Universidade de
Aveiro
Ano 2022

Edgar Guilherme Silva Morais

Identificação e análise de estados de saúde em mensagens do Twitter

Identification and analysis of health states in Twitter messages

Universidade de Aveiro
Ano 2022

Edgar Guilherme Silva Morais

Identificação e análise de estados de saúde em mensagens do Twitter

Identification and analysis of health states in Twitter messages

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica da Doutora Alina Trifan, Professora Auxiliar Convidada do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor José Luís Oliveira, Professor Catedrático do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e colaboração da Doutora Olga Fajarda, Investigadora Doutorada, do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president

Prof. Doutora Ana Maria Perfeito Tomé
Professora Associada da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor José Alberto Benítez-Andrades
Assistant Professor da Universidade de León, Espanha

Prof. Doutora Alina Liliana Trifan
Professora Auxiliar Convidada da Universidade de Aveiro

Agradecimentos / acknowledgements

First of all, I would like to express my deepest gratitude to my supervisors, Dr. Alina Trifan, Dr. Olga Fajarda, and Dr. José Luís Oliveira for all their patience, feedback, and guidance. I am also thankful to my friends, especially my friend Inês Vinagre, for all the emotional support she has given me throughout this whole process. Lastly, I would like to thank my family, particularly my parents for believing in me and helping me every step of the way.

palavras-chave

redes sociais, informação de saúde, processamento de linguagem natural, aprendizagem automática.

resumo

As redes sociais tornaram-se muito utilizadas por todo o mundo, permitindo ligar pessoas de diferentes países e criar comunidades globais. O Twitter, uma das redes sociais mais populares, permite que os seus utilizadores partilhem segmentos curtos de texto com um máximo de 280 caracteres. Esta partilha na rede gera uma enorme quantidade de dados sobre os seus utilizadores, podendo ser analisados sobre múltiplas perspetivas. Por exemplo, podem ser utilizados para extrair informação sobre a saúde de um segmento da população tendo em vista a vigilância de saúde pública.
O objetivo deste trabalho foi a investigação e o desenvolvimento de soluções técnicas para participar no “Social Media Mining for Health Shared Task” (#SMM4H), um desafio constituído por diversas tarefas de processamento de linguagem natural relacionadas com o uso de dados provenientes de redes sociais para o propósito de investigação na área da saúde. O trabalho envolveu o desenvolvimento de modelos baseados em transformadores e outras técnicas relacionadas, para participação na tarefa 1 deste desafio, que por sua vez está dividida em 3 subtarefas: 1a) classificação de tweets relativamente à presença ou não de eventos adversos de medicamentos (ADE); 1b) reconhecimento de entidades com o objetivo de detetar menções de ADE; 1c) tarefa de normalização com o objetivo de associar as menções de ADE ao termo MedDRA correspondente (“Medical Dictionary for Regulatory Activities”). A abordagem com melhor desempenho na tarefa 1a foi um modelo *BERTweet large* treinado com dados gerados através de um processo de *data augmentation*. Relativamente à tarefa 1b, os melhores resultados foram obtidos usando um modelo *RoBERTa large* com dados de treino sobreamostrados. Na tarefa 1c utilizou-se um modelo *RoBERTa base* treinado com dados adicionais provenientes de um conjunto de dados externo. A abordagem utilizada na terceira tarefa não conseguiu alcançar resultados relevantes (F1 de 0.12), enquanto que os sistemas desenvolvidos para as duas primeiras alcançaram resultados ao nível dos melhores do desafio (F1 de 0.69 e 0.66 respetivamente).

keywords

social media, health information, natural language processing, machine learning.

abstract

Social media has become very widely used all over the world for its ability to connect people from different countries and create global communities. One of the most prominent social media platforms is Twitter. Twitter is a platform where users can share text segments with a maximum length of 280 characters. Due to the nature of the platform, it generates very large amounts of text data about its users' lives. This data can be used to extract health information about a segment of the population for the purpose of public health surveillance. Social Media Mining for Health Shared Task is a challenge that encompasses many Natural Language Processing tasks related to the use of social media data for health research purposes. This dissertation describes the approach I used in my participation in the Social Media Mining for Health Shared Task. I participated in task 1 of the Shared Task. This task was divided into three subtasks. Subtask 1a consisted of the classification of Tweets regarding the presence of Adverse Drug Events. Subtask 1b was a Named Entity Recognition task that aimed at detecting Adverse Drug Effect spans in tweets. Subtask 1c was a normalization task that sought to match an Adverse Drug Event mention to a Medical Dictionary for Regulatory Activities preferred term ID. Toward discovering the best approach for each of the subtasks I made many experiments with different models and techniques to distinguish the ones that were more suited for each subtask. To solve these subtasks, I used transformer-based models as well as other techniques that aim at solving the challenges present in each of the subtasks. The best-performing approach for subtask 1a was a BERTweet large model trained with an augmented training set. As for subtask 1b, the best results were obtained through a RoBERTa large model with oversampled training data. Regarding subtask 1c, I used a RoBERTa base model trained with data from an additional dataset beyond the one made available by the shared task organizers. The systems used for subtasks 1a and 1b both achieved state-of-the-art performance, however, the approach for the third subtask was not able to achieve favorable results. The system used in subtask 1a achieved an F1 score of 0.698, the one used in subtask 1b achieved a relaxed F1 score of 0.661, and the one used in the final subtask achieved a relaxed F1 score of 0.116.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ADE	Adverse Drug Effect
ADR	Adverse Drug Reaction
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
BioBERT	Bidirectional Encoder Representations from Transformers for Biomedical Text Mining
BOW	Bag of Words
BRNN	Bidirectional Recurrent Neural Network
CADEC	CSIRO Adverse Drug Event Corpus
CNN	Convolutional Neural Networks
EHR	Electronic Health Records
FN	False Negatives
FP	False Positives
GloVe	Global Vectors
IDF	Inverse Document Frequency
IREP	Incremental Reduced Error Pruning
LSTM	Long Short-Term Memory
MADE	Medications and Adverse Drug Events from Electronic Health Records
MedDRA	Medical Dictionary for Regulatory Activities
ML	Machine Learning
MLM	Masked Language Model
MLP	Multilayer Perceptron
MTL	Multi-Task Learning
NER	Named Entity Recognition
NLP	Natural Language Processing
NN	Neural Network
NSP	Next Sentence Prediction
POS	Part of Speech

RIPPER	Repeated Incremental Pruning to Produce Error Reductions
RNN	Recurrent Neural Network
RoBERTa	Robustly optimized BERT approach
SMILES	Simplified Molecular Input Line-entry System
SMM4H	Social Media Mining for Health Applications
SVM	Support Vector Machine
TDNN	Time Delay Neural Network
TF-IDF	Term Frequency – Inverse Document Frequency
TN	True Negatives
TP	True Positives
tsv	tab-separated values
UGC	User-Generated Content
UMLS	Unified Medical Language System
URL	Uniform Resource Locator

1 Introduction

Social media has become very widely used by people all over the world for all sorts of purposes, whether it be connecting with other people, checking the news, or sharing their opinions and experiences. Because of this, social networks have managed to produce very high amounts of data relating to a plethora of subjects. The data produced can contain valuable information, however, due to its high quantity, it is impossible to analyse it all manually.

Twitter[1] is a social media platform in which users mainly publish text segments with a maximum length of 280 characters referred to as tweets. Due to its widespread use and high user count, it generates high amounts of user-generated content very quickly. Furthermore, through Twitter data analysis it is also possible to track public opinion on several issues [1].

Public health surveillance consists of monitoring public health for the good of the population. A part of public health surveillance is pharmacovigilance, which focuses on the detection of adverse events related to medicinal products [2]. An example of an application of Twitter data analysis in public health surveillance is the detection of health conditions and disorders in tweets, which is researched by Prieto et al. [1]. In the more specific case of pharmacovigilance, an example of an application of data analysis is the detection of Adverse Drug Events (ADE) in tweets.

1.1 Motivation

There has been previous work on the identification of health states and specific vocabulary from electronic health records (EHRs) [3]. Electronic health records are digital records of patients and include information such as the patient's medical history, medications, test results, etc. The Medications and Adverse Drug Events from Electronic Health Records (MADE) 2018 challenge [3] invited participants to create and submit systems for solving three shared tasks. These shared tasks encompassed the Named Entity Recognition (NER) of entities on EHRs as well as the classification of the relationships between those mentions. This challenge helped reveal state-of-the-art approaches towards solving the mentioned shared tasks. This challenge works with EHR, meaning that it deals with data generated by healthcare professionals that contains technical medical vocabulary.

The Social Media Mining for Health Applications (#SMM4H) Shared Task [4] consists of Natural Language Processing (NLP) challenges related to using social media data for

[1] https://twitter.com/

health research. The challenges presented consist mainly of classification tasks related to the identification of health states in Twitter messages. This year was the seventh edition of the shared task and each of the previous editions included the publishing of a plethora of articles related to the different approaches used for each of the tasks proposed in that year's edition. This means that each edition presents a general idea of the state-of-the-art approaches existent for the tasks proposed for that year. These tasks include the identification of ADEs, adverse pregnancy outcomes, symptoms and other health states in social media text and encompass some other languages besides English. Contrary to the MADE challenge, this shared task uses social media data that introduces challenges present solely in this kind of user-generated data, such as the use of informal language, non-technical vocabulary, and the occurrence of social media exclusive elements (user mentions, URLs). The existence of these challenges and the usefulness of using social media data for public health surveillance, makes this specific use of NLP in social media data for health research a very relevant research topic.

1.2 Objectives

The main objective of this dissertation is the development of computational models, using NLP techniques, machine learning algorithms, and deep learning to analyse social media data and extract health-related information. In order to accomplish the mentioned objective, I have participated in a task of the Social Media Mining for Health Applications Shared Task in the 2022 edition [4].

The chosen task is task 1, which consists of the classification, extraction, and normalization of ADEs in English tweets. This task was separated into three subtasks, the classification task, the extraction task, and the normalization task. The classification task consists of a binary classification of tweets relating to the presence of an ADE mention. The extraction task consists of a NER task in which the span of text that contains the detected ADE mention is extracted. Finally, the normalization subtask consists of labelling the detected ADEs with a Medical Dictionary for Regulatory Activities (MedDRA) term id. This task introduces a couple of challenges that require additional methods beyond simple applications of deep learning. The first challenge is present in subtasks 1a and 1b and relates to the class imbalance present in the training datasets, in the sense that tweets containing ADE are the minority in the datasets. The use of unbalanced training data negatively influences the training process of the used models, resulting in worse results. The second challenge has to do with the fact that MedDRA contains 23,000 preferred terms [4]. This implies that the third classification subtask has a very large label space that is not represented in the training sets requiring the system to recognize labels it has never seen before.

1.3 Thesis structure

This work is comprised of four additional chapters after this introduction:

- Background. This chapter introduces a variety of NLP and machine learning concepts and algorithms. Furthermore, it also presents some works that aimed at solving the same task as the one tackled in this dissertation.
- Methodology. In this chapter, I described the experiments I used to discover the most effective approach towards solving each of the subtasks. In addition, I also presented the system I used to obtain the final submission results for the shared task.
- Results. This chapter presents the results of the experiments described in the methodology chapter. Additionally, I also compare and analyse the results obtained to discern the best methods to use for each subtask. Finally, I presented the performance measures for my SMM4H Shared Task submission and compared them with the average submission results.
- Conclusion. This chapter summarizes this dissertation and presents some suggestions for future work.

2 Background

The main problems this dissertation proposes to tackle are binary and multi-label classification and NER tasks. A classification problem is a problem that proposes the labelling of each data point with one of multiple classes. Depending on the problem, these classes can hold different meanings. In the case of the subtask 1a, mentioned in the previous chapter, tweets are labelled with the classes ADE and noADE that represent, respectively, the presence of at least an ADE mention in the tweet and the absence of an ADE mention in the tweet. Subtask 1c is also considered a multiclass classification problem where the classes are the dictionary preferred term ids. A NER problem is a problem where certain entities are identified in text segments. These entities can vary widely depending on the domain of the task at hand. Subtask 1b is a NER task where the entities detected are ADE mentions, meaning that the task aims at the detection of spans of ADE mentions in tweets.

2.1 Natural Language Processing

In this chapter several concepts necessary in order to understand the work done in this dissertation are defined. This dissertation revolves around the use of NLP to solve some tasks. NLP is the use of artificial intelligence in conjunction with linguistics to develop systems capable of performing advanced language comprehension challenges [5]. Initially, NLP consisted of the usage of handmade rules and grammars to solve natural language comprehension problems, however, the complexity of natural language raised issues and challenges not solvable by the stated methods. These problems led to the origin of statistical NLP methods that, instead of relying on handcrafted rules, relied on methods that build statistical rules based on a set of annotated data. As such, these methods are the ones that I am going to focus on in this chapter.

The statistical NLP methods encompass mainly machine learning algorithms. Machine learning algorithms are algorithms that allow a program to detect patterns in data and, through those patterns, create generalizations that can be used to generate predictions about new data [5]. These algorithms must be trained using a set of training data for the machine learning model to compute the optimal parameters to use for a specific task. Only after this training phase is the program able to make accurate predictions regarding a specific task.

An important subset of machine learning methods is deep learning [6]. These methods are neural networks characterized by having a substantial number of layers. These methods use a large number of non-linear processing units distributed across many layers

to extract features from data. I will present some of these algorithms in the following sections of this chapter.

Solving NLP problems requires a text processing pipeline with different phases for machine learning models to be used. The base NLP pipeline that uses machine learning algorithms is mainly defined by the following phases:

- Pre-processing phase.
- Feature extraction phase.
- Prediction phase (Classification or detection).

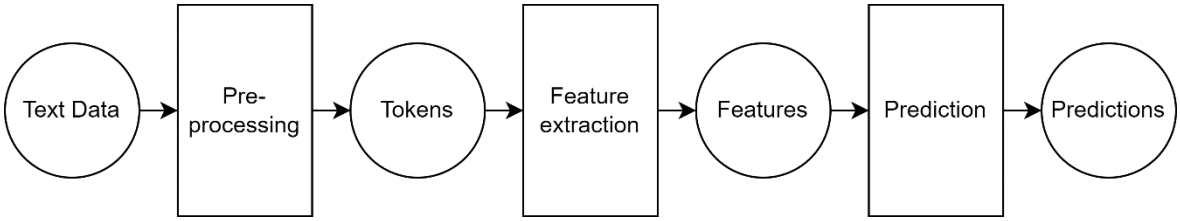

Figure 2.1- Base NLP pipeline.

The pipeline phases and their inputs and outputs are represented in Figure 2.1. In this chapter, I will present approaches and techniques regarding each of these phases in this chapter. This includes several NLP techniques and machine learning algorithms used in each of these stages. In addition, I will present several techniques and machine learning algorithms used in NLP problems.

2.2 Pre-processing

This sub-section presents some techniques used to pre-process the data. The pre-processing consists of ways in which the text of the dataset is transformed in order to turn it into a format usable by machine learning algorithms.

2.2.1 Tokenization

Tokenization is the process of dividing text into multiple segments called tokens [7]. Most of the time these tokens are words. This process has to deal with language-specific and even source-specific characteristics of the text.

Different languages use different punctuation in several ways, possibly leading to some problems when dividing the text into words. For the English language, an example of a problem of this nature is how to tokenize the text “I’ve” or “merry-go-round”. Furthermore, the source of the text can also bring some issues when tokenizing. In the case of user-

generated text from social networks, the use of emojis, abbreviations, and URLs might introduce some issues in the tokenization process that have to be dealt with.

This tokenization process sometimes also deals with the removal of stop words. In every language, there are very common words that do not add any information to the sentence they are used in. An example of these words in the case of the English language is the words "the", "a", "I" and "of". Since they are not critical to the understanding and meaning of the message, in information retrieval tasks they are often ignored when processing text.

2.2.2 Stemming and Lemmatization

Lemmatization and stemming are text-processing techniques used in information retrieval. Both techniques aim at reducing different variations of certain words as a way to improve the performance of information retrieval tasks.

Stemming simplifies different variations of words into their stem (grammatical root) by removing common prefixes and suffixes. An example of this technique is the reduction of "conditional" to "condition" or "glasses" to "glass". The main difference between different stemmers lies in the degree to which the word is altered. More aggressive stemmers change words into shorter stems while more subtle stemmers apply less drastic changes to the words. Some common stemmers are the Porter Stemmer [8] and Snowball Stemmer also known as Porter 2 Stemmer [9]. The Snowball Stemmer is more aggressive than the Porter Stemmer and fixes some issues present in the Porter Stemmer.

On the other hand, lemmatization reduces words using more intricate methods that also consider context and morphological analysis [10]. In other words, while stemming simply reduces the word without understanding its meaning, lemmatization considers the meaning of the word and its connection with other words. For example, lemmatization considers the connection between the words good and well, despite them not being similar.

2.3 Feature extraction

Feature extraction is the phase in which pre-processed data is turned into feature vectors that can be used by machine learning algorithms to execute a task such as classification. These feature vectors are used as representations of the data and, as such, heavily influence the effectiveness of the algorithms. In the specific case of textual data, which is the focus of this work, these extraction methods essentially turn the data into a set of numerical features that can be used as input for machine learning algorithms, since these can't interpret raw text data.

Bag of words

The bag of words model (BOW) is a simple way to extract features from the text [11]. In this model the only things considered are the words that appear in the documents and the frequency with which they appear in each document, meaning that the order in which they appear is irrelevant to this model. This model can represent a document using a vector in which each element corresponds to the frequency of appearance of each word present in the established vocabulary. Table 2.1 exemplifies the information contained in a bag of words model.

Table 2.1 – Example of the information contained in a bag of words model.

Documents	*I*	*have*	*a*	*cat*	*do*	*not*	*like*	*dogs*
I have a cat	1	1	1	1	0	0	0	0
I do not have a cat	1	1	1	1	1	1	0	0
I do not like dogs	1	0	0	0	1	1	1	1

This way of feature extraction will not work well with words that are overrepresented in the corpus. Since those words appear very frequently, they will impact the model despite not containing significant information. Possible solutions to this problem are the removal of stop words discussed in a previous section and the use of TF-IDF weighting.

TF-IDF

Term Frequency – Inverse Document Frequency (TF-IDF) is a term weighing scheme for information retrieval purposes [12]. This scheme allows for the calculation of the weight of terms in each document through their term frequency and inverse document frequency. In this context, the document frequency of a term is the number of documents in which the term appears. In information retrieval, a term's document frequency (*df*) is an inverse measure of its informativeness because the occurrence of rare terms (that appear in fewer documents) contains more information than the appearance of very common terms among the documents. Thus, the inverse document frequency is a good measure of informativeness and can be calculated through the following formula when there are N documents:

$$\boldsymbol{idf = log_{10}\left(\frac{N}{df}\right)} \quad (2.1)$$

Term frequency is the number of times a term appears in a document. This helps ascertain the importance of a term since the more a term appears in a document, the more likely the document has information about the term and thus the more relevant it becomes.

TF-IDF uses both the term frequency and the inverse document frequency to gauge the importance of a certain term in each document and, although it appears calculated in different forms in different sources, a widely used formula is the following one [12]:

$$\boldsymbol{w = (tf) \times log_{10}\left(\frac{N}{df}\right)} \quad (2.2)$$

As was previously stated, this weighting scheme can be used in feature extraction to overcome the issue of overrepresented words of the bag of words model. Since this scheme takes into account the document frequency of the terms, the overrepresented terms will not heavily affect the model. Table 2.2 exemplifies the same example portrayed in Table 2.1 but using TF-IDF weighting.

Table 2.2 – Example of document representation using TF-IDF.

Documents	*I*	*have*	*a*	*cat*	*do*	*not*	*like*	*dogs*
I have a cat	0	0.1761	0.1761	0.1761	0	0	0	0
I do not have a cat	0	0.1761	0.1761	0.1761	0.1761	0.1761	0	0
I do not like dogs	0	0	0	0	0.1761	0.1761	0.4771	0.4771

Word2vec

Word2vec is a technique that allows for the vectorization of words, meaning that it generates vectorial representations for words. By learning the word embeddings (vectors) using their contexts, words with similar contexts are mapped into vectors close to one another in the vector space [13], thus enabling the use of a similarity function to calculate the similarity between the words through their representations. This technique can use two neural network-based learning algorithms to generate features: continuous bag of words and continuous skip-grams [14]. While the continuous bag of words algorithm learns word representations by predicting a certain word using the surrounding words, the continuous

skip-grams algorithm learns word representations by using a certain word to predict the surrounding words.

GloVe

GloVe (Global Vectors) [15] is a model for word representation that generates real-valued vector representations for words. GloVe builds upon word co-occurrence statistics and uses a weighted least squares regression model to generate word representations. These statistics are obtained using word-word co-occurrence counts that can be represented by a matrix. In this matrix, both the rows and the columns represent words while the entries represent the number of times that a certain word (row) appears in the context of another word (column). Through these counts, it is possible to calculate word co-occurrence probabilities that can be used to extract some meaning about the relationship between words.

Table 2.3 – Example of co-occurrence probabilities and ratio between probabilities [15].

Probability and Ratio	*k = solid*	*k = gas*	*k = water*	*k = fashion*
P(k|ice)	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
P(k|steam)	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
P(k|ice)/P(k|steam)	8.9	8.5×10^{-2}	1.36	0.96

Table 2.3 presents an example of co-occurrence probabilities and the ratio between probabilities. Table 2.3 shows that, for example, the co-occurrence of the words "ice" and "solid" is higher than the co-occurrence of the words "ice" and "fashion", since the first pair of words are more related to each other. In addition, the ratio between probabilities allows for further exploration of the relationships between words. Taking the example of the ratio present in Table 2.3, it is possible to observe that when the ratio is significantly higher than one, the word "ice" is more related to the word in the column (case of "solid"), when the ratio is significantly smaller than 1, the word "steam" is more related to the word in the column (case of "gas") and when the ratio is approximately one, both words of the ratio are equally related to the word in the column (case of "water" and "fashion").

Part-of-speech

A part of speech tagger (POS tagger) is a tool used in NLP. This tool allows for the tagging of words/tokens in segments of text that contain information relating to the structure of the

sentence. These tags can be, for example, noun, verb, adjective, adverb, etc. Figure 2.2 shows an example of POS tagging. This tool can be implemented in many ways.

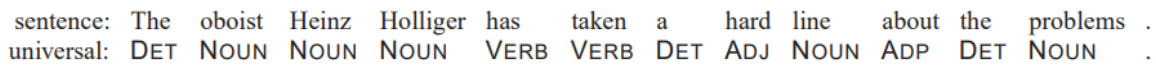

Figure 2.2 - Example of POS tagging [16].

There are supervised methods in which the tagger needs previously a tagged corpus to retrieve information regarding the use of the tags. Depending on the type of tagger this can be statistical information related to the tag sequences or a ruleset to aid the tagging process [17]. There are also unsupervised methods in which there is no need for a previously tagged corpus. These methods can create a set of rules, tag sets and other information automatically through an untagged corpus [17].

Regardless of whether the tagging method is supervised or unsupervised, they can also vary in their implementation. Some are stochastic, in the sense that they choose the tag for a given word through probabilities and frequencies calculated through a previously processed corpus, that can be tagged or untagged. Others are rule-based, in which they follow a set of rules to tag words. Finally, there are hybrid taggers that use both statistical methods and rules to tag [17].

BERT

BERT (Bidirectional Encoder Representations from Transformers) is a language representation model that is capable of achieving state-of-the-art performance in many NLP tasks such as classification, question answering, and language inference. Unlike other models, BERT is bidirectional, meaning that the representation of each word is influenced by both the preceding and the following words. The framework used in BERT has two phases, the pre-training phase, and the fine-tuning phase [18].

The pre-training phase is composed of two unsupervised tasks. The first task is a procedure called "Masked Language Model" (MLM) that starts by masking 15% of the tokens of each sequence randomly. This masking can have three possible outcomes. In 80% of the cases, the token is replaced by a [MASK] token, in 10% of the time the token is replaced with another random one and in the remaining 10%, it is left as is. Then the masked tokens are predicted to train the parameters of the model. The second task is "Next Sentence Prediction" (NSP) in which the model predicts if two sentences are consecutive or not. In this task, only 50% of the pairs of sentences are truly consecutive [18].

The fine-tuning phase can be tailored to many different NLP tasks, meaning that the process will be slightly different depending on the task the model is being used to solve.

For different tasks, there is a need to "plugin" task-specific inputs and outputs to tune the parameters [18].

As previously mentioned, the BERT model can be tailored to many different NLP tasks from distinct domains, meaning that the model is not specialized for a specific task. NLP tasks of a specific domain, particularly the healthcare domain, display unique characteristics not present in other subjects that can be exploited in order to improve the performance of models. Therefore, by specializing the model for certain NLP tasks it is possible to obtain better performance in those specific tasks. With this in mind, there has been research that aimed to create improved BERT-based models for NLP tasks in specific domains. Some of the most relevant ones for this dissertation are the following:

- RoBERTa, (Robustly optimized BERT approach) is a modified BERT model with an improved pre-training procedure [19]. The main differences are that in this model the masking patterns in the MLM procedure are dynamically generated, the NSP procedure was removed, and the model was pre-trained longer with more data using bigger batches.
- BERTweet is a modified BERT model with a pre-training procedure based on RoBERTa that is pre-trained with Twitter data [20].
- BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) is a modified BERT model pre-trained with biomedical data [21].

2.4 Dataset balancing

To train machine learning models, it is vital to have a large amount of data to feed to the models. In addition, when using supervised algorithms there is a need for labelled data. In a lot of NLP tasks, the data has to be manually labelled, which requires manpower and time. Furthermore, for some tasks, the resulting labelled datasets end up being highly imbalanced. In other words, one or more of the classes in the dataset is overrepresented while others are underrepresented, affecting the trained models negatively.

To mitigate the effect of class imbalance in the models it is possible to use some techniques on the training data. Two of these techniques are random data oversampling and undersampling [22]. In random data oversampling random examples of the minority class are duplicated and in random data undersampling, random examples of the majority class are deleted.

Another technique is the use of a weighted cross-entropy loss function [23]. This function is used when adjusting weights in some machine learning algorithms in the sense that the weights are adjusted to minimize the value of the function. By using a weighted version of the cross-entropy function the minority class instances in the dataset can influence the model to a greater extent. One more technique is increasing the gradient weight of the minority class [24]. This technique, just like the use of a weighted cross-

entropy loss function, affects the weight the minority class has on the learning process of the model. Additionally, there is a technique of paraphrasing the data to extend it by translating it into a different pivot language and then translating it back [24].

2.5 Machine learning algorithms

In this sub-section, some machine learning algorithms capable of solving tasks such as classification are presented. Machine learning algorithms can be supervised and unsupervised. The main difference between them is that supervised machine learning algorithms are trained using labelled data while unsupervised algorithms are not trained [25]. Supervised algorithms use labelled data to train a model that adapts itself to fit the training data, in other words, it changes its parameters to classify the training data correctly. On the other hand, unsupervised algorithms identify hidden patterns in unlabelled data and, thus, can organize information without the need for labelled data.

2.5.1 Naïve Bayes

Naïve Bayes is a supervised machine learning algorithm widely used in classification tasks. This algorithm has as its base the Bayes' rule, which states that given two events *A* and *B* [26]:

$$\boldsymbol{p(A|B) = \frac{p(B|A) \times p(A)}{p(B)}} \qquad (2.3)$$

In the classification context, the rule is applied as follows, given a document *E* characterized by a feature vector $(x_1, x_2, x_3, \dots, x_n)$ and a class *c*, the probability of that vector being of class c is [26]:

$$\boldsymbol{p(c|E) = \frac{p(E|c)p(c)}{p(E)} = \frac{p(c) \times \prod_{i=1}^{n} p(x_i|c)}{p(x_1, x_2, \dots, x_n)}} \qquad (2.4)$$

In the Naïve Bayes algorithm, it is assumed that all features are independent, and that assumption is what allows for the development of the presented formula. The class used to classify a vector is given by the class that maximizes the previous expression:

$$\hat{c} = \underset{c}{\operatorname{argmax}}\left(p(c) \times \prod_{i=1}^{n} p(x_i|c)\right) \quad (2.5)$$

Different implementations of the Naïve Bayes algorithm generally differ in the assumption of the distribution of the events. For example, in the case of Gaussian Naïve Bayes, the events are assumed to happen with a Gaussian distribution [27], in other words, the probability of a certain event is calculated using the following expression where μ is the mean and σ is the standard deviation of the distribution:

$$p(x_i|c) = \frac{1}{\sqrt{2\pi\sigma^2}} \times e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \quad (2.6)$$

2.5.2 Decision Trees

Decision tree classifiers are supervised machine learning classifiers. These allow for the partitioning of the data using if-else conditions applied to the objects' features [28]. These classifiers can be represented as a graph, for example, the one in Figure 2.3. The figure illustrates a decision tree that classifies objects between 2 classes, "Yes" and "No", using three conditions. The first condition is if the age of the entity is higher than 30, the second is if the value of the entity's gender attribute, and the third is the value of the attribute Last R.

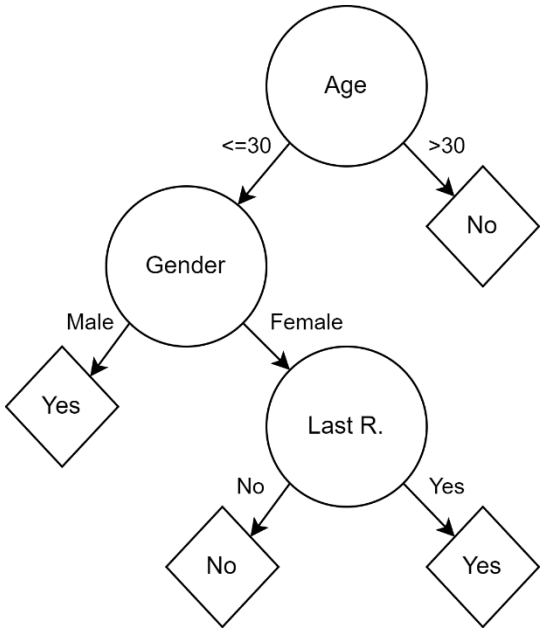

Figure 2.3 – Example of a representation of a decision tree [28].

The learning algorithm, at the training phase, chooses the best feature to use to split the objects in order to increase the homogeneity of the results in each node of the tree. These algorithms often use an impurity function to measure the homogeneity of each class at each node to choose the decisions that maximize the purity of each node.

Some often-used impurity functions are the entropy or the Gini index [28]. Both functions originate values between zero and one where zero represents high purity and one represents a high level of disorder. The formula of the entropy function is represented in formula 2.7 and the formula of the Gini index is represented in formula 2.8 where, in both cases, p_i is the probability of choosing an element from class I (relative frequency).

$$\boldsymbol{E = -\sum_{i=1}^{n} \log(p_i) \times p_i} \qquad (2.7)$$

$$\boldsymbol{GI = 1 - \sum_{i=1}^{n} p_i^2} \qquad (2.8)$$

2.5.3 Repeated Incremental Pruning to Produce Error Reductions

Repeated Incremental Pruning to Produce Error Reductions (RIPPER) is a rule-based learning algorithm that can be used on the task of text categorization. This algorithm is an improved version of the incremental reduced error pruning (IREP) algorithm [29].

In this algorithm, the dataset is first divided between a growing set and a pruning set. The growing set had 2/3 of the dataset and the pruning had the remaining subset. Then the growing set is used to generate rules to separate the data points into the different classes. The rule generations can be done using similar techniques to the ones previously mentioned in decision trees. Each rule is grown using the growing set, in the sense that conditions are continuously added to the rule until it stops covering negative examples. Then it is pruned using the following expression, where *PrPos* and *PrNeg* are, respectively, the positive and negative data points relating to a binary classification belonging to the pruning set and where *p* and *n* are, respectively, the number of examples in *PrPos* and *PrNeg* covered by the rule *Rule* [30].

$$\boldsymbol{v(Rule, PrPos, PrNeg) = \frac{p - n}{p + n}} \qquad (2.9)$$

The pruning of each rule is made to maximize the previous function, in other words, a final sequence of conditions of the rule is eliminated to maximize the previous function that is meant to represent the value/predictiveness of the rule. Each time a rule is added to the ruleset, the vectors of the growing set and of the pruning set that are covered by the added rule are eliminated from these sets. Additionally, there is a stopping condition for the addition of rules that takes into account the total description length of all the rules in the ruleset and of the examples computed as well as the positive examples not yet covered by the rules [30].

2.5.4 Support Vector Machine

Support vector machine (SVM) is a supervised machine learning algorithm commonly used for classification problems. This algorithm uses support vectors to form a hyperplane that separates the classes and allows for the classification of new data points. What differentiates this algorithm from the others is that the hyperplane obtained through it achieves maximum separation, in other words, the margin of the classifier is maximized. This is useful since even in the chance of an error in the hyperplane location it is less likely for a misclassification to occur [31]. A hyperplane that separates the data points between two classes is exemplified by the black line in Figure 2.4, where the margins of the classifier are represented in the orange and green lines.

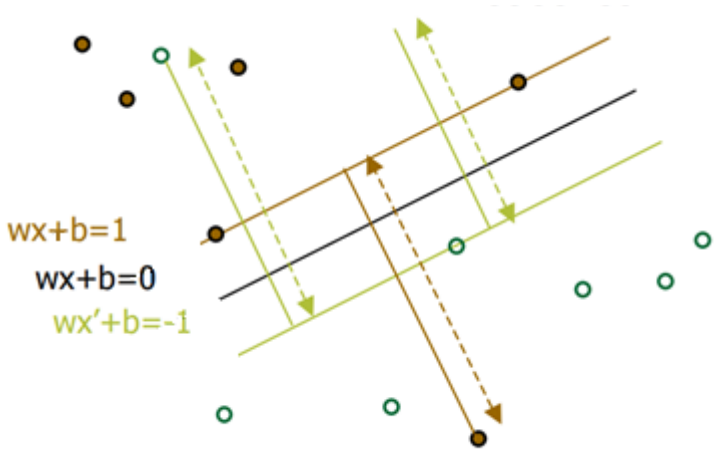

Figure 2.4 – Representation of hyperplanes [31].

2.5.5 Neural Networks

At the beginning of this chapter, I mentioned a subset of machine learning algorithms known as deep learning. I stated that these algorithms are neural networks with a large

number of layers. Neural networks are a family of machine learning algorithms that can learn in a supervised or unsupervised way. Neural networks were inspired by the structure of the human brain in the sense that the nodes exchange information in a similar way that neurons exchange signals [32]. These algorithms can be used for many tasks including classification, making them relevant to this work.

Neural networks are networks of units that are connected through their inputs and outputs where the outputs of some units are inputs of others. The basic unit of this algorithm works as a function that, given many inputs, computes a single output value. In each unit, a weighted sum of the inputs is computed in the manner shown in equation 2.10 where w_i are the weights of the inputs x_i and w_0 is the bias, which is an offset parameter.

$$a = \sum_{i=1}^{d} w_i x_i + w_0 \qquad (2.10)$$

After the weighted sum is computed an activation function is applied to that value. That activation function is a nonlinear function that determines the output of the unit. This function can vary between units, even in the same network depending on the problem that is being solved [33]. Figure 2.5 shows some commonly used activation functions.

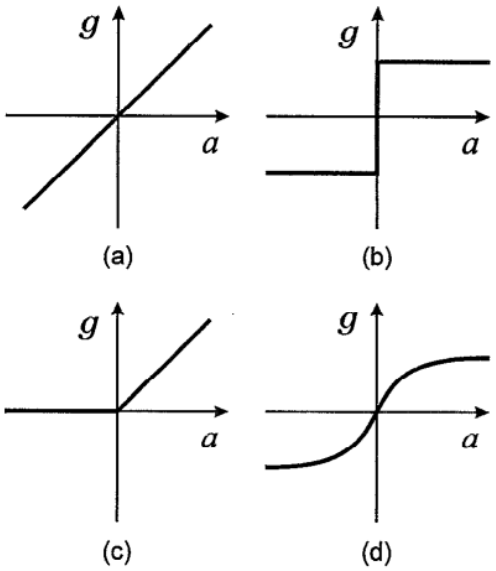

Figure 2.5 -A selection of activation functions: (a) linear, (b) threshold, (c) threshold linear, (d) sigmoidal [33].

In the case of supervised learning, in the training phase, the values of the weights are adjusted in order to minimize an error function. A commonly used error function is the

sum-of-squares error given by the expression 2.11 where n is the number of training set data points, x_q is an input, t_q is the desired output value and $y(x_q; w)$ is the value predicted.

$$\boldsymbol{E = \frac{1}{2}\sum_{q=1}^{n}\{y(x_q; w) - t_q\}^2} \qquad (2.11)$$

The minimization is usually done using the derivative of the error function relative to the weights of the network to obtain the gradient vector of the error function. Through the gradient vector, it is possible to minimize the error function through the method of gradient descent in which fixed value steps are taken towards the direction of the greatest decrease in error [33].

Recurrent neural networks

Recurrent neural networks (RNN) are a family of neural networks used in tasks where the data is sequential and each data point is correlated with the others, such as in temporal data or text data (due to the specific ordering of words) [34]. In these cases, the data points are generally not independent, meaning that there is a relation between them that can be used to improve the task at hand. It is important to note that different RNN architectures can capture different information that can be used for the prediction. Regular RNNs are generally only capable of capturing information from previous time states, but by adding a delay to the output it is possible to capture some information from future time states. Figure 2.6 shows the information captured by different RNN types on prediction. As the image states, a simple Time Delay Neural Network (TDNN) only captures information from a few states before the current one and a few states after. Furthermore, a forward RNN can capture information from all the preceding time states as well as some of the next states if the output is delayed. Lastly, the image also shows that a Bidirectional Recurrent Neural Network (BRNN) is capable of capturing the information of all preceding and future time states to use for prediction.

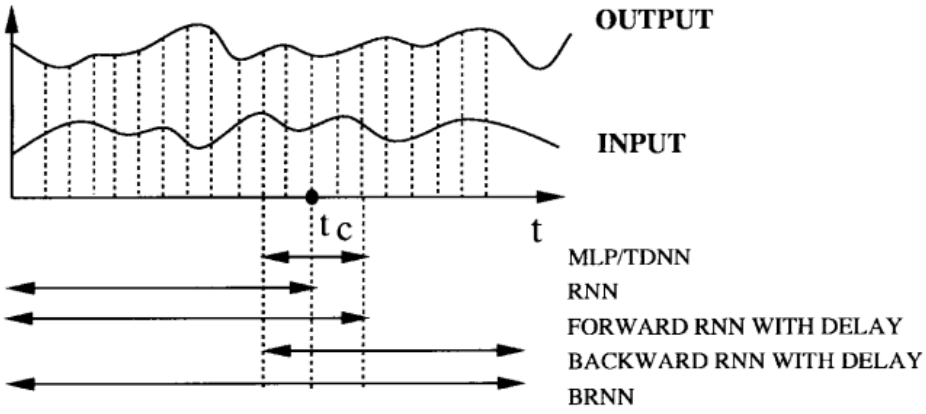

Figure 2.6 – Visualization of the amount of information captured by different network structures [34].

A specific architecture used in recurrent neural networks is long short-term memory (LSTM) [35]. This architecture aims to solve the vanishing gradient problem. This problem influences recurring neural networks in the sense that the gradient, as I mentioned previously, can be used in the training phase to adjust the weights of the network. In RNNs the gradient tends to vanish when backpropagated through the layers, hindering the training process. The problem is solved by adding gate units that help to avoid weight conflicts and memory cells that facilitate information storage.

Convolutional neural networks

Convolutional neural networks (CNN) are another type of neural network that is especially effective in image processing and other computer vision tasks. These networks are characterized by the use of convolution and, consequently, convolution layers.

Convolution networks are constituted by three types of layers: convolution layers, pooling layers, and fully connected layers. Convolutional layers are used as a means of feature extraction. For this purpose, they use an array of numbers called kernel to execute a convolution operation to the input array. The mentioned convolution operation consists of an element-wise product between the kernel and the input. In a single layer, this operation is executed many times moving the kernel through different elements of the input to capture features from the whole input [36]. This operation is exemplified in Figure 2.7.

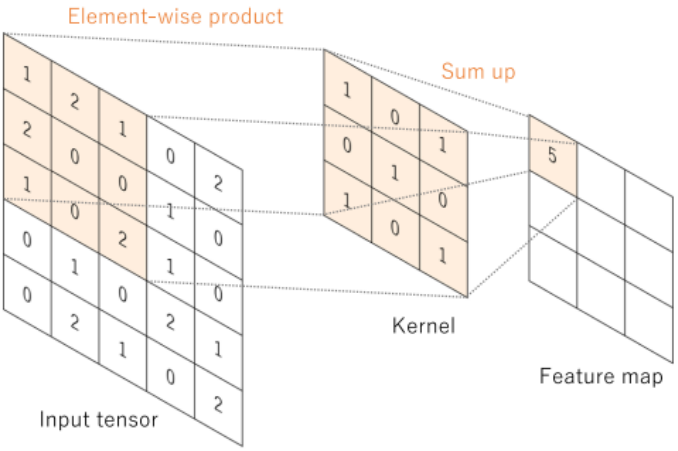

Figure 2.7 – Example of a convolution operation [36].

Pooling layers execute down-sampling operations leading to a reduction of dimensionality. In these layers, a filter is moved through the input applying an operation that leads to the aggregation of the input values. The two most common aggregation operations used are max pooling and average pooling. In the case of max-pooling, the output of the aggregation is the maximum value between the input values and in the case of average pooling, the output is an average of the input values [37].

The fully connected layers are the final layers of CNNs. In these, the output of the previous convolutional or pooling layer is transformed into a one-dimensional vector before being fed into one or more successive fully connected layers. The last fully-connected layer originates the final output of the network [36].

2.5.6 Ensemble Classifiers

Ensemble classifiers is a classification technique where many weak classifiers are used on the same object, and its final classification is decided using a function that aggregates all the classifications made by the weak classifiers. The generation of ensemble classifiers can be done in many ways, such as modifying the training parameters for each classifier, the feature set of each classifier and even by training different classifiers with different sets of the training dataset [38].

A specific method of ensemble classifiers is bagging or bootstrap aggregation. In this method, the different weak classifiers are trained samples of a training set obtained using the bootstrapping sample technique in which many documents of the main training set are chosen randomly with possible repetition. And after the training, the classification is done by majority voting on the results of the weak classifiers [39]. Due to the independence of the weak classifiers in this case their training and classifying processes can be parallelized.

A specific case of ensemble classifiers and the bagging method is called random forest. In this case, all the weak classifiers are decision trees trained in one of the ways previously mentioned, whether by training them in different sets of the training data set or by using different feature sets. The final classification is made using majority voting using the various trained decision trees [40].

Another method of using ensemble classifiers is boosting. In this method, the classifiers are trained in a way that each trained model tries to correct the mistakes made by the previously trained models.

A way to do this is by using modified training sets for different classifiers to make up for the misclassifications of the other models [23]. For example, the first classifier is trained with the original training set, the second classifier is trained with a modified training set where the misclassified examples from the first classifier were over-samples while the examples correctly classified by the first classifier were under-sampled.

2.6 Medical Dictionary

In NLP tasks in the health domain, whether it be classification or NER, medical dictionaries can be useful. As such, in this sub-section, a couple of medical language systems that have been used in NLP tasks are presented.

2.6.1 Unified Medical Language System

The Unified Medical Language System (UMLS) is a set of files and software that gathers biomedical vocabularies and standards. These resources can be accessed through different tools, such as the web browser, a local installation, or an Application Programming Interface (API) [41].

This system has been used in many different ways in the tasks of classification and extraction of adverse drug reactions and other health-related NER tasks. It has been used to tag tokens to aid in NER tasks and to filter and annotate datasets [42]. Furthermore, it has been used in the identification of medical semantic types and ids to aid in a classification task [43]. Finally, it has also been used in the construction of an adverse drug reaction (ADR) lexicon for the task of extraction of ADRs [13].

2.6.2 Medical Dictionary for Regulatory Activities

MedDRA (medical dictionary for regulatory activities) is a dictionary with standardised international medical terminology to use in registration, documentation, and safety monitoring of medicinal products through all the stages of the development cycle [44]. The existence of a single global medical dictionary comes with some advantages such as:

- Eliminates the need to perform a conversion between terminologies, preventing possible information loss or distortion originating from the conversion process.
- Improves the quality and timeliness of the available data.
- Increases the consistency of the terminology used throughout the development process of medicinal products.
- Eases the exchange of data relating to medicinal products.

It is important to note that MedDRA has been translated into 13 languages besides English. Furthermore, terms have a unique 8-digit code that is assigned to every translation of the same term. This has been used by Dima et al. [23] in a named entity resolution task in which the normalized concept of extracted ADEs was predicted.

2.7 Evaluation

In binary classification, a model, after being trained, must be evaluated in order to check its performance and compare it with other models. This is done with a test set, which is a labelled dataset that is used to check whether the algorithm is classifying the data correctly. After the model classifies the test set it is necessary to take into account the following numbers:

- Number of true positives (*TP*), which is the number of labels correctly classified as positive.
- Number of true negatives (*TN*), which is the number of labels correctly classified as negative.
- Number of false positives (*FP*), which is the number of labels incorrectly classified as positive.
- Number of false negatives (*FN*), which is the number of labels incorrectly classified as negative.

These numbers are used to calculate three performance metrics [22]. The first is precision, which is the ratio between the number of correctly classified positive labels and the number of samples classified as positive.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}} \qquad (2.12)$$

The second metric is recall, which is the ratio between the number of correctly classified positive labels and the number of positive samples in the training set.

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}} \quad (2.13)$$

Finally, the third metric is the $F1_{score}$ score, which uses the first two metrics in its formula.

$$\boldsymbol{F1_{score}} = \frac{\boldsymbol{2 \times Precision \times Recall}}{\boldsymbol{Precision + Recall}} \quad (2.14)$$

2.8 Relevant works

In this sub-section, I will present some approaches used in previous edition of the SMM4H Shared Task (2021) [45] to solve this year's task 1. This task has been a long running task in the previous editions of the challenge. For each subtask, I will start with an overview of the methods used and then highlight some specific approaches that achieved state-of-the-art performance.

Subtask 1a

In subtask 1a, the classification of tweets regarding the presence of an ADE mention, all the approaches used transformer-based models. The most prominent models in this subtask were BERT and RoBERTa since they were used in almost all the submissions. Beyond the transformer-based models, each team used other techniques to improve the effectiveness of the models, such as random undersampling and oversampling.

One of best-performing submissions in this subtask used the RoBERTa large with a binary classification head [22]. In addition, in order to overcome the class imbalance of the training dataset, the team used random oversampling with a strategy of 0.1, followed by a random undersampling with a strategy of 0.5. Another important aspect I would like to stress is that this approach did not remove mentions, hashtags, and URLs, since these were deemed beneficial for the performance of the model.

There was another system that displayed the same F1 score measure as the previous one, although with a higher performance and lower Recall. This approach used a RoBERTa large model in conjunction with a ChemBERTa model [46]. A ChemBERTa model is a model based on RoBERTa base that was trained with representations of chemical structures [47]. In this system, the training data was randomly oversampled, and the URLs and mentions were replaced with placeholders. This system started by finding the drug mention for each tweet, representing the chemical structure of the drug as a Simplified Molecular Input Line-entry System (SMILES) string, encoding it with the ChemBERTa model and taking the final [CLS] embedding. Then, for each tweet with a drug mention, the ChemBERTa embedding was concatenated to the text embeddings. Finally, these concatenated embeddings were

used as input for a fully-connected network with one hidden layer that was, in turn, used for classification.

Even though both of the previous systems displayed the best F1 score values, none of them displayed the best Recall value. The approach that exhibited the best Recall measure used an ensemble of BERT model variants [24]. Furthermore, this system also used random oversampling to bridge the class imbalance problem observed in the training dataset. Additionally, data cleaning was also performed in the form of masking Twitter handles, URLs, emails, phone numbers and money, performing hashtag expansion and codifying emojis.

Subtask 1b

In subtask 1b, the detection of ADE mention spans, all the submitted approaches used transformer-based models. The models used in the best-performing submissions on this subtask were BioBERT and RoBERTa. As with the first subtask, these approaches used techniques to improve the effectiveness of the transformer-based models.

The best-performing approach for this subtask used an ensemble of BioBERT models in Multi-Task Learning (MTL) architectures [23]. A Multi-Task Learning architecture can tackle all three subtasks (classification, extraction and normalization) simultaneously by using a shared model that is trained for all of the subtasks. The models in used ensemble were trained using the boosting learning method. More specifically, the ensemble used was constituted of five models in a MTL architecture. The first model was trained using the unaltered training data. The second one was trained with an altered training dataset that had undergone oversampling of misclassified examples from the first classifier and undersampling of the correctly classified examples. The third classifier was trained with an altered training data that had the examples classified differently by the first two classifiers oversampled and the other ones undersampled. The last two classifiers were trained with an augmented training set oversampled and undersampled according to the same approach used by the third model. Even though this approach encompassed all of the subtasks, it was only capable of achieving state-of-the-art performance in this subtask.

The second best-performing system for this subtask used a BiLSTM-CRF model [48] with many embeddings and features generated using different feature extraction methods [49]. This team's approach used contextual embeddings from BERT, Byte-Pair subword embeddings [50] and fastText sub-word embeddings [51].

Subtask 1c

In subtask 1c, the normalization of ADE mentions to MedDRA preferred term IDs, while most of the approaches included the use of a transformer-based model, one of the best-performing systems did not. This subtask also displayed a broader variety of models and

methods throughout the submissions, even though there were fewer submissions compared to the previous subtasks. Additionally, the best-performing approaches used a joint approach that tackles both subtasks 1b and 1c.

The best-performing approach for this subtask used a pipeline that joins subtasks 1b and 1c. The first step of this pipeline was the NER of ADE mentions in the tweets [46]. This step was achieved using EnRuDR-BERT [52] with certain dictionary-based features and training the model with the following two additional datasets: CSIRO Adverse Drug Event Corpus (CADEC) [53] and COMETA corpus [54]. The second step consisted in the normalization of the detected ADE mention spans. This step used two distinct models. The first model was a classifier that used BERT-generated features and a softmax layer. The second one was a neural model that used similarity distances between BERT representation vectors of the ADE mention detected and the ADE preferred term. These models are combined based on a distance threshold given by the second model. In other words, depending on the distance value displayed on the prediction of the second model, either the prediction of the second model is accepted, or the prediction of the first model is. Although this team used a joint approach on subtasks 1b and 1c, this system only displayed state-of-the-art performance in subtask 1c.

One of the second best-performing approach for this task used a Neural Transition-based Joint Model with a heterogeneous feature set [55]. The Neural Transition-based Joint Model [48], [56] was used in a NER context and extended by adding a linking action for the normalization context. The feature set used included character-level encodings generated through a CNN, non-contextual word representations produced using GloVe [15] and contextual word representations using ELMo [57]. This submission also performed some pre-processing steps in the tweets, namely lowercasing the tokens, replacing escape characters, and replacing user mentions and hashtags with single-word placeholders. Some pre-processing steps were also used in the MedDRA preferred terms. These were the replacement of numerical words to their numbers, remotion of punctuations, tokenization of the mentions and the lowercasing of the tokens. The best-performing submission from this team used a voting result obtained through the five best-performing model results.

There was another system that displayed the same F1 score as the previous model. This system used a joint approach for subtasks 1b and 1c [58]. The training data was reformulated using a labelling scheme that joined NER labels and the MedDRA preferred term tags. This system used BERT vector embeddings, POS tags and character embeddings originated through a single layer LSTM. In addition, a Bi-LSTM layer was used so as to incorporate all the embeddings and model the interactions between them. Furthermore, the CADEC dataset was added to the training data.

2.9 Summary

This chapter presented a background of the methods and algorithms used for NLP tasks. I started by introducing some general NLP concepts and the stages for a pipeline that can be used to solve this kind of problems. Then I proceeded to presenting methods used in each of the stages of the presented NER pipeline. I started with pre-processing techniques such as tokenization and stemming. After that, I presented feature extraction methods as well as dataset balancing techniques. Next, I presented machine learning algorithms, some medical dictionaries, and the most used evaluation measures in NLP tasks. I finalized this chapter by showing the best-performing approaches for the classification, NER and normalization subtasks tackled in this work that appeared in the SMM4H Shared Task 2021 edition. By analysing the best-performing approaches, it was possible to discern the most effective techniques and algorithms for each subtask. For all the subtasks, transformer-based models represented a majority of the submissions and almost all of the state-of-the-art systems. For subtask 1a, random data sampling techniques proved to be useful to overcome the class imbalance of the training dataset. When it came to subtask 1b, the use of a MTL architecture proved to be beneficial. In addition, the use of heterogeneous embeddings with a BiLSTM-CRF model also showed good results. Finally, for subtask 1c, a more diverse set of approaches were used, however, a joint training approach with subtask 1b was seen in the three best-performing systems and the use of the CADEC dataset was also present in two of the best-performing submissions.

3 Methodology

In this chapter, I will present the approaches used to solve the previously proposed tasks of the SMM4H 2022 Shared Task [4]. I will go through the datasets used, pre-processing executed, and models trained for each of the tacked subtasks.

The used models consist mainly of transformer-based models because these were used in works from previous years of the shared task and obtained high F1 scores. Furthermore, since these models have hardware requirements (particularly GPU requirements) not met by my personal computer, I used Google Collaboratory for training and testing models. The use of Google Collaboratory introduced many restrictions related to resource use such as available GPU processing time and available RAM. These restrictions can be mainly felt in the hyperparameter tunning phase since they restrict the possible algorithms and search space that can be used due to GPU use time constraints.

3.1 Social Media Mining for Health Shared Task

SMM4H Shared Task is a challenge that includes a plethora of NLP tasks related to the use of social media data in health research. As I have previously mentioned I participated in the 2022 edition of the SMM4H Shared Task [4], specifically task 1. This edition of the shared task has ten different tasks, where some of them are divided into subtasks. These tasks encompass a variety of health research areas, NLP challenges and languages. Task 1 is divided into three subtasks. The first subtask, 1a, is a binary classification task where tweets are classified regarding the presence of at least one ADE mention in the tweet. The second subtask, 1b, is a NER task with the objective of extracting ADE mention spans in tweets. The third subtask, 1c, is a normalization task that aims at matching ADE mentions to MedDRA preferred term IDs. This task presents a couple of challenges. One of them is that there is a class imbalance present in the available datasets that, when used to train machine learning models, negatively impacts the effectiveness of those models. The other challenge introduced in this task is the fact that, for subtask 1c, there are a very high number of labels. MedDRA makes available 23.000 preferred terms, making it difficult to classify ADE mentions to those terms, especially since only a small part of those terms is represented in the training dataset.

The organizers of the shared task provided 3 datasets for use in the task, the training set, validation set and test set [59]. The training set was meant to be used to train the models developed by the participants. The validation set was meant to test the approaches of the participants prior to the final submission and could be added to the training data for the final submission model. The test set was meant to evaluate the approaches of the final submission of each participating team.

The final submission for task 1 of the SMM4H Shared Task consisted of three tab-separated values (tsv) files, containing predictions for each one of the subtasks. These predictions consisted of the developed system predictions for the test dataset provided by the challenge organizers. In the case of subtask 1a, the file only had to contain the labels predicted by the system for each tweet. For subtask 1b, it also had to contain the spans that contain the ADE mention, which included the text span and the character indexes where the span started and ended. And for subtask 1c, the file had to include an additional column that contained the MedDRA preferred term ID relating to the ADE mention detected. My team submitted results for subtasks 1a and 1b, however, I explored all three subtasks in this dissertation.

3.2 Datasets

3.2.1 Challenge Dataset

As previously mentioned the organizers of the shared task provided 3 datasets for use in the training, validation, and test phases [59]. The training and validation datasets were labelled regarding each of the subtasks, meaning that they were annotated with the ADE/noADE label, the ADE mention spans (includes span start and end character indexes and span text), and the MedDRA dictionary standard concept IDs mapped to the mention.

Table 3.1 - Characteristics of the provided datasets.

Dataset	***ADE***	***noADE***	***Total #***
Training set	1239	16146	17385
Validation set	65	850	915
Test set	NA	NA	10984

Table 3.1 presents the characteristics of the datasets provided by the task organizers. This table allows us to clearly see the class imbalance issue present in the datasets. It is of note that some tweets have more than a single ADE mention, meaning that the number of mentions and MedDRA standard concept used to train models in subtasks 1b and 1c is higher than the number of positive tweets. The training set had 1711 ADE spans and the validation set had 87 ADE spans. As previously mentioned, for subtask 1c each mention was labelled with a MedDRA ID and only a very small subset of MedDRA IDs appeared in each dataset. In the training set, there were only 473 unique MedDRA IDs, in the validation set, there were 67 unique MedDRA IDs and in total there were 482 unique IDs.

The test set was meant to be used to submit the results of the model predictions to the shared task. This set is unlabelled and the only way to check the precision, recall and F1

measures of the predictions on the test set is through submitting the prediction file in the Codalab[2] platform since the labels of the test set are never disclosed to the public.

3.2.2 WEBRADR Benchmark Reference Dataset

Besides the datasets provided by the organizers of the SMM4H Shared task, the WEBRADR Benchmark Reference dataset was also used [60]. This dataset contains tweets labelled regarding the presence of ADE mentions. The dataset was constructed by first retrieving tweets containing at least one of the drugs of interest focused on in the article. Further processing was done to decrease the number of tweets retrieved, such as the removal of duplicates. The annotation process was performed independently and in parallel by two teams of trained annotators. The dataset contains further information besides the classification, namely:

- Tweet ID – Unique Twitter post ID.
- Classification – Tweet label ('AE tweet' or 'Non-AE tweet').
- Event(s) as reported – ADE span.
- Event(s) coded (PT) – MedDRA preferred term relating to the ADE mention.

This dataset has 57473 tweets, with 1056 positive and 56417 negative examples. Since the dataset provided only tweet IDs without the text, to use this dataset, each tweet's text had to be retrieved using the Twitter API. To get all the tweets of the dataset, a script that retrieves each tweet's text using the API was written. The endpoint for retrieving tweets allows each call to request the text of one hundred tweets and, because of that, the script retrieves tweets in batches of one hundred. Beyond the retrieval of the tweets, the script also processed them by removing newline and tab characters and replacing user mentions and URLs with the placeholders "@USER___" and "HTTPURL___". Even though the Twitter API allowed for the retrieval of the tweets, a lot of these were no longer available to the public. Therefore, only 31.122 tweets, with 588 being positive examples were retrieved. These positive examples contained 730 ADE mentions with 474 unique MedDRA preferred terms. Five of these unique terms were not found in an updated version of the MedDRA, possibly due to a version difference between the current set of MedDRA preferred terms and the one used when building the dataset. Even though this dataset is not intended for training purposes, it was used in subtasks 1b and 1c to increase the number of spans used to train the models.

[2] https://codalab.org/

3.3 Dataset transformations

One of the main challenges presented by these subtasks was the dataset imbalance and the lack of positive (ADE) examples. To bridge the dataset imbalance issue, dataset balancing techniques like the ones mentioned in the previous chapter were used. The used balancing techniques were:

- Random data oversampling.
- Random data undersampling.
- Data augmentation.

These data balancing techniques were experimented with, to find out which of these methods is more effective. Both of the random data sampling techniques were implemented using the Python sklearn[3] library. Random data oversampling was used with a sampling strategy of 0.3 and random data undersampling was used with a strategy of 0.1. To employ data augmentation, the TextAttack Python library [61] was used to produce examples through different text transformations. For each tweet in the minority class, five different tweets were generated through the following transformations:

- Replacing characters with random characters.
- Swapping characters with QWERTY adjacent characters.
- Replacing words with synonyms provided through Wordnet [62], [63].
- Performing contractions on recognized combinations.

The generated tweets had at least half of their words modified using one of the mentioned transformations. It is of note that two constraints were used to limit the transformations made to the tweets, namely, words could not be modified more than once and stopwords would not be modified. Table 3.2 presents an example of a tweet present in the training set and 5 new tweets generated through the data augmentations transformations. Table 3.3 displays the characteristics of the test dataset after it has undergone the previously mentioned dataset transformations.

[3] https://scikit-learn.org/stable/

Table 3.2 - Examples of tweets generated through data augmentation transformations.

Original tweet	@USER____ if #avelox has hurt your liver, avoid tylenol always, as it further damages liver, eat grapefruit unless taking cardiac drugs
Generated tweet #1	@USER____ if #Rvelox has pain your lVver, obviate Panadol always, as it further harm liver, consume grapeeruit unlAss carry cardiac drugs
Generated tweet #2	@USER____ if #avelop has offend your liver, annul Datril aleays, as it further wrong liver-colored, rust vrapefruit unless carry cardiac drugs"
Generated tweet #3	@USER____ if #avelox has scathe your liver, forfend Datril aHways, as it further indemnity liver-colored, gat grapefWuit unliss deal cardiac drugs
Generated tweet #4	@USER____ if #avelox has suffering your liver-colored, forefend Panadol invariably, as it further amends lxver, ewt gralefruit unless fetching cardiac drugs
Generated tweet #5	@USER____ if #avelox has wound your liver-colored, deflect tylenol constantly, as it further redress liver-colored, corrode gqapefruit unless winning cardiac drugx

Table 3.3 - Training dataset characteristics after applying re-sampling and data augmentation techniques.

Dataset	***ADE***	***noADE***	***#***
Original Training set	1239	16146	17385
Random Oversampling	4843	16146	20989
Random Undersampling	1239	12390	13629
Dataset Augmentation	6154	16146	22300

3.4 Pre-processing

A pre-processing step was done to prepare the raw text for the models used for classification. This step varied slightly between the different subtasks. For all subtasks the special instance character '&amp.' was replaced with the '&' character and for subtask 1b the tweets were lowercased.

Beyond the mentioned pre-processing steps, the tweets were also tokenized. Transformer-based models to solve each of the subtasks and for different transformer-based models different tokenization processes are needed. Therefore, different tokenizers are used for each model. In the case of subtask 1b, the input for the model tokenizer is a list of text tokens. This means that in this case, a prior tokenization had to be executed. This additional tokenization was performed using the TweetTokenizer from the nltk[4] Python library.

In subtask 1c, the main problem is the high number of possible labels when classifying the mentions. In an attempt to overcome this problem, the labels were limited to the ones that appear in the training set. Meaning that, when using a training set with a certain number of labels, the model will predict using only the labels seen when training the model.

3.5 Transformer models

In the experiments carried out for each of the subtasks, models available on HuggingFace[5] were used. The most effective models for these tasks were identified through works relating to the approaches used on the same subtasks of the SMM4H Shared task in previous years. These models were used together with a classification or a NER head depending on the subtask and the heads varied from model to model. The models experimented with for solving these tasks were the following:

- BERT base uncased.
- BERT large uncased.
- RoBERTa base.
- RoBERTa large.
- BERTweet base.
- BERTWeet large.

The BERT uncased models [18] were the first transformer-based models used in the experiments and are versions of the BERT model which do not distinguish words with cased

[4] https://www.nltk.org/
[5] https://huggingface.co/models

or uncased letters (do not make a difference between the words 'English' and 'english'). These transformer models were pretrained on English text with the objectives of MLM and NSP. The model was trained on an English Wikipedia dataset and a book text dataset. This model has a base and a large version that vary on the number of encoder layers stacked on top of each other (12 in the case of base and 24 in the case of large). In this model, the classification and NER heads are comprised of a dropout layer and a linear layer that outputs a vector with the scores for each of the classes. Both heads take as input tokens obtained through the model, however, each of the heads takes different tokens. The classification head's input is the classification token and the input for the NER head is the whole sequence of tokens.

The RoBERTa models [19] are modified BERT models with an improved pre-training procedure. This model is also pretrained with the following additional datasets: CC-News [64] (English portion of the CommonCrawl news dataset), OpenWebText [65] (open-source recreation of web text), and the Stories dataset [66] (subset of CommonCrawl data). Contrary to the BERT uncased models, this model is case sensitive, meaning that it distinguishes words with differently cased letters (distinguishes the words 'English' and 'english'). Like the previously presented models, this model also has base and large versions. In this model, the classification head consists of a sequence of five layers that takes as input the classification token obtained through the model. These layers are, in order, a dropout layer, a linear layer, a hyperbolic tangent layer, another dropout layer and, finally, a linear layer that outputs the scores for each of the classes. The NER head used in this model is equal to the one that was used in the BERT uncased model.

The BERTweet models [20] are models with the BERT architecture that use a pretraining procedure based on RoBERTa and are meant to be used in tweet NLP tasks. The data used for pretraining this model consists of two corpora of tweets. The first one is a general collection of pre-processed English tweets (Twitter Stream by Archive Team[6]) and the second one is a collection of COVID-19 related tweets that have undergone the same pre-processing as the first corpus. Like the previous models, this one also has base and large versions. Since this model is similar to the RoBERTa model, the classification and NER heads were similar to the ones used for that model.

3.6 Performance measures

In the executed experiments, to test the models used, many performance measures were calculated and used to compare the approaches. Since subtask 1a is a binary classification task, the measures are calculated as described in the second chapter in the "Evaluation" section. However, for subtasks 1b (NER task) and 1c (multiclass classification task), the

[6] https://archive.org/details/twitterstream

numbers used to calculate the measures had to be counted differently. In this section, I will present the ways the measures used to evaluate the submissions for these subtasks were calculated. For subtasks 1b and 1c, according to the way the performance measures were calculated in the result evaluation in the shared task, the following values were used to compare performance:

- Number of true positives (*TP*), which is the number of spans correctly recognized.
- Number of false positives (*FP*), which is the number of spans incorrectly recognized.
- Number of false negatives (*FN*), which is the number of spans unrecognized by the system.

Subtask 1b also distinguishes between relaxed or overlapping measures and strict measures. In relaxed measures, the span is correctly recognized in the condition that the predicted span overlaps at least partially with the correct span. In other words, if the predicted span contains at least part of the correct span, the prediction is considered a true positive. Conversely, in strict measures, a predicted span is only considered a true positive if the predicted span starts and ends exactly at the same character as the correct span.

The evaluation of a submission for subtask 1c also includes strict and relaxed measures. This happens because this subtask's predictions are meant to be applied to the spans detected in the previous subtask. The difference when calculating the measures relative to subtask 1b is that the correct recognition of an ADE span is dependent on the additional condition that the predicted class (MedDRA ID) is the correct one. It is of note that, on the intermediate experiments made for subtask 1c, I do not present separate strict and relaxed measures. The reason is that in these experiments I test solely the system for subtask 1c, with the supposition that the spans are correctly guessed, so that I can test the multiclass classification approaches independently of the NER module used for the preceding subtask.

3.7 Experiments

For each subtask, different experiments were executed so as to ascertain which of the models and approaches are more effective. This entails testing different models trained with different data and observing which ones display better performance metrics for each of the subtasks.

In an initial experimental stage, the different models enumerated in the previous subsection were trained with the challenge training data for each of the different subtasks. In this phase, each of the models is trained with the same hyperparameters and implementation and tested using the challenge validation set. In further experiments, the best-performing model for each task is selected and trained with different data. This data

includes data generated through dataset transformations mentioned in prior sections as well as examples from the WEBRADR Reference dataset.

As mentioned previously, for subtask 1a I evaluated the different transformer-based models on a classification task, meaning that I trained each of them on the challenge training set for a classification task. In this subtask's experiments the models were trained with the following hyperparameters:

- Number of training epochs: 3.
- Batch size: 32.
- Initial learning rate: 2e-5.
- Number of warmup steps: 0.
- Weight decay: 0.01.

It is of note that when training the model BERT large uncased the batch size parameter had to be reduced to 16 due to memory constraints. I performed further testing by training the best-performing model with re-sampled or augmented training data.

For subtask 1b, the different transformer-based models were evaluated in the same way as in the prior subtask and used similar hyperparameters, the main difference being that the models were trained for a NER task instead of a binary classification task. In this subtask, further experiments focused on evaluating the random sampling techniques previously described as well as the addition of WEBRADR Reference Dataset examples to the model training data.

Finally, for subtask 1c the models were evaluated on a multiclass classification task. As in the previous subtasks, the models were trained using the same hyperparameters. Further experimentation was performed in this subtask regarding whether to train the models with the whole tweet containing an ADE mention or with just the ADE mention span. Moreover, for this final challenge, I also evaluated the impact of using WEBRADR Reference dataset ADE spans as training data.

3.8 Hyperparameter search

For implementing hyperparameter search algorithms I used the Ray Tune library [67]. This library contains many hyperparameter tunning methods and algorithms such as BOHB (Bayesian Optimization and HyperBand)[68] and Population Based training [69]. Due to memory restrictions of the development environment, the hyperparameter search algorithm used was the Population Based training algorithm.

Even though this hyperparameter search algorithm was implemented, I was not capable of observing the results of the hyperparameter search, because of the GPU use time restrictions of the Google Colab development environment. A thorough process of

hyperparameter search takes significant time, therefore I was not able to complete the search process.

3.9 System Architecture

I mentioned at the start of this chapter that the final submission for task 1 of SMM4H Shared task was comprised of three tsv files that contained the predictions made by the developed models of the tweets in the test dataset. Even though my team only participated in two of the subtasks, in this section I will describe the architecture of the system I used to obtain the predictions for all three subtasks.

Figure 3.1 presents the architecture of the system used to obtain the aforementioned predictions. Firstly, three transformer models are trained, each for one specific subtask. It is of note that, for the final challenge submission, I added the validation set provided by the organizers to the training data used to train each of the models. This implies that this data was merged with the training set, and possible dataset transformations were performed to this merged set. Then the model trained for subtask 1a is used to generate the predictions for the related subtask. After generating these predictions, the model trained for subtask 1b is used to produce the predictions of the ADE mention spans of the tweets classified positively by the model used in subtask 1a, thus originating the subtask 1b predictions. Finally, the subtask 1b predictions are used in conjunction with the model trained for subtask 1c, to generate the predictions of the MedDRA preferred term IDs relating to the ADE spans identified in the previous subtask.

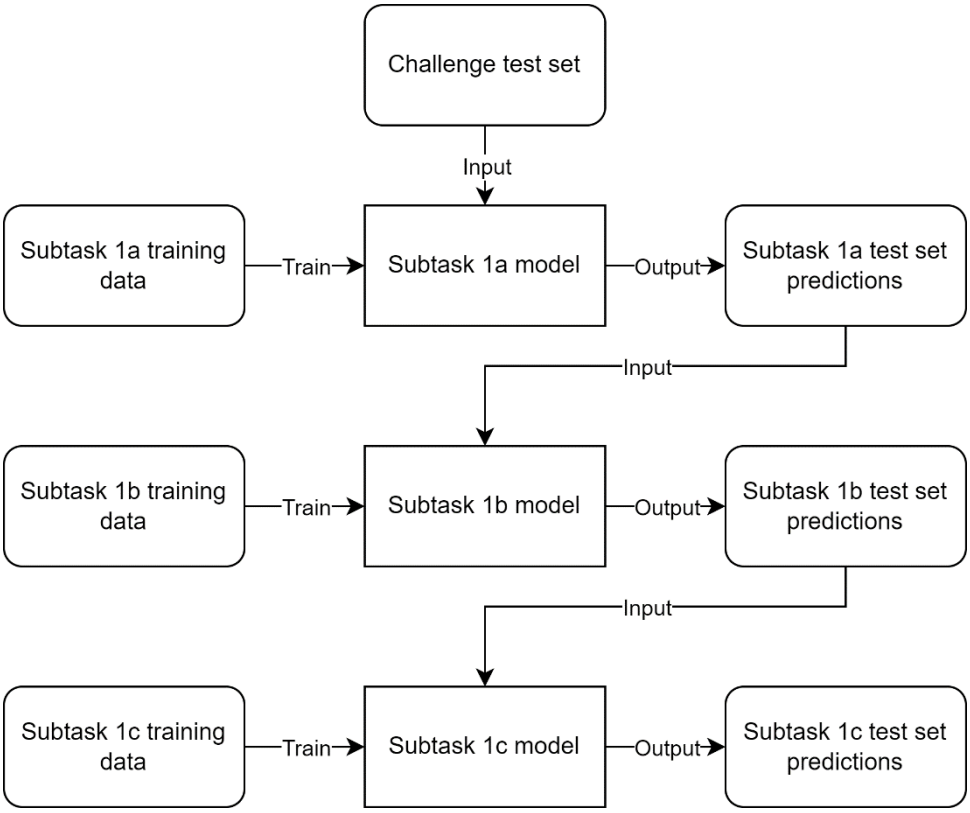

Figure 3.1 - System used to generate the predictions in a final submission of the SMM4H Shared task.

3.10 Summary

In this chapter, I presented the methods and approaches I used in this dissertation. I started by presenting the shared task in which I participated as well as analysing the shared task datasets provided by the organizers. I noted that one of the difficulties this task presents is the class unbalance present in the datasets, that ends up negatively influencing the training process of machine learning models. I also presented the characteristics of the WEBRADR Reference Dataset, that was also used in the experiments carried out. After that, I explained the implementation of the dataset balancing methods I employed. These were random oversampling, undersampling and data augmentation. Then I proceeded in presenting the pre-processing steps made in each subtask and the transformer-based models used in the experiments. Next, I explained the performance measures used in the evaluation stages for each subtask. I then proceeded to explain the experiments carried out to discover the best approaches for each of the subtasks. I ended this chapter by explaining the architecture of the system used to generate the predictions used in the submission for the SMM4H Shared Task.

4 Results

In this chapter, I will present the results obtained through the training and testing of models as described in the previous chapter. I will divide this chapter into sections, each one referring to the experiments and results of a subtask.

4.1 Subtask 1a

In this section, I will present the results pertaining to the experiments executed in relation to subtask 1a, the classification of tweets regarding the presence of an ADE mention. I will first present results regarding the different transformer models used in order to ascertain which one is most suited to this task. I will then present the results concerning the different random data sampling and data augmentation methods used to overcome the class imbalance problem present in the training dataset.

4.1.1 Model experiments

In the last chapter, I mentioned that I started by training the different transformer-based models for subtask 1a using the training data made available by the task organizers. This implies the pre-processing of the training data as formerly described. The models were tested using the challenge validation dataset.

Table 4.1 - Results for different transformer models for subtask 1a.

Model	*Precision*	*Recall*	*F1*
BERT base uncased	0.750	0.692	0.720
BERT large uncased	**0.797**	0.723	0.758
RoBERTa base	0.731	0.754	0.742
RoBERTa large	0.778	**0.862**	0.818
BERTweet base	0.754	0.662	0.705
BERTweet large	**0.797**	0.846	**0.821**

Table 4.1 shows the results of the experiments where transformer models were trained for the first subtask and tested using the challenge validation dataset. The results shown in the table indicate that the model BERTweet large presents the best performance for this task with an F1 value of 0.821. Even though the aforementioned model displayed the best F1 score when it comes to the other performance metrics, other models obtained equal or better values. For the precision metric, the model BERT large uncased presented

an equal value of 0.797. Furthermore, for the recall metric, the model RoBERTa large presents a higher value of 0.862.

The base models all present lower performance metrics than their large counterparts. In addition, the best-performing base model was the RoBERTa base model with an F1 value of 0.742 and recall value of 0.754, surpassing the recall value of the BERT large uncased model. Moreover, the model BERTweet base showed the best precision among base models at 0.750.

As observed previously, the large models are more suitable for solving the current subtask, since they present better performance values than their base counterparts. The large models use more resources, such as memory, in exchange for better performance, but since the evaluation of a given approach for this subtask does not account for resource usage, the large models have the upper hand over the base models.

4.1.2 Training dataset transformation experiments

Experiments were also executed regarding different methods of data sampling and data augmentation so as to find out the best method to overcome the best class imbalance problem. In these experiments, the best-performing transformer-based model for this task was trained using different training data. The previous section showed that the best-performing model for this subtask was BERTweet large, and thus, that is the model used in these experiments.

Table 4.2 - Results when training BERTweet large model classifier using different training dataset transformations.

Training set	*Precision*	*Recall*	*F1*
Base set	0.797	0.846	0.821
Text Augmentation	**0.909**	0.769	**0.833**
Oversampling	0.809	0.846	0.827
Undersampling	0.778	**0.862**	0.818

Table 4.2 shows the results of testing a BERTweet large model trained with different dataset transformations with the challenge validation set. The best performance was obtained when training the model with the training dataset augmented as described in the previous chapter. This approach yielded an F1 value of 0.833 and had the best precision metric among all of the methods with a precision of 0.909. The dataset transformation which showed the best recall was the random undersampling transformation with a recall value of 0.862. It is of note that, even though text augmentation showed the best overall performance, using random oversampling also showed good results having an F1 score

slightly lower than the text augmentation method with a value of 0.827. Furthermore, this approach had a better balance between precision and recall, displaying a lower precision and a higher recall than the data augmentation alternative.

4.2 Subtask 1b

In this section, I will present the results regarding the experiments performed in relation to subtask 1b, the detection of ADE mention spans on tweets. I will first the present results of the different transformer models used to find out which one is most suited to this task. I will then present results pertaining to the use of random sampling techniques on the training data and to the addition of WEBRADR Reference Dataset samples to the training data.

4.2.1 Model experiments

I started by training the different transformer-based models for subtask 1b using the training data made available by the task organizers. This implies the pre-processing of the training data as formerly described. In this subtask, the model BERTweet base was not used, because this model's tokenizer did not implement the function 'word_ids' that is responsible for mapping the tokens to their corresponding words and is necessary for pre-processing the input text for this subtask. The models were tested using the challenge validation dataset.

In this subtask, there are relaxed and strict metrics. Since the strict metrics are more representative of the effectiveness of the span detection capabilities of the model, those metrics will be more important when deciding which model is better.

Table 4.3 - Results for different transformer models for subtask 1b tested using the challenge validation dataset.

	Relaxed			*Strict*		
Model	Precision	Recall	F1	Precision	Recall	F1
BERT base uncased	0.868	0.673	0.789	0.395	0.345	0.368
BERT large uncased	**0.924**	0.629	0.748	0.379	0.287	0.327
RoBERTa base	0.875	0.769	0.819	0.612	0.563	0.587
RoBERTa large	0.892	**0.841**	**0.865**	**0.639**	**0.609**	**0.624**
BERTweet large	0.885	0.837	0.860	0.598	0.598	0.598

Table 4.3 presents the results when testing different models trained for subtask 1b. In this task, the best performing model was RoBERTa large with a strict F1 score of 0.624,

having outperformed every other model on almost all metrics. The only time this model was outperformed was by BERT large uncased with a relaxed precision metric of 0.924. Contrary to the previous subtask, this time, the base models performed a lot better when compared to their large counterparts. This is especially noticeable when BERT base uncased outperformed BERT large uncased in practically all metrics.

In both the relaxed and strict measures, the relative performance of the models is the same, in the sense that the best and worst models are the same (when evaluating them with F1 score). It is of note that, even though BERTweet was pre-trained using Twitter data, the RoBERTa large model still managed to exceed BERTweet's performance.

4.2.2 Training dataset transformations

In this subtask, different experiments were made by altering the training data used in the best-performing model from the previous experiments to increase its effectiveness. In the previous subsection, I observed that the best-performing model for this subtask was RoBERTa large. The following experiments train this model using the training dataset with the following transformations:

- Adding WEBRADR Reference Dataset positive example to the training data.
- Using random oversampling.
- Using random undersampling.
- Using random oversampling to the training dataset after the addition of WEBRADR Reference Dataset positive examples.

Table 4.4 - Results when training a RoBERTa-large model NER pipeline with different data.

	Relaxed			*Strict*		
Training data	Precision	Recall	F1	Precision	Recall	F1
Base set	0.892	0.841	0.865	0.639	0.609	0.624
Base set with reference dataset	0.887	0.798	0.840	0.650	0.598	0.623
Oversampled base set	**0.905**	**0.854**	**0.879**	**0.667**	**0.644**	**0.655**
Underampled base set	0.894	**0.854**	0.874	0.647	0.632	0.640
Oversampled base set w/ reference dataset	0.890	0.820	0.854	0.634	0.598	0.615

Table 4.4 displays the results originated from training a RoBERTa large model with different training data transformations using different training data. This table shows that the best performance is obtained when using an oversampled version of the base training dataset and this dataset transformation displayed an F1 score of 0.655. This training data transformation outperformed all the other ones in almost all performance measures. The use of random undersampling led to better performance measures relative to the base training dataset, however, it is still outperformed by the oversampled training data.

The use of the WEBRADR Reference Dataset negatively influenced the relaxed performance measures, displaying a decrease in all relaxed measures. However, when it comes to strict performance measures, the addition of reference dataset positive examples shows a negligible difference in performance (F1 score), displaying a higher precision and lower recall measures. Furthermore, using positive reference dataset examples together with random oversampling showed a decrease in all performance measures. The negative effects of the use of this dataset as training data can have a couple of explanations. The first one is that the methods used to retrieve and build this dataset were different from the ones used for the challenge datasets. The second one is that the data collection for this dataset was limited to six drugs of interest, possibly resulting in data with little variety and relevance for identifying ADE mentions related to other drugs.

4.3 Subtask 1c

In this section, I will describe results relating to the experiments performed in relation to subtask 1c, the mapping of ADE mentions to a standard concept ID in the MedDRA vocabulary. I will first present the results of the different transformer models used to find out which one is most suited to this task. Furthermore, I will compare results when encoding the full tweet or just the ADE mention when mapping the mention to the MedDRA ID. Finally, I will also investigate the impact of using the WEBRADR Reference dataset as training data, by testing the best-performing model from the previous experiments using reference dataset examples as training data.

4.3.1 Experiments using mention encoding

In this subtask, I started by training the different transformer models with the ADE mentions of the training dataset provided by the challenge organizers. These models were trained solely with the mention span without the rest of the tweet in which this mention was integrated.

Table 4.5- Results for different transformer models for subtask 1c when encoding the ADE mention.

Model	*Precision*	*Recall*	*F1*
BERT base uncased	0.138	0.138	0.138
BERT large uncased	0.138	0.138	0.138
RoBERTa base	**0.172**	**0.172**	**0.172**
RoBERTa large	0.161	0.161	0.161
BERTweet base	0.092	0.092	0.092
BERTweet large	0.103	0.103	0.103

Table 4.5 shows the results originated by training different transformer models on ADE mentions for subtask 1c. The table shows that the best-performing model was RoBERTa base with an F1 value of 0.172.

In this experiment, some base models either outperform or display a similar performance to their large counterparts. More specifically, BERT base uncased and BERT large uncased both displayed the same performance measures and RoBERTa base outperformed RoBERTa large. In addition, in this experiment, the BERTweet models displayed the worst performance. This is possibly due to the fact that these models were pre-trained with whole tweets and, in this experiment, the models were fine-tuned/trained with ADE mentions, without context of the tweet to which the mentions belong to.

4.3.2 Experiments using tweet encoding

In this experiment, I trained the different transformer models with the tweets that included the ADE mentions of the training dataset provided by the challenge organizers. This time these models were trained with the whole tweet that included the ADE span.

Table 4.6- Results for different transformer models for subtask 1c when encoding the whole ADE tweet.

Model	*Precision*	*Recall*	*F1*
BERT base uncased	0.092	0.092	0.092
BERT large uncased	0.069	0.069	0.069
RoBERTa base	**0.126**	**0.126**	**0.126**
RoBERTa large	**0.126**	**0.126**	**0.126**
BERTweet base	0.115	0.115	0.115
BERTweet large	0.103	0.103	0.103

Table 4.6 exhibits the results of the experiments where the transformer models were trained with whole tweets that contain the ADE mention. In this case, both types of RoBERTa models displayed the highest performance measures, revealing an F1 score of 0.126. As in the previous experiment, the base models had the same or better performance measures than their large counterparts.

Comparatively to the last experiment, the performance of the BERT uncased models and the RoBERTa models both decreased. Furthermore, the performance of the BERTweet models either remained unaltered (large model) or increased (base model). The fact that the performance of the BERT uncased and RoBERTa models decreased could be because, by mapping the mention using the whole tweet, different mentions that appear in the same tweet would have the same input in the pipeline and thus would have the same MedDRA ID attributed to them, leading to incorrect mappings in cases where there are different ADE mentions in the same tweet. The only case where the performance was increased relative to the last experiment was in the BERTweet models and that could be explained by the fact that, since these models were pre-trained using Twitter datasets, they are more suited to tasks where the input is a tweet, which is the case in this experiment.

By examining Table 4.5 and Table 4.6 I can see that training the models with the ADE mention instead of the whole tweet leads to better overall results. Furthermore, these two experiments show that the best approach is to train the RoBERTa base model using the ADE mention spans.

4.3.3 WEBRADR Reference Dataset

In this experiment, I trained the best performing model from the previous experiments, RoBERTa base, using solely the ADE spans of the challenge training data together with spans from the WEBRADR Reference Dataset. It is of note that introducing the WEBRADR dataset also introduced 139 new MedDRA IDs not present in the training and validation datasets.

Table 4.7- Results obtained when training RoBERTa base model for subtask 1c with different training data.

Training data	*Precision*	*Recall*	*F1*
Base set	0.172	0.172	0.172
Base set with reference dataset	**0.184**	**0.184**	**0.184**

Table 4.7 shows the results regarding the use of the WEBRADR Reference Dataset when training a RoBERTa base model for subtask 1c. It is possible to observe that the use of the reference dataset together with the training dataset when training the model improves the performance of the model.

4.4 Final approach

In this section, I will describe the results regarding two approaches. The first one is about my formal submission for the SMM4H Shared task, describing the main reasoning behind the submission as well as the results of that submission. As I have mentioned at the beginning of the chapter, this first approach only encompasses the first two subtasks. The second one relates to the best solution found after the deadline for the formal submission of the shared task had already passed. This approach accounts for all the experiments that have been described previously in this chapter and encompass all three subtasks. Both approaches generated results through systems with structures like the one described in Figure 3.1, with the exception that the first approach only included subtasks 1a and 1b.

The final results for the best approach were obtained by training the systems on the challenge test set. I reiterate that these results could only be obtained by submitting predictions to the Codalab[7] platform since the labels for the test dataset have not been made available to the challenge participants.

4.4.1 Challenge Submission

This sub-section describes the formal submission made to the shared task. This submission was accompanied with a description of the approach used to solve the task, thus, the following results are also available in a published paper [70]. Regarding the approach for subtask 1a, I trained the model BERTweet large for a binary classification task with the challenge training and validation datasets after they have undergone the data augmentation approach described in the previous chapter. The submission consisted of the predictions originated by this model through the classification of the test data.

I obtained the results for subtask 1b by training the model BERTweet large, the second-best performing model for subtask 1b, with the challenge training data together with the positive examples from the WEBRADR Reference Dataset.

Table 4.8 - Results when training a BERTweet-large model for subtask 1b with different data.

Training data	*Precision*	*Recall*	*F1*
Base training set	0.598	0.598	0.598
Base set with pos examples from WEBRADR dataset	**0.609**	**0.609**	**0.609**

[7] https://codalab.org/

To check the positive impact of using reference dataset positive tweets, I trained the BERTweet large model using the challenge training data with and without the reference dataset positive examples to check the impact of those positive examples in the performance of the model. These results are presented in Table 4.8 and they reveal that the inclusion of reference dataset positive examples in the training data led to a small increase in performance for the model.

Table 4.9- Results of the submission for subtask 1a.

	Precision	*Recall*	*F1*
Submission	**0.839**	**0.598**	**0.698**
Average	0.646	0.497	0.562

Table 4.10 - Results of the submission for subtask 1b.

	Relaxed			*Strict*		
	Precision	*Recall*	*F1*	*Precision*	*Recall*	*F1*
Submission	**0.828**	0.341	0.484	**0.560**	0.235	0.331
Average	0.539	**0.517**	**0.527**	0.344	**0.339**	**0.341**

Table 4.9 and Table 4.10 present the results obtained regarding the formal submissions for shared task for subtasks 1a and 1b respectively. For subtask 1a my approach presented higher performance measures than the average of all submissions. However, for subtask 1b, my model was only able to surpass the average performance in the precision measure (both in the relaxed and strict measure).

4.4.2 Best approach

This sub-section describes a proposed best approach accounting for the combination of models and training data that obtained the best performance measures in the previous sections of this chapter. For subtask 1a, the approach is the same as the one mentioned in the challenge submission sub-section, sharing the same performance measures described in the previous subsection.

Concerning subtask 1b, a different approach is used. This time the used model is the RoBERTa large model trained with an oversampled version of the training dataset combined with the validation dataset. Finally, for subtask 1c the RoBERTa base model was used. This model was trained solely with ADE spans from the challenge training dataset combined with the validation dataset and the WEBRADR Reference Dataset.

Table 4.11- Results of the best approach found for subtask 1b.

	Relaxed			*Strict*		
	Precision	*Recall*	*F1*	*Precision*	*Recall*	*F1*
Submission	**0.780**	**0.573**	**0.661**	**0.456**	**0.347**	**0.394**
Average	0.539	0.517	0.527	0.344	0.339	0.341

Table 4.12- Results of the best approach found for subtask 1c.

	Relaxed			*Strict*		
	Precision	*Recall*	*F1*	*Precision*	*Recall*	*F1*
Submission	0.106	0.081	0.092	**0.086**	0.065	0.074
Average	**0.120**	**0.112**	**0.116**	0.085	**0.082**	**0.083**

Table 4.11 and Table 4.12 present the results for subtasks 1b and 1c respectively obtained through the best approach found in the experiments described in this chapter. In this case, since subtask 1c is dependent on the results from subtask 1b, the measures presented for that subtask include the relaxed and strict metrics. The results for the best approach for subtask 1b both surpassed the average submission results as well as the formal submission I made for this task in all performance measures. As for subtask 1c, the results for the best approach were below the submission average. The only exception being the strict precision measure, which was slightly higher than the average.

4.5 Summary

In this chapter, I presented and discussed the results obtained through different experiments intended to pinpoint the best approach for solving the tackled shared task. The results were obtained mainly through testing different models on the challenge validation dataset. The code used to obtain the results shown in this chapter is publicly available on GitHub[8].

I started by testing the effectiveness of different transformer-based models on subtask 1a. The experiments revealed that the model that performed the best for the first subtask was BERTweet large. Further testing was performed in order to figure out the best way to overcome the class imbalance problem observed in the training dataset. Those tests revealed that using data augmentation helps to solve the problem of dataset imbalance, improving the performance of the BERTweet large model.

I then proceeded to experiment on different models to figure out the best approach for subtask 1b. This time the best-performing model was RoBERTa large. For this subtask, I

[8] https://github.com/edgargsm/SMM4H2022

experiment with random sampling techniques and investigated the effect of using positive examples from the WEBRADR Reference dataset when training the model. These experiments revealed that the random sampling techniques improved the performance of the RoBERTa large model, being random oversampling the most effective method. In addition, the effect of the positive examples of the WEBRADR dataset was found to be negligible.

For the final subtask, I experiment with different models, in the same way as with the first subtask. This time the best-performing model was RoBERTa base. I also compared the performance of different models when these were trained using only the ADE span or using the whole ADE tweet and found out that encoding just the ADE span leads to more favourable results. In this subtask, I also studied the effect of using examples from the WEBRADR Reference Dataset and found out that this dataset improves the performance of the model for this subtask.

Finally, I compared the performance of the best approaches found in this chapter with the average results of the approaches submitted by other teams for the SMM4H Shared Task. For subtasks 1a and 1b, the performance measures of my best approach were higher than the average submission results. However, the approach for subtask 1c was not able to surpass the average submission performance measures, leading me to conclude that the method used was not suitable for this subtask and, consequently, for classification tasks with a very large number of labels.

5 Conclusion

In this work, I explored a plethora of approaches towards solving task 1 from Social Media Mining for Health Applications Shared Task 2022. This task is divided into three subtasks. Subtask 1a is a binary classification task relating to the classification of tweets regarding the presence of an ADE mention. Subtask 1b is a NER task that aims to identify the ADE mention spans in tweets. The remaining subtask 1c is a multiclass classification task that classifies ADE mentions to a MedDRA dictionary preferred term ID. The main difficulties presented in this task were the class imbalance of the datasets and, in the case of the third subtask, the high number of possible classes, where most of them aren't present in the dataset.

I started this work by introducing various NLP techniques that encompass different stages of an NLP pipeline, such as data pre-processing, dataset augmentation and machine learning models. I also showed some state-of-the-art approaches that can be used to solve the proposed tasks. Following the background chapter, I described the methodology used to identify the best approaches for the shared task. I focused on using transformer-based models in all subtasks since these were considered to be state-of-the-art approaches in previous editions of the shared task.

The experiments in subtask 1a focused on figuring out the best model for the job and the most effective way to bridge the issue of class imbalance in the used datasets. As for subtask 1b, I tested different models, random sampling techniques and the impact of WEBRADR Reference Dataset positive examples on the performance of the used model. In subtask 1c, as in the previous subtask, I tested the different transformer-based models and the use of WEBRADR Reference Dataset as training data. In addition, in this last subtask, I also compared the performance of the models when they are trained with the entire tweet and when they are trained solely with the ADE mention span.

The best-performing approach for subtask 1a was the BERTweet large model trained with augmented training data. This allows me to infer that introducing variability in the data by adding new ADE tweets generated through data augmentation can be effective as a means to bridge the class imbalance issue. In addition, since the BERTweet large models are pre-trained using health-related tweets, it was expected that it would be the optimal choice for solving this subtask.

As for subtask 1b, the best performing model turned out to be RoBERTa large trained with randomly oversampled training data. Furthermore, the use of WEBRADR Reference Dataset positive examples did not improve the performance of the model and, when coupled with random oversampling worsened the effectiveness of the model. The fact that the RoBERTa large model outperformed the BERTweet large model in this subtask leads me to presume that the use of health-related tweets in the pre-training of the model is not

beneficial in this type of NER tasks. The use of positive examples from the reference dataset did not influence the performance heavily by itself possibly due to the low number of examples, however, by using random oversampling in conjunction with those examples I was able to see an amplified effect of the introduction of these examples. This effect was negative due to two possible reasons. One is that the examples of the reference dataset were retrieved in a different way than the challenge dataset. Secondly, in contrast to the challenge dataset, the data collection of that dataset focused only on six drugs of interest.

Finally, for the last subtask, the best performing model was a RoBERTa base model trained with ADE mention spans from both the training data and the WEBRADR Reference Dataset. This subtask's training data was a lot more limited since it was restricted to the ADE mention spans and this possibly contributed to the fact that a base model could outperform its large counterpart. The experiments in this subtask also revealed that the best overall performance was obtained by training the models on just the ADE mention span instead of the whole tweet. This could be because of the instances where a tweet has multiple mentions with different ADEs, which would be classified equally if the whole tweet was used to classify the mentions. This time the use of reference dataset data improved the performance of the model. There could be a few reasons for this. One is that, since there is little training data for this subtask, the impact of introducing new data points to the training process is more noticeable. Another reason might be that contrary to the previous subtask, by using just ADE mention spans for classification, the impact of the collection of the data in the reference dataset becomes irrelevant. Finally, the introduction of new examples also leads to the introduction of more labels present in the challenge training data spans as well as labels yet unknown by the model in the training phase.

5.1 Social Media Mining for Health Shared Task

My team's submission [70] results for the SMM4H Shared Task are available in this year's overview [4]. My submission for the first subtask managed to achieve state-of-the-art performance, presenting an F1 score of 0.698, surpassing all other submissions in this performance measure. The submission for the second subtask did not display high-performance measures. On the other hand, the final approach I presented for this subtask when compared with the submissions reported in the shared task overview, managed to surpass the other team's results on the test dataset, and achieve state-of-the-art results, with a relaxed F1 score of 0.661. My approach towards solving subtask 1c was not able to reach the average submission performance measures and, thus, not able to reach state-of-the-art performance, having been surpassed by approaches that do not use transformer-based models.

These results allow me to conclude that transformer-based models represent the current state-of-the-art approaches for subtasks 1a and 1b. Moreover, the use of data

augmentation and random data oversampling are effective methods for balancing training datasets. Finally, for subtask 1c, while my approach did not present state-of-the-art results, the best-performing approaches found in the shared task overview also used transformer models, albeit in a different way. This indicates that this subtask requires more intricate methods in order to obtain better results.

5.2 Future work

Through working on the proposed task of the SMM4H Shared Task, I was able to learn about many machine-learning models and techniques, with an emphasis on transformer-based models, since these were the most prominent and best-performing models presented in previous editions of the shared task. Through this process, I was able to explore a couple of libraries, particularly the scikit-learn and transformer libraries. Furthermore, during this process, I had to be wary of the resources I was using due to Google Colab GPU usage limitations.

For the task worked on in this dissertation, future work could encompass the study of the use of ensembles of transformer-based models for the referred task. Besides that, it could also explore other approaches that do not include transformer-based models, especially for subtask 1c, since some of these approaches obtained above average scores in the shared task.

Bibliography

[1] V. M. Prieto, S. Matos, M. Álvarez, F. Cacheda, and J. L. Oliveira, "Twitter: A good place to detect health conditions," *PLoS One*, vol. 9, no. 1, Jan. 2014, doi: 10.1371/JOURNAL.PONE.0086191.

[2] A. Yepes, A. MacKinlay, B. H.-P. of B. 15, and undefined 2015, "Investigating public health surveillance using Twitter," *aclweb.org*, pp. 164–170, 2015 [Online]. Available: https://www.aclweb.org/anthology/W15-3821.pdf. [Accessed: 26-Jan-2022]

[3] A. Jagannatha, F. Liu, W. Liu, H. Yu, and by Feifan Liu, "Overview of the First Natural Language Processing Challenge for Extracting Medication, Indication, and Adverse Drug Events from Electronic Health Record Notes (MADE 1.0) Part of a theme issue on 'NLP Challenge for Detecting Medication and Adverse Drug Events from Electronic Health Records (MADE 1.0)' guest edited," vol. 42, pp. 99–111, 2019, doi: 10.1007/s40264-018-0762-z. [Online]. Available: https://doi.org/10.1007/s40264-018-0762-z. [Accessed: 25-Oct-2022]

[4] D. Weissenbacher *et al.*, "Overview of the Seventh Social Media Mining for Health Applications #SMM4H Shared Tasks at COLING 2022," in *Proceedings of the Seventh Social Media Mining for Health (#SMM4H) Workshop and Shared Task*, 2022, pp. 221–241 [Online]. Available: https://aclanthology.org/2022.smm4h-1.54. [Accessed: 21-Oct-2022]

[5] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, "Natural language processing: An introduction," *Journal of the American Medical Informatics Association*, vol. 18, no. 5. Oxford Academic, pp. 544–551, 01-Sep-2011 [Online]. Available: https://academic.oup.com/jamia/article/18/5/544/829676. [Accessed: 26-Oct-2022]

[6] P. P. Shinde and S. Shah, "A Review of Machine Learning and Deep Learning Applications," in *Proceedings - 2018 4th International Conference on Computing, Communication Control and Automation, ICCUBEA 2018*, 2018, doi: 10.1109/ICCUBEA.2018.8697857.

[7] J. J. Webster and C. Kit, "Tokenization as the initial phase in NLP," 1992, p. 1106, doi: 10.3115/992424.992434.

[8] "Porter Stemming Algorithm." [Online]. Available: https://tartarus.org/martin/PorterStemmer/index.html. [Accessed: 26-Jan-2022]

[9] "Snowball." [Online]. Available: https://snowballstem.org/. [Accessed: 26-Jan-2022]

[10] J. Vilares, M. A. Alonso, and M. Vilares, "Morphological and syntactic processing for text retrieval," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3180, pp. 371–380, 2004, doi: 10.1007/978-3-540-30075-5_36.

[11] K. Juluru, H. H. Shih, K. N. K. Murthy, and P. Elnajjar, "Bag-of-words technique in natural language processing: A primer for radiologists," *Radiographics*, vol. 41, no. 5, pp. 1420–1426, Sep. 2021, doi: 10.1148/rg.2021210025. [Online]. Available: https://pubs.rsna.org/doi/10.1148/rg.2021210025. [Accessed: 30-Oct-2022]

[12] K. Chen, Z. Zhang, J. Long, and H. Zhang, "Turning from TF-IDF to TF-IGM for term weighting in text classification," *Expert Syst. Appl.*, vol. 66, pp. 1339–1351, Dec.

2016, doi: 10.1016/j.eswa.2016.09.009.
[13] A. Nikfarjam, A. Sarker, K. O'Connor, R. Ginn, and G. Gonzalez, "Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features," *J. Am. Med. Informatics Assoc.*, vol. 22, no. 3, pp. 671–681, Mar. 2015, doi: 10.1093/JAMIA/OCU041. [Online]. Available: https://academic.oup.com/jamia/article/22/3/671/776531. [Accessed: 26-Jan-2022]
[14] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," *Proc. 2015 IEEE 14th Int. Conf. Cogn. Informatics Cogn. Comput. ICCI*CC 2015*, pp. 136–140, Sep. 2015, doi: 10.1109/ICCI-CC.2015.7259377.
[15] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation" [Online]. Available: http://nlp. [Accessed: 26-Jan-2022]
[16] S. Petrov, D. Das, and R. McDonald, "A universal part-of-speech tagset," in *Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC 2012*, 2012, pp. 2089–2096, doi: 10.48550/arxiv.1104.2086 [Online]. Available: https://arxiv.org/abs/1104.2086v1. [Accessed: 02-Nov-2022]
[17] S. G. Kanakaraddi and S. S. Nandyal, "Survey on Parts of Speech Tagger Techniques," *Proc. 2018 Int. Conf. Curr. Trends Towar. Converging Technol. ICCTCT 2018*, Nov. 2018, doi: 10.1109/ICCTCT.2018.8550884.
[18] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019, vol. 1, pp. 4171–4186 [Online]. Available: https://arxiv.org/abs/1810.04805v2. [Accessed: 26-Jan-2022]
[19] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," Jul. 2019 [Online]. Available: https://arxiv.org/abs/1907.11692v1. [Accessed: 26-Jan-2022]
[20] D. Quoc Nguyen, T. Vu, A. Tuan Nguyen, and V. Research, "BERTweet: A pre-trained language model for English Tweets," pp. 9–14, Nov. 2020, doi: 10.18653/V1/2020.EMNLP-DEMOS.2. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.2. [Accessed: 26-Jan-2022]
[21] J. Lee *et al.*, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, Feb. 2020, doi: 10.1093/BIOINFORMATICS/BTZ682. [Online]. Available: https://academic.oup.com/bioinformatics/article/36/4/1234/5566506. [Accessed: 26-Jan-2022]
[22] S. Ramesh *et al.*, "BERT based Transformers lead the way in Extraction of Health Information from Social Media," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 33–38, doi: 10.18653/v1/2021.smm4h-1.5 [Online]. Available: https://aclanthology.org/2021.smm4h-1.5. [Accessed: 26-Jan-2022]
[23] G. A. Dima, D. C. Cercel, and M. Dascalu, "Transformer-based Multi-Task Learning for Adverse Effect Mention Analysis in Tweets," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 44–

51, doi: 10.18653/v1/2021.smm4h-1.7 [Online]. Available: https://aclanthology.org/2021.smm4h-1.7. [Accessed: 26-Jan-2022]

[24] A. F. Aji, H. A. Wibowo, M. N. Nityasya, R. E. Prasojo, and T. N. Fatyanosa, "BERT Goes Brrr: A Venture Towards the Lesser Error in Classifying Medical Self-Reporters on Twitter," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 58–64, doi: 10.18653/v1/2021.smm4h-1.9 [Online]. Available: https://aclanthology.org/2021.smm4h-1.9. [Accessed: 26-Jan-2022]

[25] R. Sathya, A. A.-I. J. of A. R. in, and undefined 2013, "Comparison of supervised and unsupervised learning algorithms for pattern classification," *Citeseer* [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.278.5274&rep=rep1&type=pdf#page=41. [Accessed: 26-Jan-2022]

[26] H. Zhang, "The optimality of naive Bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[27] "1.9. Naive Bayes — scikit-learn 1.0.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/naive_bayes.html. [Accessed: 26-Jan-2022]

[28] L. Rokach and O. Maimon, "Decision Trees," in *Data Mining and Knowledge Discovery Handbook*, Springer, Boston, MA, 2006, pp. 165–192 [Online]. Available: https://link.springer.com/chapter/10.1007/0-387-25465-X_9. [Accessed: 29-Oct-2022]

[29] M. Sasaki and K. Kita, "Rule-based text categorization using hierarchical categories," *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 3, pp. 2827–2830, 1998, doi: 10.1109/ICSMC.1998.725090.

[30] W. W. Cohen, "Fast Effective Rule Induction," *Mach. Learn. Proc. 1995*, pp. 115–123, Jan. 1995, doi: 10.1016/B978-1-55860-377-6.50023-2.

[31] V. Jakkula, "Tutorial on support vector machine (svm)," *Sch. EECS, Washingt. State Univ.*, vol. 37, 2006.

[32] "What are Neural Networks? | IBM." [Online]. Available: https://www.ibm.com/cloud/learn/neural-networks. [Accessed: 26-Jan-2022]

[33] C. M. Bishop, "Neural networks and their applications," *Rev. Sci. Instrum.*, vol. 65, no. 6, p. 1803, Jun. 1998, doi: 10.1063/1.1144830. [Online]. Available: https://aip.scitation.org/doi/abs/10.1063/1.1144830. [Accessed: 26-Jan-2022]

[34] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997, doi: 10.1109/78.650093.

[35] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/NECO.1997.9.8.1735.

[36] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/S13244-018-0639-9/FIGURES/15. [Online]. Available: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9. [Accessed: 26-Jan-2022]

[37] "What are Convolutional Neural Networks? | IBM." [Online]. Available: https://www.ibm.com/cloud/learn/convolutional-neural-networks. [Accessed: 26-Jan-2022]

[38] A. Rahman and S. Tasnim, "Ensemble Classifiers and Their Applications: A Review," *Int. J. Comput. Trends Technol.*, vol. 10, no. 1, pp. 31–35, Apr. 2014, doi: 10.14445/22312803/IJCTT-V10P107. [Online]. Available: http://arxiv.org/abs/1404.4088. [Accessed: 26-Jan-2022]

[39] S. jin Wang, A. Mathew, Y. Chen, L. feng Xi, L. Ma, and J. Lee, "Empirical analysis of support vector machine ensemble classifiers," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 6466–6476, Apr. 2009, doi: 10.1016/J.ESWA.2008.07.041.

[40] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, "On-line random forests," *2009 IEEE 12th Int. Conf. Comput. Vis. Work. ICCV Work. 2009*, pp. 1393–1400, 2009, doi: 10.1109/ICCVW.2009.5457447.

[41] O. Bodenreider, "The Unified Medical Language System (UMLS): integrating biomedical terminology," *Nucleic Acids Res.*, vol. 32, no. Database issue, p. D267, Jan. 2004, doi: 10.1093/NAR/GKH061. [Online]. Available: /pmc/articles/PMC308795/. [Accessed: 26-Jan-2022]

[42] E. Batbaatar and K. H. Ryu, "Ontology-Based Healthcare Named Entity Recognition from Twitter Messages Using a Recurrent Neural Network Approach," *Int. J. Environ. Res. Public Heal. 2019, Vol. 16, Page 3628*, vol. 16, no. 19, p. 3628, Sep. 2019, doi: 10.3390/IJERPH16193628. [Online]. Available: https://www.mdpi.com/1660-4601/16/19/3628/htm. [Accessed: 26-Jan-2022]

[43] A. Sarker and G. Gonzalez, "Portable automatic text classification for adverse drug reaction detection via multi-corpus training," *J. Biomed. Inform.*, vol. 53, pp. 196–207, Feb. 2015, doi: 10.1016/J.JBI.2014.11.002.

[44] "Vision for MedDRA | MedDRA." [Online]. Available: https://www.meddra.org/about-meddra/vision. [Accessed: 26-Jan-2022]

[45] A. Magge *et al.*, "Overview of the Sixth Social Media Mining for Health Applications (#SMM4H) Shared Tasks at NAACL 2021," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 21–32, doi: 10.18653/v1/2021.smm4h-1.4 [Online]. Available: https://aclanthology.org/2021.smm4h-1.4. [Accessed: 27-Oct-2022]

[46] A. Sakhovskiy, Z. Miftahutdinov, and E. Tutubalina, "KFU NLP Team at SMM4H 2021 Tasks: Cross-lingual and Cross-modal BERT-based Models for Adverse Drug Effects," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 39–43, doi: 10.18653/v1/2021.smm4h-1.6 [Online]. Available: https://aclanthology.org/2021.smm4h-1.6. [Accessed: 27-Oct-2022]

[47] S. Chithrananda, G. Grand, and B. Ramsundar, "ChemBERTa: Large-Scale Self-Supervised Pretraining for Molecular Property Prediction," 2020 [Online]. Available: http://arxiv.org/abs/2010.09885. [Accessed: 27-Oct-2022]

[48] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, 2016, pp. 260–270, doi: 10.18653/v1/n16-1030 [Online]. Available: https://aclanthology.org/N16-1030. [Accessed: 27-Oct-2022]

[49] U. Yaseen and S. Langer, "Neural Text Classification and Stacked Heterogeneous

Embeddings for Named Entity Recognition in SMM4H 2021," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 83–87, doi: 10.18653/v1/2021.smm4h-1.14 [Online]. Available: https://aclanthology.org/2021.smm4h-1.14. [Accessed: 27-Oct-2022]

[50] N. Calzolari *et al.*, "Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)," 2018 [Online]. Available: https://aclanthology.org/L18-1000. [Accessed: 27-Oct-2022]

[51] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Trans. Assoc. Comput. Linguist.*, vol. 5, no. 0, pp. 135–146, Jun. 2017, doi: 10.1162/tacl_a_00051. [Online]. Available: https://transacl.org/ojs/index.php/tacl/article/view/999. [Accessed: 27-Oct-2022]

[52] E. Tutubalina, I. Alimova, Z. Miftahutdinov, A. Sakhovskiy, V. Malykh, and S. Nikolenko, "The Russian Drug Reaction Corpus and Neural Models for Drug Reactions and Effectiveness Detection in User Reviews," doi: 10.1093/bioinformatics/xxxxxx. [Online]. Available: https://github.com/cimm-kzn/RuDReC. [Accessed: 27-Oct-2022]

[53] S. Karimi, A. Metke-Jimenez, M. Kemp, and C. Wang, "Cadec: A corpus of adverse drug event annotations," *J. Biomed. Inform.*, vol. 55, pp. 73–81, Jun. 2015, doi: 10.1016/j.jbi.2015.03.010. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25817970/. [Accessed: 27-Oct-2022]

[54] M. Basaldella, F. Liu, E. Shareghi, and N. Collier, "COMETA: A corpus for medical entity linking in the social media," in *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2020, pp. 3122–3137, doi: 10.18653/v1/2020.emnlp-main.253 [Online]. Available: https://aclanthology.org/2020.emnlp-main.253. [Accessed: 27-Oct-2022]

[55] Z. Ji, T. Xia, and M. Han, "PAII-NLP at SMM4H 2021: Joint Extraction and Normalization of Adverse Drug Effect Mentions in Tweets," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 126–127, doi: 10.18653/v1/2021.smm4h-1.26 [Online]. Available: https://aclanthology.org/2021.smm4h-1.26. [Accessed: 27-Oct-2022]

[56] B. Wang, W. Lu, Y. Wang, and H. Jin, "A neural transition-based model for nested mention recognition," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, 2018, pp. 1011–1017, doi: 10.18653/v1/d18-1124 [Online]. Available: https://aclanthology.org/D18-1124. [Accessed: 27-Oct-2022]

[57] M. E. Peters *et al.*, "Deep contextualized word representations," in *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2018, vol. 1, pp. 2227–2237, doi: 10.18653/v1/n18-1202 [Online]. Available: https://aclanthology.org/N18-1202. [Accessed: 27-Oct-2022]

[58] M. Elkaref and L. Hassan, "A Joint Training Approach to Tweet Classification and Adverse Effect Extraction and Normalization for SMM4H 2021," in *Social Media Mining for Health, SMM4H 2021 - Proceedings of the 6th Workshop and Shared Tasks*, 2021, pp. 91–94, doi: 10.18653/v1/2021.smm4h-1.16 [Online]. Available:

https://aclanthology.org/2021.smm4h-1.16. [Accessed: 27-Oct-2022]
[59] A. Magge *et al.*, "DeepADEMiner: a deep learning pharmacovigilance pipeline for extraction and normalization of adverse drug event mentions on Twitter," *J. Am. Med. Inform. Assoc.*, vol. 28, no. 10, pp. 2184–2192, Sep. 2021, doi: 10.1093/jamia/ocab114. [Online]. Available: https://academic.oup.com/jamia/article/28/10/2184/6322900. [Accessed: 31-Aug-2022]
[60] J. Dietrich *et al.*, "Adverse Events in Twitter-Development of a Benchmark Reference Dataset: Results from IMI WEB-RADR," *Drug Saf.*, vol. 43, no. 5, pp. 467–478, May 2020, doi: 10.1007/S40264-020-00912-9/TABLES/5. [Online]. Available: https://link.springer.com/article/10.1007/s40264-020-00912-9. [Accessed: 29-Jul-2022]
[61] J. X. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, "TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP," pp. 119–126, Apr. 2020, doi: 10.48550/arxiv.2005.05909. [Online]. Available: https://arxiv.org/abs/2005.05909v4. [Accessed: 29-Jul-2022]
[62] G. A. Miller, "WordNet: A Lexical Database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995, doi: 10.1145/219717.219748. [Online]. Available: https://dl.acm.org/doi/10.1145/219717.219748. [Accessed: 31-Jul-2022]
[63] C. Fellbaum, "WordNet: An Electronic Lexical Database," in *Theory and Applications of Ontology: Computer Applications*, Springer Netherlands, 2010, pp. 231–243.
[64] F. Hamborg, N. Meuschke, C. Breitinger, and B. Gipp, "News-Please: A Generic News Crawler and Extractor," in *Proceedings of the 15th International Symposium of Information Science*, 2017, pp. 218–223, doi: 10.18452/1447.
[65] A. Gokaslan and V. Cohen, "Download | OpenWebTextCorpus," *http://Skylion007.github.io/OpenWebTextCorpus*, 2019. [Online]. Available: https://skylion007.github.io/OpenWebTextCorpus/. [Accessed: 21-Oct-2022]
[66] T. H. Trinh and Q. V. Le, "A Simple Method for Commonsense Reasoning," 2018 [Online]. Available: https://github.com/tensorflow/models/tree/. [Accessed: 21-Oct-2022]
[67] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, "Tune: A Research Platform for Distributed Model Selection and Training," Jul. 2018, doi: 10.48550/arxiv.1807.05118. [Online]. Available: https://arxiv.org/abs/1807.05118v1. [Accessed: 21-Oct-2022]
[68] S. Falkner, A. Klein, and F. Hutter, "BOHB: Robust and Efficient Hyperparameter Optimization at Scale," in *35th International Conference on Machine Learning, ICML 2018*, 2018, vol. 4, pp. 2323–2341, doi: 10.48550/arxiv.1807.01774 [Online]. Available: https://arxiv.org/abs/1807.01774v1. [Accessed: 21-Oct-2022]
[69] M. Jaderberg *et al.*, "Population Based Training of Neural Networks," 2017 [Online]. Available: http://arxiv.org/abs/1711.09846. [Accessed: 21-Oct-2022]
[70] E. Morais, J. L. Oliveira, A. Trifan, and O. Fajarda, "BioInfo@UAVR@SMM4H'22: Classification and Extraction of Adverse Event mentions in Tweets using Transformer Models." pp. 65–67, 2022 [Online]. Available: https://aclanthology.org/2022.smm4h-1.19. [Accessed: 02-Nov-2022]