20k RCT. The rows represent the label of the previous sentence, the columns represent the label of the current sentence.

Sentence	Predicted	Actual
This study investigated whether oxytocin can affect attentional bias in social anxiety.	Background	Methods
The biological mechanisms by which oxytocin may be exerting these effects are [...]	Conclusions	Results
Leuprolide pharmacokinetics were characterized for 11.25 and 30 mg 3-month [...]	Conclusions	Results
While, 6%HES 130/0.4 (free flex 6%HES 130/0.4, Fresenius Kabi) infusion was [...]	Results	Methods
Arterial and central venous blood gas analyses were performed every 20 minutes [...]	Results	Methods
Cytokine responses accompanying [...] immunotherapy [...] have not previously [...]	Background	Objectives

Table 3.5: Examples of prediction errors of our model on PubMed 20k RCT. The "predicted" column indicates the label predicted by our model for a given sentence. Our model takes into account all the sentences present in the abstract in which the classified sentence appears. The "actual" column indicates the gold label of the sentence.

Our model performs better than the CRF system and the system from [78]. We hypothesize that the following four factors give an edge to our model:

No human-engineered features: Unlike most other systems, our model does not rely on any human-engineered features.

No n-grams: While other systems heavily relies on n-grams, our model maps each token to a token embedding, and feeds it as an input to an RNN. This helps combat data scarcity, as for example "chronic tendonitis" and "chronic tendinitis" are two different bigrams, but share the same meaning, and their token embeddings should therefore be very similar.

Structured prediction: The labels for all sentences in an abstract are predicted jointly, which improves the coherency between the predicted labels in a given abstract. The abla-

	PubMed 20k RCT			
	Precision	Recall	F1-score	Support
Background	71.8	88.2	79.1	3621
Conclusion	93.5	92.9	93.2	4571
Methods	93.7	96.2	94.9	9897
Objectives	78.2	48.1	59.6	2333
Results	94.8	93.1	93.9	9713
Total	90.1	89.9	90.0	30135

Table 3.6: Results for each class obtained by our model on PubMed 20k RCT.

	Backg.	Concl.	Methods	Obj.	Res.
Background	3193	28	116	277	7
Conclusions	55	4248	7	0	261
Methods	78	36	9523	35	225
Objectives	1112	1	95	1122	3
Results	11	232	426	1	9043

Table 3.7: Confusion matrix on PubMed 20k RCT obtained with our model. Rows correspond to actual labels, and columns correspond to predicted the labels. For example, 116 background sentences were predicted as method.

tion analysis presented in Table 3.4 shows that the sequence optimization layer is the most important component of the ANN model.

Joint learning: Our model learned the features and token embeddings jointly with the sequence optimization.

The sequence information is mostly contained in the transition matrix. Figure 3-6 presents an example of transition matrix after the model has been trained on PubMed 20k RCT. We can see that it effectively reflects transitions between different labels. For example, it learned that the first sentence of an abstract is most likely to be either discussing objective (0.23) or background (0.26). By the same token, a sentence pertaining to the methods is typically followed by a sentence pertaining to the methods (0.25) or the results (0.17).

Tables 3.6 and 3.7 detail the result of our model for each label in PubMed 20k RCT. The main difficulty the classifier has is distinguishing background sentences from objective sentences. In particular, a third of the objective sentences are incorrectly classified as background, which causes the recall for objectives and the precision for background to be

low. The classifier has also some difficulty in distinguishing method sentences from result sentences.

Table 3.5 presents a few examples of prediction errors. Our error analysis suggests that a fair number of sentence labels are debatable. For example, the sentence "We conducted a randomized study comparing strategies X and Y." belongs to the background according to the gold target, but most humans would classify it as an objective.

We performed two additional experiments with our model on PubMed 20k RCT: shuffling the sequences, and removing any content present in parentheses.

- In the first experiment, shuffling the sequences, within each abstract we shuffle all sentences. The point of the experiments is to assess to what extent the model relies on the sequential information to make its predictions. In that setting, our model achieves an F1-score of 84.0, which is significantly lower than the F1-score of our model when the sentences are not shuffled (90.0). This indicates that our model efficiently leverages sentence order.
- In the second experiment, we removed any content present in parentheses. For example, if the sentence is "all patients were reassessed (no dropouts) and no difference between the PRP and the saline group could be observed.", then it is transformed into "all patients were reassessed no difference between the PRP and the saline group could be observed.". The motivation behind this experiment is to quantify the usefulness of the content in parentheses. In that setting, our model achieves an F1-score of 89.9 which is slightly lower the F1-score when the content in parentheses is kept (90.0).

3.6 Conclusions

In this chapter we have presented an ANN architecture to classify sentences that appear in sequence. We demonstrate that jointly predicting the classes of all sentences in a given text improves the quality of the predictions. Our model outperforms the state-of-the-art results on two datasets for sentence classification in medical abstracts.

We have also introduced PubMed 20k RCT, a dataset for sequential sentence classification. We hope that the release of this dataset will help the development of algorithms for sequential sentence classification and increase the interest of the text mining community in the study of RCTs.

Chapter 4

Neural Network Hyperparameter Optimization

Systems based on artificial neural networks (ANNs) have achieved state-of-the-art results in many natural language processing tasks. Although ANNs do not require manually engineered features, ANNs have many hyperparameters to be optimized. The choice of hyperparameters significantly impacts models' performance. However, the ANN hyperparameters are typically chosen by manual, grid, or random search, which either requires expert experience or is computationally expensive. Recent approaches based on Bayesian optimization using Gaussian processes (GPs) is a more systematic way to automatically pinpoint optimal or near-optimal machine learning hyperparameters. Using the ANN model presented in Chapter 2, which yields state-of-the-art results for dialog act classification, we demonstrate that optimizing hyperparameters using GP further improves the results, and reduces the computational time by a factor of 4 compared to a random search. Therefore it is a useful technique for tuning ANN models to yield the best performance for natural language processing tasks.

4.1 Introduction and related work

Artificial neural networks (ANNs) have recently shown state-of-the-art results on various NLP tasks including language modeling [84], named entity recognition [18, 70, 69], text classification [106, 66, 10, 72], question answering [118, 116], and machine translation [4, 111]. Unlike other popular non-ANN-based machine learning algorithms such as support vector machines (SVMs) and conditional random fields (CRFs), ANNs can automatically learn features that are useful for NLP tasks, thereby requiring no manually engineered features.

However, ANNs have hyperparameters that need to be tuned in order to achieve the best results. The hyperparameters of an ANN model may define either its learning process (e.g., learning rate or mini-batch size) or its architecture (e.g., number of hidden units or layers). ANNs commonly contain over ten hyperparameters [7], which makes it challenging to optimize. Therefore, most published ANN-based works on NLP tasks rely on basic heuristics such as manual or random search, and sometimes do not even optimize hyperparameters.

Although most of them report state-of-the-art results without optimizing hyperparameters extensively, we argue that the results can be further improved by properly optimizing the hyperparameters. Despite this, one of the main reasons why most previous NLP works do not thoroughly optimize hyperparameters is that it may represent a significant time investment. However, if we optimize them "efficiently", we can find hyperparameters that perform well within a reasonable amount of time, as shown in this chapter.

Like ANNs, other machine learning algorithms also have hyperparameters. The two most widely used methods for hyperparameter optimization of machine learning algorithms are manual or grid search [9]. Bergstra and Yoshua [9] show that random search is as good or better than grid search at finding hyperparameters within a small fraction of computation time and suggest that random search is a natural baseline for judging the performance of automatic approaches for tuning the hyperparameters of a learning algorithm. However, all above-mentioned methods for tuning hyperparameters have some downsides. Manual search requires human experts or uses arbitrary rules of thumb, while grid and random searches are computationally expensive [104].

Recently, a more systematic approach based on Bayesian optimization with Gaussian process (GP) [119] has been shown to be effective in automatically tuning the hyperparameters of machine learning algorithms, such as latent Dirichlet allocation, SVMs, convolutional neural networks [104], and deep belief networks [8], as well as tuning the hyperparameters that features may have [37, 27]. In this approach, the model's performance for each hyperparameter combination is modeled as a sample from a GP, resulting in a tractable posterior distribution given previous experiments. Therefore, this posterior distribution is used to find the optimal hyperparameter combination to try next based on the observation.

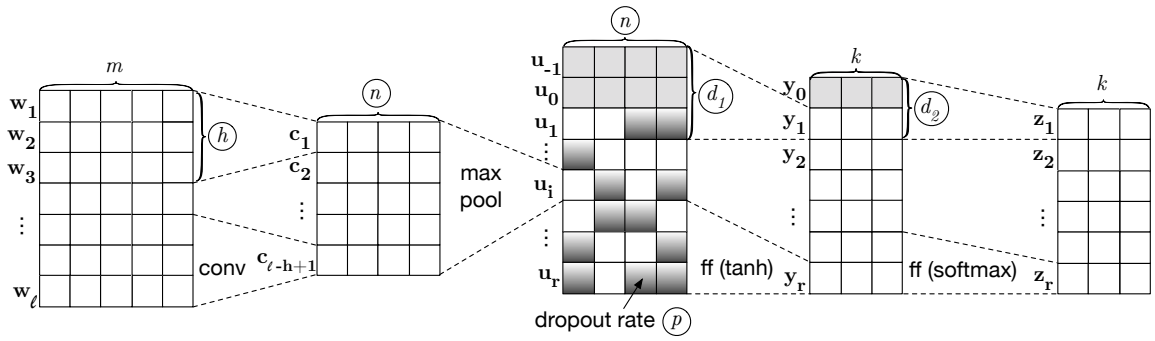


Figure 4-1: The ANN model. A sequence of words $\mathbf{w}_{1:\ell}$ corresponding to the i^{th} utterance is transformed into a vector $\mathbf{u}_i$ using a CNN, consisting of a convolution layer (conv) and a max pooling layer (max pool). Each utterance is then classified by a two-layer feedforward (ff) network with tanh and softmax activation functions. The hyperparmeters that we optimize are circled: filter size h, number of filters n, dropout rate p, history sizes d_1, d_2. In the figure, $h = 3$, $n = 4$, $p = 0.5$, $d_1 = 3$, $d_2 = 2$. The grey rows $(\mathbf{u}_{-1}, \mathbf{u}_0, \mathbf{y}_0)$ represent zero paddings.

In this work, we demonstrate the application of Gaussian Process (GP) to optimize ANN hyperparameters on an NLP task, namely dialog act classification [107], whose goal is to assign a dialog act to each utterance. The ANN model in [72] makes a good candidate for hyperparameter optimization since it is a simple model with a few architectural hyperparameters, and the optimized architectural hyperparameters are interpetable and give some insights for the task at hand. Using this model, we show that optimizing hyperparameters further improves the state-of-the-art results on two datasets, and reduces the computational time by a factor of 4 compared to a random search.

4.2 Methods

The ANN model for dialog act classification was introduced in Chapter 2 and is briefly repeated in Section 4.2.1. The GP used to optimize the hyperparameters of the ANN model is presented in Section 4.2.2.

4.2.1 ANN model

Each utterance of a dialog is mapped to a vector representation via a CNN (Section 4.2.1). Each utterance is then sequentially classified by leveraging preceding utterances (Section 4.2.1). Figure 4-1 gives an overview of the ANN model.

Utterance representation via CNN

An utterance of length ℓ is represented as the sequence of word vectors $\mathbf{w}_{1:\ell} \in \mathbb{R}^m$. Given the word vectors, the CNN model produces the *utterance representation* $\mathbf{u} \in \mathbb{R}^n$.

Let h be the size of a *filter*, and the sequence of vectors $\mathbf{v}_{1:h} \in \mathbb{R}^m$ be the corresponding filter matrix. A convolution operation on h consecutive word vectors starting from the t^{th} word outputs the scalar *feature* $c_t = \tanh\left(\sum_{i=1}^{h} \mathbf{v}_i^T \mathbf{w}_{t+i-1} + b_f\right)$, where $b_f \in \mathbb{R}$ is a bias term.

We perform convolution operations with n different filters, and denote the resulting features as $\mathbf{c}_t \in \mathbb{R}^n$, each of whose dimensions comes from a distinct filter. Repeating the convolution operations for each window of h consecutive words in the utterance, we obtain $\mathbf{c}_{1:\ell-h+1}$. The utterance representation $\mathbf{u} \in \mathbb{R}^n$ is computed in the max pooling layer, as the element-wise maximum of $\mathbf{c}_{1:\ell-h+1}$. During training, dropout with probability p is applied on this utterance representation $\mathbf{u}$.

The filter size h, the number of filters n, and a dropout probability p are the hyperparameters of this section that we optimize using the GP (Section 4.2.2).

Sequential utterance classification

Let $\mathbf{u}_i \in \mathbb{R}^n$ be the utterance representation given by the CNN architecture for the i^{th} utterance in the sequence of length r. The sequence $\mathbf{u}_{1:r}$ is input to a two-layer feedforward

neural network that classifies each utterance. The hyperparameters d_1, d_2, the history sizes used in the first and second layers respectively, are optimized using the GP (Section 4.2.2).

The first layer takes as input $\mathbf{u}_{i-d_1+1:i}$ and outputs $\mathbf{y}_i \in \mathbb{R}^k$, where k is the number of classes for the classification task, i.e. the number of dialog acts. It uses a tanh activation function. Similarly, the second layer takes as input $\mathbf{y}_{i-d_2+1:i}$ and outputs $\mathbf{z}_i \in \mathbb{R}^k$ with a softmax activation function.

The final output $\mathbf{z}_i$ represents the probability distribution over the set of k classes for the i^{th} utterance: the j^{th} element of $\mathbf{z}_i$ corresponds to the probability that the i^{th} utterance belongs to the j^{th} class. Each utterance is assigned to the class with the highest probability.

4.2.2 Hyperparameter optimization using GP

Let $\mathcal{X}$ be the set of all hyperparameter combinations considered, and let $f : \mathcal{X} \to \mathbb{R}$ be the function mapping from hyperparameter combinations to a real-valued performance metric (such as F1-score on test set) of a learning algorithm using the given hyperparameter combination. Our interest lies in *efficiently* finding a hyperparameter combination $\mathbf{x} \in \mathcal{X}$ that yields a *near-optimal* performance $f(\mathbf{x})$. In this chapter, we use Bayesian optimization of hyperparameters using GP, which we call *GP search*.

Comparison with other methods

A grid search is brute-forcefully evaluating $f(\mathbf{x})$ for each $\mathbf{x} \in \mathcal{X}$ defined on a grid and then selecting the best one. In a random search, one randomly selects an $\mathbf{x} \in \mathcal{X}$ and evaluates the performance $f(\mathbf{x})$; this process is repeated until an $\mathbf{x}$ with a satisfactory $f(\mathbf{x})$ is found. In a manual search, an expert tries out some hyperparameter combinations based on prior experience until settling on a good one.

In contrast with the other methods mentioned above, a GP search chooses the hyperparameter combination to evaluate next by exploiting all previous evaluations. To achieve this, we assume the prior distribution on the function f to be a Gaussian process, which allows us to construct a probabilistic model for f using all previous evaluations, by calculating the posterior distribution in a tractable manner. Once the model for f is computed, it

is used to choose an optimal hyperparameter combination to evaluate next.

GP search

In a GP search, we use a GP to describe a distribution over functions. A GP is defined as a collection of random variables, any finite number of which have a joint Gaussian distribution. A GP $f(\mathbf{x})$ is completely specified by its mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$, also called *kernel*, defined as:

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})], \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]. \end{aligned}$$

In our case $f(\mathbf{x})$ is the F1-score on the test set evaluated for the ANN model using the given hyperparameter combination $\mathbf{x} \in \mathcal{X}$, which is a 5-dimensional vector consisting of filter size h, number of filters n, dropout rate p, and history sizes d_1, d_2.

Let $X = (\mathbf{x}_1, \ldots, \mathbf{x}_q)$, $\mathbf{f} = (f(\mathbf{x}_1) \ldots, f(\mathbf{x}_q))$ and $X^* = (\mathbf{x}_{q+1}, \ldots, \mathbf{x}_s)$, $\mathbf{f}^* = (f(\mathbf{x}_{q+1}) \ldots, f(\mathbf{x}_s))$ be the training inputs and outputs, and test inputs and outputs, respectively. $X \cup X^* = \mathcal{X}$, and $X \cap X^* = \emptyset$. Note that $\mathbf{f}$ is known, and $\mathbf{f}^*$ is unknown. The goal is to find the distribution of $\mathbf{f}^*$ given X^*, X and $\mathbf{f}$, in order to select among X^* the hyperparameter combination that is the most likely to yield the highest F1-score.

The joint distribution of $\mathbf{f}$ and $\mathbf{f}^*$ according to the prior is

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{m} \\ \mathbf{m}^* \end{bmatrix}, \begin{bmatrix} K(X, X) & K(X, X^*) \\ K(X^*, X) & K(X^*, X^*) \end{bmatrix} \right)$$

where $\mathbf{m}$, $\mathbf{m}^*$ is a vector of the means evaluated at all training and test points respectively, and $K(X, X^*)$ denotes the $q \times q^*$ matrix of the covariances evaluated at all pairs of training and test points, and similarly for $K(X, X)$, $K(X^*, X)$ and $K(X^*, X^*)$.

Conditioning the joint Gaussian prior on the observations yields $\mathbf{f}^* | X^*, X, \mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where

$$\boldsymbol{\mu} = \mathbf{m}^* - K(X^*, X)K(X, X)^{-1}(\mathbf{f} - \mathbf{m}), \tag{4.1}$$
$$\boldsymbol{\Sigma} = K(X^*, X^*) - K(X^*, X)K(X, X)^{-1}K(X, X^*).$$

The choice of the kernel $k(\mathbf{x}, \mathbf{x}')$ impacts predictions. We investigate 4 different kernels:

- Linear: $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$
- Cubic: $k(\mathbf{x}, \mathbf{x}') = 3\left(\left(\mathbf{x}^T\mathbf{x}'\right)^2 + 2\left(\mathbf{x}^T\mathbf{x}'\right)^3\right)$
- Absolute exponential: $k(\mathbf{x}, \mathbf{x}') = e^{|\mathbf{x}-\mathbf{x}'|}$
- Squared exponential: $k(\mathbf{x}, \mathbf{x}') = e^{-0.5|\mathbf{x}-\mathbf{x}'|^2}$

To initialize the GP search, one needs to compute the F1-score for a certain number of randomly chosen hyperparameter combinations r: we investigate what the optimal number is. We then iterate over the following two steps until a specified maximum number of iterations t is reached. First, we find the hyperparameter combination in the test set with the highest F1-score predicted by the GP. Second, we compute the actual F1-score, and move it to the training set. This process is outlined in Algorithm 1.

4.3 Experiments

4.3.1 Datasets

We evaluate the random and GP searches on the dialog act classification task using the Dialog State Tracking Challenge 4 (DSTC 4) [62, 63], ICSI Meeting Recorder Dialog Act (MRDA) [55, 102], and Switchboard Dialog Act (SwDA) [60] datasets. DSTC 4, MRDA, and SwDA respectively contain 32k, 109k, and 221k utterances, which are labeled with 89, 5, and 43 different dialog acts (we used the 5 coarse-grained dialog acts introduced in [3] for MRDA). The train/test splits are provided along with the datasets, and the validation set was chosen randomly except for MRDA, which specifies a validation set.[1]

[1] See https://github.com/Franck-Dernoncourt/slt2016 for the train, validation, and test splits.

Algorithm 1 GP search algorithm

function GP-REGRESSION($X^*, X, \mathbf{f}$)
  compute $\boldsymbol{\mu}$ according to (4.1)
  return $\boldsymbol{\mu}$
end function
function GP-SEARCH($\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_s\}, f(\cdot), r, t$)
  $X \leftarrow (\emptyset)$
  $X^* \leftarrow (\mathbf{x}_1, \ldots, \mathbf{x}_s)$
  for $i = 1, \ldots, r$ **do**
    randomly choose $\mathbf{x} \in X^*$
    remove $\mathbf{x}$ from X^*
    add $\mathbf{x}$ to X and $f(\mathbf{x})$ to $\mathbf{f}$
  end for
  for $i = r + 1, \ldots, t$ **do**
    $\boldsymbol{\mu} \leftarrow$ GP-REGRESSION($X^*, X, \mathbf{f}$)
    $\hat{j} \leftarrow \underset{j=1,\ldots,|\boldsymbol{\mu}|}{\arg\max}\, \mu_j$, $\mathbf{x} \leftarrow X[j^*]$
    remove $\mathbf{x}$ from X^*
    add $\mathbf{x}$ to X and $f(\mathbf{x})$ to $\mathbf{f}$
  end for
  return $\arg\max_{\mathbf{x} \in X} f(\mathbf{x})$
end function

4.3.2 Training

For a given hyperparameter combination, the ANN is trained to minimize the negative log-likelihood of assigning the correct dialog acts to the utterances in the training set, using stochastic gradient descent with the Adadelta update rule [123]. At each gradient descent step, weight matrices, bias vectors, and word vectors are updated. For regularization, dropout is applied after the pooling layer, and early stopping is used on the validation set with a patience of 10 epochs. We initialize the word vectors with the 300-dimensional word vectors pretrained with word2vec on Google News [83, 86] for DSTC 4, and the 200-dimensional word vectors pretrained with GloVe on Twitter [91] for SwDA.

4.3.3 Hyperparameters

For each hyperparameter combination, the reported F1-score is averaged over 5 runs. Table 4.1 presents the hyperparameter search space.

Hyperparameter	Values
Filter size h	3, 4, 5
Number of filters n	50, 100, 250, 500, 1000
Dropout rate p	0.1, 0.2, . . . , 0.9
History size d_1	1, 2, 3
History size d_2	1, 2, 3

Table 4.1: Candidate values for each hyperparameter. Since h, n, p, d_1, and d_2 can take 3, 5, 9, 3, and 3 different values respectively, there are 1215 ($= 3 \times 5 \times 9 \times 3 \times 3$) possible hyperparameter combinations.

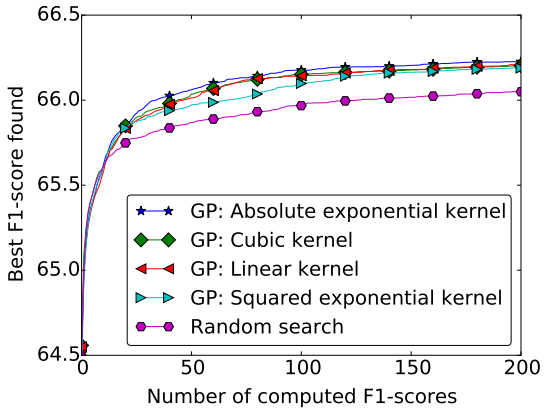
4.4 Results

GP search finds near-optimal hyperparameters faster than random search. Figure 4-2 compares the GP searches with different kernels against the random search, which is a natural baseline for hyperparameter optimization algorithms [9]. On all datasets, the F1-score evaluated using the hyperparameters found by the GP search converges to near-optimal values significantly faster than the random search, regardless of the kernels used. For example, on SwDA, after computing the F1-scores for 100 different hyperparameter combinations, the GP search reaches on average 72.1, whereas the random search only obtains 71.4. The random search requires computing over 400 F1-scores to reach 72.1: the GP search therefore reduces the computational time by a factor of 4. This is a significant improvement considering that computing the average F1-scores over 5 runs for 300 extra hyperparameter combinations takes 60 days on a GeForce GTX Titan X GPU.

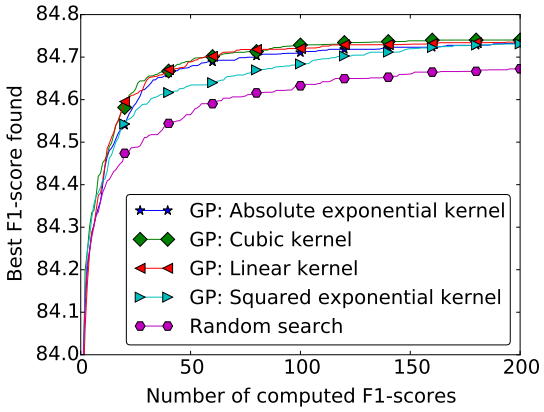
Squared exponential kernel converges more slowly than others. Even though the GP search with any kernel choice is faster than the random search, some kernels result in better performance than others. The best kernel choice depends on the choice of the dataset, but the squared exponential kernel (a.k.a. radial basis function kernel) consistently converges more slowly, as illustrated by Figure 4-2. Across the datasets, there were no consistent differences among the linear, absolute exponential, and cubic kernels.

The number of initial random points impacts the performance. As mentioned in Section 4.2.2, the GP search starts with computing the F1-score for a certain number of randomly chosen hyperparameter combinations. Figure 4-3 shows the impact of this num-

(a) DSTC 4



(b) MRDA



(c) SwDA

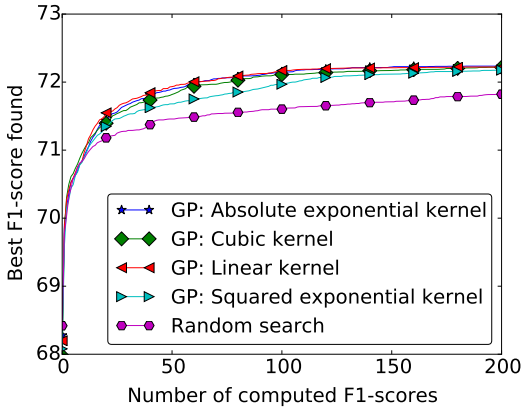
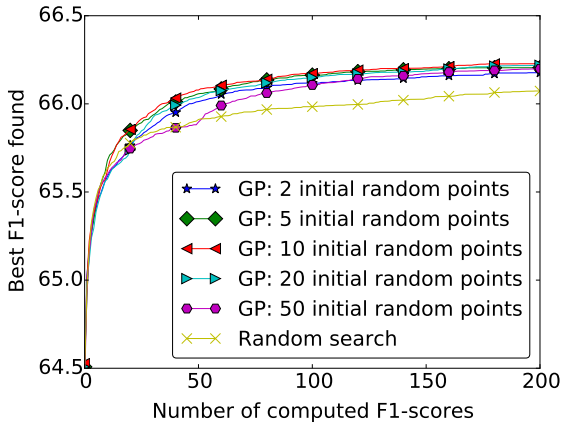
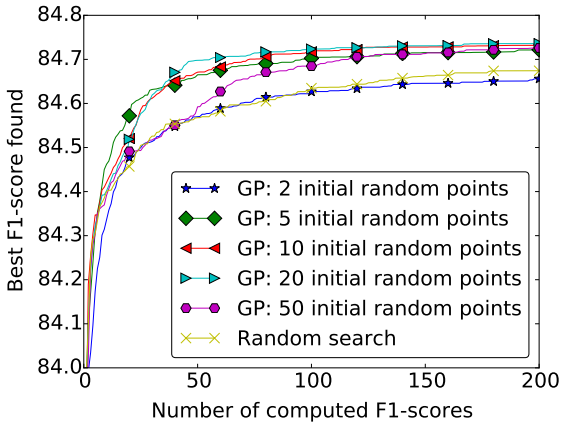


Figure 4-2: Performance of GP search with different kernels and random search for hyperparameter optimization on DSTC 4, MRDA, and SwDA. The x-axis represents the number of hyperparameter combinations for which the F1-score has been computed, and the y-axis shows the best F1-score that has been achieved by at least one of these hyperparameter combinations. Each data point is averaged over 100 runs of the specified search strategy.

(a) DSTC 4



(b) MRDA



(c) SwDA

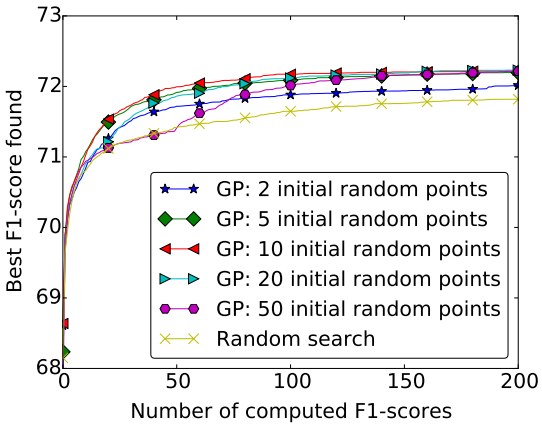


Figure 4-3: Impact of the number of initial random hyperparameter combinations on the GP search. The x-axis represents the number of hyperparameter combinations for which the F1-score has been computed, and the y-axis shows the best F1-score that has been achieved by at least one of these hyperparameter combinations. Each data point is averaged over 100 runs of the specified search strategy.

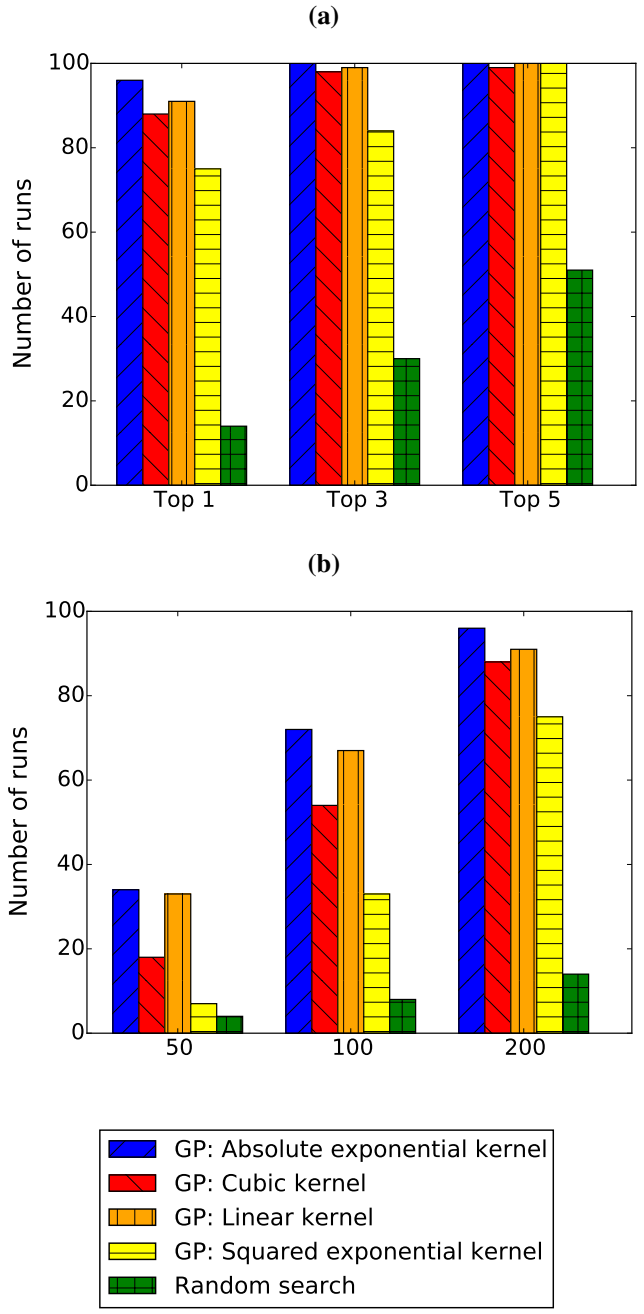


Figure 4-4: Finding near-optimal hyperparameter combinations on SwDA. Figure (a) shows how many times out of 100 runs each search strategy found a hyperparameter combination that is among the top 1, 3, and 5 best performing hyperparameter combinations. Figure (b) shows how many times out of 100 runs each search strategy found the best hyperparameter combination after evaluating 50, 100, and 200 hyperparameter combinations.

ber on all three datasets. The optimal number seems to be around 10 on average, i.e. 1% of the hyperparameter search space. When the number is very low (e.g., 2), the GP might fail to find the optimal hyperparameter combinations: it performs significantly worse on MRDA and SwDA. Conversely, when the number is very high (e.g., 50) it unnecessarily delays the convergence.

GP search often finds near-optimal hyperparameters quickly. After evaluating the F1-scores with 50 hyperparameter combinations, the GP search finds one of the 5 best hyperparameter combinations almost 80% of the time on SwDA, as shown in Figure 4-4, and even more frequently on DSTC 4 and MRDA. After computing 100 hyperparameter combinations, the GP search finds the best one over 70% of the time, while the random search stumbles upon it less 10% of the time.

Simple heuristics may not find optimal hyperparameters well. Compared to the previous state-of-the-art results that use the same model optimized manually [72], the GP search found more optimal hyperparameters, improving the F1-score by 0.5 ($= 66.3 - 65.8$), 0.1 ($= 84.7 - 84.6$), and 0.7 ($= 72.1 - 71.4$) on DTSC 4, MRDA, and SwDA, respectively. In [72], the hyperparameters were optimized by varying one hyperparameter at a time while keeping the hyperparameters fixed. Figures 4-5 and 4-6 demonstrate that optimizing each hyperparameter independently might result in a suboptimal choice of hyperparameters. Figure 4-5 illustrates that the optimal choice of hyperparameters is impacted by the choice of other hyperparameters. For example, a higher number of filters works better with a smaller dropout probability, and conversely a lower number of filters yields better results when used with a larger dropout probability. Figure 4-6 shows that, for instance, if one had first fixed the number of filters to be 100 and optimized the dropout rate, one would have found that the optimal dropout rate is 0.5. Then, fixing the dropout rate at 0.5, one would have determined that 500 is the optimal number of filters, thereby obtaining an F1-score of 70.0, which is far from the best F1-score (70.7).

The faster convergence of the GP search may stem from the capacity of the GP to leverage the patterns in the F1-score landscape such as the one shown in Figure 4-6. The random search cannot make use of this regularity.

4.5 Conclusion

In this chapter we addressed the commonly encountered issue of tuning ANN hyperparameters. Towards this purpose, we explored a strategy based on GP to automatically pinpoint optimal or near-optimal ANN hyperparameters. We showed that the GP search requires 4 times less computational time than random search on three datasets, and improves the state-of-the-art results by efficiently finding the optimal hyperparameter combinations. While the choices of the kernels and the number of initial random points impact the performance of the GP search, our findings show that it is more efficient than the random search regardless of these choices. The GP search can be used for any ordinal hyperparameter; it is therefore a useful technique when developing ANN models for NLP tasks.

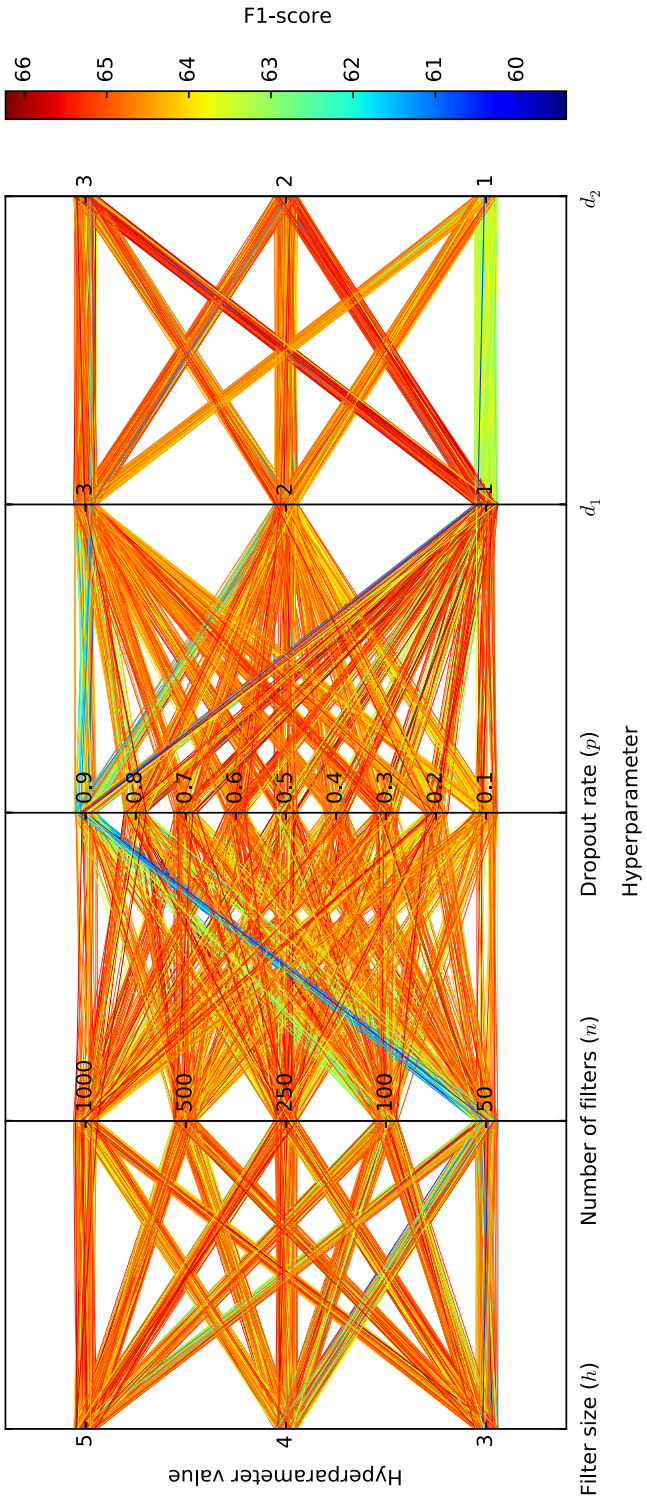


Figure 4-5: Parallel coordinate plot of all 1215 hyperparameter combinations for DSTC 4. Each hyperparameter combination in 5-dimensional search space is shown as a polyline with vertices on the parallel axes, each of which represents one of the 5 hyperparameter. The position of the vertex on each axis indicates the value of the corresponding hyperparameter. The color of each polyline reflects the F1-score obtained using the hyperparameter combination corresponding to the polyline.

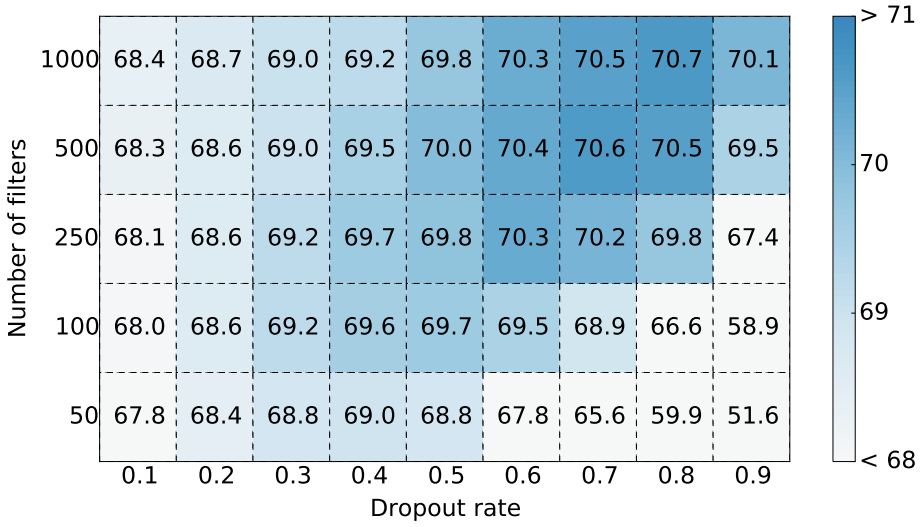


Figure 4-6: Heatmap of the F1-scores on SwDA as the number of filters and the dropout rate vary. F1-scores are averaged over all possible values of the other hyperparameters: as a result, F1-scores can be lower than the ones in Figure 4-2.

Chapter 5

Conclusions

5.1 Contributions

This thesis introduced several algorithms to perform sequential short-text classification, which outperform state-of-the-art algorithms. One challenge in the algorithms we have introduced is their number of hyperparameters. To facilitate the choice of hyperparameters, we presented a method based on Gaussian processes, which allows us to choose optimal or near optimal hyperparameters significantly faster than using random or grid search.

In order to foster research in sequential short-text classification as well as medical text mining, we released PubMed 20k RCT, a new dataset for sequential sentence classification in RCT abstracts.

5.2 Future work

Many directions can be investigated to further enhance medical text mining, and we have initiated some work in several of these directions.

Interpretability One of the most frequently mentioned limitation of ANNs is the lack of interpretability of their predictions, i.e., the lack of understanding of how the ANNs make a prediction given the input they receive. Since results directly impact health, clinicians have come to expect healthcare applications to use interpretable models [14]. Moreover, the

European Union is considering regulations that require algorithms to be interpretable [45]. We have explored a CNN-based method to perform patient phenotyping, and showed that a simple method allows us to view the phrases associated with each phenotype [43]. Much more work remains to be done in that direction to gain insights on what happens during the forward pass in an ANN. For example, developing visualization tools can help, such as LSTMVis [108], which is a visual analysis tool for recurrent neural networks with a focus on understanding these hidden state dynamics.

Going beyond classification While we have focused on classification in this thesis, the automated analysis of the medical literature requires other kind of NLP tasks as well. To that end, we have performed some work on using ANN for named-entity recognition [34, 75, 32, 74] as well as relation extraction [73]. It would be interesting to perform these two tasks jointly, as the performance of ANNs have been shown to improve when performing joint tasks [47]. Aside from the automatic creation of a knowledge base, from the user standpoint, tools to query knowledge bases need to be improved, e.g., by improving question answering systems.

Automated ANN architectures design We have analyzed the use of Gaussian processes to automatically determine of optimal or near-optimal hyperparameters. Beyond optimizing hyperparameters, one could explore algorithms that choose the entire ANN architecture, e.g., based on evolutionary computation [6, 21, 59] or reinforcement learning [126, 5].

Leveraging full-text articles We have only used paper abstracts in this thesis. This is largely due to the fact that most research papers are not open access, and even fewer can be freely used for text mining purposes. The medical field is especially plagued by the lack of open access. However, much information is present only in the body of the articles, and the lack of open access is a major impediment to structuring the medical literature. Medical researchers must think about the implications of not making research freely available when deciding to publish in pay-walled venues.

Exploring other types of literature We have focused on the medical literature, but most of the literature would benefit from sentence classification and more generally any tool that might make it more structured.

Bibliography

[1] Elaine Allen and Ingram Olkin. Estimating time to conduct a meta-analysis from number of citations retrieved. *JAMA: The Journal of the American Medical Association*, 282(7):634–635, 1999.

[2] Iman Amini, David Martinez, Diego Molla, et al. Overview of the ALTA 2012 shared task. 2012.

[3] Jeremy Ang, Yang Liu, and Elizabeth Shriberg. Automatic dialog act segmentation and classification in multiparty meetings. In *ICASSP (1)*, pages 1061–1064, 2005.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv:1409.0473*, 2014.

[5] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv:1611.02167*, 2016.

[6] Justin Bayer, Daan Wierstra, Julian Togelius, and Jürgen Schmidhuber. Evolving memory cell structures for sequence learning. In *International Conference on Artificial Neural Networks*, pages 755–764. Springer, 2009.

[7] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer, 2012.

[8] 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.

[9] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

[10] 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.

[11] Ameni Bouaziz, Christel Dartigues-Pallez, Célia da Costa Pereira, Frédéric Precioso, and Patrick Lloret. Short text classification using semantic random forest. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 288–299. Springer, 2014.

[12] Florian Boudin, Jian-Yun Nie, Joan C Bartlett, Roland Grad, Pierre Pluye, and Martin Dawes. Combining classifiers for robust pico element detection. *BMC medical informatics and decision making*, 10(1):1, 2010.

[13] Trung H. Bui, Hung H. Bui, and Franck Dernoncourt. Rule-based dialog state tracking, August 10 2017. US Patent App. 15/017,305.

[14] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730. ACM, 2015.

[15] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[16] Grace Yuet-Chee Chung. Towards identifying intervention arms in randomized controlled trials: extracting coordinating constructions. *Journal of biomedical informatics*, 42(5):790–800, 2009.

[17] Ronan Collobert. Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics*, number EPFL-CONF-192374, 2011.

[18] 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.

[19] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for natural language processing. *arXiv:1606.01781*, 2016.

[20] Patrick Davis-Desmond and Diego Mollá. Detection of evidence in clinical research papers. In *Proceedings of the Fifth Australasian Workshop on Health Informatics and Knowledge Management-Volume 129*, pages 13–20. Australian Computer Society, Inc., 2012.

[21] Franck Dernoncourt. The medial reticular formation (mRF): a neural substrate for action selection? an evaluation via evolutionary computation. Master's thesis, ENS Ulm, 2011.

[22] Franck Dernoncourt. Replacing the computer mouse. In *MIT CSAIL Student Workshop*, 2012.

[23] Franck Dernoncourt. Introduction to fuzzy logic. *Massachusetts Institute of Technology*, 2013.

[24] Franck Dernoncourt. BeatDB: an end-to-end approach to unveil saliencies from massive signal data sets. Master's thesis, Massachusetts Institute of Technology, 2014.

[25] Franck Dernoncourt. Trackmania is NP-complete. *arXiv:1411.5765*, 2014.

[26] Franck Dernoncourt. Mapping distributional to model-theoretic semantic spaces: a baseline. *arXiv:1607.02802*, 2016.

[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, Trung H. Bui, and Hung H. Bui. Adobe-MIT submission to the DSTC 4 Spoken Language Understanding pilot task. In *7th International Workshop on Spoken Dialogue Systems (IWSDS)*, 2016.

[30] Franck Dernoncourt, Ji Young Lee, Trung H. Bui, and Hung H. Bui. Robust dialog state tracking for large ontologies. In *International Workshop on Spoken Dialogue Systems*, 2016.

[31] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. Neural networks for joint sentence classification in medical paper abstracts. *European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

[32] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. NeuroNER: an easy-to-use program for named-entity recognition based on neural networks. *Empirical Methods on Natural Language Processing (EMNLP)*, 2017.

[33] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. *International Joint Conference on Natural Language Processing (IJCNLP)*, 2017.

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

[35] Franck Dernoncourt, Colin Taylor, Una-May O'Reilly, Kayan Veeramachaneni, Sherwin Wu, Chuong Do, and Sherif Halawa. MoocViz: A large scale, open access, collaborative, data analytics platform for MOOCs. In *NIPS Education Workshop*, 2013.

[36] Franck Dernoncourt, Kalyan Veeramachaneni, and Una-May O'Reilly. beatDB: A large scale waveform feature repository. In *NIPS , Machine Learning for Clinical Data Analysis and Healthcare Workshop*, 2013.

[37] Franck Dernoncourt, Kalyan Veeramachaneni, and Una-May O'Reilly. Gaussian process-based feature selection for wavelet parameters: Predicting acute hypotensive episodes from physiological signals. In *IEEE 28th International Symposium on Computer-Based Medical Systems*, 2015.

[38] Cícero Nogueira dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78, 2014.

[39] 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.

[40] Julian H Elliott, Tari Turner, Ornella Clavisi, James Thomas, Julian PT Higgins, Chris Mavergames, and Russell L Gruen. Living systematic reviews: an emerging opportunity to narrow the evidence-practice gap. *PLoS Med*, 11(2):e1001603, 2014.

[41] John W Ely, Jerome A Osheroff, Mark H Ebell, George R Bergus, Barcey T Levy, M Lee Chambliss, and Eric R Evans. Analysis of questions asked by family doctors regarding patient care. *Bmj*, 319(7206):358–361, 1999.

[42] Rebecca English, Yeonwoo Lebovitz, Robert Griffin, et al. *Transforming clinical research in the United States: challenges and opportunities: workshop summary*. National Academies Press, 2010.

[43] Sebastian Gehrmann, Franck Dernoncourt, Yeran Li, Eric T. Carlson, Joy T. Wu, Jonathan Welt, David W. Grant, John Foote Jr., Edward T Moseley, Patrick D. Tyler, and Leo A. Celi. Comparing rule-based and deep learning models for patient phenotyping. *arXiv preprint arXiv:1703.08705*, 2017.

[44] Alexander Genkin, David D Lewis, and David Madigan. Sparse logistic regression for text categorization. *DIMACS Working Group on Monitoring Message Streams Project Report*, 2005.

[45] Bryce Goodman and Seth Flaxman. Eu regulations on algorithmic decision-making and a "right to explanation". In *ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)*, 2016.

[46] Kazuo Hara and Yuji Matsumoto. Extracting clinical trial design information from medline abstracts. *New Generation Computing*, 25(3):263–275, 2007.

[47] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv:1611.01587*, 2016.

[48] 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.

[49] Erik Hemberg, Kalyan Veeramachaneni, Franck Dernoncourt, Mark Wagy, and Una-May O'Reilly. Efficient training set use for blood pressure prediction in a large scale learning classifier system. In *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation*, pages 1267–1274. ACM, 2013.

[50] Erik Hemberg, Kalyan Veeramachaneni, Franck Dernoncourt, Mark Wagy, and Una-May O'Reilly. Imprecise selection and fitness approximation in a large-scale evolutionary rule based system for blood pressure prediction. In *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation*, pages 153–154. ACM, 2013.

[51] Kenji Hirohata, Naoaki Okazaki, Sophia Ananiadou, Mitsuru Ishizuka, and Manchester Interdisciplinary Biocentre. Identifying sections in scientific abstracts using conditional random fields. In *IJCNLP*, pages 381–388, 2008.

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

[53] Ke-Chun Huang, I-Jen Chiang, Furen Xiao, Chun-Chih Liao, Charles Chih-Ho Liu, and Jau-Min Wong. Pico element detection in medical text without metadata: Are first sentences enough? *Journal of biomedical informatics*, 46(5):940–946, 2013.

[54] Ke-Chun Huang, Charles Chih-Ho Liu, Shung-Shiang Yang, Furen Xiao, Jau-Min Wong, Chun-Chih Liao, and I-Jen Chiang. Classification of pico elements by text features systematically extracted from pubmed abstracts. In *Granular Computing (GrC), 2011 IEEE International Conference on*, pages 279–283. IEEE, 2011.

[55] Adam Janin, Don Baron, Jane Edwards, Dan Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, et al. The ICSI meeting corpus. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 1, pages I–364. IEEE, 2003.

[56] Gang Ji and Jeff Bilmes. Backoff model training using partially observed data: application to dialog act tagging. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 280–287. Association for Computational Linguistics, 2006.

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

[58] Siddhartha R Jonnalagadda, Pawan Goyal, and Mark D Huffman. Automating data extraction in systematic reviews: a systematic review. *Systematic reviews*, 4(1):1, 2015.

[59] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. *JMLR*, 2015.

[60] Dan Jurafsky, Elizabeth Shriberg, and Debra Biasca. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. *Institute of Cognitive Science Technical Report*, pages 97–102, 1997.

[61] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv:1404.2188*, 2014.

[62] Seokhwan Kim, Luis Fernando D'Haro, Rafael E. Banchs, Jason Williams, and Matthew Henderson. Dialog State Tracking Challenge 4: Handbook, 2015.

[63] Seokhwan Kim, Luis Fernando D'Haro, Rafael E. Banchs, Jason Williams, and Matthew Henderson. The Fourth Dialog State Tracking Challenge. In *Proceedings of the 7th International Workshop on Spoken Dialogue Systems (IWSDS)*, 2016.

[64] Su Nam Kim, Lawrence Cavedon, and Timothy Baldwin. Classifying dialogue acts in one-on-one live chats. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 862–871. Association for Computational Linguistics, 2010.

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

[66] 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.

[67] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. *arXiv:1508.06615*, 2015.

[68] Thøger P Krogh, Torkell Ellingsen, Robin Christensen, Pia Jensen, and Ulrich Fredberg. Ultrasound-guided injection therapy of achilles tendinopathy with platelet-rich plasma or saline a randomized, blinded, placebo-controlled trial. *The American journal of sports medicine*, page 0363546516647958, 2016.

[69] 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.

[70] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv:1603.01360*, 2016.

[71] 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.

[72] 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.

[73] Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. MIT at SemEval-2017 Task 10: Relation Extraction with Convolutional Neural Networks. In *Proceedings of the 11th International Workshop on Semantic Evaluations*. Association for Computational Linguistics, 2017.

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

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

[76] Piroska Lendvai and Jeroen Geertzen. Token-based chunking of turn-internal dialogue act sequences. In *Proceedings of the 8th SIGDIAL Workshop on Discourse and Dialogue*, pages 174–181, 2007.

[77] Jimmy Lin, Damianos Karakos, Dina Demner-Fushman, and Sanjeev Khudanpur. Generative content models for structural analysis of medical abstracts. *BioNLP'06 Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis*, 6:65–72, 2006.

[78] Marco Lui. Feature stacking for sentence classification in evidence-based medicine. In *Australasian Language Technology Workshop 2012: ALTA Shared Task*, page 134, 2012.

[79] 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.

[80] Chris Mavergames. The future of knowledge: Cochranetech to 2020 (and beyond). 21st Cochrane Colloquium, 2013.

[81] Larry McKnight and Padmini Srinivasan. Categorization of sentence types in medical abstracts. In *AMIA*, 2003.

[82] Anuj Mehta, Franck Dernoncourt, and Allan Walkey. Trend analysis: Evolution of tidal volume over time for patients receiving invasive mechanical ventilation. In *Secondary Analysis of Electronic Health Records*, pages 275–283. Springer International Publishing, 2016.

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

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

[85] Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Cernocky. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201, 2011.

[86] 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.

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

[88] Naoaki Okazaki. Crfsuite: a fast implementation of conditional random fields (CRFs), 2007.

[89] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English Gigaword fifth edition. Technical report, Linguistic Data Consortium, Philadelphia, 2011.

[90] 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.

[91] 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.

[92] Tom Pollard, Franck Dernoncourt, Samuel Finlayson, and Adrian Velasquez. Data preparation. In *Secondary Analysis of Electronic Health Records*, pages 101–114. Springer International Publishing, 2016.

[93] Silvia Quarteroni, Alexei V Ivanov, and Giuseppe Riccardi. Simultaneous dialog act segmentation and classification from human-human spoken conversations. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5596–5599. IEEE, 2011.

[94] Norbert Reithinger and Martin Klesen. Dialogue act classification using language models. In *EuroSpeech*. Citeseer, 1997.

[95] David Alexander Robinson. Finding patient-oriented evidence in pubmed abstracts. *Athens: University of Georgia*, 2012.

[96] Mihai Rotaru. Dialog act tagging using memory-based learning. *Term project, University of Pittsburgh*, pages 255–276, 2002.

[97] IOM roundtable on evidence-based medicine. Learning what works best: The nation's need for evidence on comparative effectiveness in health care: an issue overview. 2007.

[98] Patrick Ruch, Celia Boyer, Christine Chichester, Imad Tbahriti, Antoine Geissbühler, Paul Fabry, Julien Gobeill, Violaine Pillet, Dietrich Rebholz-Schuhmann, Christian Lovis, et al. Using argumentation to extract key sentences from biomedical abstracts. *International journal of medical informatics*, 76(2):195–200, 2007.

[99] Raymond Francis Sarmiento and Franck Dernoncourt. Improving patient cohort identification using natural language processing. In *Secondary Analysis of Electronic Health Records*, pages 405–417. Springer International Publishing, 2016.

[100] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681, 1997.

[101] Kaveh G Shojania, Margaret Sampson, Mohammed T Ansari, Jun Ji, Steve Doucette, and David Moher. How quickly do systematic reviews go out of date? a survival analysis. *Annals of internal medicine*, 147(4):224–233, 2007.

[102] Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. The ICSI meeting recorder dialog act (MRDA) corpus. Technical report, DTIC Document, 2004.

[103] Joao Silva, Luísa Coheur, Ana Cristina Mendes, and Andreas Wichert. From symbolic to sub-symbolic information in question classification. *Artificial Intelligence Review*, 35(2):137–154, 2011.

[104] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2951–2959. Curran Associates, Inc., 2012.

[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. Citeseer, 2013.

[107] Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373, 2000.

[108] Hendrik Strobelt, Sebastian Gehrmann, Bernd Huber, Hanspeter Pfister, and Alexander M Rush. Visual analysis of hidden state dynamics in recurrent neural networks. *arXiv:1606.07461*, 2016.

[109] Dinoj Surendran and Gina-Anne Levow. Dialog act tagging with support vector machines and hidden markov models. In *INTERSPEECH*, 2006.

[110] Alma Swan. *Policy guidelines for the development and promotion of open access*. UNESCO, 2012.

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

[112] Barry N Taylor and Ambler Thompson. NIST special publication 330 - the international system of units (SI), 2008.

[113] Kay Dickersin Tianjing Li. Introduction to systematic review and meta-analysis. *Coursera*, 2015.

[114] Kalyan Veeramachaneni, Franck Dernoncourt, Colin Taylor, Zachary Pardos, and Una-May O'Reilly. Moocdb: Developing data standards for MOOC data science. In *AIED 2013 Workshops Proceedings Volume*, page 17, 2013.

[115] Kalyan Veeramachaneni, Sherif Halawa, Franck Dernoncourt, Una-May O'Reilly, Colin Taylor, and Chuong Do. Moocdb: Developing standards and systems to support mooc data science. *arXiv:1406.2015*, 2014.

[116] 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.

[117] Sida Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics, 2012.

[118] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards AI-complete question answering: A set of prerequisite toy tasks. *arXiv:1502.05698*, 2015.

[119] Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for machine learning. *the MIT Press*, 2(3):4, 2006.

[120] Joy T. Wu, Franck Dernoncourt, Sebastian Gehrmann, Patrick D. Tyler, Edward T Moseley, Eric T. Carlson, David W. Grant, Yeran Li, Jonathan Welt, and Leo A. Celi. Behind the scenes: A medical natural language processing project. *International Journal of Medical Informatics (IJMI)*, 2018.

[121] Yijun Xiao and Kyunghyun Cho. Efficient character-level document classification by combining convolution and recurrent layers. *arXiv:1602.00367*, 2016.

[122] Yasunori Yamamoto and Toshihisa Takagi. A sentence classification system for multi biomedical literature summarization. In *21st International Conference on Data Engineering Workshops (ICDEW'05)*, pages 1163–1163. IEEE, 2005.

[123] Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv:1212.5701*, 2012.

[124] 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.

[125] Jin Zhao, Praveen Bysani, and Min-Yen Kan. Exploiting classification correlations for the extraction of evidence-based practice information. In *AMIA*, 2012.

[126] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv:1611.01578*, 2016.

Abbreviations

The following abbreviations are used in this thesis proposal:

ANN	Artificial neural network
ALTA	Australasian Language Technology Association
AUC	Area under the curve
AUROC	Area under the receiver operating characteristic curve
CNN	Convolutional Neural Network
CRF	Conditional random field
DSTC 4	Dialog State Tracking Challenge 4 (dataset)
EACL	European Chapter of the Association for Computational Linguistics
EBM	Evidence-based medicine
GP	Gaussian process
GRU	Gated Recurrent Unit
HMM	Hidden Markov Model
i2b2	Informatics for Integrating Biology and the Bedside (dataset)
ICU	Intensive care unit
LR	Logistic Regression
LSTM	Long Short Term Memory network
MEDLINE	Medical Literature Analysis and Retrieval System Online
MRDA	ICSI Meeting Recorder Dialog Act Corpus (dataset)
NAACL	North American Chapter of the Association for Computational Linguistics
NB	Naive Bayes
NICTA	National ICT Australia Ltd
PIBOSO	Population (P), Intervention (I), Background (B), Outcome (O), Study Design (S), and Other (O)
PICO	Population (P), Intervention (I), Comparison(C), and Outcome (O)
RCT	Randomized controlled trial

RNN	Recurrent neural network
ROC	Receiver operating characteristic
SR	Systematic Review
SVM	Support Vector Machines
SwDA	Switchboard Dialog Act Corpus (dataset)