

Instituto Politécnico Nacional

---

---

Centro de Investigación en Computación

TESIS:

Deep-Learning Methods for Mining Social Media

QUE PARA OBTENER EL GRADO DE:

Doctorado en Ciencias de la Computación

P R E S E N T A:

M. en C. Segun Taofeek Aroyehun

Director de tesis:

Dr. Alexander Gelbukh



México, D.F.

Agosto 2021



# INSTITUTO POLITÉCNICO NACIONAL

## SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

### ACTA DE REVISIÓN DE TESIS

En la Ciudad de  siendo las  horas del día  del mes de  del  se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Posgrado del:  para examinar la tesis titulada:  del (la) alumno (a):

Apellido Paterno:	Aroyehun	Apellido Materno:	----	Nombre (s):	Segun Taofeek
-------------------	----------	-------------------	------	-------------	---------------

Número de registro:

Aspirante del Programa Académico de Posgrado:

Una vez que se realizó un análisis de similitud de texto, utilizando el software antiplagio, se encontró que el trabajo de tesis tiene 2 % de similitud. **Se adjunta reporte de software utilizado.**

Después que esta Comisión revisó exhaustivamente el contenido, estructura, intención y ubicación de los textos de la tesis identificados como coincidentes con otros documentos, concluyó que en el presente trabajo **SI**  **NO**  **SE CONSTITUYE UN POSIBLE PLAGIO.**

**JUSTIFICACIÓN DE LA CONCLUSIÓN:** *(Por ejemplo, el % de similitud se localiza en metodologías adecuadamente referidas a fuente original)*


La similitud con fuentes de terceros es baja, sólo 2%. Excluimos de la comparación los artículos del mismo autor (aún con éstos la similitud fue baja) y los textos debidamente citados y claramente identificados como fragmentos citados.

Finalmente, y posterior a la lectura, revisión individual, así como el análisis e intercambio de opiniones, los miembros de la Comisión manifestaron **APROBAR**  **SUSPENDER**  **NO APROBAR**  la tesis por **UNANIMIDAD**  o **MAYORÍA**  en virtud de los motivos siguientes:

La tesis presentada cumple con los índices de calidad establecidos en el Reglamento de Posgrado del IPN.

### COMISIÓN REVISORA DE TESIS

  
Director de Tesis  
Dr. Alexander Gelbukh


  
Dr. Ildar Batyrshin

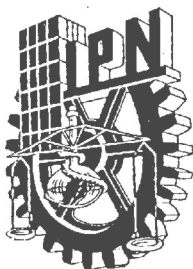
  
Dr. Grigori Sidorov

  
Dr. Marco Antonio Moreno Ibarra

  
Dra. Olga Klesnikova

  
Dr. Luis Manuel Vilches

  
Dr. Marco Antonio Moreno Ibarra  
PRESIDENTE DEL COLEGIO DE PROFESORES DE POSGRADO  
DIRECCION



**INSTITUTO POLITÉCNICO NACIONAL**  
**SECRETARÍA DE INVESTIGACIÓN Y POSGRADO**

**CARTA CESIÓN DE DERECHOS**

En la Ciudad de México el día 2 del mes agosto del año 2021, el (la) que suscribe Segun Taofeek Aroyehun alumno (a) del Programa de Doctorado en Ciencias de la Computación con número de registro B170598, adscrito a Centro de Investigación en Computación, manifiesta que es autor (a) intelectual del presente trabajo de Tesis bajo la dirección de Dr. Alexander Gelbukh y cede los derechos del trabajo intitulado Deep-Learning Methods for Mining Social Media, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección aroyehun.segun@gmail.com. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Segun Taofeek Aroyehun

Nombre y firma

# Resumen

La clasificación de texto en dominios de nicho es un desafío debido a sus requisitos típicos en términos de registro y estilo. En esta tesis, estudiamos la aplicación de redes neuronales profundas al texto en dominios especializados (principalmente redes sociales). El advenimiento de las plataformas de redes sociales ha facilitado el acceso a datos generados por los usuarios que pueden ser útiles para promover el bien social y la preservación de la civilidad en los espacios en línea. En este sentido, presentamos los desafíos y soluciones para la aplicación de redes de aprendizaje profundo a esta fuente de datos para la moderación de contenidos y la vigilancia de la salud pública. Presentamos un enfoque para lidiar con el ruido de los datos de las redes sociales en la tarea de detección de agresiones. Abordamos el desafío del rendimiento multiplataforma de nuestro método propuesto para garantizar la generalización. Para la vigilancia de la salud pública, propusimos un modelo para identificar menciones de medicamentos y reacciones adversas a medicamentos en las redes sociales. Además, evaluamos la idoneidad de los diferentes enfoques para la representación del texto. Finalmente, estudiamos la viabilidad de identificar la valoración en expresiones cotidianas mediante la predicción de dimensiones de juicio.

# Abstract

Text classification in niche domains is challenging because of their typical requirements in terms of register and style. In this thesis, we study the application of deep neural networks to text in specialized domains (mainly social media). The advent of social media platforms has facilitated access to user-generated data which can be useful for advancing societal good and the preservation of civility in online spaces. In this regard, we present the challenges and solutions to the application of deep learning networks to this source of data for content moderation and public health surveillance. We present an approach to deal with the noisiness of social media data on the task of aggression detection. We addressed the challenge of cross-platform performance of our proposed method to ensure generalization. For public health surveillance, we proposed a model to identify mention of drug and adverse drug reaction in social media. Also, we assessed the suitability of the different approaches to text representation. Finally, we study the feasibility of identifying appraisal in everyday expressions by predicting judgement dimensions.

# Acknowledgements

“Let yourself be silently drawn by the strange pull of what you really love. It will not lead you astray.” – Rumi

I thank my advisor, Dr. Alexander Gelbukh, for his patience and guidance over the period of the development of this thesis. I would also like to thank my tutorial committee members for their continuous support.

I am grateful to the CIC for providing the administrative and academic support required for the success of my research.

I am especially thankful to the government and people of Mexico for their support. I appreciate the support from CONACYT and SIP-IPN without which this work will not be a reality.

Many thanks to Google and Microsoft for finding this endeavour worthy of support through their Latin America Research Award programmes.

Last but not least, I am extremely grateful to my family, friends, colleagues, and associates whose company, advice, and encouragement I had to rely on to achieve this feat. Thank you!

# Contents

<b>Resumen</b>	<b>iv</b>
<b>Abstract</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vi</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Why Deep Learning? . . . . .	1
1.2 Objectionable Content on Social Media . . . . .	2
1.3 Pharmacovigilance in the Age of Social Media . . . . .	2
1.4 Contributions . . . . .	3
1.5 Structure of the Rest of this Document . . . . .	4
<b>2 Theoretical Framework</b>	<b>6</b>
2.1 Description of Tasks . . . . .	6
2.2 Text Pre-Processing . . . . .	7
2.3 Sequence Representation . . . . .	8
2.4 Neural Network Architectures . . . . .	8
2.4.1 Recurrent neural networks . . . . .	8
2.4.2 Convolutional neural networks . . . . .	9

2.4.3	Transformer and Language models . . . . .	9
2.5	Classification Techniques . . . . .	11
2.6	Data Augmentation . . . . .	12
2.7	Dropout . . . . .	13
2.8	Evaluation metrics . . . . .	13
<b>3</b>	<b>Deep Learning for Patent Classification</b>	<b>16</b>
3.1	Introduction . . . . .	16
3.2	Related work . . . . .	19
3.2.1	Transfer Learning . . . . .	20
3.2.2	Multi-task learning . . . . .	20
3.2.3	Vocabulary selection . . . . .	21
3.2.4	Layer normalization . . . . .	22
3.2.5	Use of additional information for text classification . . . . .	22
3.3	Hierarchical transfer approaches . . . . .	23
3.3.1	Base model (PEncoder) . . . . .	23
3.3.2	Transfer learning . . . . .	24
3.3.3	Multi-task learning . . . . .	25
3.3.4	Combination of multi-task and transfer learning . . . . .	26
3.3.5	Training . . . . .	26
3.4	Experimental Results . . . . .	29
3.4.1	Datasets . . . . .	30
3.4.2	Metrics . . . . .	31
3.4.3	Comparison of model performance . . . . .	32
3.5	Discussion . . . . .	34
3.5.1	Effect on less frequent labels . . . . .	36
3.5.2	Ablation study of the regularization techniques . . . . .	36
3.5.3	Qualitative error analysis . . . . .	38
3.5.4	Does the choice of word embeddings matter? . . . . .	39
3.6	Conclusion . . . . .	40



<b>4</b>	<b>Detection of Aggression on Social Media</b>	<b>42</b>
4.1	Introduction . . . . .	42
4.2	Related Work . . . . .	43
4.3	Data . . . . .	44
4.4	Methodology . . . . .	46
4.5	Results . . . . .	50
4.6	Conclusion . . . . .	53
<b>5</b>	<b>All-in-one Model for Multilingual Offensive Language Detection</b>	<b>55</b>
5.1	Introduction . . . . .	55
5.2	Related Work . . . . .	56
5.3	Methodology . . . . .	57
5.4	Results . . . . .	58
5.5	Conclusion . . . . .	59
<b>6</b>	<b>Automatic Identification of Social Media Posts Mentioning Drugs and Adverse Drug Reactions</b>	<b>60</b>
6.1	Introduction . . . . .	60
6.2	Data . . . . .	61
6.3	Method . . . . .	61
6.4	Results . . . . .	62
6.5	Conclusion . . . . .	63
<b>7</b>	<b>Comparison of Word Embeddings for the Detection of Adverse Drug Reaction in Social Media Messages</b>	<b>64</b>
7.1	Introduction . . . . .	64
7.2	Model and Experimental Set-Up . . . . .	65
7.3	Results . . . . .	66
7.4	Conclusion . . . . .	68
<b>8</b>	<b>Automatic Prediction of the Assessment Dimensions of Human Behavior</b>	<b>69</b>

8.1	Methodology . . . . .	71
8.2	Results . . . . .	73
8.3	Conclusion . . . . .	73
<b>9</b>	<b>Conclusion and Future Work</b>	<b>75</b>
	<b>Bibliography</b>	<b>77</b>

# List of Figures

2.1	A Chain of Recurrent Neural Network Units . . . . .	9
2.2	Convolutional Neural Network for Sentence Classification (Kim, 2014a)	10
2.3	The Transformer Encoder Block as proposed in (Vaswani et al., 2017)	11
2.4	Confusion matrix with each cell depicting possible evaluation outcome	13
3.1	The International Patent Classification scheme, a hierarchical taxonomy for categorizing the subject of an invention. . . . .	18
3.2	Base model: PEncoder. . . . .	24
3.3	Transfer learning setting: $TL_{sec}$ (section label Classifier <sub>sec</sub> and ) or $TL_{cls}$ (class label Classifier <sub>cls</sub> ). . . . .	25
3.4	Multi-task learning setup with one auxiliary task: $MTL_{sec}$ or $MTL_{cls}$ .	26
3.5	Multi-task learning setup with two auxiliary tasks: $MTL_{sec+cls}$ . . . . .	27
3.6	Combination of transfer learning and multi-task learning with an auxiliary task: $MTL_{sec} + TL_{sec}$ , $MTL_{sec} + TL_{cls}$ , $MTL_{cls} + TL_{sec}$ , and $MTL_{cls} + TL_{cls}$ . . . . .	28
3.7	Combination of transfer learning and multi-task learning with more than one auxiliary task: $MTL_{sec+cls} + TL_{sec}$ and $MTL_{sec+cls} + TL_{cls}$ . . . . .	29
3.8	Micro-average precision evaluation metric for three conditions: top prediction, three guesses, and all categories (adapted from (Fall et al., 2003)). $\hat{y}_n$ is for the $n$ -th best prediction, MC is the main subclass label, and IC is the incidental subclass label. . . . .	31
4.1	Confusion Matrix of the LSTM Predictions on the English (Facebook) Test Set . . . . .	51

4.2	Confusion Matrix of the CNN-LSTM Predictions on the English (Social Media) Test Set . . . . .	52
-----	---	----

# List of Tables

3.1	Summary of the dataset, train and test splits, in the WIPO-alpha dataset and in the part of the USPTO-2M dataset we used. <i>Subclasses</i> column show the number of unique labels at the subclass level; <i>Train</i> and <i>Test</i> respectively indicate the total number of documents in the train and test subsets. . . . .	30
3.2	Model performance for classification at the subclass level on WIPO-alpha test set. TP is top prediction, AC is all categories, and TG is three guesses. The best score is in bold; the second best score is underlined. . . . .	32
3.3	Model performance for classification at the subclass level on USPTO-2M (the part we used) test set. TP is top prediction, AC is all categories, and TG is three guesses. The best score is in bold; the second best score is underlined. . . . .	33
3.4	Statistics of the labels on the WIPO-alpha and USPTO-2M (used part) training splits indicating the minimum, 25th percentile, 50th percentile, 75th percentile, maximum, and average number of examples per label. The OOV column is the number of out-of-vocabulary words with respect to the embeddings for the patent domain. . . . .	35
3.5	Performance comparison using the top prediction (TP) MAP metric, on the 50 least frequently occurring labels in WIPO-alpha. . . . .	36
3.6	Contingency table highlighting the differences between PEncoder and the $MTL_{sec} + TL_{cls}$ model on the 50 least frequently occurring labels in WIPO-alpha. . . . .	37

3.7	Ablation study of the regularization techniques used in the PEncoder (first row), on the WIPO-alpha and USPTO-2M datasets. The metric is the micro-average precision of the top prediction (TP). . . . .	37
3.8	Performance comparison of PEncoder on WIPO-alpha when using different types of word embeddings. . . . .	40
4.1	Weighted F1 Scores on the English Development set. PL stands for Pseudo Labeling and Aug. represents Augmentation . . . . .	49
4.2	Weighted F1 Scores on the English (Facebook) Test set . . . . .	51
4.3	Weighted Macro-F1 Scores on the English (Social Media) Test set . . . . .	52
5.1	Details of the dataset for subtasks A and B for each language. Total is the number of labeled examples per language. OFF – Offensive, and PRFN – profane . . . . .	57
5.2	F1 score on the test set. The numbers in parentheses represent the difference in performance between our submission and the best model on the leaderboard. . . . .	59
6.1	Number of Examples in the Train and Test Sets for Tasks 1 and 3 . . . . .	61
6.2	F1 Score of the Positive Class on our Development Split of the Training set using NBSVM and Deep Learning Models (For deep learning models, scores are the average of three runs and the values in parentheses are for the corresponding character level model) . . . . .	62
6.3	Scores on the Evaluation Data for Task 1 (P-Precision; R-Recall; F-F1 measure) . . . . .	63
6.4	Scores on the Evaluation Data for Task 3 (P-Precision for the ADR class; R-Recall for the ADR class; F-F1 measure for the ADR class) . . . . .	63
7.1	Details of the Data . . . . .	66
7.2	Performance on the Test Set . . . . .	67
7.3	Ablation Study on the Validation Split . . . . .	67

8.1	Frequency of each label in the training set as a fraction of the total number of examples. . . . .	70
8.2	Mean F1 score on the public and private test sets. Average is the unweighted mean of the scores on the private and public leaderboards as they are approximately 50% each of the test set. . . . .	72





# Chapter One

## Introduction

This chapter outlines the background to this thesis. It presents the motivation, relevance, and the challenges tackled in this thesis.

### 1.1 Why Deep Learning?

To solve the problem of extracting useful information from social media text, traditional systems have used various pipelines to generate feature representations that are then fed into machine learning models to perform sequence classification. These pipelines often include various natural language processing (NLP) modules and resources to extract different linguistic features, which hopefully will capture important features for the target tasks. Determining which NLP modules and resources to use and which features to extract is known as feature engineering. There are several limitations of feature engineering.

Firstly, the process is manual, expensive, and requires linguistic intuition. The extracted features are generally not versatile enough to be applied to any other domain or data. Closely related to the first point is the limited coverage of feature sets. when the model encounters unseen words during training, the model can fail on such examples because most of the features are discrete representations which requires partial and/or complete matching. Another limitation is that the feature sets might be sub-optimal due to limited domain knowledge or omission of important characteristics. In addition, the difficulty of realizing possible interaction among

feature sets could lead to information redundancy. Finally, the modules used in generating the features may be error-prone (or non-existent in languages other than English which might necessitate the use of ad-hoc methods or heuristics) resulting in less effective feature sets that can affect the performance of the target task.

In this work, we address the limitations above by adapting recent developments in deep learning (DL) for social media mining. DL has the huge advantage of automatically inducing effective feature representation from data. Thus, it requires a sizable amount of data for effective representation learning. The two main areas of focus are on the identification and classification of objectionable content and pharmacovigilance (the monitoring of adverse drug reactions).

## 1.2 Objectionable Content on Social Media

objectionable content poses a huge challenge to the stability and progress of society. Such contents are of various forms (insult, hate, racism, sexism, derogatory/condescending remarks) which are directed from one person to another person (or group) and at the same time it is in the public domain (for others to consume or act on) which fuels a culture of hatred, intolerance, and public discourse going awry (exclusion in public discourse which is leading to the emergence of "echo chambers"). The latter hurts the process of public opinion forming that requires rational and critical examination of opposing viewpoints. No open society can thrive where such a culture is the order of the day. As such, it's become imperative for social media and technology companies to develop systems and processes to ensure civil conduct of its users devoid of abusive language and behaviour.

## 1.3 Pharmacovigilance in the Age of Social Media

Pharmacovigilance is the detection, assessment, understanding and prevention of adverse effects related to the administration of drugs (WHO, 2021). Adverse drug reactions (ADR) are adverse patient outcomes caused by medications. Globally,

adverse drug reactions are responsible for morbidity and mortality. A number of ADRs are unreported (Sloane et al., 2015). Users of the world wide web use it to meet their health information needs and social media is an expansive component of the read-write web. Social media has been utilized in domains such as disaster response, sentiment analysis of product reviews, trend analysis of disease outbreaks and financial assets among others. Hence, social media has potential for pharmacovigilance applications. The amount of data and the real-time nature of social media can allow for accelerated monitoring, detection and reporting of ADR. User posts on social media contain information on outcomes of drug administration, potentially providing early access to reports of ADR. This kind of user-generated, unsolicited and recent information may not be easily accessible by other means. The volume of data has made social media a useful resource for ADR monitoring. With the volume comes the challenge of identifying the most relevant data.

There are several technical challenges to the realization of these benefits. Among these challenges are the informal nature of social media, the difference between layperson description and medical professionals' description of drug-related terms, and the need for large-scale annotated data to train accurate supervised machine learning models which is expensive and time-consuming. In addition, only a small proportion of data related to drugs tends to contain information associated with ADRs (Sarker et al., 2015). NLP research present an opportunity to harness the huge amount of data for the benefit of public health.

## 1.4 Contributions

This thesis made the following contributions:

- Proposed approaches to leverage label information in a deep neural network classifier for a niche domain (patent text) which led to performance gains.
- Improved classification of objectionable content via the combination of data augmentation and pseudo labeling.

- Demonstrated the effectiveness of developing a single competitive model to detect objectionable content in multiple languages by building on recent advances in distributed multilingual representations.
- Demonstrated that the combination of heterogeneous embeddings provides performance gains for detection of adverse drug reactions in social media data.
- Proposed a hybrid approach for identification of adverse drug reactions – a deep learning model is used for learning feature representation and standard machine learning classifier is employed for the classification decision in a non-end-to-end fashion.
- Achieved the state-of-the-art performance on the identification of appraisal judgement dimensions using a recent language model representation with decision thresholding.

## 1.5 Structure of the Rest of this Document

- Chapter 2 – outlines fundamental concepts employed in subsequent chapters
- Chapter 4 – describes the improved classification model for the identification of objectionable content
- Chapter 6 – describes the proposed hybrid approach for identification of adverse drug reactions
- Chapter 7 – presents the analyses of the combination of heterogeneous embeddings
- Chapter 5 – describes the all-in-one model for the identification objectionable content across languages
- Chapter 8 – presents state-of-the-art model for the identification judgement appraisal dimension in social media posts

- Chapter 9 – ends the thesis with a summary of the contributions and avenues for future work.

# Chapter Two

## Theoretical Framework

This chapter provides the background necessary for the subsequent chapters. It provides a description of the tasks covered in this thesis followed by text pre-processing, sequence representation approaches, and neural network architectures. Then, it outlines classification and regularization techniques. It closes with a description of relevant evaluation metrics.

### 2.1 Description of Tasks

**Detection and classification of objectionable content.** A multiclass classification of social media text into one of three categories.

**Detection of adverse drug reaction in social media posts.** A binary classification of posts as to whether they contain drug name(s) and a binary classification of posts to determine whether they contain mention of adverse drug reactions or not.

**Identification of appraisal judgement dimensions.** Given a short text, predict one or more judgement dimensions expressed in the given text. This is a multilabel classification problem where the labels consist of the five judgement dimensions.

## 2.2 Text Pre-Processing

Text pre-processing aims to make the input text more consistent (predictable and analyzable) to facilitate text representation. As social media text is particularly noisy, extensive pre-processing steps require well-developed tools the availability of which varies across languages. Notably, English is well-supported with tools such as Ekphrasis (Baziotis et al., 2017). Standard text pre-processing methods include tokenization, stop word removal, lemmatization, stemming, and normalization.

Tokenization is the splitting up of input text (which is one long sequence) into sub-units referred to as tokens. Tokens can be defined at different levels of granularity: fine-grained and coarse-grained. At the fine-grained level, tokens can be in the form of characters, sub-words, or words. Coarse-grained tokens consist of multiple fine-grained tokens such as n-grams, phrases, and multi-word expressions. These sub-units constitute the input to subsequent text processing pipelines. Stop word removal eliminates words using a stop word list, (words which are considered more general and bears no meaning). Examples in English include “a”, “the”, and “is”. Stemming reduces inflected (or derived) words to their stem, base or root. The stemmed equivalent may not be a valid word (Consumes, consumed → consume). It helps with standardizing the vocabulary. Text normalization is a technique useful for transforming noisy text (such as social media data) which contains abbreviations, misspellings, emoticons (or emojis), user handles, URLs, and out-of-vocabulary words into a canonical form. There are several techniques for text normalization the most common approaches include dictionary mappings and spelling correction. (u → you; www.example.com → URL; @example, @example1 → @user). The presence of punctuation marks, digits, and special characters can have an impact on text representation as such their removal is usually essential. The need for and the nature of text pre-processing can vary from task to task. Also, the sequencing of text pre-processing needs to be carefully chosen depending on the nature of the datasets. For example, stemming followed by punctuation removal on “consumes..” will give “consumes..”, whereas the removal of punctuation marks followed by stemming of “consumes..” will result in “consume”. A potential outcome of aggressive pre-processing is modification of the

sentence structure which can lead to loss of syntactic information and the intended meaning.

## 2.3 Sequence Representation

Word vector representations and their composition for classification. Word vector is a projection from a sparse one-hot encoding representation to a lower dimensional vector space. The dimensions capture semantic properties of words such that semantically similar words are close in the vector space. When pre-trained on large enough text collection, the word vectors are considered "universal feature extractors" that can be applied to a variety of downstream tasks. It is well-known that the unsupervised pre-training of word vectors is essential for the success of deep learning for natural language processing.

Distributed representation of sequence use real-valued vectors to represent meaning. Typical examples are bag-of-words (BOW) and sequence representations. In BOW, representations do not take into account word order e.g, by averaging word embeddings or aggregating TF-IDF weights. On the contrary, sequence representation relies on the order of the sequence of tokens to generate a representation. Models that are oblivious of the sequence order are not able to reflect changes in meaning of natural language due to simple permutation of word order (e.g. cats eat mice vs mice eat cats).

## 2.4 Neural Network Architectures

### 2.4.1 Recurrent neural networks

Recurrent neural networks (RNNs) are able to process input sequences by applying recursively a transition function on the hidden state vector at a given time step. The hidden state vector is a function of the current input vector and the previous hidden state vector. The hidden state vector at a particular time step can be considered as the contextual distributed representation of the sequence of tokens upto the current



time step. RNNs are suitable for modeling sequences. RNNs with Long Short-Term Memory (LSTM) units (Hochreiter and Schmidhuber, 1997a) are effective for modeling long-range dependencies. Hence their wide application in natural language processing to harness their representational power for sequence modeling and classification.

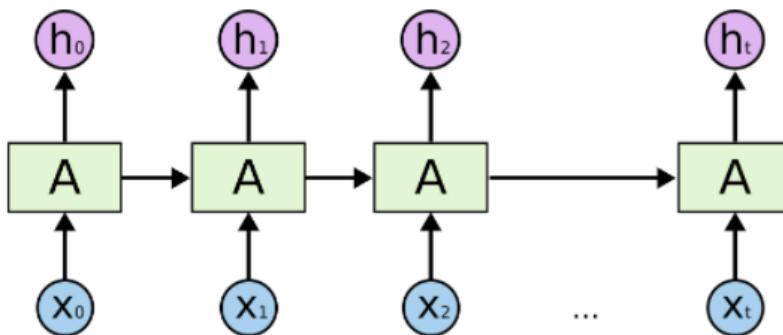


Figure 2.1: A Chain of Recurrent Neural Network Units

## 2.4.2 Convolutional neural networks

Convolutional neural networks (CNN) consist of convolving filters which process local features (LeCun et al., 1998). A convolution operation consists of a filter applied to a window of tokens to produce a feature. The application of the filter to all possible window of tokens generate a feature map. To capture the most important feature, a maximum operation is applied on the feature map to yield the salient feature representation for the filter. In practice, multiple filters are used to generate multiple features. The combination of these features constitute the input to the classification layer. Figure 2.2 illustrates a convolutional neural network.

## 2.4.3 Transformer and Language models

Transformer is a simple network architecture based on attention mechanisms which compute an attention score to capture the influence each word has on every other word and vice-versa in the case of self-attention. It does without recurrence and convolution operations in RNNs and CNN which are less amenable to parallelization and require

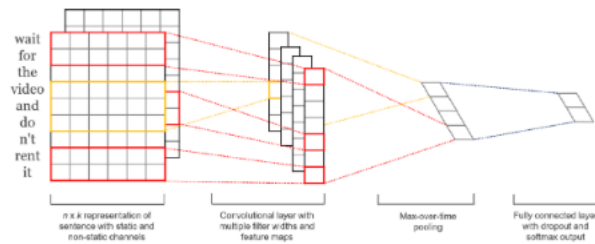


Figure 2.2: Convolutional Neural Network for Sentence Classification (Kim, 2014a)

significantly more time to train on large datasets. With this, transformer allows for the training of big models on a massive collection of data on several massively parallel hardware such as GPUs and TPUs.

A transformer encoder block shown in Figure 2.3 is a stack of two components: self-attention and position-wise feed-forward neural network. Also, there is a residual connection around each component with layer normalization. Since the transformer architecture does not contain recurrence or convolution, the position encodings is added to the input embeddings to introduce position information of the tokens in a sequence. Transformer-based language models (LMs) consist of  $N$  layers of transformer blocks. For example, the  $BERT_{base}$  architecture consists of 12 transformer layers while the  $BERT_{large}$  model consist of 24 transformer layers (Devlin et al., 2019).

The pre-training step: training is done with different pre-training objectives. BERT was trained with two unsupervised objectives: masked LM and next sentence prediction (NSP). The masked LM objective predict masked tokens given their left and right contexts. The NSP aims to incorporate the relationship between two sequences in the LM. This is achieved with a binary classification over a pair of sequence. The pre-training step is expensive in time and resources required. The BERT model pre-training data was extended to include text in multiple languages resulting in multilingual BERT.

The fine-tuning step: the mode is initialized with the pre-trained weights and all the weights are optimized using labelled data for the target task(s). In contrast to pre-training, the finetuning step is relatively less expensive.

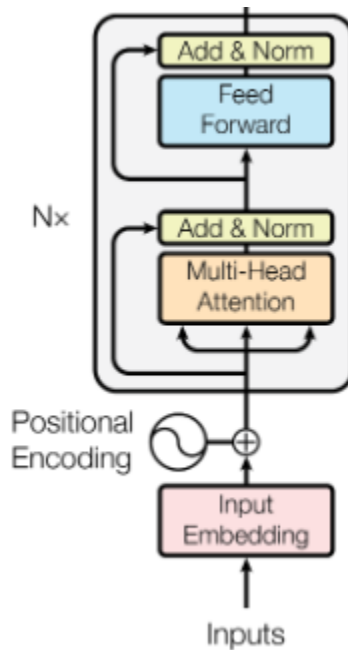


Figure 2.3: The Transformer Encoder Block as proposed in (Vaswani et al., 2017)

## 2.5 Classification Techniques

**Supervised learning.** Given a set of data points consisting of some inputs and corresponding output(s), supervised learning aims to learn a function that transforms input data to output value(s) using the labeled data. The expectation is that the function can estimate with high accuracy the output value(s) on unseen inputs.

**Unsupervised learning.** Unsupervised learning train systems that learn to represent particular inputs in a way that reflects the underlying statistical structure of the inputs. There is no explicit output associated with each entry in the inputs. Unsupervised learning methods work with only the observed input pattern and some prior explicit or implicit information about what is important. The resulting output is some form of hidden knowledge such as clusters, topics, and latent representations.

**Semi-supervised learning.** When faced with practical classification problems, it often happens that we have access to a collection of data with unknown labels. Semi-supervised learning aims to use unlabeled data to train a classifier that will perform

better than a classifier that relies only on labeled data. Semi-supervised classification methods are useful in scenarios where there is a large amount of unlabeled data and a small amount of labeled data. This happens when access to or construction of labeled data is expensive and/or difficult. In a scenario where labeled data is available for learning reliable classifiers, and unlabeled data can provide additional information, using them can improve classification performance.

## 2.6 Data Augmentation

Data augmentation refers to strategies to increase the diversity of training examples without collecting new data. Most data augmentation techniques generate synthetic examples by modifying the original data. The goal is for the augmented data to act as a regularizer by mitigating overfitting in optimizing machine learning models. As the study of natural language processing progresses, there are more and more tasks, domains, and languages to explore, many of which are resource-poor. It is difficult to find large datasets for training reliable models. In supervised classification, the newly generated data is expected to preserve the meaning and label of the original data from which it was derived. Therefore, successful data augmentation techniques must balance the fidelity of the generated samples and their variability (not too similar and not too different). Data augmentation techniques vary in their complexity. One simple approach that is easy to implement is easy data augmentation (Wei and Zou, 2019). It consists of simple token-level manipulations: synonym replacement, random insertion, random swap, and random deletion. This approach mainly introduces noise. We hypothesize that for a noisy domain as social media, the application of such an approach will significantly impact model performance. A more involved technique is backtranslation (Sennrich et al., 2016). It translates a sequence of text into an intermediate language and then back into the original language. This assumes that there exists accurate translation models. In effect, this technique generates a paraphrase of the original examples. Depending on the accuracy of the translation models, it should generate less noise than easy data augmentation.

		Actual Class	
		p	n
Predicted Class	Y	True Positives	False Positives
	N	False Negatives	True Negatives
Totals:		P	N

Figure 2.4: Confusion matrix with each cell depicting possible evaluation outcome

## 2.7 Dropout

Deep learning models with multiple hidden layers can learn complicated input-output relationships. However, they tend to overfit the training data when the data is of limited size. The result is a poor performance on unseen test data. One method to overcome overfitting is dropout which randomly sets to zero some units along with their connections in the network architecture during training. The key idea is to prevent the co-adaptation among the units (Srivastava et al., 2014).

## 2.8 Evaluation metrics

**Confusion matrix:** It is a tabular representation for assessing the performance of a classification model. A comparison is made between the ground truth (actual) values and the predicted values generated by the model.

The possible evaluation outcomes in Figure 2.4 are defined below:

- True positives (TP) predicted true and actually true
- False positives (FP): predicted true and actually false
- True negatives (TN): predicted false and actually false
- False negatives (FN): predicted false and actually true

We can compute the following metrics using the confusion matrix.

**Accuracy:** The fraction of all observations that the system labels correctly.

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

With unbalanced datasets, accuracy is problematic. It is a poor metric when the samples of interest are those that belong to less-frequent categories.

**Precision:** The fraction of samples predicted as positive that are actually positive for a particular class.

$$precision = \frac{TP}{TP + FP}$$

It maximizes the possibility that all positive predictions are indeed true, with the risk of predicting some positive samples as false.

**Recall:** The fraction of samples that are actually positive that are predicted to be positive for a specific class.

$$recall = \frac{TP}{TP + FN}$$

It maximizes the possibility that all positive samples are predicted to be positive, with the risk of predicting many samples as false.

**F-score:** It combines both precision and recall with harmonic mean:

$$F_{\beta} = \frac{(1 + \beta^2).precision.recall}{(\beta^2.precision) + recall}$$

The  $\beta$  parameter gives weight to the relative importance of precision over recall:

$$\left\{ \begin{array}{l} \text{if } \beta < 1, \text{precision is more important} \\ \text{if } \beta = 1, \text{precision and recall are equally important} \\ \text{if } \beta > 1, \text{recall is more important} \end{array} \right.$$

**Aggregation of class-specific metrics:** When samples can be classified into more than one label out of  $N$  ( $N > 2$  labels (multilabel classification) or only one out of  $N$  labels (multiclass classification). The overall score across all classes can be derived by with an average over each of the class-specific metrics. This can be done in one of three ways:

- micro-averaging: is calculated as the average of the corresponding TP, FP, TN, and FN for each class to derive the aggregated metric.
- macro-averaging: calculate the performance for each class and take a simple average.
- weighted-averaging: similar to macro-averaging except that weights are used when computing average. The weights for the different classes are usually proportional to the number of examples in the test set (the support) per class. This is suitable when there is class imbalance.

# Chapter Three

## Deep Learning for Patent Classification

### 3.1 Introduction

Text classification is the task of assigning tags to a given text that adequately represents its meaning. The representation of the sequence of tokens constituting a piece of text is the input to a natural language processing (NLP) model. Traditional approaches use a representation that is sparse, usually manually crafted features such as bag-of-words and n-grams. Recently, neural approaches learn text representation using architectures such as convolutional neural network (CNN) and recurrent neural network (RNN) to compose text sequence into a fixed length representation. Neural models have been shown to perform better on various NLP tasks. Word embeddings which capture the relationships (syntactic, semantic, and pragmatic) among words in vector space drives the performance of neural models. Typically, word embeddings are pre-trained on a large text collection. The resulting word representations are transferred to downstream tasks. In addition, performance on downstream tasks is typically determined by the similarity of the source corpus with the target task datasets. To improve the performance of classification models, one can add external information that cannot be derived from the text sequence. The structural arrangement of labels is a source of such information. A meaningful label hierarchy is an additional source of information relevant for classification and cannot be derived from the raw text:



for example, it can be an indication that examples from different classes share some common attributes. This extra information can be useful when the training data is limited.

Patent applications are examined for their novelty, non-obviousness, and usefulness. This requires the evaluation of subject-matter experts. Patent offices route applications to subject-matter experts according to the subject of the invention based on a standard hierarchical classification scheme. One commonly used scheme is the International Patent Classification (IPC). It is a standard hierarchical taxonomy organized into levels. The highest is the section level. Each section is divided into classes; each class is divided into subclasses; subclasses include groups and a hierarchy of subgroups. For example, the IPC code G06N 5/02 refers to section G, class G06, subclass G06N, group G06N 5, and subgroup G06N 5/02; see Figure 3.1. The tag at each level has a description, e.g., the description of the subclass G06N 5 is *Computer Systems Based on Specific Computational Models*. The IPC scheme, administered by the World Intellectual Property Organization (WIPO), is subject to constant reviews. However, the higher levels – section, class, and subclass – are fairly stable. The current version of the IPC<sup>1</sup> consists of 8 sections, 131 classes, 646 subclasses, 7518 main groups, and 68030 subgroups. As the depth of the taxonomy increases, the number of categories grows and each category becomes more sparse, that is, the number of documents per category decreases.

Given a hierarchical tree-like taxonomy of labels, such as IPC, parent-child relationship between the upper and lower level categories can be used to improve model performance at the lower, more granular levels. In fact, this is very similar to the recommendation for human experts who assign IPC tags to patents following the guidelines for the assignment of IPC tags. The manual (World Intellectual Property Organization, 2020) suggests following the hierarchy step by step. To assign the subclass of the subject of an invention, it is advised to first identify the relevant section, followed by the most suitable class and subclass. Indeed, to solve complex new problems, humans apply the skills acquired in solving related problems. Also, to learn

---

<sup>1</sup>[www.wipo.int/classifications/ipc/en/ITsupport/Version20200101/transformations/stats.html](http://www.wipo.int/classifications/ipc/en/ITsupport/Version20200101/transformations/stats.html)

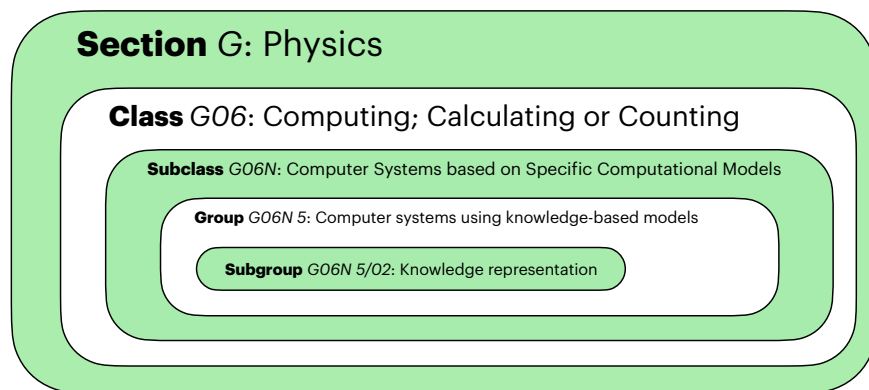


Figure 3.1: The International Patent Classification scheme, a hierarchical taxonomy for categorizing the subject of an invention.

complex concepts, we learn simple concepts first. We believe that the upper levels of the categorization hierarchy are simple. With this a coarse-grained representation of the document can be learned, which can be used at the lower levels.

Classification at the lower levels is particularly difficult due to the large number of labels. Furthermore, the distribution of the number of samples per category is not uniform, with some categories having more data and others with fewer examples. Therefore, automatic categorization faces problems of data sparsity and class imbalance. An approach that transfers knowledge from a higher level of the hierarchy will be of advantage to the model at the lower level by providing a useful initialization for model parameters rather than starting with random weights. Given the rapid progress in various fields of science and technology, the number of patent applications is increasing dramatically. Consequently, there is a need for more effective automated systems to help process and analyze patent applications.

This chapter examines how information obtained from the label hierarchy can be used in a deep neural network to classify patent documents. The categorization of patent documents is a multiclass classification problem at each level of the standard taxonomy. Existing patent classification studies have used conventional machine learning models such as k-Nearest Neighbour, Support Vector Machines, Naïve Bayes, and Artificial Neural Network (Fall et al., 2003; Guyot et al., 2010). Within these

approaches, features are extracted with bag-of-words, term frequency, n-grams or phrases (D’hondt et al., 2013) to represent a document. The representations so derived, are known to largely ignore the semantics and word order in the document. They mainly depend on frequency of occurrence and/or co-occurrence. Progress in deep neural networks on training word embeddings and architectures such as CNN (Kim, 2014b) and RNN (Cho et al., 2014; Graves et al., 2013; Hochreiter and Schmidhuber, 1997b) address these shortcomings. Recent papers by S. Li et al., (2018) and Risch and Krestel, (2018), who used CNN and GRU respectively, are examples of using deep-learning techniques to classify patent documents. For this task, we first present a neural network architecture that performs better than existing ones. Our model is based on the GRU architecture (Cho et al., 2014) combined with layer normalization, a regularization method proposed by J. L. Ba et al., (2016), and vocabulary size selection with frequency cut-off. We further improve the performance by transferring knowledge along the taxonomy of labels using transfer learning, multi-task learning, and a combination of both. Applying this approach using another architecture should be straightforward.

We evaluate the proposed approach on two datasets and compare with the current state-of-the-art model. As far as we know, previous work did not consider the transfer of knowledge based on a categorization scheme for the classification of patent documents. Our method should be applicable to a wide range of other classification problems, for example, for automatic categorization based on hierarchically controlled vocabularies for information management in medicine and law. More research is needed to evaluate its usefulness in such areas.

## 3.2 Related work

There are mainly two approaches to knowledge transfer: transfer learning and multi-task learning. Here, the applications of these two techniques to natural language processing are briefly considered. Next, we describe two regularization techniques: layer normalization and vocabulary selection. Finally, we highlight relevant approaches that seek to consider additional information to boost performance of text classification.

### 3.2.1 Transfer Learning

Transfer learning is a technique that uses knowledge derived from learning source task(s) to solve a different but related target task(s) in order to improve performance. Pre-trained word embeddings (Bojanowski et al., 2016; Chaturvedi et al., 2016; Mikolov, Sutskever, et al., 2013; Pennington et al., 2014a) and language models (Devlin et al., 2019; Peters et al., 2018) are known to improve performance on various downstream tasks. Mou et al., (2016) investigated the use of transfer learning in deep neural network models for natural language processing and showed that transfer learning improves model performance. Al-Stouhi and Reddy, (2016) showed that transfer learning can improve predictive performance when there is class imbalance using conventional machine learning methods. Howard and Ruder, (2018a) used language model pre-training as source task to improve text classification by fine-tuning with the target dataset. They introduced the idea of discriminative learning rates in the fine-tuning phase, where layers that are initialized via pre-training are optimized with a learning rate smaller than other layers in the network. This is to avoid catastrophic forgetting. This approach is a middle ground between freezing layers and updating all the weights using the target task dataset. Usually, the learning rate for the network is multiplied by a scaling factor to obtain learning rates for the pre-trained layers. In contrast, we used the supervision that is naturally available in the label hierarchy to select our source tasks, classification in this case.

### 3.2.2 Multi-task learning

Multi-task learning is pioneered by Caruana, (1997) with the aim of using signals from related tasks to improve generalization of machine learning models. This is based on hard or soft parameter sharing of hidden representation (Ruder, 2017). Hard parameter sharing uses the same hidden layers across all tasks and each task only has its specific prediction head. In contrast, with soft parameter sharing, each task is independent to the extent that it maintains its own model and/or parameters, while the interaction among tasks is achieved by enforcing similarity of parameters. Hard parameter sharing is efficient and avoids the problem of overfitting (Baxter, 1997).

However, it can lead to the problem of negative transfer. For efficiency and simplicity, our experiments with multi-task learning are based on hard parameter sharing.

Multi-task learning is a multi-objective method: it aims to solve a set of tasks in parallel by combining the objectives of each task. For this combination, weights are used to determine the importance of each task. Optimal weights for each task is critical for the effectiveness of multi-task learning. Finding the optimal values requires a lot of computational resources and time. Hence, it is desirable to learn these weights as part of the parameters for the model. Kendall et al., (2018) suggests a multi-task objective that takes into account the uncertainty of each task to adjust the relative weight. Although this idea was originally proposed for computer-vision tasks, there are examples of its usage in natural language processing, such as J. Choi et al., (2019). We experiment with this relative weighting approach and simple average.

Søgaard and Goldberg, (2016) arrange tasks in terms of a hierarchy that follows different levels of linguistic processing. They recommend the use of low-level auxiliary tasks such as part-of-speech tagging, parsing, and named entity recognition at lower layers of deep neural networks. Similarly, we implicitly use a hierarchy of tasks through transfer learning where weights learned from pre-training at a coarse level in the label hierarchy is transferred to a lower level.

### 3.2.3 Vocabulary selection

The size of the vocabulary is of interest to researchers who use neural networks for natural language processing as the embedding layer is a significant part of the size of the model parameters. Excluding words that contribute very little to solving the task at hand can reduce the size and complexity of the model. A common heuristics is to keep the most frequent  $K$  words and map other words to a unique token *UNK* (Jean et al., 2015; Luong et al., 2015). Chen et al., (2019) conducted a systematic study to empirically show that vocabulary size affects model performance on classification tasks. In the end, the authors proposed a sophisticated task-dependent solution to determine the optimal vocabulary size. In this chapter, we use the simpler frequency-based approach by ranking words in the vocabulary using frequency of occurrence

and apply a cut-off frequency threshold.

### **3.2.4 Layer normalization**

It is a method to normalize the activation levels of neurons in a layer so as to stabilize hidden-state activation levels in a Recurrent Neural Network (RNN). J. L. Ba et al., (2016) experimentally confirmed the effectiveness of layer normalization on RNN – reduction in training time and better generalization.

### **3.2.5 Use of additional information for text classification**

The source of additional information different from the raw text can be either from the label set or an existing knowledge base. The label set can be meaningful in its structure or description. The corresponding structural, semantic, and conceptual information can improve classification models. There are three main approaches of using additional information to improve text classification. The first approach is based on learning better representation using hybrid models that combine implicit information from raw text with explicit background knowledge in knowledge bases (Ma et al., 2018; J. Wang et al., 2017). Also in (G. Wang et al., 2018), the authors used the semantics of the labels to learn a label-embedding attentive representation. The second approach focuses on designing neural architectures that models the structure of the labels (Aly et al., 2019; Zhao et al., 2019) using capsule networks (Sabour et al., 2017). The third approach focuses on class-based synthetic data generation to facilitate classification with limited data (Kaushik et al., 2020; Y. Li et al., 2018). Data augmentation by employing supposedly label-preserving transformations to increase the number of training examples has been shown to improve the performance of neural networks (Aroyehun and Gelbukh, 2018; Wei and Zou, 2019; Xie et al., 2020).

### 3.3 Hierarchical transfer approaches

For the task of assigning an IPC subclass tag to each patent document: given a document consisting of  $L$  words  $[w_1, w_2, \dots, w_L]$ , we make a prediction of its most likely IPC subclass label.

To do this, we use the parent-child relationship between levels in the IPC hierarchy, transferring knowledge from two higher levels (section and class) to a lower level (subclass). We use transfer learning, multi-task learning, and a combination of both. Transfer learning and multi-task learning are known to provide inductive bias and regularization effects (Baxter, 1997; H. Choi and Lee, 2019; McCann et al., 2018). We show that these approaches translate into improved performance gains at the lower level of the IPC.

#### 3.3.1 Base model (PEncoder)

Drawing on the findings of Adhikari et al., (2019) that well-executed simple models are usually more effective than complex ones, we applied two regularization strategies: layer normalization and frequency-based vocabulary selection to the bidirectional GRU model of Risch and Krestel, (2018). To our knowledge, this model has the best performance on the two patent classification datasets we experimented with. Our base architecture is shown in Figure 3.2. The text sequence is provided as input to the GRU model in the form of word-embedding vectors; a single layer bidirectional GRU (indicated by directional arrows in Figure 3.2) is used to encode the input sequence in both forward and backward directions. We use the domain-specific embeddings provided by Risch and Krestel, (ibid.). The latent representation at each time step is a combination of the forward and backward hidden states; a contextual representation of each word within a given sequence in both directions. We combine these hidden states using the mean operation. Furthermore, to obtain a representation of a document as a fixed-size vector,  $r$ , we take the average (indicated with *Avg* in Figure 3.2) of the contextual representation of the sequence of tokens. The classification layer takes  $r$  as input, then the softmax activation function returns the probability on the number of labels for a given task. In addition, we employed two regularization strategies:

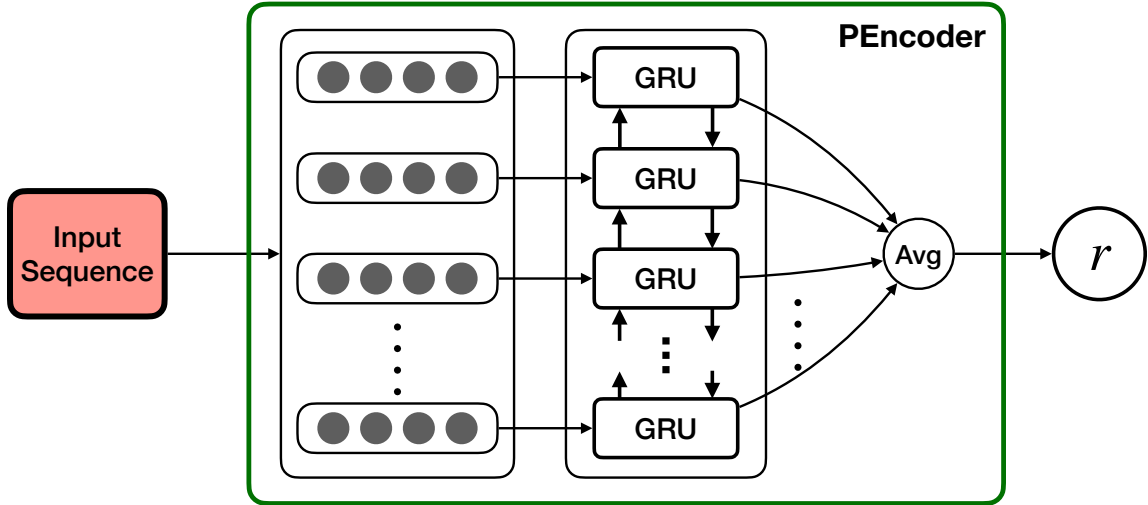


Figure 3.2: Base model: PEncoder.

vocabulary selection and layer normalization. We study their effects in Section 3.5.2. We use the PEncoder in our transfer learning and multitask learning experiments.

### 3.3.2 Transfer learning

In the transfer-learning (TL) setup, we pre-train the PEncoder on an upper level categorization task and then initialize the GRU component for the classification at the lower level using the model parameters from the upper level classification; only the classifier, the last layer, is randomly initialized. The left side of Figure 3.3 represents the pre-training phase. The transfer of the pre-trained model parameters is shown in Figure 3.3 by *Weight Transfer* from a source task, where the source task is either section-level or class-level classification. All model parameters are optimized on the target task. We denote this transfer learning configuration with  $TL_{sec}$  (the source task is section-level classification) and  $TL_{cls}$  (the source task is class-level classification), respectively.

Knowledge transfer from the section to class level did not improve performance in our experiments.



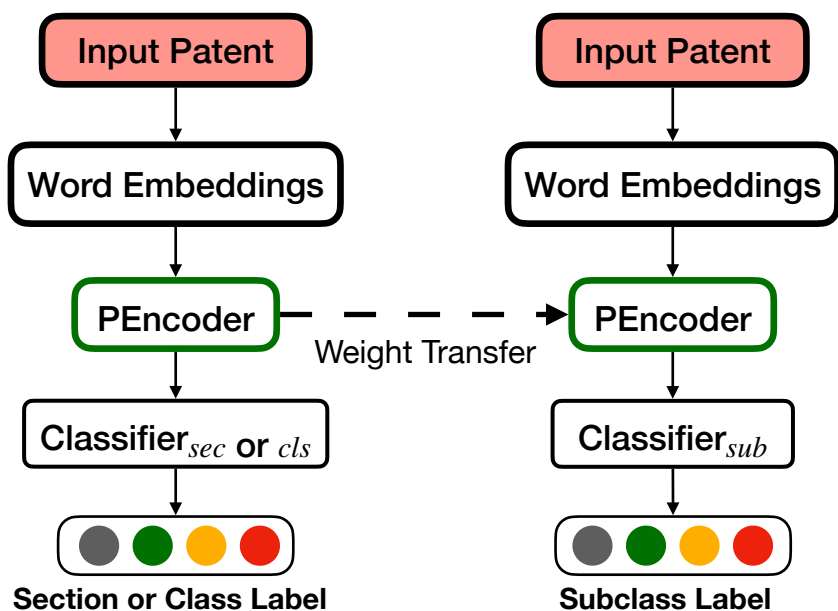


Figure 3.3: Transfer learning setting:  $TL_{sec}$  (section label  $Classifier_{sec}$  and ) or  $TL_{cls}$  (class label  $Classifier_{cls}$ ).

### 3.3.3 Multi-task learning

In the multi-task learning (MTL) setup, we train on multiple tasks in parallel. Figure 3.4 depicts a setting where two tasks are learned in parallel. The tasks share the same hidden layer, PEncoder. Each task has a task-specific classification layer. With two auxiliary tasks and one main task, we have the following task combinations: section-subclass and class-subclass, and section-class-subclass. Figure 3.5 shows the configuration where both auxiliary tasks are used simultaneously. We refer to these options as  $MTL_{sec}$  (multi-task learning with section-level classification as additional task),  $MTL_{cls}$  (multi-task learning with class-level classification as auxiliary task), and  $MTL_{sec+cls}$  (multi-task learning with section and class-level classification as additional tasks), respectively.

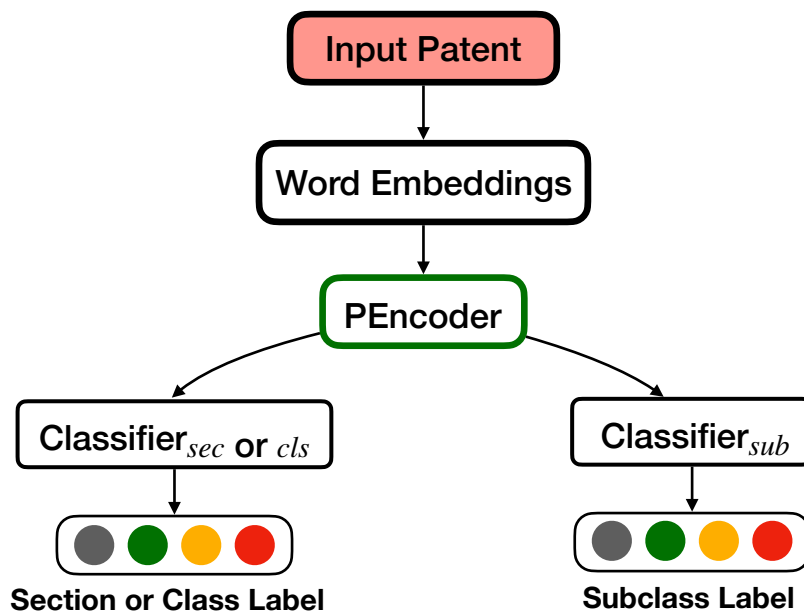


Figure 3.4: Multi-task learning setup with one auxiliary task:  $MTL_{sec}$  or  $MTL_{cls}$ .

### 3.3.4 Combination of multi-task and transfer learning

We combine the TL and MTL (MTL+TL) configurations. As in Figure 3.6 and Figure 3.7, we transferred model parameters from a source task to the target task, where the target task is trained jointly with auxiliary task(s). The pre-trained weights are used to initialize the shared hidden layer. The three multi-task combinations— $MTL_{sec}$ ,  $MTL_{cls}$ , and  $MTL_{sec+cls}$ —and two source tasks— $TL_{sec}$  and  $TL_{cls}$ —yield six possible settings: for example, by  $MTL_{sec} + TL_{cls}$  we mean multi-task learning with classification at *section* and subclass levels, and the PEncoder is initialized with weights from *class*-level classification.

### 3.3.5 Training

Our implementation uses the PyTorch library (Paszke et al., 2019). The model is a single-layer bidirectional GRU with 256 hidden units. For the WIPO-alpha dataset, we used 300-dimensional fastText embeddings (Bojanowski et al., 2016) Risch and

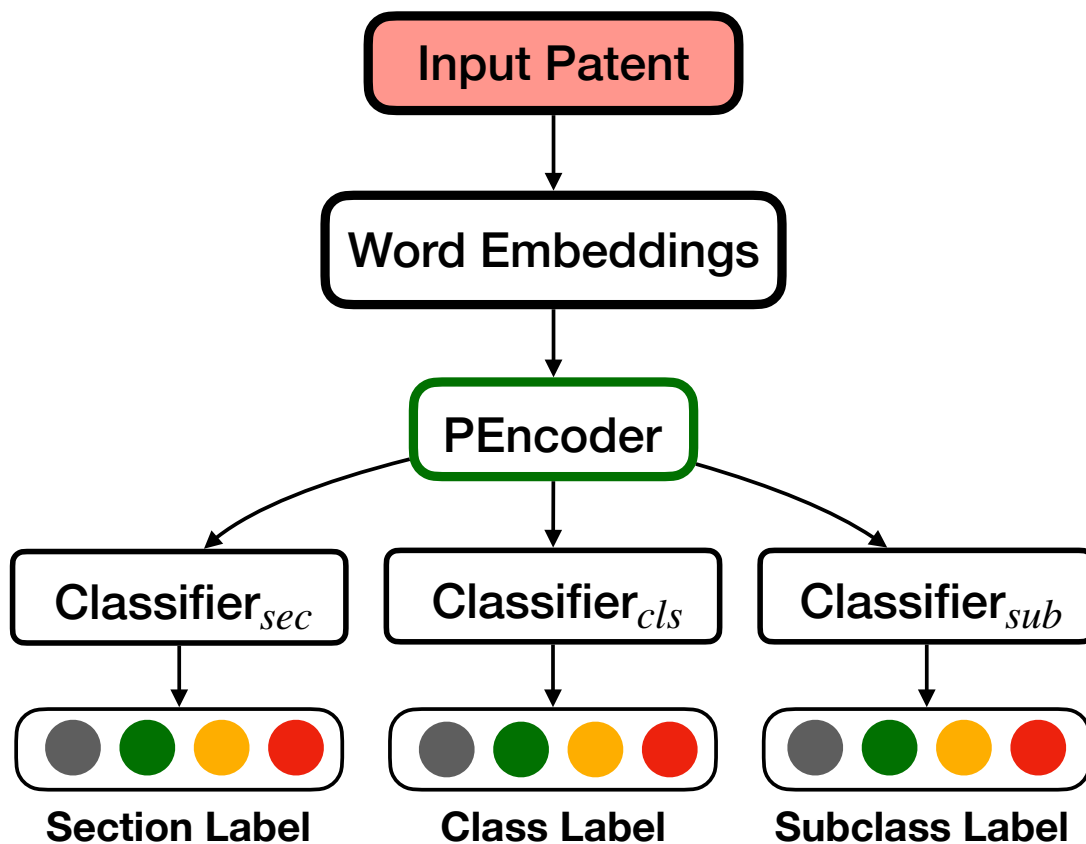


Figure 3.5: Multi-task learning setup with two auxiliary tasks:  $MTL_{sec+cls}$ .

Krestel, (2018) pre-trained embeddings on patent text. For the USPTO-2M dataset, we applied domain-specific embeddings of dimensions 100 pre-trained on a collection of documents released by the United States Patent and Trademark Office (USPTO).

For experiments with WIPO-alpha dataset we used the first 300 words and the first 30 words in experiments with the USPTO-2M dataset. In addition, we limit the vocabulary to words with a minimum document frequency of five. In the pre-processing step, we deleted tokens consisting of fewer than three characters, non-ASCII characters, digits, and alphanumeric characters. All tokens are converted to lowercase.

We used standard dropout on the output of the bidirectional GRU and spatial dropout on the output of the embedding layer. Both with a probability 0.1. We used

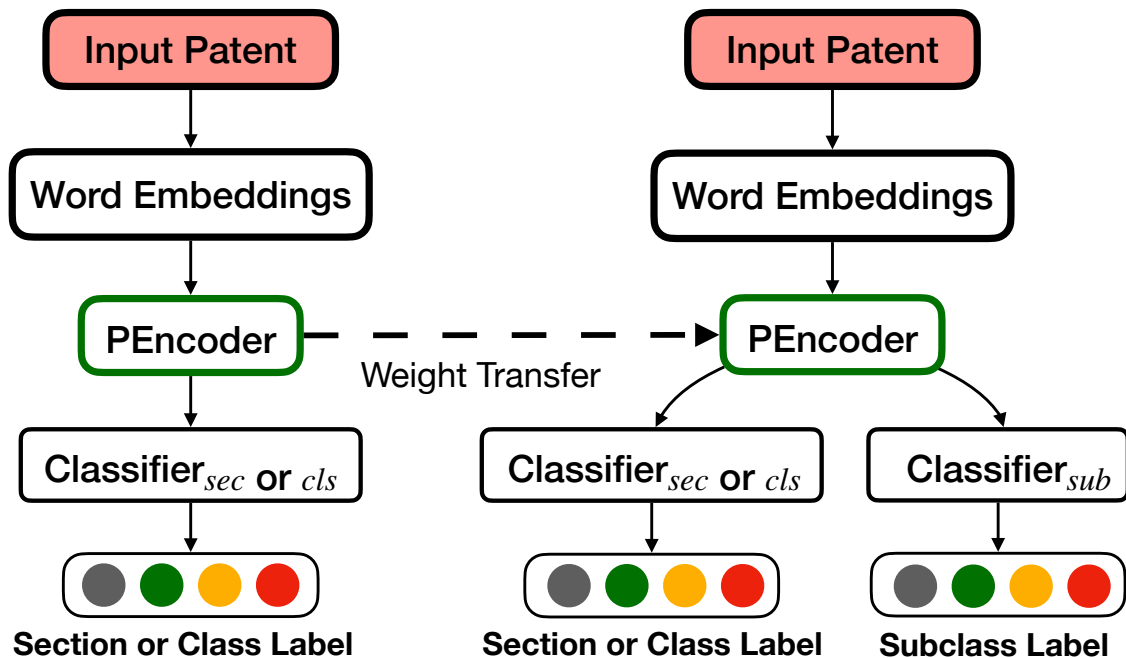


Figure 3.6: Combination of transfer learning and multi-task learning with an auxiliary task:  $MTL_{sec} + TL_{sec}$ ,  $MTL_{sec} + TL_{cls}$ ,  $MTL_{cls} + TL_{sec}$ , and  $MTL_{cls} + TL_{cls}$ .

the categorical cross-entropy as classification loss:

$$\mathcal{L} = -\frac{1}{M} \sum_{i=1}^M \sum_{j=0}^{N-1} y_{ij} \log p_{ij}, \quad (3.1)$$

where  $M$  is the number of examples,  $N$  is the number of classes,  $y_{ij}$  is the binary indicator of the ground-truth for whether example  $i$  belongs to class  $j$ , and  $p_{ij}$  is the predicted likelihood of example  $i$  belonging to class  $j$ . We use a learning rate of 0.001 for all experiments except for the MTL experiment on the WIPO-alpha dataset, where we set the learning rate to 0.01.

In training the model, we use AdamW (Loshchilov and Hutter, 2019) as optimizer. AdamW is a new implementation of the Adam (Kingma and J. Ba, 2014) optimizer that uses weight decay in the place of L2 regularization used in the previous implementation. The combination of training objectives in the multi-task setting is based on the

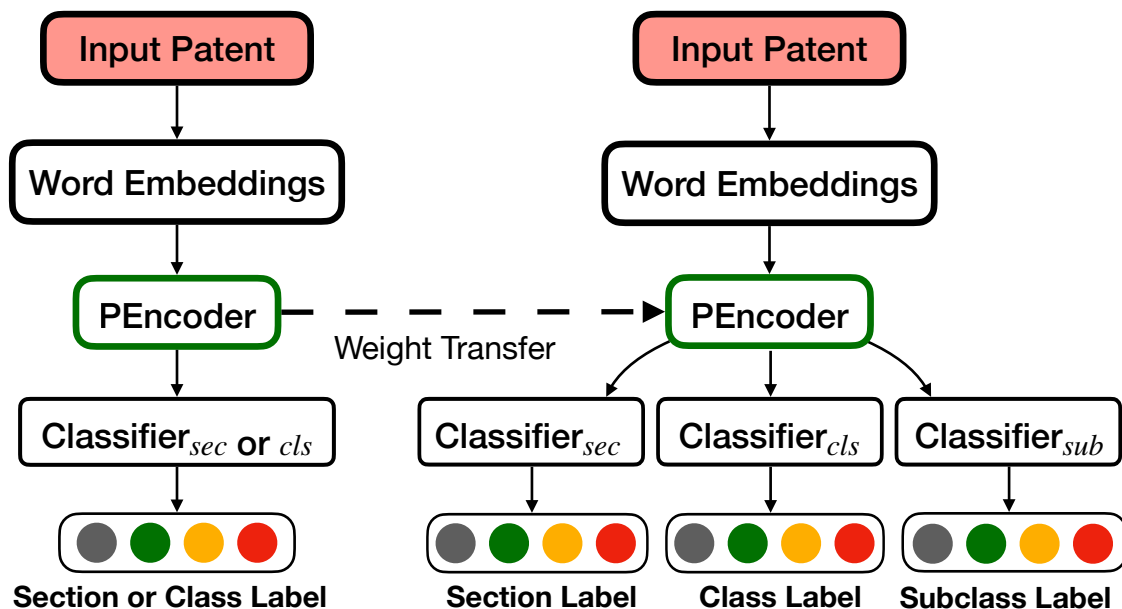


Figure 3.7: Combination of transfer learning and multi-task learning with more than one auxiliary task:  $MTL_{sec+cls} + TL_{sec}$  and  $MTL_{sec+cls} + TL_{cls}$ .

weighted average strategy suggested by Kendall et al., (2018) for WIPO-alpha, and by simple average for USPTO-2M. We find the simple average worse on WIPO-alpha, while weighted average performed poorly for USPTO-2M. We set the batch size to 256. We use an NVIDIA GTX1080 GPU to train our models for 5 epochs; the results we report are at the end of 5 epochs. These configurations are to a large extent our attempt to be consistent with the experimental setup of Risch and Krestel, (2018) for a fair comparison as there is no publicly available implementation of their model.

### 3.4 Experimental Results

We conducted experiments on two patent classification datasets. Our results are a significant improvement over the current state-of-the-art.

### 3.4.1 Datasets

Statistics of these datasets are shown in Table 7.1.

Table 3.1: Summary of the dataset, train and test splits, in the WIPO-alpha dataset and in the part of the USPTO-2M dataset we used. *Subclasses* column show the number of unique labels at the subclass level; *Train* and *Test* respectively indicate the total number of documents in the train and test subsets.

Dataset	Subclasses	Split		
		Train	Test	Total
WIPO-alpha	451	46,324	28,926	75,250
USPTO-2M used	632	1,645,176	303,332	1,948,508

WIPO-alpha is a publicly available collection (released by WIPO) of patent applications written in English for the development and evaluation of automated approaches for assigning IPC tag(s) to patent applications. This benchmark was provided by Fall et al., (2003) with a pre-defined train and test splits, thus facilitating comparisons among automated systems. It consists of full text of approximately 75,000 patent applications collected by various patent offices around the globe from 1998 to 2002, as well as the IPC codes assigned by human patent examiners—subject matter experts. Documents belonging to a subclass that has between 20 and 2000 documents have been included in this collection; the exclusion of categories with fewer documents resulted in 451 subclasses out of 632 existing at the time.

USPTO-2M is a publicly available<sup>2</sup> patent classification benchmark dataset created by S. Li et al., (2018), with nearly two million utility patent documents collected from the USPTO. It includes patents in 632 subclass categories<sup>3</sup> from the 2006 to 2015. Patent applications are generally structured into standard sections, the most informative are title, abstract, claim, and description. The USPTO-2M includes only the title and the abstract sections. As in previous work of Risch and Krestel, (2018), we take as training set documents from 2006 to 2013 and as test set documents from 2014 (so documents from 2015 were not used). However, we had to discard 1,737

---

<sup>2</sup><http://mleg.cse.sc.edu/DeepPatent/>

<sup>3</sup>S. Li et al., (2018) indicate 637, but we found 632 in the dataset.

badly-formed records from the training set and 2 from the test set, resulting in the numbers shown in Table 7.1.

### 3.4.2 Metrics

During training, the classification setup is multi-class. However, the model prediction is considered a ranked list. This is done in order to fit the micro-average precision (MAP) evaluation metric for three conditions: top prediction, three guesses, and all categories. The variations are considered appropriate for a scenario where documents can be assigned more than one category, but there is only one main category. These metrics are shown in Figure 3.8 according to the proposal of Fall et al., (2003).

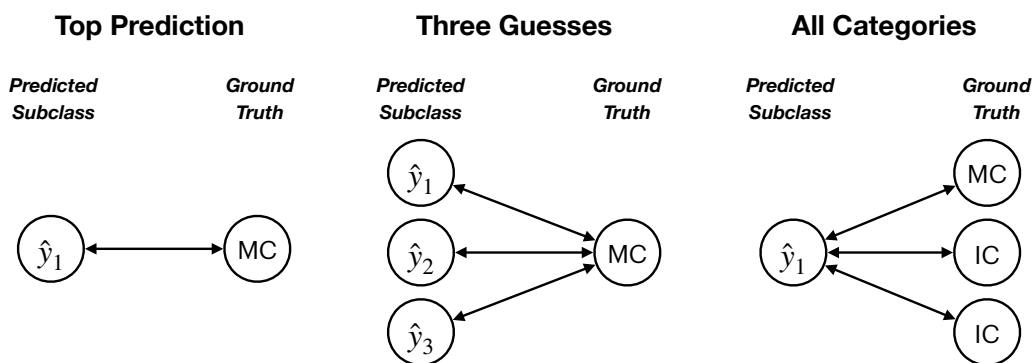


Figure 3.8: Micro-average precision evaluation metric for three conditions: top prediction, three guesses, and all categories (adapted from (Fall et al., 2003)).  $\hat{y}_n$  is for the  $n$ -th best prediction, MC is the main subclass label, and IC is the incidental subclass label.

The top prediction compares the classifier top prediction and the main IPC category (MC). In the three guesses, comparison is made between the top-3 predictions of the classifier and the MC; if one of the top-ranked predictions matches the MC, then the classifier has made a correct prediction. In the all-categories, the top prediction of the classifier is compared with all IPC categories associated with the document, both the MC and the incidental (or secondary) categories (IC).

### 3.4.3 Comparison of model performance

Table 3.2 shows the results obtained for the WIPO-alpha test set at the subclass level. The scores for USPTO-2M is in Table 3.3. The results were obtained using an experimental setup similar to that used by Risch and Krestel, (2018), we compare our models with the results reported in (ibid.) for their model *RNN-patent*. The results show that PEncoder, a regularized implementation of the same model, yields significant performance gain: it provides a 2.2% top prediction improvement in MAP for WIPO-alpha and a 4.4% top prediction gain in MAP for USPTO-2M. This is a confirmation of the effectiveness of the regularization approaches, consistent with conclusions reached in previous work (J. L. Ba et al., 2016; Chen et al., 2019; Luong et al., 2015).

Table 3.2: Model performance for classification at the subclass level on WIPO-alpha test set. TP is top prediction, AC is all categories, and TG is three guesses. The best score is in bold; the second best score is underlined.

Model	WIPO-alpha		
	Micro-averaged precision		
	TP	AC	TG
1. RNN-patent (Risch and Krestel, 2018)	0.49	0.57	0.72
2. PEncoder (base model)	0.5191	0.6101	0.7557
3. $TL_{sec}$	0.5119	0.6008	0.7445
4. $TL_{cls}$	0.5220	0.6128	0.7551
5. $MTL_{sec}$	0.5304	0.6205	0.7616
6. $MTL_{cls}$	0.5266	0.6160	0.7567
7. $MTL_{sec+cls}$	0.5170	0.6040	0.7417
8. $MTL_{sec} + TL_{sec}$	0.5278	0.6175	0.7551
9. $MTL_{sec} + TL_{cls}$	<b>0.5376</b>	<b>0.6265</b>	<b>0.7697</b>
10. $MTL_{cls} + TL_{sec}$	0.5274	0.6170	0.7592
11. $MTL_{cls} + TL_{cls}$	<u>0.5332</u>	<u>0.6240</u>	<u>0.7629</u>
12. $MTL_{sec+cls} + TL_{sec}$	0.5243	0.6129	0.7508
13. $MTL_{sec+cls} + TL_{cls}$	0.5304	0.6205	0.7616

Table 3.2 and Table 3.3 show the results of experiments where two additional sources of knowledge are considered (section and class levels) using the transfer-



Table 3.3: Model performance for classification at the subclass level on USPTO-2M (the part we used) test set. TP is top prediction, AC is all categories, and TG is three guesses. The best score is in bold; the second best score is underlined.

Model	USPTO-2M		
	Micro-averaged precision		
	TP	AC	TG
1. RNN-patent (Risch and Krestel, 2018)	0.53	0.64	0.75
2. PEncoder (base model)	<u>0.5740</u>	<b>0.6886</b>	<u>0.7951</u>
3. $TL_{sec}$	0.5733	0.6844	0.7949
4. $TL_{cls}$	<b>0.5743</b>	<u>0.6859</u>	<b>0.7959</b>
5. $MTL_{sec}$	0.5731	0.6872	0.7931
6. $MTL_{cls}$	0.5737	0.6849	0.7950
7. $MTL_{sec+cls}$	0.5700	0.6812	0.7905
8. $MTL_{sec} + TL_{sec}$	0.5727	0.6868	0.7921
9. $MTL_{sec} + TL_{cls}$	0.5726	0.6875	0.7929
10. $MTL_{cls} + TL_{sec}$	0.5718	0.6831	0.7921
11. $MTL_{cls} + TL_{cls}$	0.5713	0.6844	0.7920
12. $MTL_{sec+cls} + TL_{sec}$	0.5708	0.6815	0.7910
13. $MTL_{sec+cls} + TL_{cls}$	0.5722	0.6851	0.7921

learning approach, TL (Section 3.3.2), multi-task learning models, MTL (Section 3.3.3), and their combination, MTL+TL (Section 3.3.4), based on the regularized implementation PEncoder (Section 3.3.1).

In the experiments with TL (rows 3 to 4 of Table 3.2 and Table 3.3), the model is pre-trained on a source task and the model parameters are transferred with further training for classification at the subclass level. On WIPO-alpha, the model with knowledge transfer from the class level performs better than that with transfer from the section level, with a difference of about 1% top-prediction MAP, which is a gain of 0.03 compared to the regularized model, PEncoder. On USPTO-2M, the transfer from the class level is slightly better than transfer from the section level, with a higher top-prediction MAP margin of 0.001, which is a slight performance gain of 0.0003 over PEncoder.

For the MTL experiments (rows 5 to 7 of Table 3.2 and Table 3.3), the main

task is classification at the subclass level, and the auxiliary tasks are section, class, and a combination of classification at the section and class levels. On WIPO-alpha, the combination of section and subclass in multi-task configuration offers the best top-prediction MAP performance of 0.5304, better than class-subclass and section-class-subclass combinations, which is a 0.01 better than the best result with transfer learning and a gain of 0.02 over PEncoder. On USPTO-2M, the the best result is from the combination of class-subclass, followed by the combination of section-subclass. With the best MAP top-prediction of 0.5737, this is slightly lower than the best result of the transfer learning strategy and PEncoder by 0.0006 and 0.0004, respectively.

The MTL+TL experiments (rows 8 to 13 of Table 3.2 and Table 3.3) is a combination of transfer learning with multi-task learning. For WIPO-alpha, this approach gives the best top-prediction MAP, 0.5376, with a combination of classification at the section and subclass levels and weight transfer from class level classification. This is a performance improvement of 0.0072, 0.0156, and 0.0185 compared to the best performance obtained when using multi-task learning, transfer learning, and the PEncoder, respectively. Conversely, on the USPTO-2M, this approach is slightly less effective. The setting where we use section and subclass tasks and section as source of knowledge gives the highest top-prediction MAP of 0.5727. The second best is a similar configuration with the exception of using class level classification as the knowledge source. It achieves a top-prediction MAP of 0.5726. Therefore, this strategy is slightly below our best results with a performance drop of 0.0004, 0.0006, and 0.0013 on the three metrics, respectively.

### 3.5 Discussion

Through the simple regularization and knowledge transfer approaches, the results are better than the state-of-the-art RNN-patent model using the three variants of the micro-average precision metric, with an absolute gain of approximately 5% using the top prediction MAP metric. For WIPO-alpha, the combination of transfer learning and multi-task learning gives the best results, while the transfer learning approach is more effective for USPTO-2M.

The difference in performance between the two datasets can be attributed to the size and some specifics of the datasets outlined in Table 3.4. The size of the USPTO-2M dataset—approximately 2,000,000 samples (on both train and test sets), although sparse—seems to supersede the regularization effect of using MTL which can be observed for the WIPO-alpha dataset. Another important difference between the two datasets is that the labels on the long-tail of USPTO-2M are extremely sparse: there are labels with number of examples as low as 1, while the WIPO-alpha dataset has a minimum label frequency of 20. For USPTO-2M, using a simple average of the task losses instead of the weighted version provides a slight advantage over the transfer learning model. Therefore, with a larger dataset, transfer learning from a higher level seems to be more effective than multi-task learning.

Table 3.4: Statistics of the labels on the WIPO-alpha and USPTO-2M (used part) training splits indicating the minimum, 25th percentile, 50th percentile, 75th percentile, maximum, and average number of examples per label. The OOV column is the number of out-of-vocabulary words with respect to the embeddings for the patent domain.

Dataset	min	25p	50p	75p	max	avg	OOV
WIPO-alpha	20	33	61	124	2,000	103	377,331
USPTO-2M	1	109	492	1634	209,254	2,624	1,624 <sup>†</sup>

<sup>†</sup>Computed with the first 30 words.

It seems that the difference in the number of out-of-vocabulary (OOV) words for the two datasets is also a factor in the observed difference in performance of MTL. The number of OOV words for WIPO-alpha is over 300,000, while USPTO-2M has very few (about 1,000). On one hand, this huge difference can be linked to our experimental setup: we use the first 300 words for WIPO-alpha and the first 30 words for USPTO-2M. On another hand, the patent embeddings we used was trained with a large collection of patent documents collected by the USPTO. We think that USPTO-2M is most likely a subset of that collection as such it has very few OOV words. After checking few of the OOV words for WIPO-alpha, we find that most are misspellings caused by OCR errors that cannot be easily corrected with a

simple spellchecker such as the omission of space between two words. It follows that WIPO-alpha is a more challenging dataset. Therefore, a challenging task will benefit the most from the regularization effects of TL, MTL, and MTL+TL.

### 3.5.1 Effect on less frequent labels

To identify whether our proposed approach is beneficial to categories with fewer documents, we select 50 least-occurring labels in the WIPO-alpha training set. The corresponding documents in the test set contains 742 samples. We compare PEncoder with the best model,  $MTL_{sec} + TL_{cls}$ .

Table 3.5 shows that the MTL+TL transfer approach achieves a micro-average precision of about 33%, while the PEncoder achieves a score of approximately 24%. According to the McNemar’s test statistics (McNemar, 1947) for comparison of machine-learning models (Dietterich, 1998), the performance gap is statistically significant at  $\alpha = 0.05$  with a p-value of 0.000000015. This is in line with our expectation that transfer and multi-task learning can be beneficial when some labels are sparsely populated.

Table 3.5: Performance comparison using the top prediction (TP) MAP metric, on the 50 least frequently occurring labels in WIPO-alpha.

PEncoder	$MTL_{sec} + TL_{cls}$	P-value
0.2372	0.3288	0.000000015

The contingency table, Table 3.6, reveals that both models made incorrect predictions on 462 out of 742 samples. This shows that there is a need for techniques that can further alleviate the data sparsity problem.

### 3.5.2 Ablation study of the regularization techniques

We study the effect of the regularization techniques—vocabulary size selection and layer normalization in Table 3.7. On the WIPO-alpha, the choice of vocabulary size has little impact on performance, its removal results in a performance drop of

Table 3.6: Contingency table highlighting the differences between PEncoder and the  $MTL_{sec} + TL_{cls}$  model on the 50 least frequently occurring labels in WIPO-alpha.

$MTL_{sec} + TL_{cls}$	PEncoder	
	Correct	Wrong
Correct	140	104
Wrong	36	462

Table 3.7: Ablation study of the regularization techniques used in the PEncoder (first row), on the WIPO-alpha and USPTO-2M datasets. The metric is the micro-average precision of the top prediction (TP).

Vocabulary size selection	Layer normalization	WIPO-alpha	USPTO-2M
+	+	<b>0.5191</b>	0.5740
-	+	0.5100	<b>0.5741</b>
+	-	0.4701	0.5730
-	-	0.4501	0.5726

0.0091. A significant drop in performance of 0.05 can be observed without layer normalization. The removal of both regularization techniques results in a decrease of 0.07. On USPTO-2M, there is a marginal performance improvement of 0.0001 without selection of vocabulary size. Without layer normalization, the performance decreases by 0.0010. When both techniques are removed, the performance reduces by 0.0014.

Indeed, the combination of the regularization methods can be linked with the higher performance achieved by the PEncoder, on both datasets; the magnitude of this effect is greater on WIPO-alpha. The apparently negative impact of setting the vocabulary size on USPTO-2M can be attributed to our use of a shorter sequence length, 30 words, due to limitations on computational resources. This is in line with our observation that WIPO-alpha is a more challenging dataset, in addition to being smaller. Therefore, it benefits more from the regularization techniques.

**Abstract:** An improved infant formula resulting in reduced constipation, abdominal discomfort and gastrointestinal problems, comprises at least one protein component having a phosphorus content of less than 0.75 g P/100 g protein, and at least one lipid component that can be easily digested by an infant. Preferably, it further comprises at least one prebiotic component, and at least one viscosity-improving component. The protein fraction of the formula is preferably a hydrolysate prepared by hydrolysing a protein starting material, especially a whey protein with a combination of at least one endo- and at least one exoproteinase.

**Gold label:** A23D (edible oils or fats)

**PEncoder:** A23L (foods, foodstuffs, or non-alcoholic beverages, not covered by subclasses A23B - A23J; their preparation or treatment)

**MTL<sub>sec</sub> + TL<sub>cls</sub>:** A61K (preparations for medical, dental, or toilet purposes)

Case 3.1: Incorrect prediction by both models

### 3.5.3 Qualitative error analysis

Looking at the most common errors made by the models on the subset we discussed in Section 3.5.2, related or similar labels represent a common source of confusion. Related labels are those labels that share the same parent section or class in the label hierarchy. The example in Case 3.1 depicts a situation where both models made incorrect predictions. However, the predictions are closely related to the ground truth. Both models consider that the sample is about a kind of preparation. The PEncoder correctly identified the section and class while the MTL<sub>sec</sub> + TL<sub>cls</sub> could only correctly identify the section. In Case 3.2, the MTL<sub>sec</sub> + TL<sub>cls</sub> made a correct prediction while the prediction of PEncoder was incorrect. It predicted a similar label *G06F–electric digital processing* instead of *G06N–computer system based on specific computational models*.

Next, we study the relative effectiveness of both models by taking a count of how many examples which PEncoder predicted wrongly are correctly predicted by MTL<sub>sec</sub> + TL<sub>cls</sub> and vice versa. In the first case, the knowledge transfer approach can make correct predictions on about 20% of failure cases of PEncoder. For the converse,

**Abstract:** A method and a system are presented, which assist classifiers in gathering information in a cost-effective manner by determining which piece of information, if any, to gather next. The system includes an explicit system, an implicit system, a classifier, and a profit module. A feature set is inputted into the explicit system, which uses the feature set to determine tests to perform to gather information useful for classifying the system state. The relative profit of each test performed is determined by the profit module and the profit determined is used to determine which test or tests to select for a particular feature set. The results of the explicit system, which is generally an exhaustive or semi-exhaustive search algorithm, are used to train the implicit system. The implicit system is then able to process decisions much faster than the explicit system when circumstances require time-critical operation.

**Gold label:** G06N (computer systems based on specific computational models)

**PEncoder:** G06F (electric digital data processing)

**MTL<sub>sec</sub> + TL<sub>cls</sub>:** G06N

Case 3.2: Knowledge transfer makes correct prediction and PEncoder fails

we count the number of samples that were incorrectly predicted by the knowledge transfer approach but correctly predicted by the PEncoder. The PEncoder can make correct predictions on about 9% of the failure cases of the knowledge transfer approach. This is an indication that the knowledge transfer method is relatively better at differentiating among similar labels. This analysis demonstrates the robustness of the proposed approach. It also confirms our intuition on the benefits of knowledge transfer. Additionally, this error analysis signals an area of focus for future research.

### 3.5.4 Does the choice of word embeddings matter?

We examine the effect of the choice of word embeddings on the performance of PEncoder by conducting experiments using different word embeddings. In Table 3.8, we present results of experiments with a fasttext embedding pre-trained on patent text and another pre-trained on text from English Wikipedia. We also use a

randomly initialized word embeddings. The dimension of the embeddings is 300. The domain-specific word embeddings yields the best performance across all metrics. The general domain embeddings are better than the randomly initialized embeddings on the top prediction and all categories MAP metrics. The random embeddings are slightly better than the Wikipedia embeddings on the three guesses MAP metric.

Table 3.8: Performance comparison of PEncoder on WIPO-alpha when using different types of word embeddings.

Embeddings	TP	AC	TG
fasttext-patent	0.5191	0.6101	0.7557
fasttext-wiki	0.4935	0.5808	0.7171
random	0.4738	0.5626	0.7176

### 3.6 Conclusion

Using deep learning techniques, we examined the question of how to effectively transfer knowledge from upper level task(s) to a lower level task by leveraging the parent-child relationship present in a given categorization scheme—such as IPC—to improve automatic categorization at the lower—more granular—level. We considered three approaches: transfer learning, multi-task learning, and the combination of both, based on PEncoder, the regularized bidirectional GRU model. The evaluation of the approaches on two patent classification datasets, WIPO-alpha and USPTO-2M, shows improvement in performance, the combination of multi-task and transfer learning yields the best performance on WIPO-alpha, while transfer learning is the most effective on USPTO-2M. It is worthy of note that the approaches has outperformed the state-of-the-art RNN-Patent model (Risch and Krestel, 2018) on all metrics with an absolute increase of up to 5% on the top-prediction MAP metric.

There are a number of avenues for future work. For example, the experiments in this chapter are on the subclass level classification, classification at the more detailed group and subgroup levels has not received much attention. The IPC guideline offers alternative strategies to assign IPC labels, such as using the labels of similar



documents and searching for IPC codes using the so-called “catchword index” to find a likely match. Adding such alternatives to the knowledge transfer techniques may improve model performance especially on less-frequent labels. Given that patent documents are available in a number of languages, the application of the approaches proposed here to patent classification in other languages is interesting. Beyond the patent domain, the findings in this work could have useful implications for automatic categorization based on hierarchically controlled vocabularies in other domains using deep learning techniques.

# Chapter Four

## Detection of Aggression on Social Media

### 4.1 Introduction

The web has enabled user-generated content on various online platforms. An essential part of these platforms is the social media websites. These websites facilitate social interactions by posting comments and responding to comments from other users. As with offline interactions, online interactions are subject to anti-social behaviours such as trolling, flaming, abuse, bullying, and hate speech. The global increase in internet penetration has allowed unprecedented access to the web, the illusion of 'invisibility' by web users has led to increasing anti-social behaviour on digital platforms. As a result, web users are exposed to mental, psychological, and emotional distress if such behaviour is not curbed.

In an effort to make the web more civilized, automatic detection of content that contains or could have the effect of abuse, aggression or hate speech on social media platforms is an important task. However, most studies (with publicly available datasets) on automatic hate speech detection has focused on Twitter. The task on aggression detection (Kumar et al., 2018) aims to compare classifiers developed for the automatic identification aggression on social media platforms by making annotated data collected from Facebook available. The task is to develop a multi-class classifier that can make a subtle distinction between texts categorized as one of three categories:

overtly aggressive, non-aggressive and covertly aggressive.

Motivated by existing works that apply machine learning to automatic hate speech detection, we examined the effectiveness of models that rely on little or no feature engineering. The presence of noisy content such as misspellings, acronyms, code-mixing, and incorrect grammar in social media posts makes extracting linguistic features using Natural Language Processing (NLP) tools error-prone. Consequently, we experimented with a linear model (NBSVM with n-grams) and deep learning models (CNN, LSTM, BiLSTM, and combinations thereof). The linear model serves as our baseline (hard to beat) while we refine our deep learning models for this task. Our goal is to examine (1) the relationship between model complexity and effectiveness and (2) the size of dataset required to accurately detect aggression in social media posts.

For the remainder of this chapter, Section 4.2 outlines related work. Sections 4.3 and 4.4 present the data and our methodology respectively. The results of our experiments are in Section 4.5 and we conclude in Section 4.6.

## 4.2 Related Work

The task of detecting aggression on social media can be seen as a document classification task. This task can also be divided into binary classification or multi-class classification. Within the framework of detection of aggressiveness, the binary classification would imply the presence or the absence of certain anti-social phenomena such as abuse (Nobata et al., 2016) in a particular example (abusive or not abusive). In the multi-class scenario, some types of anti-social behaviour are interesting (Waseem et al., 2017) such as racism, sexism, hate speech, and bullying. It has been observed that contents containing anti-social phenomena occur sparingly in a collection of social media posts. This usually results in unbalanced datasets where posts devoid of the phenomena of interest are in abundance. This problem is more pronounced in the multi-class scenario, which leads to difficulties in learning features which are discriminative enough by classifiers. In (Malmasi and Zampieri, 2018), it was concluded that making a subtle distinction between types of anti-social behaviour:

profanity and hate speech is a difficult task for machine learning classifiers. As a result, we reckon that it will be difficult to differentiate between overt and covert aggression. Additionally, (Davidson et al., 2017) indicates that posts that do not contain explicitly aggressive words are likely to be difficult to identify. This would likely apply to the covertly aggressive class in this study.

In (Gao et al., 2017), the authors recognized the cost of annotating data for hateful comments in social media and proposed a system that exploits unlabeled data. Their result shows improvement over systems which are based solely on manually annotated data. This approach is most closely linked to our work.

The identification of aggression in social media is closely linked to existing studies on the detection of hate speech, abuse, and cyberbullying. The methods used to address these tasks as a supervised classification problem can be broadly divided into two categories. One approach is based on manual feature engineering. With the feature engineering approach, the extracted features serve as input for classic machine learning algorithms such as naive bayes, logistic regression, support vector machines, and random forest (Malmasi and Zampieri, 2017; Schmidt and Wiegand, 2017). The other approach is based on deep neural networks that automatically learn features from input data. (Gambäck and Sikdar, 2017) used convolutional neural networks to classify hate speech and (Zhang et al., 2018) used a combination of convolutional neural network and gated recurrent unit (GRU) for the same task.

Rather than relying solely on the textual content of social media posts to identify hate speech, (Founta et al., 2018; Qian et al., 2018) examined the use of metadata of the users or the posts. However, the metadata may not be on every social media platforms. Additionally, this approach can fail if a social media platform allows anonymous posting.

## 4.3 Data

In this section, we describe the datasets used in this chapter and the data augmentation strategy. Additionally, details of the pseudo labeling approach are presented.

**Dataset** We used the annotated data provided by the task organizers for the 2018 edition of the TRAC challenge. The details of the annotation procedure can be found in (Kumar, Reganti, et al., 2018). The dataset consists of posts collected from Facebook. The posts relate to entities (organizations, individuals or issues) in India. The dataset includes posts written in English and Hindi. We used the English part in this work. The dataset is annotated with three high-level tagsets namely: covertly aggressive, non-aggressive, and overtly aggressive. Each example is annotated with one of these tagsets. The English dataset consists of 12000 training examples, 3001 development examples, 916 test examples from Facebook, and 1257 examples from an unspecified social media platform. On the training set, the number of examples per class is unbalanced. The covertly aggressive class accounts for about 34% of the training examples; non-aggressive class represents about 42% of the training examples and overtly aggressive class represents approximately 24% of the training examples. The minimum and maximum number of tokens are respectively 1 and 1200 on the training set. We found that the training set contains redundant posts: different post ID but duplicate content.

**Data Preprocessing** We have used well-formed texts in our experiments. We converted all alphabetic characters to lowercase. We removed punctuation marks, digits, URLs, repeated characters, non-English characters, alphanumeric characters, and usernames. Additionally, we have decoded emoji(emoticons) into their text equivalents and converted hashtags into their constituent tokens where possible. To correct spelling mistakes, we used a spell checker.

**Data Augmentation and Pseudo Labeling** A common technique to improve the generalization of neural network models in computer vision is to enlarge the dataset using transformations that preserve the labels (Simard et al., 2003). This is usually done by well-known invariant operations such as rotating and scaling images in the training set. Inspired by this technique, we simulate transformations by translating each example into an intermediate language and back to English. We have translated into four intermediate languages namely French, Spanish, German,

and Hindi. We observed a slight increase in model performance when using all four languages. It is known that machine translation provides conceptually equivalent output in a target language. We used the Google Translate API for the translation. We rely on a weak notion of label-preserving transformation for the text by translating the original training text to an intermediate language and translating it back to English. We anticipate that the translation into multiple intermediate languages will help diversify our augmented training set in terms of various lexical choices, each using the intermediate languages as a source language and English as the target language in the backtranslation phase. Although, the process of translation and backtranslation is error prone, this process increases the size of our training set by a factor of 5. The resulting augmented training set is highly interdependent. With the aforementioned procedure, the DNN models outperform the linear baseline model.

One of the test sets provided for evaluation is data collected from an unspecified social media platform. To diversify our training examples, we used a related dataset that was collected from Twitter for hate speech detection which is publicly available<sup>1</sup>. There are two datasets with a total of 22075 tweets (subject to change based on availability on the Twitter platform). We assume that the dataset contains the phenomenon of interest for the task under consideration. So, we used the model with the highest performance on the development dataset to label the Twitter dataset. The examples with the pseudo labels were added to the original training dataset. Then, we train the deep learning models with the original, augmented, and pseudo-labeled examples.

## 4.4 Methodology

Our approach was to develop a baseline model and a set of deep neural network models. The models are presented in this section.

**Baseline Model** In (S. Wang and Manning, 2012) a support vector machine (SVM) model which uses naive bayes (NB) log-count ratio features (NBSVM) is proposed.

---

<sup>1</sup><https://github.com/ZeeraKW/hatespeech>

NBSVM shows consistent performance on several text classification tasks. We used the logistic regression classifier in place of the SVM. We consider this model to be a strong linear baseline. We implemented two types of the NBSVM model. One is based on word n-grams and the other is based on character n-grams. We found the character n-grams model to be superior on the development dataset for aggression detection. The character n-grams NBSVM serves as our baseline based on which we compare results from deep neural network models in the experiments conducted.

**Input Representation** The representation of inputs in deep neural network models is the embedding layer. The embedding layer encode each word in the vocabulary used in the model. We experimented with various pre-trained word vectors for the embedding layer including word2vec, Glove, SSWE, and fastText. We found the coverage of the pre-trained embeddings to be limited. FastText has the highest vocabulary coverage in our experiment (with about 5000 missing entries). So, instead of using a pre-trained word vector file to locate the vector representation of each word, we used the pre-trained fastText model (Mikolov, Grave, et al., 2018) to derive the vector representation of each word in the vocabulary of our model. This allows us to use the information at the sub-word level to derive vector representation of words that are not part of the vocabulary of the corpus (Wikipedia) on which the fastText model has been pre-trained. This feature is especially important for social media posts, which usually contain typos and abbreviations. Additionally, we used a randomly initialized embedding which is trained with the network to learn task-specific word representations. The input representation of the deep neural network models is therefore a concatenation of 300-dimensional word vectors, derived from fastText model, and a 50 dimensional task-specific word vector.

**Deep Neural Network Models** We tested seven deep learning models for aggression detection: CNN, LSTM, BiLSTM, CNN-LSTM, LSTM-CNN, CNN-BiLSTM, and BiLSTM-CNN. The models vary in complexity, with the combination of BiLSTM and CNN: CNN-BiLSTM and BiLSTM-CNN being the most complex neural architecture. CNNs have been used for text classification (Kim, 2014a). CNN is able to

learn features from words or phrases in different locations in the text. LSTMs model long-term dependencies in text and have been found to be useful for text classification . BiLSTMs consist of two LSTMs. One encodes the information in a sequence in the forward direction and the other in the reverse direction. Past and future information is available to the model when making a classification decision. CNNs and (Bi)LSTMs complement each other in their modeling capacities. Each of CNNs and (Bi)LSTMs capture information at different scales. Therefore, we investigated whether there could be potential improvements by combining multiple pieces of information of different scales for aggression detection. We examined inputting the word representation into the CNN and feeding the local features learned by the CNN into the (Bi)LSTM in the CNN-(Bi)LSTM model. Conversely, we also passed the input representation to the (Bi)LSTM and passed the long-term features learned by the (Bi)LSTM to the CNN in the (Bi)LSTM-CNN model. The representations learned from the CNN, (Bi)LSTM, and their combinations serve as input for the fully-connected layer. The output of the fully-connected layer is a softmax (probability) output which indicates the likelihood of belonging to each of the three classes. The class with the highest probability is the predicted class.

For the training of the deep neural network models, we used the sparse categorical cross-entropy as objective function. We chose the RMSprop optimizer to minimize the loss function in 5 epochs. We used feature dropout and earlystopping as regularization mechanisms to avoid overfitting. A spatial dropout (Tompson et al., 2015) probability of 0.2 was applied to the embedding layer.

**Submissions to the leaderboard** We submitted three systems for evaluation on the unseen test dataset. Our first submission is the predictions of the LSTM model. This model was trained with the augmented dataset. The second submission uses the representations learnt by the CNN and LSTM in the CNN-LSTM configuration. This submission was trained with the larger training set, which consists of augmented and pseudo labeled examples. Additionally, we calculated the sentiment score for each example and used the score as an additional feature to the representations learned from the CNN-LSTM layers before making predictions. The third submission is an



ensemble of the predictions made by the deep neural network models trained on (1) augmented data and (2) combination of augmented and pseudo labeled data, and sentiment score for each post. The final predictions were made by majority voting.

Table 4.1: Weighted F1 Scores on the English Development set. PL stands for Pseudo Labeling and Aug. represents Augmentation

<b>System</b>	<b>No Aug.</b>	<b>Data Aug.</b>	<b>Data Aug. + PL</b>
NBSVM	<b>0.5116</b>	0.5135	–
CNN	0.4989	0.5520	0.5741
LSTM	0.4940	<b>0.5679</b>	0.5561
BiLSTM	0.4714	0.5274	0.5776
CNN-LSTM	0.4702	0.5296	<b>0.5822</b>
LSTM-CNN	0.3679	0.5624	0.5729
CNN-BiLSTM	0.3839	0.5272	0.5573
BiLSTM-CNN	0.4836	0.5523	0.5440

## 4.5 Results

This section presents the results obtained on the English development and the test datasets (Facebook and Social Media) using weighted macro-F1 scores.

Table 4.1 shows the performance of the baseline model (NBSVM) and the DNN models are presented. Using the training dataset as given for training, the NBSVM model performs best. The table shows the effects of data augmentation and pseudo labeling on the models. It can be observed that when using data augmentation, the performance gain with the NBSVM is very small (0.0019) compared to the DNN models where the maximum performance gain of 0.1433 is observed with the LSTM model. The combination of data augmentation and pseudo labeling leads to a slight improvement in the performance of the DNN models with the exception of the LSTM model. Overall, data augmentation results in about 5% weighted macro-F1 improvement on the development set over the NBSVM baseline model. Similarly, the combination of data augmentation and pseudo labeling results in a weighted macro-F1 gain of about 7% over the linear baseline model. Pseudo labeling gives a slight improvement of about 2%. The significant performance gain (around 5%) is due to the introduction of data augmentation. These gains demonstrate the effectiveness of the data augmentation approach and the complementarity of pseudo labeling in our approach. We also experimented with and without the sentiment score as a feature. We did not observe any significant gain or drop in performance.

Table 4.2 shows the performance scores of the three systems submitted for evaluation on the Facebook test set. All submissions are better than the random baseline score. All our systems have comparable performance; they are all within 0.05 F1 score of each other. The LSTM model trained with the augmented training set obtained the best weighted F1 score of **0.6425**. This score ranks **first** on the shared task which attracted **30** submissions. It is clear that a complex model (CNN-LSTM) does not necessarily give the best result. Figure 4.1 shows the confusion matrix for our best system, the LSTM model. From the confusion matrix, it can be observed that the most of the model errors in absolute terms occur on the non-aggressive (NAG) class; the model predicts covertly aggressive (CAG) when the true label is NAG. In relative

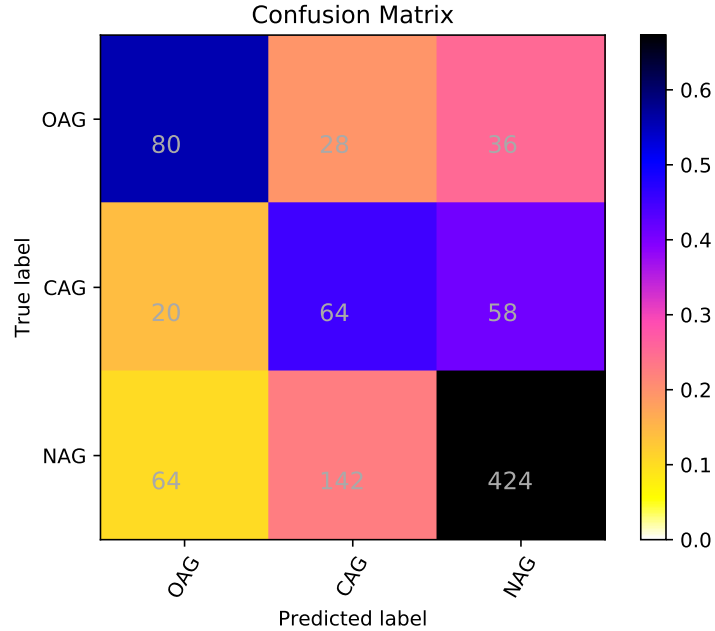


Figure 4.1: Confusion Matrix of the LSTM Predictions on the English (Facebook) Test Set

Table 4.2: Weighted F1 Scores on the English (Facebook) Test set

System	F1 (weighted)
Random Baseline	0.3535
LSTM	<b>0.6425</b>
CNN-LSTM	0.6058
Ensemble	0.5897

terms, the most difficult class for the model is the CAG. Most of the errors here result from the model predicting NAG when the true labels are CAG. In addition, it can be observed that the model does better on the NAG class, followed by the overtly aggressive class (OAG) and the worst performance is on the CAG class.

Table 4.3 gives an overview of the results obtained on the surprise test dataset collected from an unnamed social media platform. On this dataset, the CNN-LSTM model which was trained on the combination of the augmented training data, and pseudo labeled Twitter data along with sentiment score received a weighted F1 score

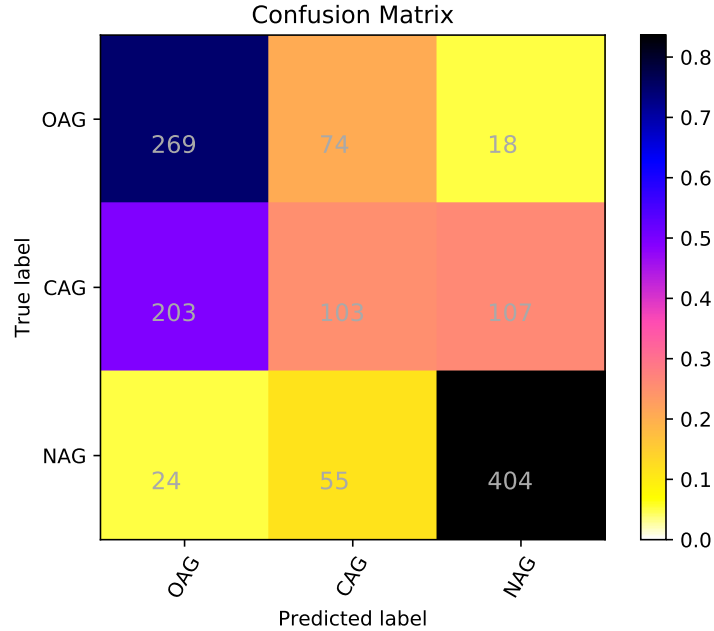


Figure 4.2: Confusion Matrix of the CNN-LSTM Predictions on the English (Social Media) Test Set

Table 4.3: Weighted Macro-F1 Scores on the English (Social Media) Test set

System	F1 (weighted)
Random Baseline	0.3477
LSTM	0.5400
CNN-LSTM	<b>0.5920</b>
Ensemble	0.5682

of **0.5920**. This score places the system on an overall ranking of **third** among 30 systems that participated in this track. In this case, a complex model supported by an out-of-domain data is superior. In Figure 4.2, the confusion matrix gives us an overview of the performance of the CNN-LSTM model for each class. We can observe that the model made the most errors on the CAG class. It predicts OAG when the real class is CAG. The model does best on the NAG class, followed by the OAG class and has the lowest performance on the CAG class.

Based on the performance trend observed in our results, the performance of our

best model for each track appears to reflect the distribution of the number of examples per class in the original training set. Our models achieve better performance on classes with more training examples than on classes with fewer training examples. Also, it is not surprising that although the OAG class has the least number of examples in the training set, the performance of our models on the OAG class is better than on the CAG class which has more examples. The performance on the OAG class confirms an earlier conclusion that a message which contains explicitly aggressive word(s) is easier to detect. Considering our result, the conclusion is true even if the category containing explicitly aggressive word(s) is the minority class.

## 4.6 Conclusion

In this chapter, we tackle the challenge of automatically detecting aggression in social media posts. We have developed a linear baseline classifier using NBSVM with character n-grams as features. We performed experiments with deep neural network (DNN) models of varying complexity, ranging from CNN, LSTM, BiLSTM, CNN-LSTM, LSTM-CNN, CNN-BiLSTM to BiLSTM-CNN. To do better than the linear baseline with deep neural networks, we experimented with data augmentation, pseudo labeling, and sentiment score feature. We found that the largest improvement is due to the data augmentation strategy. The improved DNN models were better than the linear baseline model on the development set. Therefore, the DNN models require more data points than a non-DNN model (in this case NBSVM) for this task. As the results show, a complex model does not necessarily improve the performance on this task. The LSTM-based model obtained the best weighted F1 score on the English Facebook test data. With a score of 0.6425 on the Facebook test set we achieved first place out of 30 submissions on the leaderboard. The CNN-LSTM classifier receives a weighted F1 score of 0.5920 on the social media test set. With this score, we achieved an overall ranking of third out of 30 teams on the leaderboard.

The performance trend of our models on all three classes shows that the models have the best performance on the NAG class and least on the CAG class. This is consistent with previous work on the nature of implicit offensive language. The trend

reflects the distribution of examples per class in the training set. The performance on classes with more training examples are better than those with fewer examples. A natural path for future work is to examine approaches that can improve performance on classes with smaller amount of training examples.

# Chapter Five

## All-in-one Model for Multilingual Offensive Language Detection

### 5.1 Introduction

The impact of objectionable content on internet users ranges from subtle discomfort to more serious psychological and emotional problems, that if left unchecked, can lead to violent actions by/towards affected individuals. To make the web a safe place for everyone, platforms such as Twitter and Facebook pay a lot of attention to content moderation. To aid in this difficult task of removal of objectionable content, it becomes necessary to build efficient and effective systems capable of identifying and classifying such content for automatic or human-assisted content moderation. A standard approach is to automatically mark such content for removal or review by human moderators. There are several studies on the English language due to availability of datasets and distributional representation with which to develop models. While significant progress has been made in the English language, the same cannot be said of other languages. With shared task series such as HASOC providing data in other languages, this opens the way for further research in other languages. With the availability of datasets in multiple languages, it becomes expensive to design a robust system for each language. An alternative strategy will be to train a single model for languages for which annotated data is available.

We base our approach on recent advances in the development of multilingual

language models. In particular, the observation by Conneau et al., (2020) that a multilingual model can achieve the performance of multiple language-specific models, at least after pre-training. Can we say the same for fine-tuning on a downstream classification task? We investigate whether jointly fine-tuning a multilingual model on a multilingual dataset is feasible for the task of objectionable content identification and classification. We think that this approach will be more energy efficient and computationally less expensive.

Specifically, we explore the possibility of using a multilingual pre-trained language model (BERT) to train a single model for the three languages with datasets provided for the HASOC 2020 shared task (Mandl, Modha, Shahi, et al., 2020) using transfer learning.

## 5.2 Related Work

Several approaches have been explored to automatically identify offensive content. Traditionally, feature engineering in combination with classical machine learning models such as Support Vector Machines, Logistic Regression, and Naive Bayes have shown competitive performance (Malmasi and Zampieri, 2017). Recently, neural networks have shown better performance than the traditional approaches by using architectures such as GRU, LSTM, and CNN in combination with word embeddings (Aroyehun and Gelbukh, 2018). The introduction of contextual word embeddings based on pre-trained language models (Devlin et al., 2019) and the transformers (Vaswani et al., 2017) architecture has led to state-of-the-art results in several areas of NLP including offensive content identification (Risch and Krestel, 2020). Typically, current approaches are based on pre-trained language models that are adapted to the downstream task (Sun et al., 2019). On the English language, significant progress has been made on the detection of objectionable content, which cannot be said of other languages. Recently, shared tasks such as TRAC 2018 (Kumar, Ojha, et al., 2018), HASOC 2019 (Mandl, Modha, Majumder, et al., 2019), TRAC 2020 (Kumar et al., 2020), and Offenseval (Zampieri et al., 2020) have introduced datasets in languages other than English. However, the mode of evaluation at those venues is still carried



Table 5.1: Details of the dataset for subtasks A and B for each language. Total is the number of labeled examples per language. OFF – Offensive, and PRFN – profane

	A		B				Total
	OFF	NOT	OFF	HATE	PRFN	NONE	
EN	1856	1852	321	158	1377	1852	3708
DE	673	1700	140	146	387	1700	2373
HI	847	2116	465	234	148	2116	2963

out in a monolingual manner. It would be interesting to see evaluation settings where models are evaluated on their multilingual and/or cross-lingual capabilities, such as Pamungkas and Patti, (2019) and Ranasinghe and Zampieri, (2020).

### 5.3 Methodology

**Task.** Given a text (tweets in this case) predict (1) For subtask A, whether the text is offensive or not. and (2) For subtask B, classify the text into one of the following categories: none, offensive, hate, and profane.

**Data.** The HASOC 2020 dataset contains annotated data in English, German, and Hindi. The data contains hierarchical annotations at two levels. Level one has binary labels (offensive VS. non-offensive) and level two has four mutually exclusive labels. Table 5.1 shows the details of the training set.

**Approach.** We train a single model with a combination of labeled datasets for each language. We therefore have a single model for each task that covers all the three languages.

As validation set, we use the test set (gold labels) from the 2019 edition of HASOC. We observe that applying language-independent pre-processing: removing URL, normalizing repeated characters, converting emoji to text, and removing punctuation marks resulted in performance degradation on the validation set. Therefore, we did not apply any pre-processing. It appears that a contextual model such as BERT

is able to leverage the information that would have otherwise been removed. We experimented with both the multilingual BERT (Devlin et al., 2019) and the XLM-R (Conneau et al., 2020) pre-trained models. We find that the XLM-R model had unstable and worse performance across runs. This is likely due to the size of the model and thus requires careful fine-tuning. Based on this observation, we choose multilingual BERT for our experiments. We use as representation the embedding of the [CLS] token with dimension 768 and feed it to a single layer perceptron with softmax activation. This yields a probability distribution over the number of classes to predict (two for subtask A and four for subtask B). The training set up uses the following hyperparameter settings: learning rate of  $3e - 5$ , batch size of 128, Adam as optimizer, and a maximum number of epochs of 5. We choose the model with the best performance on the validation set to make prediction on the test set. For subtask B, we continue fine-tuning on the best model from subtask A with the same hyperparameters stated above. The implementation uses the Flair library (Akshik et al., 2018).

## 5.4 Results

Table 8.2 shows the results obtained per task on each language on the test set. For subtask A in English, we obtained a macro-average F1 of 0.4980, and 0.2537 for subtask B. These results are within 2 F1 points of the highest ranked submission for English on the leaderboard. On the German data, the difference in performance for subtask A is the smallest, 0.0058, about 1% F1 points. On the Hindi dataset, the largest difference in performance is found on subtask B, about 10% F1 points. The second largest difference is also found on the Hindi subtask A, which is about 3% F1 points lower than the best performance on the leaderboard. We suspect that there is a negative transfer from English or German to Hindi. This finding deserves further investigation in the future. Overall, the scores on subtask B are consistently lower than the scores on subtask A in all languages. Perhaps, this is an indication of the difficulty of the task.

Table 5.2: F1 score on the test set. The numbers in parentheses represent the difference in performance between our submission and the best model on the leaderboard.

Task	EN	DE	HI
A	0.4980 (-0.0172)	0.5177 (-0.0058)	0.5005 (-0.0332)
B	0.2537 (-0.0115)	0.2687 (-0.0256)	0.2374 (-0.0971)

## 5.5 Conclusion

We investigated the feasibility of using a single multilingual model to detect and classify offensive language in three Indo-European languages. We conducted fine-tuning experiments with the multilingual BERT model. We record a competitive macro-average F1 score of 0.4980 on English subtask A. We find that the performance difference between our submission for tasks on Hindi and the best model on the leaderboard is the largest. In the future, we would like to experiment further with a mixture of more language-specific datasets and determine the limitations of using a mixed-language dataset to fine-tune multilingual encoders for the task of identifying objectionable content.

## Chapter Six

# Automatic Identification of Social Media Posts Mentioning Drugs and Adverse Drug Reactions

### 6.1 Introduction

The emergence of social media platforms such as Twitter has led to the availability of huge amount of data for research purposes. Public health surveillance using this non-traditional mode of communication has received attention in recent times. The third edition of Social Media Mining for Health Applications (SMM4H) (Davy et al., 2018) challenge aims to facilitate pharmacovigilance research using social media data.

The challenge consists of several tasks. Our focus in this chapter is on tasks 1 and 3. The purpose of task 1 is to identify tweets that contain drug name(s) while task 3 focuses on recognizing Twitter posts mentioning adverse drug reactions (ADR). Both tasks are binary classification tasks. We believe that both of these tasks are necessary steps in extracting ADR from social media posts as filtering out tweets that are not related to drugs and/or ADR could result in a more accurate ADR extraction system. The evaluation scores for both tasks are the precision, recall, and F1 values of the positive class.

In the following sections, we describe the data, our approach, results, and conclusion.

## 6.2 Data

The datasets consisted of IDs of tweets and their corresponding label, as well as a script to download the text of the tweets. Using these IDs, textual data was collected from Twitter. For task 1, tweets were annotated for the presence of at least one mention of drug name(s). The presence of ADR mention was also annotated for task 3. We downloaded a total of 9056 tweets out of 9625 expected tweets as training data for task 1. We also retrieved 16677 tweets out of the expected 25630 tweets for task3. Table 7.1 shows the number of examples per label in the training data for tasks 1 and 3. For task 1, the number of examples per class is almost the same. For task 3, the number of examples per label is very unbalanced with about 92% of the examples belonging to the negative class (non-ADR) and about 8% of the training data belonging to the positive class (ADR). The test set consists of 5382 tweets and 5000 tweets for tasks 1 and 3, respectively. We performed data cleaning by getting rid of special and repeated characters, numbers, URL, and hashtags. To deal with spelling errors, we performed spell checking.

Table 6.1: Number of Examples in the Train and Test Sets for Tasks 1 and 3

Task	Train set		Test set
	neg class	pos class	
1	4356	4700	5382
3	15326	1351	5000

## 6.3 Method

Our approach for tasks 1 and 3 is very similar. We have experimented with NBSVM, deep learning models, and the combination of a deep learning model as a feature extractor and SVM as a classifier. NBSVM is a strong baseline (S. Wang and Manning, 2012). The choice of which deep learning model to use as a feature extractor was determined by the average performance across three runs on our development split. The train/development split used for task 1 is 90% training and 10% development.

Table 6.2: F1 Score of the Positive Class on our Development Split of the Training set using NBSVM and Deep Learning Models (For deep learning models, scores are the average of three runs and the values in parentheses are for the corresponding character level model)

Task	Classifiers			
	NBSVM	CNN	LSTM	BiLSTM
1	<b>0.909</b>	0.877 (0.888)	0.848 (0.781)	0.876 (0.798)
3	<b>0.624</b>	0.619 (0.549)	0.591 (0.391)	0.622 (0.321)

For task 3, the development set was generated after a random undersampling of the majority class. We set the class imbalance to ratio 1:3 between the minority and the majority class. As shown in Table 6.2, the best deep learning model for task 1 was CNN and BiLSTM for task 3.

In our experiments, the NBSVM model uses logarithmic count ratios over character n-grams ranging from 1 to 5 characters as features. In the deep learning models, we used pre-trained fastText word embedding <sup>1</sup>. We train the SVM model using the RBF kernel.

For the deep learning models, we use the binary cross-entropy loss function as the objective function. To optimize the loss function, we use the ADAM optimizer with a learning rate of 0.001. We run the models for 100 epochs with earlystopping and dropout with a probability of 0.2 to avoid overfitting.

## 6.4 Results

Table 6.3 shows the performance of our systems on the evaluation data for task 1. The NBSVM model achieved the best recall (0.899) and F1 (0.910). These scores are above average. The mean values for precision, recall, and F1 scores on the leaderboard for the competition are 0.8904, 0.872, and 0.880 respectively. The CNN model slightly outperforms the NBSVM by 0.002 in terms of the precision score. For task 3, Table 6.4 shows that our BiLSTM+SVM model performs best out of our submissions by

---

<sup>1</sup><https://s3-us-west-1.amazonaws.com/fasttext-vectors/wiki.en.zip>

Table 6.3: Scores on the Evaluation Data for Task 1 (P-Precision; R-Recall; F-F1 measure)

<b>System</b>	<b>P</b>	<b>R</b>	<b>F</b>
NBSVM	0.920	<b>0.899</b>	<b>0.910</b>
CNN	<b>0.922</b>	0.786	0.848
CNN+SVM	0.909	0.803	0.853

Table 6.4: Scores on the Evaluation Data for Task 3 (P-Precision for the ADR class; R-Recall for the ADR class; F-F1 measure for the ADR class)

<b>System</b>	<b>P</b>	<b>R</b>	<b>F</b>
NBSVM	0.258	<b>0.795</b>	0.390
BiLSTM	0.293	0.586	0.390
BiLSTM+SVM	<b>0.314</b>	0.529	<b>0.394</b>

obtaining the best precision (0.314) and F1 (0.394) scores for the ADR class. The NBSVM model obtains a better recall for the ADR class (0.795). The difference in recall scores obtained by the models suggests that an ensemble of classifiers could lead to a better F1 score.

## 6.5 Conclusion

In this chapter, we developed three classifiers for tasks 1 and 3 on the SMM4H shared tasks with NBSVM, deep learning models (CNN, LSTM, and BiLSTM), and the combination of a deep learning model and SVM. For task 1, our best submission is with the NBSVM. The BiLSTM+SVM model received our best F1 score for the ADR class in task 3. The NBSVM model performed better in terms of recall in task 3.

In the future, we would like to explore the use of informed sampling techniques to deal with class imbalance. Furthermore, we would like to investigate the enrichment of the training data with semantic and conceptual domain-specific knowledge that could provide relevant priors for the classifiers.

# Chapter Seven

## Comparison of Word Embeddings for the Detection of Adverse Drug Reaction in Social Media Messages

### 7.1 Introduction

The goal of the social media mining for health care applications competition is to provide a benchmark for validating and comparing methods for healthcare applications using data from social media (Weissenbacher et al., 2019). Task 1 focuses on identifying adverse drug reaction as a drug-related outcome. For this task, successful systems are expected to distinguish between tweets that report side effects and tweets that do not. The evaluation metric is the F1 score. This task requires that a drug's adverse effect be distinguished from a similar and usually confusing expression of a drug's indication. The former is usually associated with the use of the drug, while the latter is a specification of the reason for using a drug. In addition, the task of detecting mention of an adverse reaction to a drug is a highly unbalanced binary classification. About 1% of the training set are positive examples and about 99% are negative examples. Our approach is based on the combination of three different types of word embedding representations, namely: character (Lample et al., 2016), non-contextual(Glove pre-trained on Twitter data) (Pennington et al., 2014b), and contextual(BERT) (Devlin et al., 2019).



In the next section, we provide details of our model and training set-up. Section 3 shows the results of our experiments, while we conclude and indicate possible directions for future work in Section 4.

## 7.2 Model and Experimental Set-Up

We hypothesize that the different types of embeddings capture diverse relationships and their combination could help in the identification of adverse drug reaction in tweets. In our experiments, word representations differ along two dimensions: whether they are pre-trained (Glove and Bert) or not (character embedding) and whether they are contextual (Bert) or otherwise (Glove and Character embeddings). We briefly describe each of them:

- Character embedding: this is a 50 dimensional representation of the characters in a word (they are combined to form a word embedding). This representation is learnt while training the model. It is based on a bidirectional LSTM. The advantage of the character-based representation for social media text is that it eliminates the problem of out-of-vocabulary words resulting from noise in the form of misspellings and abbreviations in word-based representations such as Glove. In addition, this representation is task and domain specific for the training set.
- Glove (twitter) - is a 100-dimensional representation pre-trained on a large Twitter corpus. We expect this embedding to reflect the language of Twitter users. However, the embedding is not contextual-dependent: a word has the same representation even though it appears in the company of other words.
- BERT (en, base-uncased) - is a contextual word representation for a general domain, here the representation of a word is determined by other words in its context (sentence). The base-uncased BERT model yields a word embedding of dimension 768. With this representation, several state-of-the-art results have been achieved on NLP tasks. However, to our knowledge, its successful application to texts in the social media domain is limited.

To take advantage of some of the above representations, we concatenated these representations for words in a tweet. This combination has a dimension of 918. A linear layer then projects this representation to a dimension of 256. This projection is meant to serve as a distillation and/or fine-tuning step. The resulting representation is fed into an LSTM layer with a hidden size of 512 to sequentially model a tweet. Finally, a dense layer is used as a classifier.

The model was trained for 100 epochs with a learning rate annealing factor of 0.5, using SGD as the optimizer and a batch size of 8. We used a ratio of 80:20

Table 7.1: Details of the Data

	No. of Examples
train	24202
dev	6051
test	4575

between training and development splits. Table 7.1 shows the number of training, validation, and evaluation examples used in our experiment. Weissenbacher et al., (2019) contains details on the collection and annotation of the dataset. Based on the validation split, a model with the best F1 score during training is saved as the best model. Using the best model, we made predictions for the evaluation examples as our first submission (sub1 in Table 7.2). Our second submission (sub2 in Table 7.2) was based on the model at the 100th epoch or the last epoch, as training is stopped if learning rate becomes too small. Our experiments were conducted using the Flair framework (Akbik et al., 2018).

## 7.3 Results

Table 7.3 shows the results obtained on the evaluation split. We achieved our best performance with the final model, an F1 of 0.5209. This result is higher than the average score of all participants on the task with average F1, precision, and recall of 0.5019, 0.5351, 0.5054 respectively (Weissenbacher et al., 2019). Table 7.3 shows the results obtained from our ablation experiments with respect to the contributions of

the different embedding representations and the distillation/fine-tuning step. The F1 scores shown are based on the model that obtained the best F1 score on the validation set during training. One can observe a slight decrease in performance (0.0045) when we did not use the fine-tuning layer. This suggests that either the fine-tuning layer is detrimental to performance or that the size of the resulting fine-tuned representation is an important parameter to tune with our approach. We evaluate the contribution of the three embedding representations to performance by removing each one after the other from the model and maintaining our fine-tuning strategy. When the representation of words with character embedding word is removed, we observe a decrease in performance of 0.0238. When the BERT representation is removed, the performance improves by 0.0025. Without the incorporation of the Glove embedding, the performance increases by 0.0005. The results are consistent with our perceived advantages and disadvantages of the three embedding representations. Character embedding contributes the most to the performance of the model. In particular, the removal of BERT and Glove leads to an improvement in performance. This can be attributed to the out-of-vocabulary problem with Glove and the domain mismatch in the case of BERT.

Table 7.2: Performance on the Test Set

	<b>P</b>	<b>R</b>	<b>F1</b>
sub1	0.6145	0.4457	0.5167
sub2	<b>0.6203</b>	<b>0.4489</b>	<b>0.5209</b>

Table 7.3: Ablation Study on the Validation Split

Model	F1
emb comb w/ fine tuning	0.9015
emb comb w/o fine tuning	0.9060
emb comb w/ fine tuning w/o character	0.8777
emb comb w/ fine tuning w/o Glove	0.9020
emb comb w/ fine tuning w/o BERT	0.9040

## 7.4 Conclusion

Our focus in this chapter is on identifying reports of adverse drug reactions in microblogs, tweets. We conducted experiments with the datasets provided for the 2019 edition of the social media mining for healthcare application shared tasks. Our approach is based on the combination of three different types of embedding representations and a fine-tuning strategy. Using this approach, we made two submissions using a model that achieved the best F1 score on the validation data and with a model trained till the last possible epoch. The latter performed better. Through ablation experiments, we observed that, contrary to our expectation, our fine-tuning strategy leads to a slight decrease in performance. Moreover, the different word representations contribute to different degrees. The character embedding representation makes the largest contribution, without it the performance of the model decreases while there is a marginal improvement in performance when the Glove and BERT representation are removed from the model.

As follow-up work, we would like to investigate other fine-tuning or distillation approaches as well as hyperparameter optimization of the size of the fine-tuning layer. It is also interesting to investigate the impact of normalizing tweets and identifying expressions of drug usage as an auxiliary task.

# Chapter Eight

## Automatic Prediction of the Assessment Dimensions of Human Behavior

Language allows us to express the evaluation of people, actions, events, and things. This manifests itself as emotion and assessment of human behaviour and artefacts. The study of evaluative language has benefited from efforts in several disciplines such as linguistics, philosophy, psychology, cognitive science and computer science (Benamara et al., 2017). In linguistics, the appraisal framework of Martin and White, (2003) provides a detailed classification scheme for understanding how evaluation is expressed and implied in language. In computer science, affective computing studies evaluative language under the umbrella term of sentiment analysis with common tasks related to polarity detection and classification, emotion recognition, and aspect-based sentiment analysis, among others. Sentiment analysis has benefited from the availability of user-generated content on online platforms.

The appraisal theory proposed by Martin and White, (ibid.) has three categories of evaluative text: affect, judgement, and appreciation. These categories respectively model opinions in terms of emotions, norms, and aesthetics. Utterances are seen as an indication of a positive (“praising”) or negative (“blaming”) attitude towards a particular object (person, thing, action, situation, or event). The dimensions of judgement are normality, capacity, tenacity, veracity, and propriety. Each of the

dimensions represents an answer to the following corresponding questions:

- Normality: How special?
- Tenacity: How dependable?
- Capacity: How capable?
- Veracity: How honest?
- Propriety: How far beyond reproach?

The corpus used in this chapter is annotated with the above judgement dimensions. Taboada and Grieve, (2004) automatically categorized appraisal into affect, judgement, and appreciation using a lexical approach that groups adjectives based on their semantic orientation. Benamara et al., (2017) examined linguistic and computational approaches to the study of evaluative text. Their analysis suggested that appraisal is a richer and more detailed task amenable to computational approaches depending on availability of data. They envision that appraisal analysis can contribute to advances in affective computing. Recently, Hofmann et al., (2020) has shown that the dimensions of appraisal can improve the detection of emotion in text. A similar observation was made by Whitelaw et al., (2005) who found appraisal phrases to be useful features for sentiment analysis.

This chapter explores the capabilities of machine learning models to predict dimensions of human judgement expressed in short texts (tweets) using the dataset of the ALTA-2020 shared task on assessing human behaviour (Mollá, 2020). The aim of the task is to advance computational techniques for analysing evaluative language.

Table 8.1: Frequency of each label in the training set as a fraction of the total number of examples.

Label	Normality	Capacity	Tenacity	Veracity	Propriety
Proportion	0.11	0.16	0.11	0.015	0.18

The use of neural networks has significantly improved performance on NLP tasks. However, neural networks require a large amount of labeled data. In contrast, traditional machine learning models such as NBSVM are competitive in domains

with small amount of data (Aroyehun and Gelbukh, 2018; S. Wang and Manning, 2012). The recently introduced contextual representation learning models (Devlin et al., 2019; Peters et al., 2018) are pre-trained with language modeling objective on a large and diverse text collection. The learned representation can be applied to downstream tasks through fine tuning (Howard and Ruder, 2018b). We consider the effectiveness of using NBSVM and fine tuning a Roberta-large model (Liu et al., 2019) to predict the dimensions of judgement short texts.

## 8.1 Methodology

**Task.** The task is to predict one or more judgement dimensions expressed in a given text. This is a multi-label classification problem where the labels consist of the five judgement dimensions.

**Data.** We use the data provided for the ALTA-2020 shared task (Mollá, 2020). There are 198 tweets in the training set. Each example is labeled with the presence or absence of each judgement dimension described in Section ???. Table 8.1 shows the proportion of each label in the training set. The proportion of each label ranges from 2% to 18%. The test set consists of 100 examples. Approximately 50% of the test set is used for the public leaderboard while the remainder is reserved for the private leaderboard on the Kaggle<sup>1</sup> In-class platform.

The private leaderboard is used for the final ranking, while the public leaderboard is used by participants to evaluate their models. In our experiment using the Roberta-large model, we created a validation split by randomly sampling 10% of the examples from the training set.

**Data Pre-processing.** We clean the text of each tweet by removing punctuation marks, digits, and repeated characters. We normalize URLs and usernames (tokens that starts with the @ symbol). Hashtags are converted to their constituent word(s) after removing the # symbol.

---

<sup>1</sup><https://www.kaggle.com/>

**NBSVM.** S. Wang and Manning, (2012) proposed a support vector machine (SVM) model that uses the naive bayes logarithmic count ratio as features. NBSVM is a strong linear model for text classification. In our implementation we use the logistic regression classifier instead of the SVM. The features are based on n-grams of words (unigrams and bigrams). We experiment with and without the data pre-processing step. In the multi-label classification setting, we train a binary classifier per label with the same classifier settings.

**Roberta-large.** An optimized BERT (Devlin et al., 2019) model trained longer and with a larger and more diverse text collection of 160GB. In addition, the pre-training tasks did not include the prediction of the next sentence and the tokenizer is based on the byte-pair encoding model(Liu et al., 2019). We fine tune the model on the data provided for the ALTA 2020 shared task without the step of data pre-processing . We used the simpletransformers library <sup>2</sup> for implementation. The classifier is a linear layer with a sigmoid activation function. The hyperparameters are: maximum learning rate of  $4e - 5$ , maximum number of epochs is 20 with early stopping on the validation loss using patience of 3, batch size of 64, model parameters are optimized by AdamW with a linear schedule and a warm-up steps of 4. We set the maximum sequence length to 128.

Table 8.2: Mean F1 score on the public and private test sets. Average is the unweighted mean of the scores on the private and public leaderboards as they are approximately 50% each of the test set.

Method	Public leaderboard	Private leaderboard	Average
NBSVM	<b>0.1600</b>	0.0000	0.0800
NBSVM w/ prep.	0.1600	0.0000	0.0800
Roberta-large	0.1167	0.0667	0.0917
Roberta-large w/ threshold	0.1429	<b>0.1547</b>	<b>0.1488</b>

**Prediction threshold.** Lipton et al., (2014) studied the difficulty of associating the maximum achievable F1 score with the decision thresholds for predicting conditional

---

<sup>2</sup><https://simpletransformers.ai/>



probabilities. They observed that the predictions chosen to maximize the F1 score are a function of the conditional probability assigned to the example and the conditional probability distribution of the other examples. Based on this observation, we choose a decision threshold for each label to track the distribution of conditional probabilities on the validation set without referring to the gold labels to avoid overfitting. The default decision threshold is 0.5 and we find that the conditional probabilities are much smaller. We apply this heuristic to the model output of the Roberta-large model. Specifically, we set 0.2 as the decision threshold of the *capacity* label, and set 0.1 as the decision threshold of the remaining labels.

## 8.2 Results

Table 8.2 shows the results obtained on the test set, divided into two equal halves for the public and private leaderboards. Using the NBSVM model, we achieved the best score of 0.16 on the public leaderboard. The application of the data pre-processing step did not affect the performance of the NBSVM model, possibly because the removed tokens are not the relevant lexical units for the task. Based on this observation, we did not apply the pre-processing step to experiments using the Roberta-large model. The Roberta-large model achieved a relatively low score on the public leaderboard and performed better on the other half of the test set, as shown by the scores on the private leaderboard. Due to the decision thresholding on the output of the Roberta-large model, the performance has seen significant improvement. With this approach, we achieved the best overall score on the ALTA-2020 competition leaderboard.

## 8.3 Conclusion

We address the task of automatically predicting judgement dimensions using the data and leaderboards provided for the ALTA-2020 shared task. We evaluated the performance of a strong linear classifier, NBSVM with n-grams as features, and a recent pre-trained language model, Roberta-large. We observed that the NBSVM achieved our best results on the public leaderboard, but the performance degrades

on the private test set. Hence, it did not generalize well. The Roberta-large model with the decision thresholding strategy showed consistent performance on the public and private leaderboards. Using this model, we achieved the best overall score on the leaderboard.

Although we have achieved better performance using the Roberta-large model, we believe that the statistical power (Card et al., 2020) of the test set is limited due to the small sample size (100 examples). Therefore, it is difficult to distinguish accidental performance improvement from substantial model advantage. We hope to test our method on a larger test set to check the robustness of our method.

# Chapter Nine

## Conclusion and Future Work

This thesis studied the application of deep learning to social media mining with a focus on the tasks of identification of objectionable content, pharmacovigilance, and the prediction of appraisal judgement dimensions. Chapter 3 explores using label information to improve text classification in the patent domain. We apply this insight also in Chapter 5. Chapter 4 introduces an improved classification model for the detection of objectionable content using a combination of data augmentation and pseudo labeling. This approach achieved the best performance on this task without the need for manual feature engineering. Evaluation on a test dataset from another social media platform shows a drop in performance. Building on the advances in pre-trained multilingual language models, Chapter 5 proposes a competitive all-in-one model for the detection of offensive content in multiple languages using a single model. This approach tradeoffs performance for efficiency in comparison to monolingual approaches. Chapter 6 explores a two-step approach to the identification of relevant social media posts and messages mentioning adverse drug reactions by using a deep learning model as feature extractor and feeding same to a standard machine learning model in a non-end-to-end fashion. The results indicate improved performance, though there is still a lot of room for improvement. Chapter 7 introduces analyses of the combination of heterogeneous word embedding for the identification of posts mentioning adverse drug reactions. The results reveal that the combination of contextual and non-contextual word representations provides performance improvement. Chapter 8 presents a state-of-the-art model for the identification of appraisal judgement dimensions using a

recent language model representation with decision thresholding.

For future work, we envision two pertinent directions. The first is the relaxation of the data requirements for the supervised models. This can vary from using unlabeled data as in unsupervised learning or learning with very few examples as in few-shot learning. The requirement of large labeled data is a limitation of the current work. Such labeled data are expensive to create. In addition, this requirement hampers the adaptation of the proposed models to new domains. The second avenue deals with multitask learning where several tasks can be jointly learned together. For example, the task of identifying posts containing drug name(s) and those mentioning adverse drug reactions can be learned together as they appear to be related. In addition, the all-in-one-model for identification of objectionable content can be trained using multitask learning framework rather than the data combination approach used in the present work.

# Bibliography

- Adhikari, Ashutosh, Achyudh Ram, Raphael Tang, and Jimmy Lin (2019). “Rethinking Complex Neural Network Architectures for Document Classification”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4046–4051.
- Akbik, Alan, Duncan Blythe, and Roland Vollgraf (2018). “Contextual String Embeddings for Sequence Labeling”. In: *COLING 2018, 27th International Conference on Computational Linguistics*, pp. 1638–1649.
- Aly, Rami, Steffen Remus, and Chris Biemann (2019). “Hierarchical multi-label classification of text with capsule networks”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pp. 323–330.
- Aroyehun, Segun Taofeek and Alexander Gelbukh (2018). “Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling”. In: *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pp. 90–97.
- Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton (2016). “Layer Normalization”. In: *arXiv preprint arXiv:1607.06450*.
- Baxter, Jonathan (1997). “A Bayesian / Information Theoretic Model of Learning to Learn via Multiple Task Sampling”. In: *Machine Learning* 28.1, pp. 7–39.

- Baziotis, Christos, Nikos Pelekis, and Christos Doukeridis (2017). “DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 747–754.
- Benamara, Farah, Maite Taboada, and Yannick Mathieu (2017). “Evaluative language beyond bags of words: Linguistic insights and computational applications”. In: *Computational Linguistics* 43.1, pp. 201–264.
- Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2016). “Enriching Word Vectors with Subword Information”. In: *arXiv preprint arXiv:1607.04606*.
- Card, Dallas, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky (2020). “With Little Power Comes Great Responsibility”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9263–9274.
- Caruana, Rich (1997). “Multitask Learning”. In: *Machine Learning* 28.1, pp. 41–75.
- Chaturvedi, Iti, Yew-Soon Ong, Ivor W. Tsang, Roy E. Welsch, and Erik Cambria (2016). “Learning word dependencies in text by means of a deep recurrent belief network”. In: *Knowledge-Based Systems* 108. New Avenues in Knowledge Bases for Natural Language Processing, pp. 144–154. ISSN: 0950-7051. DOI: <https://doi.org/10.1016/j.knosys.2016.07.019>.
- Chen, Wenhui, Yu Su, Yilin Shen, Zhiyu Chen, Xifeng Yan, and William Yang Wang (2019). “How Large a Vocabulary Does Text Classification Need? A Variational Approach to Vocabulary Selection”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3487–3497.

- Cho, Kyunghyun, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (2014). “On the Properties of Neural Machine Translation: Encoder–Decoder Approaches”. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111.
- Choi, HongSeok and Hyunju Lee (2019). “Multitask Learning Approach for Understanding the Relationship Between two Sentences”. In: *Information Sciences* 485, pp. 413–426.
- Choi, Jihun, Taeuk Kim, and Sang-goo Lee (2019). “A Cross-Sentence Latent Variable Model for Semi-Supervised Text Sequence Matching”. In: *Proceedings of the 2019 Meeting of the Association for Computational Linguistics*, pp. 4747–4761.
- Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov (2020). “Unsupervised Cross-lingual Representation Learning at Scale”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 8440–8451. DOI: [10.18653/v1/2020.acl-main.747](https://doi.org/10.18653/v1/2020.acl-main.747). URL: <https://www.aclweb.org/anthology/2020.acl-main.747>.
- Davidson, Thomas, Dana Warmusley, Michael Macy, and Ingmar Weber (2017). “Automated Hate Speech Detection and the Problem of Offensive Language”. In: *Proceedings of ICWSM*.
- Davy, Weissenbacher, Sarker Abeed, Paul Michael, and Gonzalez-Hernandez Graciela (2018). “Overview of the Third Social Media Mining for Health (SMM4H) Shared Tasks at EMNLP 2018”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the*

*Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: [10.18653/v1/N19-1423](https://doi.org/10.18653/v1/N19-1423). URL: <https://www.aclweb.org/anthology/N19-1423>.

D’hondt, Eva, Suzan Verberne, Cornelis Koster, and Lou Boves (2013). “Text representations for patent classification”. In: *Computational Linguistics* 39.3, pp. 755–775.

Dietterich, Thomas G. (1998). “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms”. In: *Neural Computation* 10.7, pp. 1895–1923.

Fall, Caspar J, Atilla Töröcsvári, Karim Benzineb, and Gabor Karetka (2003). “Automated Categorization in the International Patent Classification”. In: *ACM SIGIR Forum*. Vol. 37. New York, NY, USA: Association for Computational Machinery, pp. 10–25.

Founta, Antigoni-Maria, Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Athena Vakali, and Ilias Leontiadis (2018). “A Unified Deep Learning Architecture for Abuse Detection”. In: *CoRR* abs/1802.00385. arXiv: [1802.00385](https://arxiv.org/abs/1802.00385). URL: <http://arxiv.org/abs/1802.00385>.

Gambäck, Björn and Utpal Kumar Sikdar (2017). “Using Convolutional Neural Networks to Classify Hate-Speech”. In: *Proceedings of the First Workshop on Abusive Language Online*. Vancouver, BC, Canada: Association for Computational Linguistics, pp. 85–90. DOI: [10.18653/v1/W17-3013](https://doi.org/10.18653/v1/W17-3013). URL: <https://aclanthology.org/W17-3013>.

Gao, Lei, Alexis Kuppersmith, and Ruihong Huang (2017). “Recognizing Explicit and Implicit Hate Speech Using a Weakly Supervised Two-path Bootstrapping Approach”. In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Vol. 1. Taipei, Taiwan, pp. 774–782.



- Graves, Alex, Navdeep Jaitly, and Abdel-rahman Mohamed (2013). “Hybrid speech recognition with deep bidirectional LSTM”. In: *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, pp. 273–278.
- Guyot, Jacques, Karim Benzineb, Gilles Falquet, and Simple Shift (2010). “my-Class: A Mature Tool for Patent Classification.” In: *CLEF (notebook papers/labs/workshops)*.
- Hochreiter, Sepp and Jürgen Schmidhuber (1997a). “Long Short-Term Memory”. In: 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735).
- Hochreiter, Sepp and Jürgen Schmidhuber (1997b). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780.
- Hofmann, Jan, Enrica Troiano, Kai Sassenberg, and Roman Klinger (2020). “Appraisal Theories for Emotion Classification in Text”. In: *arXiv preprint arXiv:2003.14155*.
- Howard, Jeremy and Sebastian Ruder (2018a). “Universal Language Model Fine-tuning for Text Classification”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339.
- Howard, Jeremy and Sebastian Ruder (2018b). “Universal Language Model Fine-tuning for Text Classification”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 328–339. DOI: [10.18653/v1/P18-1031](https://doi.org/10.18653/v1/P18-1031). URL: <https://www.aclweb.org/anthology/P18-1031>.
- Jean, Sébastien, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio (2015). “On Using Very Large Target Vocabulary for Neural Machine Translation”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1–10.

- Kaushik, Divyansh, Eduard Hovy, and Zachary C Lipton (2020). “Learning the Difference that Makes a Difference with Counterfactually Augmented Data”. In: *International Conference on Learning Representations (ICLR)*.
- Kendall, Alex, Yarin Gal, and Roberto Cipolla (2018). “Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7482–7491.
- Kim, Yoon (2014a). “Convolutional Neural Networks for Sentence Classification”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pp. 1746–1751. DOI: [10.3115/v1/D14-1181](https://doi.org/10.3115/v1/D14-1181). URL: <http://anthology.aclweb.org/D/D14/D14-1181.pdf>.
- Kim, Yoon (2014b). “Convolutional Neural Networks for Sentence Classification”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1746–1751. DOI: [10.3115/v1/D14-1181](https://doi.org/10.3115/v1/D14-1181).
- Kingma, Diederik P. and Jimmy Ba (2014). “Adam: A Method for Stochastic Optimization”. In: *arXiv preprint arXiv:1412.6980*.
- Kumar, Ritesh, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri (2018). “Benchmarking Aggression Identification in Social Media”. In: *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC)*. Santa Fe, USA.
- Kumar, Ritesh, Atul Kr Ojha, Shervin Malmasi, and Marcos Zampieri (2018). “Benchmarking aggression identification in social media”. In: *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pp. 1–11.
- Kumar, Ritesh, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri (2020). “Evaluating Aggression Identification in Social Media”. In: *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*. Marseille, France:

European Language Resources Association (ELRA), pp. 1–5. URL: <https://www.aclweb.org/anthology/2020.trac-1.1>.

Kumar, Ritesh, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari (2018). “Aggression-annotated Corpus of Hindi-English Code-mixed Data”. In: *Proceedings of the 11th Language Resources and Evaluation Conference (LREC)*. Miyazaki, Japan.

Lample, Guillaume, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer (2016). “Neural Architectures for Named Entity Recognition”. In: *Proceedings of NAACL-HLT*, pp. 260–270.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (1998). “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11, pp. 2278–2324. DOI: [10.1109/5.726791](https://doi.org/10.1109/5.726791).

Li, Shaobo, Jie Hu, Yuxin Cui, and Jianjun Hu (2018). “DeepPatent: Patent Classification with Convolutional Neural Networks and Word Embedding”. In: *Scientometrics* 117.2, pp. 721–744.

Li, Yang, Quan Pan, Suhang Wang, Tao Yang, and Erik Cambria (2018). “A generative model for category text generation”. In: *Information Sciences* 450, pp. 301–315.

Lipton, Zachary C., Charles Elkan, and Balakrishnan Naryanaswamy (2014). “Optimal Thresholding of Classifiers to Maximize F1 Measure”. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 225–239.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692*.

- Loshchilov, Ilya and Frank Hutter (2019). “Decoupled Weight Decay Regularization”. In: *Proceedings of the International Conference on Learning Representations*, pp. 1–8.
- Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning (2015). “Effective Approaches to Attention-based Neural Machine Translation”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421.
- Ma, Yukun, Haiyun Peng, Tahir Khan, Erik Cambria, and Amir Hussain (2018). “Sentic LSTM: a hybrid network for targeted aspect-based sentiment analysis”. In: *Cognitive Computation* 10.4, pp. 639–650.
- Malmasi, Shervin and Marcos Zampieri (2017). “Detecting Hate Speech in Social Media”. In: *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pp. 467–472.
- Malmasi, Shervin and Marcos Zampieri (2018). “Challenges in Discriminating Profanity from Hate Speech”. In: *Journal of Experimental & Theoretical Artificial Intelligence* 30 (2), pp. 1–16.
- Mandl, Thomas, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel (2019). “Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages”. In: *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pp. 14–17.
- Mandl, Thomas, Sandip Modha, Gautam Kishore Shahi, Amit Kumar Jaiswal, Durgesh Nandini, Daksh Patel, Prasenjit Majumder, and Johannes Schäfer (2020). “Overview of the HASOC track at FIRE 2020: Hate Speech and Offensive Content Identification in Indo-European Languages”. In: *Working Notes of FIRE 2020 - Forum for Information Retrieval Evaluation*. CEUR.
- Martin, James R. and Peter R. White (2003). *The language of evaluation*. Vol. 2. Springer.

- McCann, Bryan, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher (2018). “The Natural Language Decathlon: Multitask Learning as Question Answering”. In: *arXiv preprint arXiv:1806.08730*.
- McNemar, Quinn (1947). “Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages”. In: *Psychometrika* 12.2, pp. 153–157.
- Mikolov, Tomas, Edouard Grave, Piotr Bojanowski, Christian Puhersch, and Armand Joulin (2018). “Advances in Pre-Training Distributed Word Representations”. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.
- Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013). “Distributed Representations of Words and Phrases and their Compositionality”. In: *Advances in Neural Information Processing Systems*. Ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Vol. 26. Curran Associates, Inc., pp. 3111–3119.
- Mollá, Diego (2020). “Overview of the 2020 ALTA Shared Task: Assess Human Behaviour”. In: *Proceedings of the 18th Annual Workshop of the Australasian Language Technology Association*.
- Mou, Lili, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin (2016). “How Transferable are Neural Networks in NLP Applications?” In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 479–489.
- Nobata, Chikashi, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang (2016). “Abusive Language Detection in Online User Content”. In: *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, pp. 145–153.
- Pamungkas, Endang Wahyu and Viviana Patti (2019). “Cross-domain and Cross-lingual Abusive Language Detection: A Hybrid Approach with Deep Learning

and a Multilingual Lexicon”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. Florence, Italy: Association for Computational Linguistics, pp. 363–370. DOI: [10.18653/v1/P19-2051](https://doi.org/10.18653/v1/P19-2051). URL: <https://www.aclweb.org/anthology/P19-2051>.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., pp. 8024–8035.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014a). “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: [10.3115/v1/D14-1162](https://doi.org/10.3115/v1/D14-1162).

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014b). “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Peters, Matthew, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (2018). “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: [10.18653/v1/N18-1202](https://doi.org/10.18653/v1/N18-1202). URL: <https://www.aclweb.org/anthology/N18-1202>.

- Qian, Jing, Mai ElSherief, Elizabeth M Belding, and William Yang Wang (2018). “Leveraging Intra-User and Inter-User Representation Learning for Automated Hate Speech Detection”. In: *arXiv preprint arXiv:1804.03124*.
- Ranasinghe, Tharindu and Marcos Zampieri (2020). “Multilingual Offensive Language Identification with Cross-lingual Embeddings”. In: *arXiv preprint arXiv:2010.05324*.
- Risch, Julian and Ralf Krestel (2018). “Learning Patent Speak: Investigating Domain-Specific Word Embeddings”. In: *Proceedings of ICDIM 2018, Thirteenth International Conference on Digital Information Management*. IEEE, pp. 63–68.
- Risch, Julian and Ralf Krestel (2020). “Bagging bert models for robust aggression identification”. In: *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pp. 55–61.
- Ruder, Sebastian (2017). “An Overview of Multi-Task Learning in Deep Neural Networks”. In: *arXiv preprint arXiv:1706.05098*.
- Sabour, Sara, Nicholas Frosst, and Geoffrey E Hinton (2017). “Dynamic Routing Between Capsules”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., pp. 3859–3869.
- Sarker, Abeed, Rachel Ginn, Azadeh Nikfarjam, Karen O’Connor, Karen Smith, Swetha Jayaraman, Tejaswi Upadhaya, and Graciela Gonzalez (2015). “Utilizing social media data for pharmacovigilance: A review”. In: *Journal of Biomedical Informatics* 54, pp. 202–212. ISSN: 1532-0464. DOI: <https://doi.org/10.1016/j.jbi.2015.02.004>. URL: <https://www.sciencedirect.com/science/article/pii/S1532046415000362>.
- Schmidt, Anna and Michael Wiegand (2017). “A Survey on Hate Speech Detection Using Natural Language Processing”. In: *Proceedings of the Fifth International*

*Workshop on Natural Language Processing for Social Media. Association for Computational Linguistics*. Valencia, Spain, pp. 1–10.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2016). “Improving Neural Machine Translation Models with Monolingual Data”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 86–96. DOI: [10.18653/v1/P16-1009](https://doi.org/10.18653/v1/P16-1009). URL: <https://www.aclweb.org/anthology/P16-1009>.

Simard, Patrice Y, David Steinkraus, John C Platt, et al. (2003). “Best practices for convolutional neural networks applied to visual document analysis.” In: *ICDAR*. Vol. 3, pp. 958–962.

Sloane, Richard, Orod Osanlou, David Lewis, Danushka Bollegala, Simon Maskell, and Munir Pirmohamed (2015). “Social media and pharmacovigilance: A review of the opportunities and challenges”. In: *British Journal of Clinical Pharmacology* 80.4, pp. 910–920.

Søgaard, Anders and Yoav Goldberg (2016). “Deep Multi-Task Learning with Low Level Tasks Supervised at Lower Layers”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 231–235.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *The Journal of Machine Learning Research* 15.1, pp. 1929–1958. ISSN: 1532-4435.

Al-Stouhi, Samir and Chandan K. Reddy (2016). “Transfer Learning for Class Imbalance Problems with Inadequate Data”. In: *Knowledge and Information Systems* 48.1, pp. 201–228.



- Sun, Chi, Xipeng Qiu, Yige Xu, and Xuanjing Huang (2019). “How to fine-tune bert for text classification?” In: *China National Conference on Chinese Computational Linguistics*. Springer, pp. 194–206.
- Taboada, Maite and Jack Grieve (2004). “Analyzing appraisal automatically”. In: *In Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*.
- Tompson, Jonathan, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler (2015). “Efficient object localization using convolutional networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 648–656.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). “Attention is All you Need”. In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., pp. 5998–6008. URL: <http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf>.
- Wang, Guoyin, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin (2018). “Joint Embedding of Words and Labels for Text Classification”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 2321–2331. DOI: [10.18653/v1/P18-1216](https://doi.org/10.18653/v1/P18-1216).
- Wang, Jin, Zhongyuan Wang, Dawei Zhang, and Jun Yan (2017). “Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification”. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 2915–2921. DOI: [10.24963/ijcai.2017/406](https://doi.org/10.24963/ijcai.2017/406).

- Wang, Sida and Christopher D Manning (2012). “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*. Association for Computational Linguistics, pp. 90–94.
- Waseem, Zeerak, Thomas Davidson, Dana Warmusley, and Ingmar Weber (2017). “Understanding Abuse: A Typology of Abusive Language Detection Subtasks”. In: *Proceedings of the First Workshop on Abusive Language Online*.
- Wei, Jason and Kai Zou (2019). “EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 6382–6388. DOI: [10.18653/v1/D19-1670](https://doi.org/10.18653/v1/D19-1670). URL: <https://www.aclweb.org/anthology/D19-1670>.
- Weissenbacher, Davy, Abeed Sarker, Arjun Magge, Ashlynn Daughton, Karen O’Connor, Michael Paul, and Graciela Gonzalez-Hernandez (2019). “Overview of the Fourth Social Media Mining for Health (SMM4H) Shared Task at ACL 2019.” In: *Proceedings of the 2019 ACL Workshop SMM4H: The 4th Social Media Mining for Health Applications Workshop & Shared Task*.
- Whitelaw, Casey, Navendu Garg, and Shlomo Argamon (2005). “Using appraisal groups for sentiment analysis”. In: *Proceedings of the 14th ACM International Conference on Information and knowledge management*, pp. 625–631.
- WHO (2021). *Pharmacovigilance*. [https://www.who.int/medicines/areas/quality\\_safety/safety\\_efficacy/safety/en/](https://www.who.int/medicines/areas/quality_safety/safety_efficacy/safety/en/). (accessed on May 31, 2021).
- World Intellectual Property Organization (2020). *Guide to the International Patent Classification*. Tech. rep. Geneva, Switzerland: World Intellectual Property Organization.

- Xie, Qizhe, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le (2020). “Unsupervised Data Augmentation for Consistency Training”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 6256–6268.
- Zampieri, Marcos, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin (2020). “SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020)”. In: *arXiv preprint arXiv:2006.07235*.
- Zhang, Ziqi, David Robinson, and Jonathan Tepper (2018). “Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network”. In: *The Semantic Web*. Ed. by Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam. Cham: Springer International Publishing, pp. 745–760. ISBN: 978-3-319-93417-4.
- Zhao, Wei, Haiyun Peng, Steffen Eger, Erik Cambria, and Min Yang (2019). “Towards Scalable and Reliable Capsule Networks for Challenging NLP Applications”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 1549–1559. DOI: [10.18653/v1/P19-1150](https://doi.org/10.18653/v1/P19-1150).