Weakly supervised Deep Learning for Natural Language Processing

by

Cristóbal Colón-Ruiz

A dissertation submitted by in partial fulfillment of the requirements for the degree of Doctor of Philosophy in

Computer Science and Technology

Universidad Carlos III de Madrid

Advisor:

Isabel Segura-Bedmar

June 2023

This thesis is distributed under license “Creative Commons **Atributtion - Non Commercial - Non Derivatives**”.

A mis padres, Cristóbal y María Rosa, por haberme dado la posibilidad de llegar hasta aquí, así como su apoyo y comprensión.

A mi hermano, Sebastián, por haberme ayudado a desconectar en los momentos que más lo necesitaba.

A mi pareja, Serezade, por su apoyo, comprensión y paciencia en todo momento, sobre todo en los momentos difíciles.

A mi tutora y directora, Isabel, por haber sido mi guía durante todo el proceso.

Por último, a mis compañeros, Rodrigo y Judith, por haberme aguantado tanto.

Published and submitted content

The author's research publications are partially or fully included in the methods proposed in the present document and are listed below.

1. **Journals:**

   - **C. Colón-Ruiz** and I. Segura-Bedmar. Comparing deep learning architectures for sentiment analysis on drug reviews. Journal of Biomedical Informatics, 2020, vol. 110, p. 103539. [Journal Impact Factor: 6.317, Q1]. Partially included in Chapters 3, and 4.
     https://doi.org/10.1016/j.jbi.2020.103539
   - I. Segura-Bedmar, **C. Colón-Ruiz**, M. Á. Tejedor-Alonso, and M. Moro-Moro (2018). Predicting of anaphylaxis in big data emr by exploring machine learning approaches. Journal of Biomedical Informatics. [Journal Impact Factor: 2.753, Q1]. Partially included in Chapters 3, and 4.
     https://doi.org/10.1016/j.jbi.2018.09.012
   - **C. Colón-Ruiz**, I. Segura-Bedmar, and P. Martínez (2019). Análisis de Sentimiento en el dominio salud: analizando comentarios sobre fármacos. Procesamiento del Lenguaje Natural, 63, 15-22. ISSN: 1135-5948. 2019, Septiembre. [SCOPUS, Computer Science Applications, 0,235, Q3]. Partially included in Chapters 3, and 4.
     http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/6090

2. **Preprints:**

   - **C. Colón-Ruiz**, T. W. Fitzgerald, I. Segura-Bedmar, E. Birney, and M. Herrero-Zazo. Automated Extraction and Classification of Drug Prescriptions in Electronic Health Records: Introducing the PRESNER Pipeline. medRxiv, 2023. Submitted for publication in Bioinformatics [Journal Impact Factor: 5.8, Q1]. Partially included in Chapters 3, and 5.
     https://doi.org/10.1101/2023.10.04.23296481
   - **C. Colón-Ruiz** and I. Segura-Bedmar. Generative Strategies for Low Annotated Data Scenarios: Sentiment Analysis in Drug Reviews. 2023. Submitted for publication in Journal of Biomedical Informatics [Journal Impact Factor: 4.5, Q2]. Partially included in Chapters 3, and 4.

- **C. Colón-Ruiz** and I. Segura-Bedmar. Semi-Supervised Generative Adversarial Network for Sentiment Analysis of drug reviews. TechRxiv, 2021. Partially included in Chapters 3, and 4.

  https://doi.org/10.36227/techrxiv.17075054

3. **Conferences:**

- **C. Colón-Ruiz** and I. Segura-Bedmar (2019). Protected Health Information Recognition by BiLSTM-CRF. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019). Bilbao, Spain. 2019, Septiembre. CEUR, ISSN 1613-0073. 2421, 679- 686. Partially included in Chapter 3.

  https://ceur-ws.org/Vol-2421/MEDDOCAN_paper_6.pdf

- **C. Colón-Ruiz** and I. Segura-Bedmar (2019). Hulat-TaskAB at eHealth-KD Challenge 2019: Knowledge Recognition from Health Documents by BiLSTM-CRF. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019). Bilbao, Spain. 2019, Septiembre. CEUR, ISSN 1613-0073. 2421, 35 - 42. Partially included in Chapter 3.

  http://ceur-ws.org/Vol-2421/eHealth-KD_paper_3.pdf

- **C. Colón-Ruiz**, I. Segura-Bedmar, and P. Martínez (2017). Combining the BANNER tool with the DINTO ontology for the CEMP task of BioCreative V.5., BioCreative, Barcelona, España, April, 2017, Proceedings of the BioCreative V.5 Challenge Evaluation Workshop, pp: 104-107. Partially included in Chapter 3.

  https://biocreative.bioinformatics.udel.edu/media/store/files/2017/BioCreative_V5_paper14.pdf

- I. Segura-Bedmar, **C. Colón-Ruiz**, and P. Martínez (2017). LABDA at SemEval-2017 Task 10: Extracting Keyphrases from Scientific Publications by combining the BANNER tool and the UMLS Semantic Network, Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval), Vancouver, Canada, August, 2017, Association for Computational Linguistics, pp: 938-941. Partially included in Chapter 3.

  https://aclanthology.org/S17-2164/

Abstract

The main hypothesis of this Ph.D. dissertation is that the use of transfer learning mechanisms, semi-supervised approaches and pre-trained generative language models can improve the performance of models in scenarios with limited annotated data. This can be achieved by transferring the learned knowledge from one task to another, reducing the need for large amounts of annotated data, and generating synthetic instances or representations that enrich the training set, improving the robustness and generalization of the models. In particular, using Deep Learning algorithms that allow acquiring knowledge from tasks with similar features can mitigate the bottleneck problem of lack of rich annotated corpora, thus improving the learning of relevant features for specific Natural Language Processing tasks.

The increasing availability of Electronic Health Records (EHR) and patient reviews has resulted in a large volume of clinical documents where some information is unstructured. Due to the high cost of time and resources to extract information from clinical texts, there has been an increased interest in research and development of Natural Language Processing techniques to automate the process and optimize research into new clinical solutions and approaches to improve patient results. However, clinical documents present additional challenges compared to generic texts due to the difference in the language features used, specific acronyms, and non-standardized jargon by each system or clinical center. In addition, the need to de-identify and anonymize texts to guarantee data privacy means that access to clinical documents is limited, leading to a scarcity of annotated corpora. In the present document, datasets consisting of drug reviews and EHRs are employed for Sentiment Analysis and Named Entity Recognition tasks in few-data scenarios in order to validate the hypothesis.

The results show that pre-trained models on large corpora based on transformers overcome other Deep Learning algorithms. However, its performance declines when the number of annotated data is limited or when classes are under-represented. Furthermore, the results suggest that employing unlabeled data in semi-supervised approaches or including synthetic instances of pre-trained generative language models improves the performance of discriminative models fine-tuned in a few-data scenario. However, their performance decreases when there is enough training data to learn task-specific relevant features.

Resumen

La hipótesis principal de esta tesis doctoral es que el uso de mecanismos de aprendizaje por transferencia, enfoques semisupervisados y modelos del lenguaje generativos preentrenados puede mejorar el rendimiento de los modelos en escenarios con pocos datos anotados. Esto se puede conseguir transfiriendo el conocimiento aprendido de una tarea a otra, reduciendo la necesidad de grandes cantidades de datos anotados, y generando instancias o representaciones sintéticas que enriquezcan el conjunto de entrenamiento, mejorando la robustez y generalización de los modelos. En particular, el uso de algoritmos de Deep Learning que permiten adquirir conocimiento de tareas con características similares puede mitigar el problema del cuello de botella que supone la falta de corpus ricos anotados, mejorando así el aprendizaje de características relevantes para tareas específicas de Procesamiento del Lenguaje Natural.

La creciente disponibilidad de Historias Clínicas Electrónicas (HCE), así como de las reseñas electrónicas de pacientes, ha dado lugar a un gran volumen de documentos clínicos donde parte de la información está incluida de forma no estructurada dentro de textos. Debido al elevado coste en tiempo y recursos para extraer la información de los textos clínicos, se ha incrementado el interés en investigar y desarrollar técnicas de Procesamiento del Lenguaje Natural con el fin de automatizar el proceso y optimizar la investigación en nuevas soluciones y enfoques clínicos que mejoren los resultados con pacientes. No obstante, los documentos clínicos plantean retos añadidos frente a textos de carácter genérico debido a la diferencia de las características del lenguaje empleado, uso de acrónimos específicos y jergas no estandarizadas por parte de cada sistema o centro clínico. Además, la necesidad de anonimizar los textos, conlleva que el acceso a documentos clínicos sea limitado propiciando la escasez de corpus anotados. En el documento presente, conjuntos de datos formados por reseñas de medicamentos y HCE son empleados para tareas de análisis de sentimientos y de reconocimiento de entidades nombradas en escenarios de pocos datos con el objetivo de validar la hipótesis.

Los resultados muestran que los modelos preentrenados basados en transformadores superan a otros algoritmos de Deep Learning. Sin embargo, su rendimiento decae cuando el número de datos anotados es limitado o cuando hay clases infrarrepresentadas. Ademas, los resultados sugieren que emplear datos no etiquetados en enfoques semi-supervisados o incluir instancias sintéticas de modelos generativos del lenguaje preentrenados mejora el rendimiento de los modelos discriminativos ajustados en un escenario de pocos datos.

No obstante, su rendimiento disminuye cuando hay datos de entrenamiento suficientes para aprender las características relevantes para una tarea específica.

List of Abbreviations

ALM Autoregressive Language Modelling

AM Amazon Reviews

BERT Bidirectional Encoder Representations from Transformers

BiLSTM Bidirectional Long Short-Term Memory

BT Backtrainslation

CE Cross-Entropy

CNN Convolutional Neural Networks

CRF Conditional Random Fields

DA Data Augmentation

DRD Drug Review Dataset

DT Decision Trees

EHR Electronic Health Records

GAN Generative Adversarial Networks

GPT Generative pre-trained transformer

GRU Gated Recurrent Unit

IE Information Extraction

IMDB Internet Movie Database

KL Kullback-Leibler

KNN K-Nearest Neighbours

LR Logistic Regression

LSTM Long Short-Term Memory

MLM Masked Language Modelling

MLP Multilayer Perceptron

MPQA Multi-Perspective Question Answering

MR Movie Review

MSE Mean Squared Error

MT Machine Translation

NB Naive Bayes

NER Named Entity Recognition

NLG Natural Language Generation

NLP Natural Language Processing

NLU Natural Language Understanding

NSP Next Sentence Prediction

PHI Protected Health Information

PPL Perplexity

QA Question Answering

RE Relation Extraction

ReNN Recursive Neural Networks

RF Random Forest

RNN Recurrent Neural Networks

SA Sentiment Analysis

SL Supervised Learning

SSGAN Semi-Supervised Generative Adversarial Networks

SSL Semi Supervised Learning

SST Stanford Sentiment Treebank

SVM Support Vector Machines

VAE Variational Autoencoders

Contents

Published and submitted content **iv**

Abstract **vi**

1 Introduction **1**

1.1. Motivation 1

1.2. Objetives 4

1.3. Methodology 4

1.4. Outline 5

2 Background **8**

2.1. Input Text Representation: Word Embeddings 9

2.1.1. Static Word Embeddings 9

2.1.2. Contextual Word Embeddings 11

2.2. Deep Learning 12

2.2.1. Convolutional Neural Network 13

2.2.2. Recursive Neural Network 14

2.2.3. Recurrent Neural Network 15

2.2.4. Transformers: "Attention is All You Need" 18

2.3. Weakly Supervised learning. 24

2.3.1. Semi-supervised approaches 26

2.3.2. Data augmentation approaches 28

3 Related Work **31**

3.1. Standardized Datasets 31

3.1.1. Datasets for Sentiment Analysis 31

3.1.2. Datasets for Named Entity Recognition 34
3.2. Evaluation Metrics 37
3.3. Supervised approaches for Sentiment Analysis 38
3.4. Supervised approaches for Named Entity Recognition 51
3.5. Semi-Supervised approaches for Sentiment Analysis. 59
3.6. Semi-Supervised approaches for Named Entity Recognition 69
3.7. Data Augmentation approaches for Sentiment Analysis 76
3.8. Data Augmentation approaches for Named Entity Recognition 85
3.9. Conclusions. 95

4 Sentiment Analysis in Drug Reviews 99

4.1. Supervised Approaches 100
4.1.1. Introduction. 100
4.1.2. Dataset 101
4.1.3. Data Preprocessing. 102
4.1.4. Methods. 103
4.1.5. Evaluation. 109
4.1.6. Conclusions. 118
4.2. Semi-Supervised Approaches: Generative Adversarial Network 119
4.2.1. Introduction. 119
4.2.2. Dataset 120
4.2.3. Method 122
4.2.4. Evaluation. 126
4.2.5. Conclusions. 132
4.3. Data Augmentation approaches: Generative Pre-trained Transformers 134
4.3.1. Introduction. 134
4.3.2. Dataset 135
4.3.3. Methods. 136
4.3.4. Evaluation. 142
4.3.5. Conclusions. 145
4.4. Conclusions. 146

5 Named Entity Recognition in Clinical Texts **149**

5.1. Datasets 151

5.1.1. 2018 n2c2 dataset 151

5.1.2. UK Biobank dataset 152

5.2. Annotation of the UK Biobank dataset: UKB dataset 154

5.3. NER model 156

5.4. Evaluation. 158

5.4.1. Baseline results. 159

5.4.2. Results of CBB-CRF fine-tuned only on the UKB dataset 160

5.4.3. Results of CBB-CRF fine-tuned only on the n2c2 dataset 162

5.4.4. Results of CBB-CRF fine-tuned on the n2c2 and UKB datasets. 163

5.4.5. Overview comparison between different approaches 164

5.5. Conclusions. 165

6 Conclusions **169**

6.1. Conclusions on sentiment analysis in drug reviews 169

6.2. Conclusions on the named entities recognition in medical prescriptions 171

6.3. Overall conclusions 172

6.4. Main Contribution 172

6.5. Publications. 173

6.5.1. Journals 173

6.5.2. Preprints. 174

6.5.3. Conferences. 174

6.6. Future Work 175

6.7. Funding 177

Bibliography **179**

List of Figures

2.1 Overview of the CBOW and Skip-gram models of Word2Vec. w represents a one-word vector, and t is the order of occurrence in the sequence. . 10

2.2 Convolutional neural network applied to text classification. Image retrieved from Colón-Ruiz et al., 2019 [94] 14

2.3 Overview of Recursive Neural Network for SA: The vectors of the parent nodes are computed bottom-up using the compositionality function described in 2.3 (This function varies for different models). The root node obtains the sentence-level representation, which is useful for sentence-level SA. 15

2.4 Recurrent Neural Network 16

2.5 Long Short-Term Memory cell. 16

2.6 Gated Recurrent Unit cell. 18

2.7 Transformer architecture. Image retrieved from Vaswani et al. [76] 19

2.8 Overview of the BERT-base architecture and the pre-training procedure. The CLS token is used for the NSP task, and the masking tokens are used for the MLM task. 22

2.9 Illustration of types of weak supervision. Each bar represents the feature vector, while the tick marks represent the label. The intermediate graphs denote situations with mixed types of weak supervision. (*) represents labels that could be inaccurate. 24

2.10 Illustration of the effect of unlabeled data in binary classification. Image retrieved from Van et al., 2019. [111] 26

2.11 Overview of the taxonomy of data augmentation techniques. 29

3.1 Illustration of a non-terminal RAE tree node. The red nodes consist of the softmax layer for label distribution prediction. Image retrieved from Socher et al. 2011 [158] 39

3.2 Overview of an RNTN layer. Each dashed box represents a slice of the last dimension of the tensor. Each captures the words' influence in their parent nodes. Image retrieved from Socher et al, 2013. [29] 41

3.3 Overview of a DCNN network with a filter window size of 3 and 2, respectively. The (dynamic) layers have k-values of 5 and 3, respectively. Image retrieved from kalchbrenner et al, 2014. [161] 42

3.4 Overview of the Multi-task model with shared layer architecture. Image retrieved from Liu et al., 2016. [101] 44

3.5 Overview of the 3W3DT model, where LM is a trained model and CF is the model confidence function. Image retrieved from Basiri et al., 2020. [168] 47

3.6 Overall pre-training and fine-tuning procedures for BERT. Image retrieved from Devlin et al., 2018. [54] 48

3.7 Overview of permutation language modelling training. Image retrieved from Yang et al., 2019. [56] 50

3.8 General architecture of the winning entity recognition model in task 10 of SemEval-2017 (AI2). Image retrieved from Ammar et al, 2017. [183] . . 54

3.9 Overview architecture of the Hulat-TaskAB model in subtask A of ehealthkd-2019. Image retrieved from Colon et al, 2019. [184] 55

3.10 Overview of SSVAE architecture. Image retrieved from Xu et al., 2017. [198] 62

3.11 Overview of CVAE-Attention architecture. Image retrieved from Yu et al., 2019. [16] 63

3.12 Overview of M with Π-model (top) and TE (bottom). 66

3.13 Overview of Variational Sequential Labelers with Hierarchical Latent Variables (VSL-GG-Hier). Image retrieved from Chen et al., 2018. [210] . . . 71

3.14 Overview of prediction auxiliary modules for sequence labeling of the CVT model. Image retrieved from Clark et al., 2018. [215] 73

3.15 Overview of the architecture of the conditional BERT model. The tag embeddings correspond to the BERT segment embeddings. Image retrieved from Wu et al., 2018. [104] 79

3.16 Illustration of interpolation using SMOTE method. 83

3.17 Overview of the language model architecture with LSTM Image retrieved from Ding et al., 2020. [230] 87

3.18 Overview of Masked Entity Linguistic Modeling (MELM) for data augmentation. On the left without sequence linearization and on the right with the proposed sequence linearization. Image retrieved from Zhou et al., 2022. [233] 92

4.1 Class distribution in training and test datasets (10 classes) 101

4.2 Class distribution in training and test datasets (3 classes) 102

4.3 Probability density function and Cumulative distribution function over the length of reviews 103

4.4 Overview architecture of the concatenation of the CNN and LSTM models 105

4.5 Overview architecture of the LSTM + CNN system 106

4.6 Overview architecture of BERT + LSTM model 107

4.7 Overview architecture of Clinical BioBERT model 108

4.8 Model confusion matrices (3 classes) 118

4.9 Class distribution in training and test datasets 121

4.10 Probability density function and Cumulative distribution function over the length of reviews 123

4.11 Overview architecture of SSGAN-BL 124

4.12 Overview architecture of SSGAN-CBB 125

4.13 Macro-F1 and Micro-F1 curves of BERT-small BiLSTM and Clinical BioBERT classifiers 127

4.14 Macro-F1 and Micro-F1 curves of BERT-mini LSTM and SSGAN-BL classifiers 128

4.15 F1 curves of BERT-mini LSTM and SSGAN-BL classifiers for each class labels 129

4.16 BERT-LSTM and SSGAN-BL classifiers confusion matrices 130

4.17 Macro-F1 and Micro-F1 curves of SSGAN-BL and Clinical BioBERT classifiers 131

4.18 Macro-F1 and Micro-F1 curves of Clinical BioBERT and SSGAN-CBB classifiers 132

4.19 F1 curves of Clinical BioBERT and SSGAN-CBB classifiers for each class labels 132

4.20 Clinical BioBERT and SSGAN-CBB classifiers confusion matrices 133

4.21 Latent space visualisation of real and synthetic reviews. 140

4.22 Overview of data augmentation method 141
4.23 Macro-F1 and Micro-F1 curves of Clinical BioBERT with Data Augmentation 143
4.24 F1 curves of Clinical BioBERT with Data Augmentation for each class labels 144
4.25 Clinical BioBERT Macro-F1 and Micro-F1 curves with balanced data augmentation. 145

5.1 Probability density function and Cumulative distribution function over the length of EHR sentences (2018 n2c2 dataset). 152
5.2 Probability density function and Cumulative distribution function over the length of prescriptions (UK Biobank dataset). 153
5.3 Example of the contents of a file with ann extension and its IOB tagging. (a) Example of the content of a file with the ann extension. Tabs separate the columns. (b) Example of IOB tagging for prescription drugs. 155
5.4 Overview architecture of CBB-CRF model 157
5.5 Overview of the CBB-CRF model fine-tuning 158
5.6 Baseline and CBB-CRF models confusion matrices on the UKB test dataset 166

List of Tables

3.1 Overview table of datasets for SA. The size of each dataset, the average length (number of tokens) of the texts, the number of classes, and the language of the sentences are included. 33

3.2 Overview table of datasets for NER. The size of each dataset, the average length (number of tokens) of the texts, the number of entity classes, the number of entities, and the language of the texts are included. 37

3.3 Accuracy of text classification models on SST, IMDB and MR datasets. The best results are shown in bold. 52

3.4 Results of the model evaluation on the DRD dataset. The best results are shown in bold. (*) The dataset was divided into two sentiment classes (median), i.e., into classes above- and below-average. 52

3.5 Results of evaluating NER approaches on test sets from different datasets in micro-F1. The best results are shown in bold. 58

3.6 Accuracy of semi-supervised text classification models on the IMDB dataset. The best results are shown in bold. 68

3.7 Accuracy of semi-supervised text classification models on the MR dataset. The best results are shown in bold. 68

3.8 Results of the SSGAN text classification model on the SST-5 dataset. Results are shown in terms of accuracy. The best results are shown in bold. . 68

3.9 Results of VAE-based text classification models according to the amount of unlabeled data. The first part of the table is in terms of accuracy and the second is in terms of F1. The best results are shown in bold. 69

3.10 Micro-F1 of the semi-supervised NER models on the CoNLL-2003 dataset. The best results are shown in bold. 76

3.11 Sentences generated using EDA. 77

3.12 Accuracy of the BERT-base model on the SST-2 test dataset from [220] paper. The classifier has been trained with different data augmentation approaches on 10 examples of each class. The best results are shown in bold. 81

3.13 Accuracy obtained by models trained with data augmentation methods based on the amount of labeled data available. Results are obtained on the SST-2 test dataset. The best results are shown in bold. 84

3.14 Accuracy achieved by models trained using data augmentation methods based on the amount of labeled data available from the SST-5 and IMDB datasets. The evaluation is conducted on the respective test datasets. The most outstanding results are highlighted in bold. 85

3.15 Original example and different transformations of augmented examples. The changes are highlighted in bold. 86

3.16 Micro-F1 on the CoNLL-2003 test dataset from the work [230]. The classifier has been trained with different data augmentation approaches with 20k synthetic examples. The best results are shown in bold. 89

3.17 Micro-F1 on the CoNLL-2003 test dataset from the [232] work. The NER models have been trained with different data augmentation approaches: LC adds 10 augmented data per original data, and QA adds 100,000 augmented data in each configuration. The best results are shown in bold. . . 91

3.18 Micro-F1 on the CoNLL-2003 test dataset from the [233] work. The NER models have been trained with different data augmentation approaches. The best results are shown in bold. 92

3.19 Micro-F1 on the CoNLL-2003 test dataset from the [69] work. The NER models have been trained with different data augmentation approaches. The best results are shown in bold. 94

4.1 Results of Linear SVM (baseline) 110

4.2 Results of the CNN models using different initializations. The best results are shown in bold 110

4.3 Results of the LSTM models using different initializations. The best results are shown in bold 111

4.4 Results of the hybrid architectures (using random initialization). The best results are shown in bold 112

4.5 Training time in minutes for 200 epochs using the Keras 2.0 library (10-class dataset) 114

4.6 Results of Linear SVM with tf-idf representation (baseline) 115

4.7 Results of the CNN/LSTM architectures (using random initialization). The best results are shown in bold 115

4.8 Results of the hybrid architectures (using random initialization). The best results are shown in bold 115

4.9 Results of BERT architectures. The best results are shown in bold 116

4.10 Training time in minutes for 200 epochs using the Tensorflow 2.3.0-tf library (3-class dataset) 117

4.11 Training time in minutes for 10 epochs using the Tensorflow 2.7.0-tf library (3-class dataset) 117

4.12 Ratios between labeled and unlabeled reviews. For example, the ratio 1:99 indicates that only 1% of the dataset is labeled. 122

4.13 Percentage of labeled reviews out of the total dataset used for the training phase. 136

4.14 Example of text pre-processing. The examples shown are short reviews due to a lack of space. 138

4.15 PPL of the GPT-J model on the test set. GPT-J was fine-tuned on different percentages of annotated data from the training set. 139

4.16 Examples of synthetic drug reviews. The examples shown are short reviews due to a lack of space. 142

4.17 Clinical BioBERT results using data augmentation (100% of annotated data). Due to computational cost, the comparison is performed using only one synthetic sample for each instance. 145

5.1 Statistics on the filtered annotated dataset from the n2c2 dataset. 152

5.2 Statistics on the annotated dataset from the UKB dataset. 156

5.3 The evaluation results of the baseline approach on the UKB test dataset. The baseline approach consists of the union of the ChEMBL dictionaries, Med7_lg,and Med7_trf results ($ChEMBL(x) \cup Med7_lg(x) \cup Med7_trf(x), x \in test$). 160

5.4 The evaluation results of the CBB-CRF model on the UKB test dataset. The CBB-CRF model has not been previously fine-tuned on the n2c2 dataset. 161

5.5 Evaluation results of the CBB-CRF model on the n2c2 test dataset. The CBB-CRF model has only been fine-tuned on the n2c2 dataset. 162

5.6 Evaluation results of the CBB-CRF model on the UKB test dataset. The CBB-CRF model has only been fine-tuned on the n2c2 dataset. 163

5.7 The evaluation results of the CBB-CRF model on the UKB test dataset. The CBB-CRF model has been previously fine-tuned on the n2c2 dataset and subsequently on the UKB dataset. 164

5.8 Strict evaluation results of the different approaches on the UKB test dataset (F1-score). The best results are shown in bold. 165

5.9 Lenient evaluation results of the different approaches on the UKB test dataset (F1-score). The best results are shown in bold. 165

Chapter 1

Introduction

1.1. Motivation

Artificial intelligence (AI) has proven to be a cross-cutting tool that helps to perform tasks typically performed by humans automatically [1]. Specifically, supervised approach algorithms have achieved great success in learning implicit patterns in data and extracting information by learning from annotated training datasets consisting of two-element tuples: a feature vector describing an example and a label or set of labels indicating the desired output of the model to be trained given that particular example [2].

In Machine Learning, models are trained with features defined by researchers and experts to provide relevant information on each example for a given task, resulting in a resource-intensive and time-consuming task [3]. On the other hand, deep learning algorithms can learn these features by themselves from the raw data [4], outperforming traditional machine learning algorithms [5], [6] but at the cost of being data-hungry algorithms that require large annotated datasets [7] and, in many tasks, such sets can be difficult to obtain, not being sufficient in quantity or quality due to the high cost of labeling the data or due to its nature [8].

The above being said, weakly supervised learning consists of techniques employed when access to the labeled dataset is limited [9]. Generally, weakly supervised learning can be categorized in three scenarios: i) the given labels are not always ground-truth (inaccurate supervision) [10], e.g., a silver dataset is used, i.e., an automatically labeled dataset being susceptible to contain errors in the labels; ii) the labels are coarse-grained (inexact supervision), e.g., this can be observed in computer vision, where labeling is performed at the image level instead of labeling each at the level of objects contained within the image [11], or in text classification, where custom labeling may be related to the target labels but lacks accuracy due to subjectivity [12]; and finally, iii) only a small subset of annotated data is available while the rest of the set is unlabeled (incomplete supervision) [13], [14].

Weak supervision also occurs in Natural Language Processing (NLP), as access to

fully labeled datasets is limited despite increasing data availability [12], [15], [16]. NLP is the field that studies the interaction between human language and computers and how computers may be able to understand, interpret or generate human language to perform tasks such as Natural Language Understanding (NLU), Natural Language Generation (NLG), Machine Translation (MT), Information Extraction (IE), Relation Extraction (RE), Question Answering (QA) and text classification among others [17].

Within the different tasks existing in NLP, the Sentiment Analysis (SA) task can be considered a text classification task that attempts to determine the corresponding category of a text according to its emotional polarity [18]. Specifically, SA refers to the determination of the opinion or emotion expressed about something in a text, which allows obtaining information about whether it supports a particular viewpoint of the text and can contribute to various applications, such as financial forecasting [19], political strategy planning [20], public opinion analysis [21], online decision making [22], and pharmacovigilance [23], among many others. On the other hand, the Named Entity Recognition (NER) task focuses on automatically identifying concepts or designators belonging to predefined semantic types within texts, such as places, people, organizations, quantities, etc. [24]. Such a task can provide valuable information to help search engines deliver more relevant results by understanding user intent and extracting specific information from texts, assisting in a wide range of tasks such as IE [25], [26], MT [27], text summarisation [28], and others.

In order to deal with the challenges of NLP tasks, several standardized corpora and tasks for the study and investigation of different techniques have been proposed over time (e.g., in the case of SA and NER, Stanford Sentiment Treebank [29], SemEval 2017 Task 4: Sentiment Analysis in Twitter [30], CoNLL 2003 Shared Task [31]). However, despite the existence of the corpora constructed by the different tasks, they are often not sufficient or do not fully meet the specific needs of more specific tasks in real-life scenarios [32].

The source of all these corpora usually comes from documents for general purposes such as newspapers, Wikipedia, Twitter, or generic product reviews. However, one of the most valuable fields of knowledge is the clinical field and how the use of NLP techniques can help improve the extraction of useful information to support medical and pharmacological research areas to improve patient outcomes [33]. In the field of pharmacovigilance, early detection of side effects depends on specific clinical trials and testing protocols, which are usually performed under specific conditions with a limited number of test subjects and time [34]. Therefore, automatic analysis of online user reviews and automatic information extraction could help obtain relevant information on the efficacy of medicines and their side effects. For example, automatically detecting negative user reviews could alert about unexpected drug behaviors and obtain relevant information to improve pharmacovigilance systems [35]. In contrast, identifying positive user reviews could help assess drug efficacy and even discover new uses for a drug [36].

Furthermore, the clinical field poses additional challenges for NLP tasks due to: a

greater complication in the use of specific jargon and terminology; a great variability in the structure, distribution, and terminology used between clinical centers; a great variability in how patients and health professionals express symptoms, adverse effects or diseases; and the confidentiality of the information, requiring additional security measures, de-identification and anonymization of texts, implying limited access to the data [37]–[40]. Thus, specific corpora for the analysis of clinical texts have been created to address these challenges and to facilitate the development and evaluation of NLP algorithms and techniques adapted to this domain. E.g., IE tasks in the clinical domain include corpora from i2b2 [41], [42], n2c2 [43]–[45] and ShARe/CLEF eHealth Challenges [46], while for SA tasks Drug Review Dataset (Drugs.com) [47] stands out. These corpora are designed to address the particularities of medical and pharmacological language, making them ideal for developing and evaluating NLP techniques that can be applied in the diagnosis, treatment [48] and monitoring of diseases [49] and therapies.

However, while using these corpora with deep learning techniques is a promising trend, there is great variability between documents, where texts are typically heterogeneous, making it difficult for models to generalize in extracting information from unstructured sources [50]–[52]. As a result, the need to adjust models to specific applications of interest implies using new annotated datasets, facing the challenge of weak supervision [51].

Transfer of learning [53] approach is one of the most successful approaches to mitigate the lack of task-specific annotated data and weak supervision scenarios. This approach attempts to reduce dependency by reusing knowledge acquired in similar tasks for the target tasks. Specifically, one of the most widespread techniques is using pre-trained models in similar domains [37], [51], [54]–[57]. However, this technique requires that there is no high mismatch in the domain of the datasets used, such as the size of the inputs to the models or significant differences in the language used in the case of NLP. Furthermore, depending on the data available for the target task, a pre-trained model may face difficulties that lead to "*rethinking the prior training*" or the "*catastrophic forgetting dilemma*" (loss of prior learning during the learning of new information) and may suffer from adjustments to local minima or systematic biases, being replicated in the new task [53], [58]. Due to the heterogeneity of texts, such problems are accentuated in the clinical domain, and model generalization to new datasets is problematic even when fine-tuned to the target task (downstream task) [51], [52].

Therefore, the main hypothesis of this Ph.D. dissertation is that the use of transfer learning mechanisms with semi-supervised approaches and pre-trained generative language models that provide synthetic instances or synthetic latent representations, as well as fine-tuning models to tasks with similar text distribution could improve model performance in few-annotated data scenarios. This hypothesis is based on the assumption that these mechanisms can transfer learned knowledge from one task to another, reducing the need for large amounts of annotated data and generating instances or synthetic representations that can enrich the training set, improving the robustness and generalization of

models. To test this hypothesis, the performance of different models for SA on drug reviews, as well as models for NER on medical prescriptions, is evaluated using supervised and weakly supervised approaches. The performance of these models when trained on different datasets, including original and synthetic instances generated by transfer learning mechanisms, is compared to determine whether these approaches effectively improve models' performance with limited annotated data. Overall, the results of this research could contribute to developing more efficient and effective models for NLP tasks, specifically SA and NER, for clinical applications, which could positively impact patient care and medical research.

1.2. Objetives

Considering the above, the main objective of the Thesis is to apply NLP and deep learning techniques to process drug reviews and medical records associated with patients. The aim is to develop models for SA of drug reviews and automatic extraction of relevant information, both cases in weakly supervised scenarios.

The main objective is composed of the following specific objectives (SO):

- SO1: Investigate, design, and evaluate supervised approaches to SA in drug reviews.
- SO2: Investigate, design and evaluate semi-supervised approaches for SA in drug reviews.
- SO3: Investigate, design and evaluate data-augmented approaches with pre-trained generative models for SA in drug reviews.
- SO4: Investigate, design and evaluate the above approaches for NER in clinical texts.

1.3. Methodology

In order to achieve the objectives of the Thesis and validate the hypothesis, the following methodology has been followed:

- **State-of-the-art study**: Study on the current state of the literature reviewing supervised and weakly supervised learning methods applied to NLP tasks and previous related works. Special emphasis is put on analyzing the performance of the approaches in the clinical area, focusing on SA and NER tasks.
- **Design approaches and architectures**: The next stage consists of designing the architecture of the models and methods to be used to leverage transfer learning

mechanisms from other similar corpora, unlabeled data, and pre-trained models in similar domains to solve the two tasks: SA and NER.

- **Experimentation**: Once the design is finalized, it is implemented to carry out experimentation with annotated datasets in different scenarios (few labeled data, relying on unlabeled or augmented data), observing the impact on the models as a function of the amount of annotated training data available whenever possible.

- **Analysis of errors**: After a quantitative evaluation of the models trained on the two different tasks, the main causes of false positives and false negatives in the test datasets are also analyzed to find the weak points of our method and design future lines of work.

- **Discussion**: The results obtained during the evaluation and the problems detected in the analysis of errors are used to discuss whether the techniques employed manage to outperform the results obtained by other techniques in weak supervision scenarios. Furthermore, weaknesses are identified, and different lines of future research are proposed to address the problem of weakly supervised scenarios in NLP tasks.

1.4. Outline

This Thesis is organized into six Chapters as follows:

Chapter 1 contains an introduction where the motivation behind the research is established, detailing and justifying the importance of addressing the issues described, as well as the main and specific objectives designed to respond to the hypothesis. It explains the methodology used and, finally, the document's structure.

Chapter 2 provides a detailed review of weakly supervised learning approaches and deep learning models relevant to NLP. The review aims to provide a theoretical framework for understanding the approaches and models subsequently used in SA and NER in few-annotated data scenarios.

Chapter 3 describes an overview of recent work in SA and NER, emphasizing deep learning in supervised and weakly supervised scenarios. This chapter focuses on works performed on standardized datasets and on SA in drug reviews and NER in the clinical domain.

Chapter 4 presents the research conducted on SA task focusing on drug reviews using deep learning approaches. Although, as shown in Chapter 3, a wide variety of research work with deep learning based systems for SA has been conducted in recent years, the analysis of patient sentiment, specifically patient experience with drugs, has received less attention in research [47], [59], [60]. This chapter addresses this gap by conducting a comprehensive study on various deep learning approaches and their combinations, assessing their suitability for capturing pertinent information related to drug reviews. The study

encompasses different scenarios involving varying amounts of annotated data and investigates how the proposed approaches can alleviate the limitations of purely supervised methods in few-annotated data scenarios. Furthermore, a comparative analysis of the implemented deep learning approaches is presented. Initially, the performance evaluation of the models is conducted on the Drug Review Dataset (Drugs.com) [47] using supervised approaches. Subsequently, the best-performing models are trained under low annotated data scenarios and compared against semi-supervised techniques and augmented data approaches that employ pre-trained generative models. Finally, the chapter concludes with a discussion and analysis of the experimental results obtained.

Chapter 5 presents a specific case for NER from a set of clinical prescriptions from the UK Biobank of the UK National Health System (NHS) without previously annotated data. Despite available models trained for information extraction from clinical texts and Electronic Health Records (EHRs), the transferability in the clinical domain of the models presents a challenge when applied in similar contexts in different medical centers [51]. Due to the heterogeneity of texts, an accurate predictive model may not generalize to the same level on new data when the data distribution used for training differs [52]. In this chapter, a study is conducted on different deep learning models with a supervised approach for the NER task in a few-annotated data scenario. Specifically, the effect of transfer learning on pre-trained models based on transformers and prior fine-tuning on tasks with similar datasets (track 2 of the n2c2 2018 [44]) is evaluated when only a few annotated data is available for the target task. Finally, the chapter concludes with a discussion, analysis of the experimental results obtained, and conclusions on the study conducted.

Chapter 6 draws the conclusions from the studies conducted and proposes future work in order to improve the results in weakly supervised scenarios. In addition, this chapter provides details on the main contributions and funding.

Chapter 2

Background

In machine learning, Supervised Learning (SL) approaches use algorithms that fit functions to training examples. These examples consist of a pair of elements, including an input vector and the desired output. This approach aims to learn a model capable of generalizing from training data and making accurate predictions on new data. Within this context, the problems to be addressed can be categorized into classification and regression problems [2]. In NLP, we also have to consider sequence-to-sequence (seq2seq) tasks, such as text summarization or machine translation, where the input consists of a sequence of tokens, and the output is another sequence of tokens. Nonetheless, the primary focus of this thesis lies in the domain of classification tasks.

While in regression, the desired output consists of continuous values (prices, temperature, among others) [61], in classification, the output consists of discrete categorical values (classes/labels) [62]. Within classification problems, three main scenarios are to be considered: binary, multi-class, and multi-label classification problems [63]. In binary classification problems, the model must decide between two mutually exclusive classes based on input vectors (examples). In contrast, multi-class problems involve classifying examples into one of several distinct classes, usually with more than two classes in total. Finally, in multi-label problems, an example may be associated with several classes instead of just one.

For example, in SA, models receive a sequence of words as input and can predict categories as a binary classification problem, distinguishing between positive and negative categories, or can be multi-class, including positive, neutral, negative, or even multiple sentiment levels [64]–[66]. Furthermore, SA can be applied to the whole document or specific sections about various objectives and aspects expressed in the document [67]. In the case of NER, the classification can be binary, distinguishing between named and unnamed entities within the text. It can also be multi-class and classify entities into different classes. Finally, in multi-label NER problems, an entity can simultaneously belong to several categories, such as an organization and a geographical location [68], [69].

Since this thesis focuses on supervised and weakly supervised learning with deep

learning for SA and NER tasks, this chapter provides a theoretical framework to understand the most relevant approaches and models used in this work and the literature. First, it will provide a detailed description of how text inputs are represented. Next, it will describe deep learning and the main models relevant to NLP, and finally, it will detail weakly supervised learning, showing how it differs from supervised learning.

2.1. Input Text Representation: Word Embeddings

As previously described, models are given a sequence of words as input in the context of NLP. However, these sequences must have numerical representations for the algorithms to work with them. Traditionally, the most commonly employed approaches were based on simple vector representations such as bags of words (BoW) or TF-IDF representations to which specific features (n-grams, part-of-speech, etc.) could be included to aid the task. However, such representations do not capture the semantics and context of the words and are sparse representations of high dimensionality.

Over the past decade, there have been significant advancements in representations based on word embeddings, wherein each word is associated with a fixed-length vector. These approaches effectively tackle the previously described issue and can be broadly categorized into two main categories: static and dynamic (also known as contextual) word embeddings.

2.1.1. Static Word Embeddings

In static word embedding models, the values of the word vectors are adjusted by taking into account the co-occurrence of words in large corpora [70]. Inspired by the distributional hypothesis [71], these word embedding models have the advantage that words with similar semantics have similar vector representations. However, the words will have the same representation regardless of the context in which they appear. In the following, the most relevant word embedding models are described.

Word2Vec [72] employs a neural network composed of an input layer, a projection layer (where vectors are averaged), and an output layer. Considering the architecture, there are two different approaches: Skip-Gram and Continuous Bag of Words (CBOW).

As shown in Figure 2.1, the Skip-Gram model maximizes the classification of neighboring words given another word in the text. In contrast, CBOW focuses on maximizing the classification of a word given neighboring words.

While both Word2Vec approaches are able to capture semantics and analogies between words, the CBOW model requires less time to converge, while Skip-Gram is less sensitive to overfitting frequent words and provides good results when trained on small data sets.

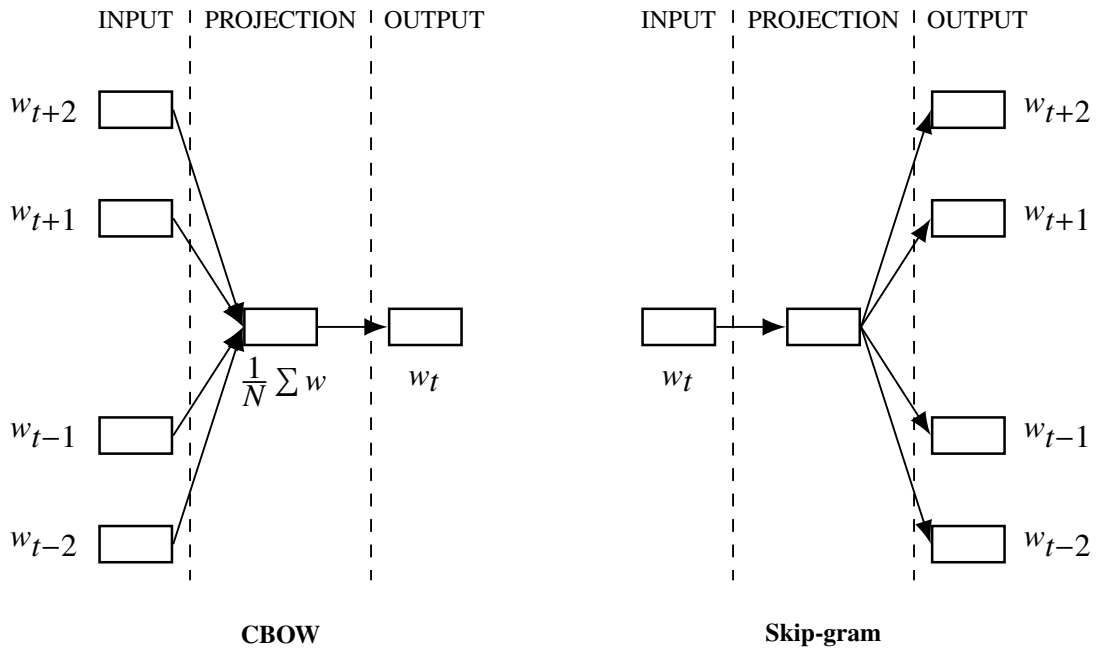


Figure 2.1: Overview of the CBOW and Skip-gram models of Word2Vec. w represents a one-word vector, and t is the order of occurrence in the sequence.

Global vectors for word representation (Glove) [73] employs weighted least squares to fit vectors from the inter-word ratios of the global counts matrix of word-word co-occurrences. The authors point out that, while from the co-occurrence matrix, the probabilities of word j given word i ($P(j|i)$) can be obtained, the relationship between words can be better examined when the ratios of the probabilities between two words are compared. Thus, Glove is trained by adjusting the values of the word vectors using the equation 2.1:

$$\begin{aligned} F\left(w_i^T \tilde{w}_k\right) &= P_{ik} = \frac{X_{ik}}{\sum_k X_i} \\ F\left((w_i - w_j)^T \tilde{w}_k\right) &= \frac{F\left(w_i^T \tilde{w}_k\right)}{F\left(w_j^T \tilde{w}_k\right)} \end{aligned} \tag{2.1}$$

Where w are word vectors, $\tilde{w}$ are the vectors of the context words, and X is the co-occurrence matrix. With this, Glove forces word embeddings to capture sublinear relationships in vector space, showing great performance in word analogy tasks. However, using the word co-occurrence matrix requires more memory for training than Word2Vec.

So far, the word embedding models described are not able to work with Out-Of-Vocabulary (OOV) words, as these models have a fixed vocabulary. Therefore, if a word is absent in the model's vocabulary, information about it is lost.

FastText [74], based on the Skip-gram model, addresses this problem by representing words as a bag of characters, where each word vector consists of the sum of the n-gram representations of the characters.

For example, with an n-gram length of three, the word “this” would be represented by the entire combination of its consecutive characters with a length of three: “<th thi his is> this”, where the characters “<” and “>” describe the word boundaries, and “this” is used as a special sequence. Note that the sequence “<his>”, corresponding to the word *his*, differs from the trigram “his”.

This representation of the words not only addresses the OOV problem but also provides morphological information about the words, making words with overlapping n-grams closer together in vector space.

2.1.2. Contextual Word Embeddings

One of the main problems described in static word embedding models is that they cannot capture the nuances that differentiate a word semantically depending on the context in which it occurs. Thus, they do not represent situations such as polysemy (multiple meanings) of words. For example, the word "bank" in the sentences "I sat on the river bank" and "I deposited money in the bank" is represented with the same vector in both sentences.

In recent years, different models have emerged that are able to provide word embeddings that address the problem described above [54], [75]. These types of embeddings, called dynamic word embeddings or contextual word embeddings, take into account the context in which the words occur so that, for example, different vectors are obtained for each instance of "bank," reflecting the different implicit meanings.

Unlike static embedding models that rely on shallow neural networks to acquire vector representations, dynamic word embeddings are derived from deep learning models trained in language modeling tasks. The concept of dynamic word embeddings was introduced with the appearance of the "Transformer" model published in [76]. This model is able to obtain contextual information from words by using attention layers when encoding and decoding text sequences. Moreover, models such as Embeddings from Language Models (ELMo) [75], which employs bidirectional layers of Long Short-Term Memory (LSTM) [77] recurrent networks trained to predict the next word in a text given the previous sequence, also helped popularise dynamic word embeddings in NLP tasks. However, transformer-based models such as Bidirectional Encoder Representations from Transformers (BERT) [54] have provided a breakthrough in this respect [37], [54], [66], [78].

To understand in more detail the previously mentioned models capable of providing contextualized embeddings, the following section describes the features of deep learning and the most popular models used in NLP tasks.

2.2. Deep Learning

Deep learning is a machine learning technique using neural network architectures to process information in an automated way [4]. The aim is to create models with several hidden layers to generate higher levels of abstraction, as in the human brain. Today, deep learning algorithms are used in many data processing tasks and offer better results than other machine learning approaches [3].

The approach to data processing has experienced a significant evolution, transitioning from manual rule-based systems to automated ones [79]. Traditional machine learning approaches employ feature engineering, involving hand-designed features extracted from input data, which are subsequently utilized to train models with algorithms such as Naive Bayes (NB), Support Vector Machines (SVM), Logistic Regression (LR), and Decision Trees (DT), among others [47], [80], [81]. Meanwhile, representation learning models automatically extract and learn relevant features [82]. In this way, deep learning algorithms have the capacity to generate intricate knowledge from simple feature representations by incorporating multiple layers capable of computing features at varying levels of abstraction [83].

The main advantage of deep learning is that the neural network can compute a new representation of the outputs for each layer per composition [4]. This means that each hidden layer of the network learns a new representation of the data, which is used as input for the next hidden layer. As a result, the network can learn more complex and abstract representations of the input data as it goes deeper into the hidden layers, allowing for a more efficient approximation of the objective function. In addition, deep architectures are trained by backpropagation, allowing the chain rule to transfer derivatives from higher layers to the first layers.

Deep learning techniques include Convolutional Neural Networks (CNN) [84], Recursive Neural Networks (ReNN) [85], and Recurrent Neural Networks (RNN) [86], which differ in the type of features they learn during training. CNNs are designed to capture spatial features and patterns, ReNNs to extract hierarchical features and patterns, and RNNs to obtain temporal features and patterns. In addition to these networks, transformer-based models [76], such as BERT [54] and Generative pre-trained transformer (GPT) [57], have revolutionized NLP [64], [87], [88]. These models are able to capture the relation between words in text sequences, allowing a better understanding of the meaning of the text and improving the results in tasks such as text classification [62] or NLG [89].

The following subsections review the main deep learning models relevant to NLP. The review aims to provide a theoretical framework for understanding the deep learning approaches and models subsequently used in SA and NER.

2.2.1. Convolutional Neural Network

CNN [84] was developed specifically to address the problem of image classification, which is a major problem in the field of machine learning and artificial intelligence. Before the advent of CNNs, traditional image classification methods typically used manual features extracted from images, such as edge, corner, and texture detection [90], [91]. These features were fed to classification algorithms such as SVMs [92] or DT to do the classification. However, as in NLP, this approach had some limitations. First, the manual feature extraction process was very laborious and time-consuming [3]. Second, the extracted features were not always the most relevant to the task [93].

A CNN is a neural network that uses convolutions to process input information. The basic idea behind convolution is that it takes a window of fixed size (kernel) and slides through the input to produce a new representation of the input. In the NLP context, the input is usually a sequence of words, representing each word as a vector of words (i.e., a pre-trained word embedding). The convolution window slides through these word sequences, extracting relevant features at different locations. This means that the CNN can learn specific patterns in different parts of the sequence, such as common word combinations in a text classification task.

$$Y_{i,j,k} = f(\sum_{c} \sum_{r} W_{c,r,k} X_{i+c,j+r} + b_k) \tag{2.2}$$

In equation 2.2, X represents the input to the CNN, which is usually a sequence of words or an image. W represents the convolutional filter or kernel, which has a size of c by r (two-dimensional inputs). b is the bias term and f is the activation function. The output of the convolution operation is represented by Y, which is a feature map. The convolution operation is performed by sliding the kernel over the input and taking the dot product between the kernel and the input at each position. This operation extracts a feature map that highlights specific patterns in the input. The bias term is added to this result, and the activation function is applied to produce the final output.

Thus, the output of a convolutional layer represents the input, where each unit represents a specific pattern. This is achieved by applying multiple filters to the input, each focusing on a different pattern. The output of each filter k is then applied to a nonlinear activation function (e.g., the ReLU function [95]) to introduce nonlinearities into the network. After the convolutional layer, a pooling layer is usually added that reduces the spatial dimension of the output representation. This is typically done by taking the units' maximum (max-pooling) or average (average-pooling) value in a given window. The pooling layer helps reduce the network size and capture more general characteristics of the input. Finally, fully connected layers can be aggregated to perform a classification or prediction task (see Figure 2.2).

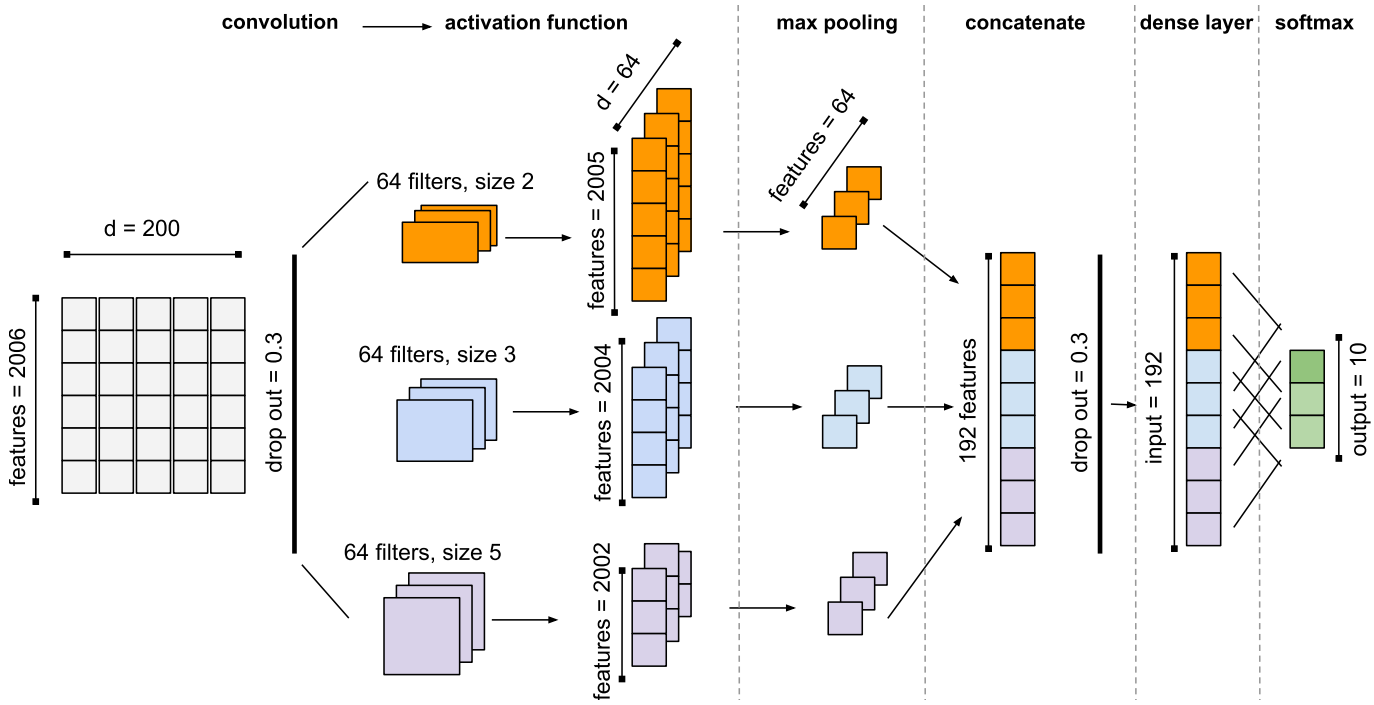


Figure 2.2: Convolutional neural network applied to text classification. Image retrieved from Colón-Ruiz et al., 2019 [94]

2.2.2. Recursive Neural Network

One of the first deep learning models to overcome traditional modeling approaches was ReNN [85]. These models represent the words of the sentences using word embeddings, being able to learn the semantic features of the words in the text recursively, as well as the syntactic structure of the texts without prior feature design.

Specifically, ReNNs are a class of neural networks that use a hierarchical tree structure, usually acyclic, to represent the input (see Figure 2.3). The network structure is built recursively, starting from individual words and gradually combining the representations to form larger representations of phrases and sentences. Specifically, the representations are obtained by combining the information from the child nodes recursively using the following compositionality function:

$$h_i = f\left(W\begin{bmatrix}u_l\\u_r\end{bmatrix} + b\right) \tag{2.3}$$

Where h_i is the subtree representation at node i, u_l and u_r are the vector representations of the subtrees merging at node i, W is the weight matrix shared by all nodes, b is the bias vector and f is the nonlinear activation function. The concatenation of the vectors u_l and u_r is denoted as $\begin{bmatrix}u_l\\u_r\end{bmatrix}$.

One of the main problems with these models is the a priori dependence on knowing the tree structure, using different rule-based parsers or other models previously trained on

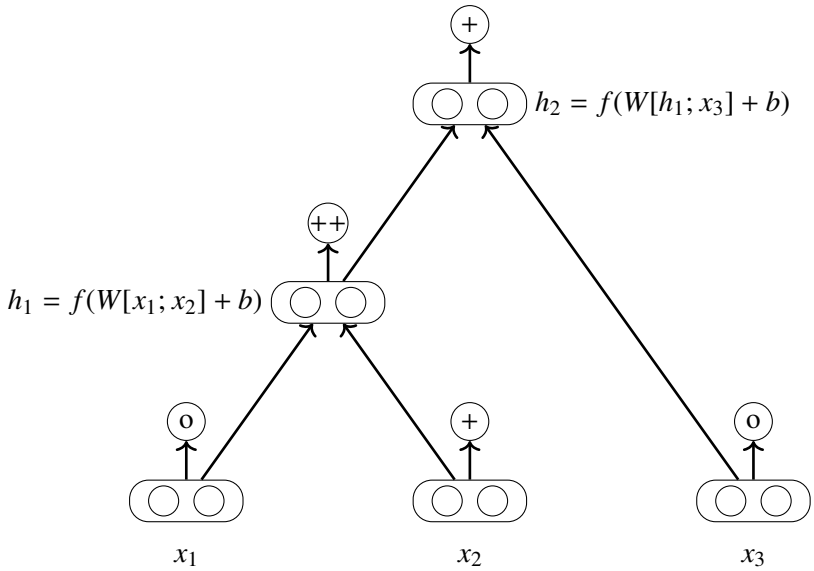


Figure 2.3: Overview of Recursive Neural Network for SA: The vectors of the parent nodes are computed bottom-up using the compositionality function described in 2.3 (This function varies for different models). The root node obtains the sentence-level representation, which is useful for sentence-level SA.

syntactically labeled text datasets such as [96], among others.

2.2.3. Recurrent Neural Network

RNNs [86] are a type of neural network architecture that processes input sequences, such as time series or sequences of words in natural language. Unlike CNNs, RNNs are composed of recurrent units that maintain an internal state or memory for processing sequential inputs. Each unit receives an input, combines it with its previous internal state and produces an output along with a new internal state used to process the next input. In this way, the network learns sequence patterns.

$$h_t = f(Wx_t + Uh_{t-1} + b) \tag{2.4}$$

In the equation 2.4, x_t corresponds to the input at time instant t, h_{t-1} is the previous internal state, W and U are the weight matrices, and b is the bias vector. For example, the activation function f can be the sigmoid function or the hyperbolic tangent function. Figure 2.4 shows a simple RNN where the state at each time step is calculated from the current input and the previous output.

However, RNNs have a limitation known as the gradient vanishing or exploding problem, making it difficult to train on long sequences [97]. During backpropagation, the gradient can become extremely small or large, difficulting to train on long sequences, as the network struggles to learn from the information in earlier time steps. In order to over-

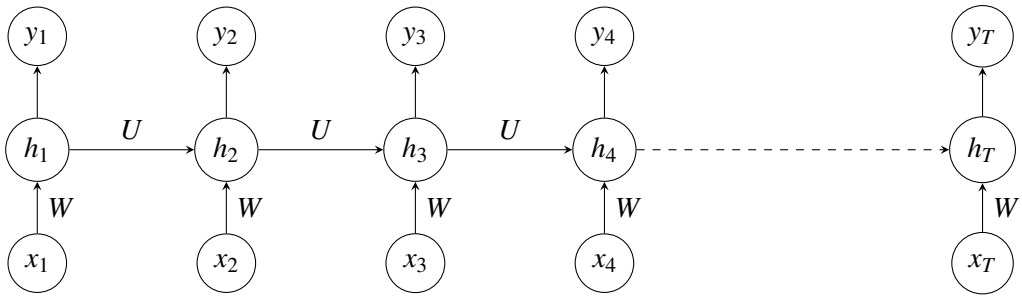


Figure 2.4: Recurrent Neural Network

come this problem, variants of RNNs have been developed, such as LSTM [77] and Gated Recurrent Unit (GRU) neural networks [98], which use special memory units to mitigate the gradient vanishing or exploding problem.

Long Short-Term Memory

LSTMs are a type of RNN architecture designed to overcome the limitations of traditional RNNs. LSTMs were first proposed by Hochreiter and Schmidhuber [77] and have since become a popular choice for processing sequential data, such as speech recognition, NLP, and time series analysis [99]–[102].

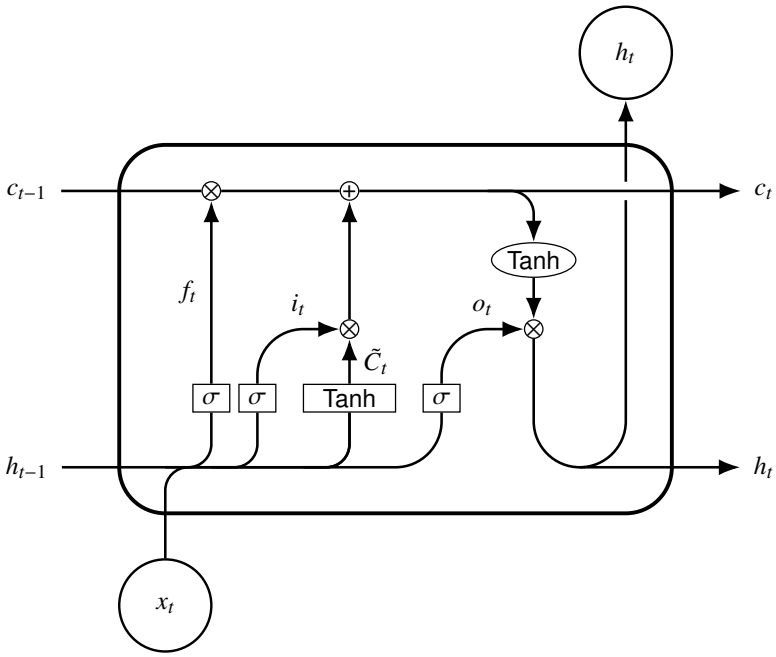


Figure 2.5: Long Short-Term Memory cell.

The main innovation of LSTMs is using memory cells (see Figure 2.5) that allow the network to remember or forget information over time selectively. Each memory cell comprises three gates: the input gate, the forgetting gate and the output gate. The input gate controls how much new information is added to the memory cell, the forgetting gate controls how much information is retained or discarded from the memory cell, and the

output gate controls how much information is sent from the memory cell to the next layer as can be seen in equation 2.5.

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
h_t &= o_t * \tanh(C_t)
\end{aligned}
\tag{2.5}
$$

Where f_t, i_t and o_t are the forgetting, input and output gates, respectively. $\tilde{C}_t$ is the candidate memory state, C_t is the current memory state and h_t is the output of the cell. x_t is the input at time t and h_{t-1} is the cell's output at the previous time. W_f, W_i, W_C and W_o are the weight matrices for each gate and the candidate memory state, and b_f, b_i, b_C and b_o are the respective bias terms. *sigma* and *tanh* are the sigmoid functions and the hyperbolic tangent, respectively.

LSTMs are suitable for processing long-term dependencies on sequential data as memory cells can selectively remember or forget information over time. This allows the network to retain important information and discard irrelevant information, especially when processing long sequences.

Gated Recurrent Unit

GRU [98] is a type of RNN that addresses the vanishing or exploding gradient problem of traditional RNNs. Like LSTMs, GRUs use special memory units to store information about a long sequence of inputs.

The GRU has a similar structure to the LSTM but with fewer parameters (see Figure 2.6), which makes it computationally more efficient. The main difference between LSTM and GRU is how they handle memory units. In GRU, the memory unit has fewer gates than in LSTM, and there is no separate output gate. Instead, there is a single update gate and a reset gate that control the flow of information. The update gate determines how much of the old memory to keep and how much of the new input to add, while the reset gate determines how much of the old memory to forget.

Equation 2.6 describes how the GRU Cell output is updated at each time step t, given the current input x_t and the previous output h_{t-1}. The cell has two gates that control how the current output h_t is calculated: the update gate z_t and the reset gate r_t. First, the update gate z_t and the reset gate r_t are calculated from the concatenation of the current input x_t and the previous output h_{t-1} using the weight matrices W_z and W_r and the biases b_z and b_r, respectively.

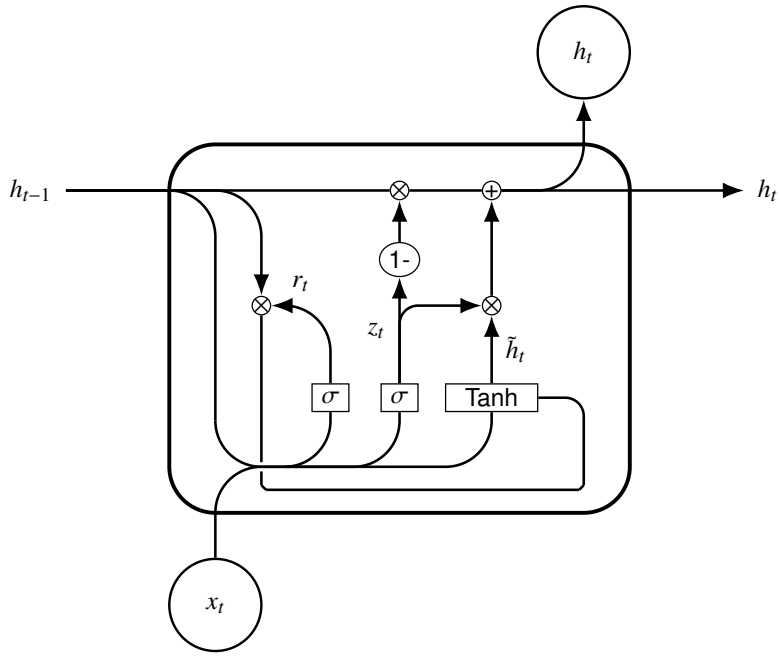


Figure 2.6: Gated Recurrent Unit cell.

$$
\begin{aligned}
z_t &= \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \\
r_t &= \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)) \\
\tilde{h}_t &= \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + b_h) \\
h_t &= (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t
\end{aligned}
\tag{2.6}
$$

These gates have values between 0 and 1, which determine how much new information and how much information from the previous output should be incorporated into the current output. Next, the "candidate" for the current output $\tilde{h}_t$ is calculated from the concatenation of the current input x_t and the previous output weighted by the restart gate r_t using the weights matrix W_h and the bias b_h, and applying the hyperbolic tangent function. Finally, the current output h_t is updated as a weighted combination of the previous output h_{t-1} and the "candidate" current output $\tilde{h}_t$, determined by the update gate z_t. If z_t is close to 1, then most of the information from the "candidate" to the current output $\tilde{h}_t$ is incorporated into the current output h_t. In contrast, if z_t is close to 0, the current output h_t will be similar to the previous output h_{t-1}.

In summary, GRU is a variant of RNN that uses fewer parameters than LSTM but achieves similar results. GRU is computationally efficient and can be used to process long sequences of input data [98].

2.2.4. Transformers: "Attention is All You Need"

Transformers [76] are a type of neural network architecture based on the innovation of self-attention mechanisms, which allow the model to weigh the importance of different

parts of the input sequence. This mechanism allows capturing long-range dependencies in the input sequence outperforming RNNs such as LSTM and GRU.

The first transformer model [76] consists of an encoder that receives the input sequence and a decoder that generates the output sequence (seq2seq). The encoder comprises multiple layers of self-attention and Feedforward Neural Networks (FNN). At the same time, the decoder also includes an additional layer of attention on top of the encoder output (see Figure 2.7).

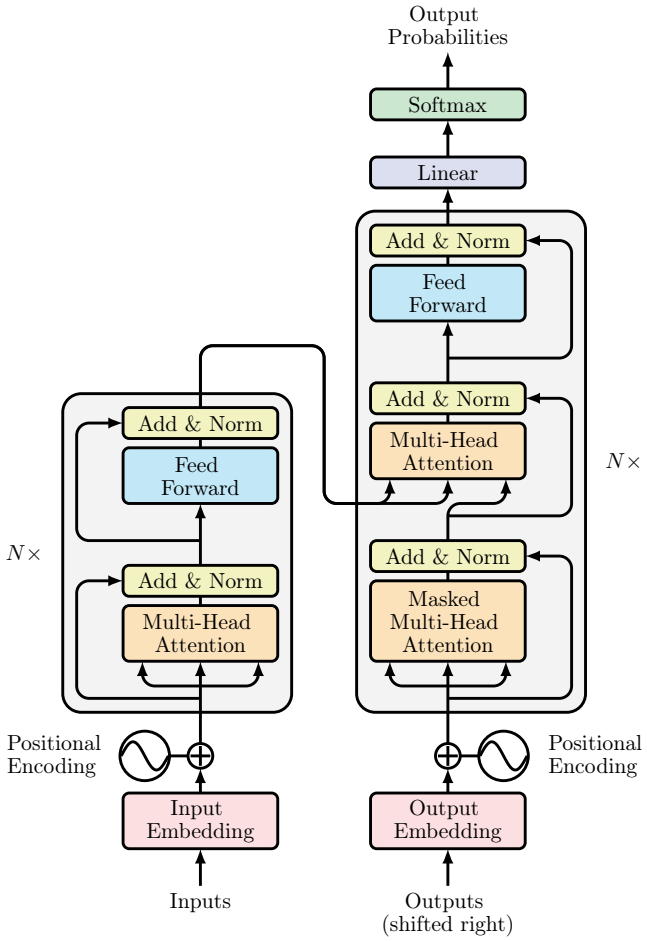


Figure 2.7: Transformer architecture. Image retrieved from Vaswani et al. [76]

In each self-attention layer, the input sequence is mapped into the Queries (Q), Keys (K), and Value (V) matrices through linear transformations by multiplying the weight matrices W_Q, W_K, and W_V by the input sequence X. In the equation, 2.7, Q, K, and V are the queries, keys, and values, respectively, and d_k is the dimensionality of the key vectors. Specifically, Q is a matrix with relevant information about each element of the input sequence, while K is another similar matrix, with a separate linear transformation, used to obtain the degree of similarity of each element with the components of Q. The softmax function is applied along the rows of the resulting matrix to determine the degree of attention to be paid to each position of the matrix V when generating the output.

In multi-head attention, the output is obtained by concatenating the matrices of each

attention layer (equation 2.8) and then performing a linear transformation for a residual sum with the original input, along with a normalization. This allows the original information to be retained more easily to prevent vanishing gradient problems.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.7}$$

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(head_1, \ldots, head_h)W^O \\ head_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.8}$$

In addition to the multi-head attention layers, as shown in Figure 2.7, the transformers also include FNNs that process the output of each multi-head attention layer. These networks consist of two linear transforms with a non-linear activation function in between (2.9), where x is the input to the feedforward network, W_1, W_2, b_1 and b_2 are the weight matrices and bias vectors, respectively, and $\max(0, \cdot)$ is the rectified linear unit activation function (ReLU) [95]. Finally, the decoder includes an additional layer of multi-head attention on the encoder output, which determines how much attention should be paid to each position in the encoder output when generating the decoder output.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.9}$$

However, both the encoder and decoder are composed of networks with no intrinsic notion of the position of each element of the input stream, resulting in models that do not know the relationship between tokens based on their position in the sequence. This information is relevant to understanding the meaning and structure of a text for any NLP task, as it provides the spatial context and temporal order, allowing differentiation between, for example, “the dog chases the cat” and “the cat chases the dog”.

In order to add information about the position of each element in the input sequence, positional encoding is used by adding position vectors to the word embedding vectors representing the inputs. Each position in the sequence is associated with a unique position vector added to the corresponding word embedding. These position vectors have a dimension equal to the dimension of the word embedding and contain information about the absolute or relative position of each word in the sequence. The equation for generating the position vectors in the Transformer position encoding is as follows (equation 2.10):

$$\begin{aligned} \text{Position}(p, 2i) &= \sin\left(\frac{p}{10000^{2i/d}}\right) \\ \text{Position}(p, 2i+1) &= \cos\left(\frac{p}{10000^{2i/d}}\right) \end{aligned} \tag{2.10}$$

Where p is the position in the sequence, i is the dimension of the position vector, and d is the dimension of the input (or embedding) vectors. The equation generates a unique

position vector for each position in the sequence and each dimension in the vector. The sinusoidal and cosinusoidal part of the equation allows the position vectors to be encoded, allowing nearby positions in the sequence to be similar in terms of cosine similarity.

Having described transformers, several models based on their architecture have emerged after their development. Among these, autoencoder models leverage encoding layers to acquire bidirectional text representations, with BERT being one of the most prominent examples of such models (detailed in the subsequent subsection 2.2.4). On the other hand, autoregressive models use decoding layers to generate text sequentially, with GPT being one of the most notable examples (described in the subsection 2.2.4).

Such transformer-based models, however, have a complexity regarding adjustable parameters, which requires large corpora for training. Therefore, a strategic approach is followed, divided into two crucial phases: pre-training and finetuning. The pre-training phase consists of training the model on one or several NLP tasks, for which large amounts of unlabeled text are used, for example, predicting the next word or predicting whether two sentences are consecutive. The models learn to capture linguistic relations and complex language patterns during this stage. Subsequently, in the finetuning phase, the models' weights are adjusted to a specific task, usually using a smaller and more specific dataset. This allows the model to leverage its prior knowledge as a transfer learning mechanism while reducing the computational requirements for task-specific tuning.

Considering the above, the two most popular pre-trained transformer-based models developed so far, BERT and GPT, are described in more detail below.

Bidirectional Encoder Representations from Transformers

BERT [54] is a pre-trained transformer-based model that has obtained state-of-the-art results in several NLP tasks. Specifically, BERT models use a transformer-based multiple encoder architecture [76] to pre-train a language model on a large corpus of text data (see Figure 2.8).

In order to pre-train the model, two different tasks are used. The first task is Masked Language Modelling (MLM). In this task, a certain percentage of the input tokens of a sentence are randomly selected and masked. The tokens that are masked are replaced by a special "[MASK]" token. The model is then trained to predict the masked tokens based on the context of the surrounding words. In this way, the model learns the relationships between different words in a sentence and understands their context within the larger text.

The second pre-training task is Next Sentence Prediction (NSP). In this task, the model is trained to determine whether two given sentences are consecutive in a larger text. For this purpose, the model processes two consecutive sentences taken as input during pre-training and determines whether the second sentence follows the first. In this way, the model learns to understand the logical and semantic relationships between sentences, which helps it to develop a broader understanding of the underlying structure of written

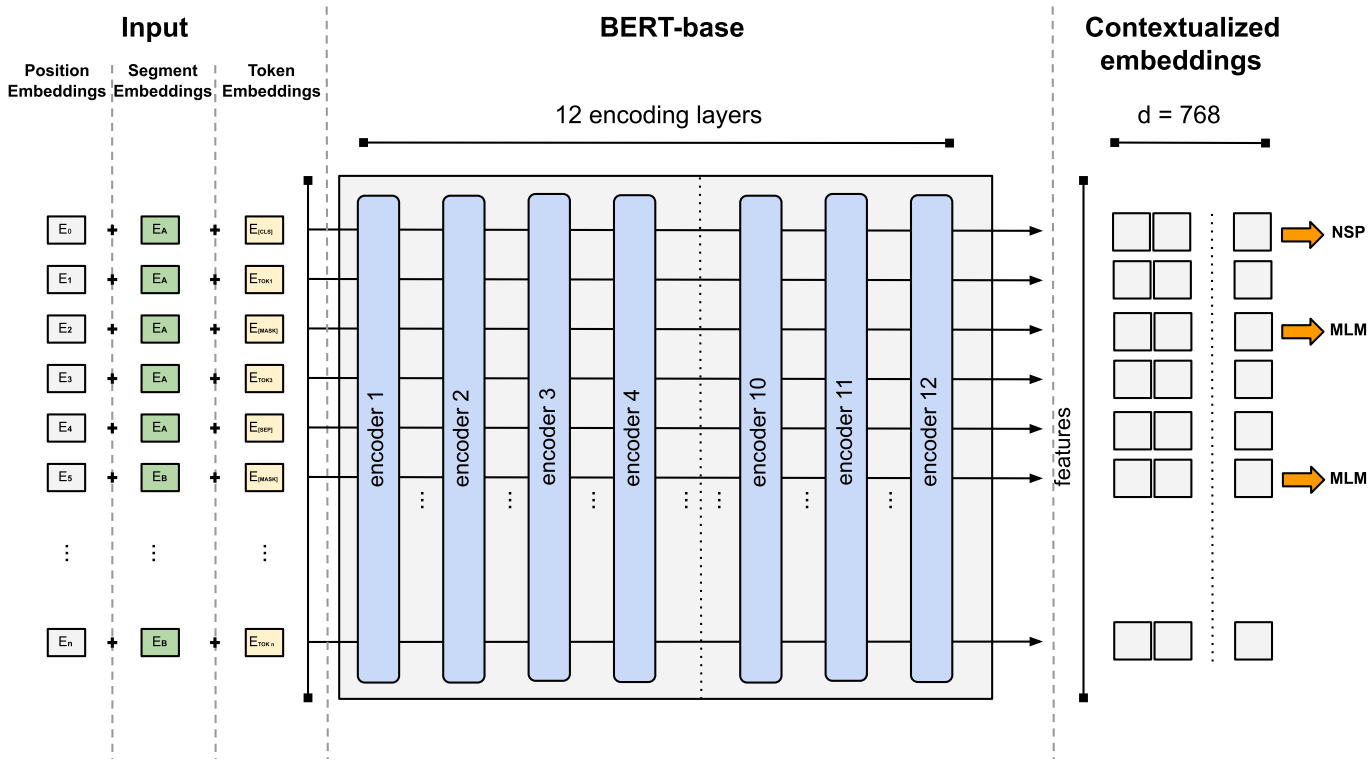


Figure 2.8: Overview of the BERT-base architecture and the pre-training procedure. The CLS token is used for the NSP task, and the masking tokens are used for the MLM task.

language.

In both pre-training tasks, the special tokens "[CLS]" and "[SEP]" are used. The "[CLS]" token is added at the beginning of each input sequence and is used for the classification task, where a label is predicted for the entire sequence. The "[SEP]" token separates two input sequences in the NSP task and separates the input text and the answer in question-answer tasks. Furthermore, the "[SEP]" token distinguishes which sequence vector to use. The sequence vector, like the positional vector, provides information about the position of both sequences.

In both pre-training tasks, the softmax layers generate a probability distribution over the set of possible masked words or the possible relationship between two consecutive sentences. In the MLM task, the softmax layer predicts the correct masked word from a probability distribution over the vocabulary. On the other hand, in the NSP task, the softmax layer is used to predict the correct relationship between the two sentences (i.e., whether they are consecutive or not).

Thanks to pre-training in these two tasks, BERT is able to develop a deep understanding of the relationships between different words and phrases in a text. This makes it very effective in a wide range of NLP tasks [54], [64], [103], [104]. In addition, BERT models can be easily adjusted to specific NLP tasks by adding a task-specific output layer and fine-tunning the model with the dataset for this specific task data, taking advantage not only of the transfer learning but also reducing the computational cost.

Generative Pre-trained Transformer

GPT [57] is a pre-trained transformer-based model used to generate text using a multi-decoder architecture [76] and is pre-trained on large corpora with the task of Autoregressive Language Modelling (ALM). An ALM model generates sequences of words individually, in a given order, using a statistical conditional probability model. That is, the model estimates the probability of each word in the sequence conditional on the previous words in the sequence and uses that probability to generate the next word.

$$P(y_{1:T}) = \prod_{t=1}^{T} P(y_t \mid y_{1:t-1}) \tag{2.11}$$

where $y_{1:T}$ represents the sequence of tokens to be generated, $P(y_t|y_{1:t-1})$ is the conditional probability of the next token y_t given the history of tokens generated $y_{1:t-1}$ so far.

The unidirectional training approach of GPT, in an autoregressive manner, distinguishes it significantly from BERT, which is trained bidirectionally using autoencoding techniques. This distinction makes GPT particularly well-suited for text-generation tasks, where maintaining coherence and fluency is paramount [105]. GPT achieves this by closely simulating the natural text generation process, progressively building upon prior context without assuming conditional independence among generated words. On the other hand, BERT's features make it better suited for language understanding tasks like text classification and IE, where a comprehensive understanding of token relationships in both directions is crucial for optimal performance [54].

In GPT, to only use the previous information in the sequence, masking matrices (M) are used for one-way attention. In M, values above the main diagonal are set to $-\infty$, while values below the main diagonal and on the main diagonal are set to zero. This ensures that attention only flows in one direction, from left to right, in the case of unidirectional left-to-right attention.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V \tag{2.12}$$

$$M_{i,j} = \begin{cases} 0 & \text{if } i < j \\ -\infty & \text{otherwise} \end{cases} \tag{2.13}$$

In this way, when the masking matrix is added to the attention matrix, the $-\infty$ values ensure that the attention values in the top positions do not contribute to the final attention calculation. This ensures that the model only looks to the left during attention, making attention unidirectional.

Once the GPT model has been pre-trained with large corpora of text, it can be fine-tuned to specific tasks by adjusting the weights of the pre-trained model with a smaller

task-specific dataset.

2.3. Weakly Supervised learning

In the literature, different supervised approaches have been successfully proposed to solve a wide variety of tasks in NLP, such as text classification [64], [106], NER [107], [108], RE [103], [109], or MT [27], [88], among others [17]. However, the most successful techniques, such as deep learning, require ground truth labels for a large training dataset [4]. In real-world scenarios, obtaining enough labels or quality labels, i.e., strong supervised information, can be difficult due to the high cost of labeling processes [8]. Therefore, several research efforts have been directed toward approaches that allow machine learning algorithms to work in weakly supervised scenarios.

Typically, there are three types of weak supervision [9]: **incomplete supervision**, **inexact supervision** and **inaccurate supervision**. Figure 2.9 represents an overview of the types of weak supervision.

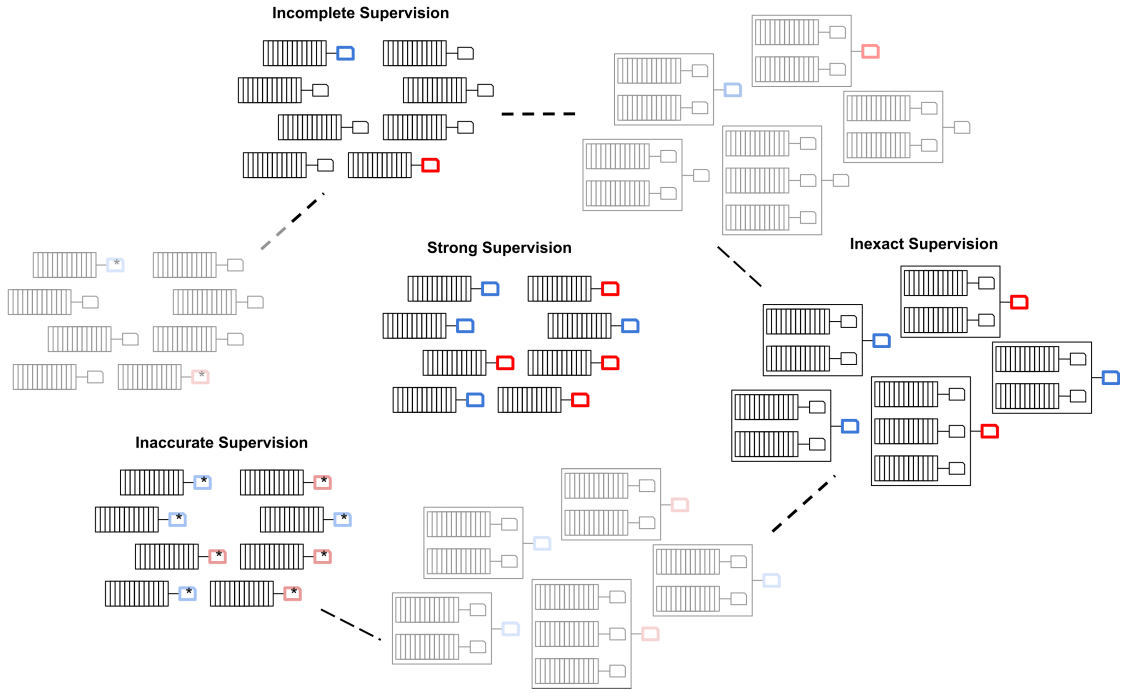


Figure 2.9: Illustration of types of weak supervision. Each bar represents the feature vector, while the tick marks represent the label. The intermediate graphs denote situations with mixed types of weak supervision. (*) represents labels that could be inaccurate.

In **incomplete supervision**, only a subset of the training data is usually labeled, while the rest remains unlabeled. For example, in the case of SA, ground-true labels of reviews can be provided by the authors or human annotators. However, the number of labeled reviews will be limited compared to all available reviews due to the human cost. The small subset of labeled data is generally insufficient to train well-performing models. Thus, the task is to learn $f : X \rightarrow Y$ from a training dataset $T = \{(x_1, y_1), ..., (x_l, y_l), x_{l+1}, ..., x_u)\}$,

where l is the number of labeled examples and u is the number of unlabeled examples less labeled ones.

There are two main techniques in this scenario. Concretely, Active Learning [110], and Semi Supervised Learning (SSL) [13]. Active learning techniques are based on training a model with the available labeled examples and using this model to label new examples. Once the new examples have been labeled, an "oracle" (this can be a human expert) is asked, which is the ground-truth label for those labeled with the lowest confidence. Once this phase is finished, the labeled new examples are added to the training set.

On the other hand, semi-supervised learning is conceptually situated between supervised and unsupervised learning. It aims to improve the performance of machine learning models when supervised learning is complicated by the difficulty of obtaining data with its associated output. Suppose that unlabeled data can provide additional information useful for model prediction. In that case, those trained with semi-supervised learning can improve task performance with little previously labeled data [111]. Therefore, semi-supervised learning techniques attempt to exploit unlabeled data without performing any queries.

In the **inexact supervision** scenario, supervised information is available, although it differs from the desired information for the task, e.g., the coarse-grained label information is available. The approaches used to deal with this scenario aim to use the available label information to obtain fine-grained information. Formally, it can be defined as the task of learning $f : X \rightarrow Y$ from a training dataset $T = \{(X_1, y_1), ..., (X_n, y_n)\}$, where each $X_i = \{x_{i1}, ..., x_{im}\}$ consists of a bag of unknown label instances x_{nm}. For example, in image classification, a set of images may be available where elements of interest appear, and each image is labeled with the element of interest (X, y). However, within the image, there is much more information that does not determine the belonging of that image to its label. In this case, each image is a group of instances consisting of smaller parts of the image $X_i = \{x_{i1}, ..., x_{im}\}$. The model aims to learn the group (image) label , considering that the group will be positive if one of its instances x is positive [112]. In text classification and SA, it is common to find texts labeled at the document level when the interest may lie at the sentence level. Such approaches can be applied by forming bags (documents) of sentences as instances x. The models, in this case, aim to obtain sentence-level information using bag-level tagging, as in [113]. For example, in a financial report providing the sentiment label at the document level, we may be interested in obtaining sentiment on sales growth forecast: “The interim report indicates that sales were up 25.2% from the prior-year quarter despite challenging market conditions. However, given the strong sales in the third and fourth quarters of the previous year, it will not be possible to maintain this growth rate for the current year as a whole.” While the document-level report has a positive label, the sales growth forecast expresses a negative sentiment.

Finally, in the **inaccurate supervision** scenario, the labels provided for each example may not always be ground-truth labels. This situation may occur due to annotator failures,

annotation discrepancies due to the difficulty of the text to be categorized, or due to the use of automatic annotation processes where the generated labels do not always match with ground-truth labels. In this scenario, many studies have focused on evaluating the types of noise in the label and the robustness of different algorithms assuming label noise [114]. In practice, one of the main approaches is to identify potentially mislabelled examples to correct or eliminate them. In work such as [115], a relative neighborhood graph with the latent representations of the data is used to filter out those nodes whose neighborhood is dominated by nodes of different classes. In contrast, in [116], they use assembly methods over multiple labeling sources to obtain the reliability of the labels.

Having detailed the background of weakly supervised learning, and since this thesis addresses problems with few-annotated data given in incomplete supervision scenarios, the following subsections focus on describing the two approaches used in this thesis to deal with such a scenario: Semi-supervised and Data Augmentation (DA) approaches. While semi-supervised approaches focus on using both labeled and unlabeled data to provide information to predictive models, data augmentation approaches focus on using the labeled training data to increase the diversity of the training with new synthetic data and provide complementary information to the models.

2.3.1. Semi-supervised approaches

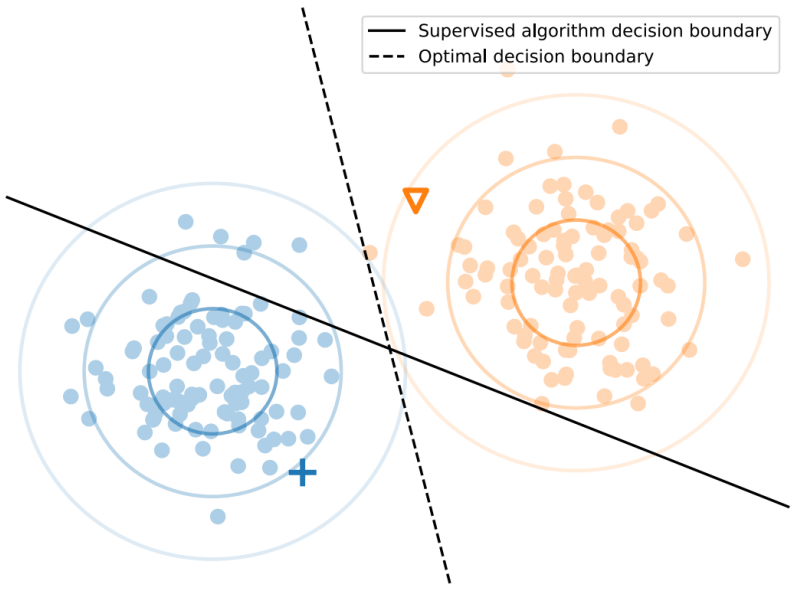


Figure 2.10: Illustration of the effect of unlabeled data in binary classification. Image retrieved from Van et al., 2019. [111]

As initially introduced, semi-supervised learning attempts to use labeled and unlabeled data to provide information for a specific task. However, in semi-supervised learning, the condition that the distribution of the data $p(x)$ over the input space contains information about the posterior distribution $p(y|x)$ must be met. In the affirmative case,

unlabeled data could be used to obtain information about $p(x)$ and thus about $p(y|x)$. If not, it would be impossible to improve the performance of predictive models by using additional unlabeled data. An intuitive illustration can be seen in Figure 2.10 with a binary classification problem. The dots represent the unlabeled data colored according to their true label. Without unlabeled data, a supervised algorithm does not have enough information to form a suitable decision barrier. In the presence of unlabeled data, a semi-supervised algorithm can improve performance based on some assumptions. In particular, smoothness, low-density, and manifold assumptions.

The **smoothness assumption** indicates that if two points of $X = \{x_1, ..., x_n\}$ are close in the input space, their corresponding labels should be the same. In semi-supervised learning, this assumption can be applied transitively to unlabeled data, e.g., having two nearby unlabeled points $x'_1, x'_2 \in X_U$, and a labeled point $x_1 \in X_L$ that is only close to x'_2, both x'_2 and x'_1 should have the label of x_1, as this is propagated transitively from x'_2.

The **low-density assumption** indicates that the decision barrier of a classifier should preferably pass through low-density regions. This assumption is closely linked to the smoothness assumption since if the decision barrier passes through low-density areas, it is expected that there are only a few points with different labels. However, if the decision barrier passes through dense areas, the smoothness assumption should be infringed since two nearby points in the input space should have different labels.

The **manifold assumption** indicates that in problems where data can be represented in Euclidean space, the observed data points tend to be concentrated in lower dimensional structures. Therefore, the manifold assumption indicates that data points concentrated in the same structure have the same label as those labeled data within that structure (manifold). For example, if a three-dimensional input space is considered, where a group of points lies on the surface of a sphere, such data form a 2-dimensional manifold and share the same label.

Considering the assumptions, however, the main question would be: *When are these conditions met, and therefore, is it possible to apply semi-supervised learning techniques?* The main disadvantage of semi-supervised learning is that it cannot be seen as a guaranteed way to obtain improved predictions by introducing unlabeled data [111]. For example, in work such as [117], performance degradations have been identified, and, in particular, this can be observed in scenarios where supervised classifiers can achieve a good performance. Therefore, semi-supervised learning should be treated as another set of approaches within the suite of algorithms considered for further evaluation in a particular scenario.

Within semi-supervised learning, there are four main categories of approaches: generative, graph-based, low-density separation, and disagreement. Generative methods aim to model the underlying distribution of data using labeled and unlabeled data. They often use latent variables to capture the underlying structure and use implicit information to improve model performance [16], [65].

Alternatively, graph-based methods consist of constructing graphs in which each node represents training examples. Relationships between nodes are determined by similarity measures between training examples, and these connections are used to propagate the label between different subgraphs to the unlabeled examples. These graph-based models can be either transductive or inductive. Transductive graph-based models assume a "closed world", where the test data is known in advance and their goal is to optimize performance on the test dataset, i.e., the test data is part of or exactly the unlabeled dataset. Conversely, inductive graph-based models aim to generalize the graph structure to unseen data points and have the ability to make predictions on new, unseen test data [118].

In contrast, approaches based on low-density separation focus on the classification boundaries being in the least dense areas within the input space consisting of both labeled and unlabeled data. A very illustrative example is the semi-supervised support vector machines (S3VM) that force the classification boundary to divide by the least dense areas while correctly classifying the labeled data [119].

Finally, disagreement-based approaches consist of training multiple models with different capabilities and/or features of the training dataset. The most representative approaches consist of self-training and co-training, where unlabeled data are pseudo-labeled by the models, and those with higher confidence are added to the training set in an iterative process [120].

2.3.2. Data augmentation approaches

Data augmentation focuses on generating new examples from the original training set. These synthetic examples aim to increase the training set diversity without needing more labeled data. These approaches have been mostly employed in computer vision research, where techniques such as flipping, color alteration, and cropping of images are standard techniques in model training. However, in NLP tasks, the generation of augmented data is less obvious due to the discrete nature of texts [121]. Despite this, several data augmentation techniques have been proposed in the literature for NLP, ranging from approaches based on rule-based manipulation of the available data [122] to approaches based on interpolation techniques and generative models that introduce more diversity to the training data [123]. Furthermore, in NLP, data augmentation techniques can be performed at different levels, being categorized within the data space, where techniques focus on the input data, and within the feature space, where techniques focus on processing the representations of the input data (see Figure 2.11).

Within the feature space, noise insertion methods are based on introducing controlled perturbations to the data, while interpolation methods are based on generating new examples by gradually mixing existing data values [124], [125]. On the other hand, within the data space, noise insertion at the character and word level is based on introducing deliberate variations in existing characters or words [126]. Replacement synonyms are based on substitutions from thesaurus or terms with semantic similarity provided by contextual

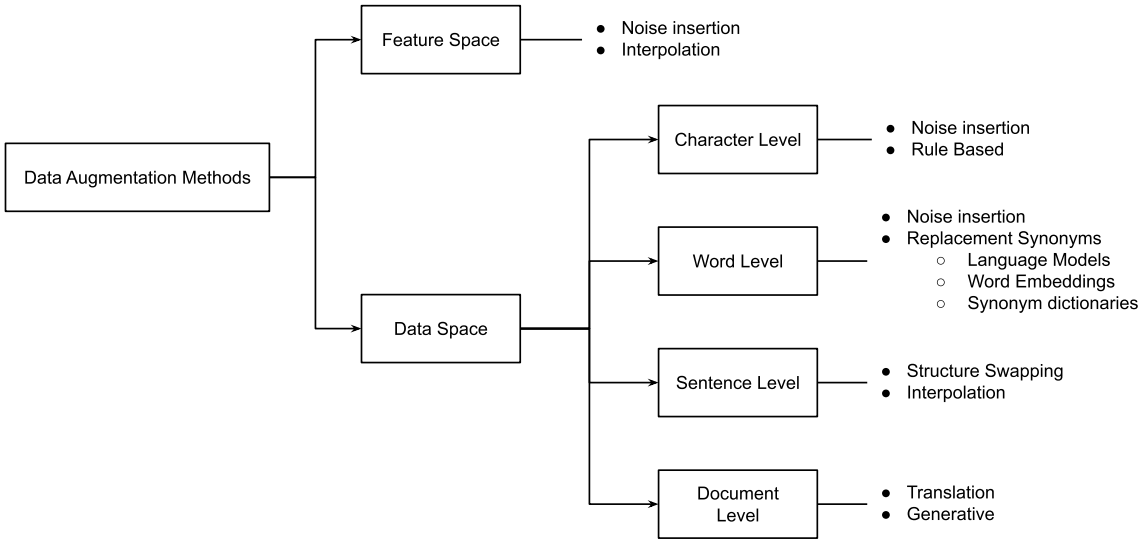


Figure 2.11: Overview of the taxonomy of data augmentation techniques.

language or word embedding models [104], [122], [127]. At the sentence level, structure swapping is based on rearranging the syntax or arrangement of words in the sentence, while interpolation is based on combining fragments of different sentences to create new structures [15]. Finally, at the document level, translation is based on back-translating a text to obtain substantial changes (also used at the sentence level), while generative approaches are based on generating new content from language models or networks trained for that purpose [128].

However, one of the challenges common to all approaches is the need to maintain a distribution of the augmented data in an equilibrium where such distributions must not be too similar but not too different from the distribution of the original data. Suppose the augmented data are too similar to the original data. In that case, these new data may not bring diversity to the training set, and the resulting model may overfit and not generalize well to the new data. On the other hand, if the augmented data is very different from the original data, the model may be unable to capture the features relevant to the specific task and make accurate predictions. Finding this balance can be challenging and may require combining different data augmentation techniques and data selection strategies to ensure the final model is robust and accurate. Moreover, in addition to equilibrium between data distributions, automatic data augmentation in NLP faces the difficulty of maintaining the quality of labeling, as small changes in the input data can change the semantics of the texts and, consequently, lead to mislabeling [129].

Chapter 3

Related Work

This Chapter describes an overview of NLP task of SA and NER, focusing on supervised and weakly supervised approaches. Furthermore, this chapter describes different systems proposed for these tasks in various corpora and, finally, in the clinical domain.

3.1. Standardized Datasets

The increasing availability of labeled datasets has provided a great tool to advance the field of NLP research, and a breakthrough in research has been achieved [83]. Therefore, describing the different datasets used for the different techniques and approaches is important. This section summarises and overviews some main domain- and task-specific standardized datasets.

3.1.1. Datasets for Sentiment Analysis

The datasets used for the SA task consist of texts associated with a certain opinion or emotion expressed about a target. Training and testing on these datasets allow fitting predictive models that obtain information about a certain point of view of the text. In the datasets used for SA, the categorization of texts can be binary (positive, negative categories) or multi-class (positive, neutral, negative or multi-level). Below is a summary of some of the main datasets used for SA and a table with an overview of them (see Table 3.1).

Stanford Sentiment Treebank (SST) [29] consists of an extension of Movie Review (MR) [130], forming a dataset of movie reviews where each review corresponds to a sentence. The SST dataset consists of two versions: SST-2, set with binary labels consisting of 6,920 sentences for the training set, 872 for the validation set and 1,821 for the test dataset, all with a balanced class distribution. SST-5, where each sentence can be labeled among five classes (very negative, negative, neutral, positive, very positive) and consists of 8,544 sentences for the training dataset, 1,101 for the validation dataset and 2,210 sen-

tences for the test dataset, where the classes distribution is slightly unbalanced, with the positive and negative ones being the most represented.

Internet Movie Database (IMDB) [131] consists of a dataset of film reviews for binary sentiment classification. Unlike SST, where the labels correspond to sentences, IMDB's labels are at the review level, constituting a longer text length than the SST examples. IMDB contains the same amount of labeled reviews for each class and comprises more than 50,000 annotated examples (reviews).

Yelp [132] is a dataset of reviews of local businesses, such as restaurants and shops. The Yelp dataset is the result of the Yelp Dataset Challenges datasets in 2013, 2014 and 2015 [133], and comprises two categories. On Yelp.P, reviews have a binary sentiment rating (negative and positive) and include 560,000 training and 38,000 test examples (reviews), both sets with a balanced class distribution. On Yelp.F, reviews are labeled among five classes from 1 (negative) to 5 (positive), including 650,000 training and 50,000 test examples, both sets also with a balanced class distribution.

Amazon Reviews (AM) [134] is a dataset of reviews on various products, such as electronics, books, and toys, collected from Amazon's website. It comprises more than 7.6 million reviews and associated metadata, such as product category, review date, and usefulness score. This dataset is divided into two categories: AM.P, which includes 4 million reviews labeled with binary classification, with 3.6 million in the training set and the rest in the test set. AM.F, comprising 3.65 million reviews categorized across five distinct classes, ranging from 1 (negative) to 5 (positive), with 3 million in the training set and the remaining reviews in the test set. Both AM.P and AM.F exhibit balanced class distributions.

Multi-Perspective Question Answering (MPQA) [135] is a dataset containing more than 10,000 sentences extracted from English news and opinion papers, where sentiments on different aspects and targets, either entities or specific events or topics on which an opinion is expressed in the text, are labeled in eight classes representing the authors' sentiment: positive, negative, both, neutral, uncertain-positive, uncertain-negative, uncertain-both, and uncertain-neutral. Although MPQA does not include labels at the document or sentence level, the annotations were employed by the authors in [136] to derive, at the sentence level, 3,311 positive and 7,293 negative examples using the previous annotations denoting positive or negative polarities.

SemEval 2017 Task 4: Sentiment Analysis in Twitter (SS-Twitter) [30] consists of a dataset of tweets annotated on a binary class, three classes (Positive, Neutral, Negative) and five classes (Strongly Positive, Weakly Positive, Neutral, Weakly Negative, Strongly Negative) and information about their topic. The dataset contains tweet texts in English and Arabic and consists of more than 50,000 tweets for training and more than 12,000 tweets for testing in English, while it includes more than 3,000 and 6,100 tweets in Arabic for training and testing, respectively. The class distribution within both the English and Arabic datasets reveals a notable imbalance, with the positive class being the most

prevalent in binary classification. Conversely, the neutral category emerges as the most abundant for the three-class rating, while in the context of the five-class rating, neutral and positive classes have a significant presence.

In the biomedical domain, Drug Review Dataset (DRD) [47] contains around 215,000 drug reviews collected from the Drugs.com website[1], a widely used online resource for detailed information on drugs and pharmaceuticals, as well as for access to reviews and user experiences on different drugs. Each review includes information such as the name of the drug, the medical condition treated, and the degree of patient satisfaction with the drug, along with the review itself. This dataset is labeled in 10 different classes for patient satisfaction and medicine efficacy, from 1 (most negative) to 10 (most positive), making it suitable for SA and data mining tasks. However, the class distribution is unbalanced, with those with more polarised scores (1, 8, 9, and 10) predominating.

Datasets	**Classes**	**Avg. Length**	**Dataset size**	**Language**	**Sources**
MR	2	20	10,662	English	[130]
SST-5	5	18	11,855	English	[29]
SST-2	2	19	9,613	English	[29]
IMDB	2	294	50,000	English	[131]
Yelp.P	2	153	598,000	English	[132]
Yelp.F	5	155	700,000	English	[132]
AM.P	2	91	4,000,000	English	[134]
AM.F	5	93	3,650,000	English	[134]
MPQA	2	25	10,605	English	[135]
SS-Twitter.E2	2	19	26,693	English	[30]
SS-Twitter.E3	3	19	62,617	English	[30]
SS-Twitter.E5	5	19	43,011	English	[30]
SS-Twitter.A2	2	19	4,413	Arabic	[30]
SS-Twitter.A3	3	19	9,455	Arabic	[30]
SS-Twitter.A5	5	19	9,455	Arabic	[30]
DRD	10	89	215,063	English	[47]

Table 3.1: Overview table of datasets for SA. The size of each dataset, the average length (number of tokens) of the texts, the number of classes, and the language of the sentences are included.

It must be noted that there is a wide variety of datasets for SA, and new ones are being generated continuously [137]–[139]. Since a complete review of all of them would be beyond the scope of this work, the section has only proposed a review of those datasets most commonly used in the SA task [140], [141].

[1]https://www.drugs.com/

3.1.2. Datasets for Named Entity Recognition

The datasets used for NER consist of texts with entities belonging to predefined semantic types. These entities are composed of words, sets of words or phrases that refer to particular elements present in the text, such as names of people, places, organizations and other relevant concepts depending on the domain. For example, a text may contain the entity "Amoxicillin" marked as a drug. Training and testing on these datasets allow researchers to assess the performance of their systems in accurately identifying and classifying named entities within text documents, which involves identifying their boundaries within the text and assigning them the corresponding semantic label [142]. Below is a summary of some of most popular datasets used for NER and a table with an overview of them (see Table 3.2).

The 2003 Conference on Computational Natural Language Learning shared task dataset (CoNLL-2003) [143] is a widely used reference dataset in NER. It consists of text data from English news articles from the Reuters Corpus[2], annotated with named entity labels (a German language dataset is also available). The dataset consists of a training set of 14,987 sentences, a validation set of 3,466 sentences and a test set of 3,684 sentences. The named entities are classified into four categories: persons (PER), organizations (ORG), places (LOC), and miscellaneous names (MISC). Specifically, the training set comprises 7,140 entities categorized as LOC, 3,438 as MISC, 6,321 as ORG, and 6,600 as PER. Meanwhile, the validation set contains 1,837 entities under LOC, 922 under MISC, 1,341 under ORG, and 1,842 under PER. Finally, the test set encompasses 1,668 LOC entities, 702 MISC entities, 1,661 ORG entities, and 1,617 PER entities. Thus, the distribution of entities among these sets is generally balanced, except for MISC-type entities, which are less represented across all sets. Furthermore, the entities are labeled following the IOB tagging scheme (I = inside an entity, O = outside an entity, B = beginning of an entity) [144].

The 2017 Semantic Evaluation Task 10 dataset (SemEval-2017 Task 10) [145] consists of texts extracted from scientific articles in English from ScienceDirect[3] open access publications and has been annotated with key phrase labels. The training set contains 350 examples, the validation set consists of 50, and the test set contains 100. Each example comprises a sequence of words containing the relevant key phrases labeled in three categories: PROCESS (including methods and equipment), TASK, and MATERIAL (including corpora and physical materials). The training set comprises 2,992 entities categorized as PROCESS, 1,132 as TASK, and 2,608 as MATERIAL. Meanwhile, the validation set contains 458 entities under PROCESS, 137 under TASK, and 567 under MATERIAL. Finally, the test set encompasses 954 PROCESS entities, 194 TASK entities, and 904 MATERIAL entities. Thus, the distribution of entities among these sets is generally balanced, except for TASK-type entities, which are less represented across all sets. Furthermore, the

[2]https://www.reuters.com/researchandstandards/
[3]http://www.sciencedirect.com/

dataset is provided annotated in BRAT format[4] [146] but without any tagging scheme such as IOB or similar.

In the clinical domain, EHRs are among the richest resources for epidemiological studies to plan and evaluate disease identification and prevention strategies [147]. However, researchers have to manually review a large amount of information, which is very costly and time-consuming [148]. Thefore, automatic recognition of entities or concepts within a text could help to automatically process the information in EHRs in a more efficient way. Providing systems capable of automatically identifying and classifying key entities or phrases of interest in these records is a vital task [149], as it enables the automatic extraction of diagnoses, demographic data, laboratory results, and treatments, among others, accelerating the identification of disease patterns, outbreaks, or epidemiological trends. Therefore, datasets for the task of NER in the clinical domain have also been constructed [42], [44], [150]–[152]. Below we describe some of the main datasets for NER in the clinical domain.

The 2010 Informatics for Integrating Biology and the Bedside (i2b2-2010) [42] task dataset comprised 826 clinical care discharge summaries provided by Partners Healthcare, Beth-Israel Deaconess Med Center (BIDMC), and University of Pittsburgh Med Center (UPMC). The data were divided into training and test sets, in which 349 annotated examples were used as training and 477 for testing. In the context of NER, each example consists of a sequence of words whose entities have been labeled in three categories: referring to a health problem (Problem), to a treatment (Treatment), and tests performed (Test). The training set comprises 11,967 entities categorized as Problem, 7,368 as Test, and 8,496 as Treatment. On the other hand, the test set contains 18,550 entities under Problem, 12,899 under Test, and 13,560 under Treatment. Thus, the distribution of named entities among these sets is unbalanced, with Problem entities being over-represented. Furthermore, the dataset does not provide any tagging scheme such as IOB or similar.

The 2018 National NLP Clinical Challenges shared task dataset (n2c2-2018) [44] comprises 505 discharge summaries from the MIMIC-III (Medical Information Mart for Intensive Care-III) clinical care database [151]. The data were divided into training and test sets, where 303 annotated examples were used as training and 202 for testing. In the context of NER, each example consists of a sequence of words whose entities have been labeled in nine categories: the name of the drug (Drug), the strength (Strength), the drug form (Form), the dose of the drug (Dosage), the frequency of the drug administration (Frequency), the route of administration (Route), the drug intake period (Duration), the reason for prescribing the drug (Reason) and adverse drug event (ADE). The training set contains 16,225 entities of the Drug type, 6,691 of the Strength type, 6,651 of the Form type, 4,221 of the Dosage type, 6,281 of the Frequency type, 5,476 of the Route type, 592 of the Duration type, 3,855 of the Reason type, and 959 of the ADE type. Meanwhile, the test set contains 10,575 entities of the Drug type, 4,230 of the Strength type, 4,359 of the Form type, 2,681 of the Dosage type, 4,012 of the Frequency type, 3,513 of the

[4]https://brat.nlplab.org/index.html

Route type, 378 of the Duration type, 2,545 of the Reason type, and 625 of the ADE type. Therefore, the distribution of named entities among these sets is unbalanced, with an over-representation of Drug entities and an under-representation of Duration and ADE entities. The annotated dataset is also provided in BRAT format without any tagging scheme such as IOB or similar.

Moreover, although most of the resources in the health domain are in English, there are recent efforts to create datasets in other languages. The eHealth Knowledge Discovery Challenge task dataset at IberLEF 2019 (ehealthkd-2019) [152] comprises 1,000 sentences of electronic health documents written in Spanish. The data were divided into training, validation, and test sets, where 600 annotated sentences were used for training, 100 for validation, and 300 for testing. In the NER context, each sentence consists of a sequence of words where an entity can be labeled in four categories: term, concept, idea relevant to the knowledge domain of the sentence (Concept), process or modification of other concepts (Action), function or filter of another set of elements (Predicate), and textual element referring to a concept (Reference). The training set contains 2,381 entities of the Concept type, 976 of the Action type, 330 of the Predicate type, and 131 of the Reference type. On the other hand, the validation set contains 368 entities of the Concept type, 167 of the Action type, 45 of the Predicate type, and 24 of the Reference type. Finally, the test set contains 1,162 entities of the Concept type, 517 of the Action type, 161 of the Predicate type, and 58 of the Reference type. Thus, the distribution of named entities among these sets is unbalanced, with the Concept entities being the most represented and the Reference entities the least. The annotated dataset is also provided in BRAT format without any tagging scheme such as IOB or similar.

The Medical Document Anonymization (MEDDOCAN) dataset [153] comprises synthetic clinical cases written in Spanish with Protected Health Information (PHI). The anonymization datasets are particularly relevant for research, as they can assist in training models for de-identification or exhaustive removal/substitution of PHI-related items, a precondition for accessing clinical records outside hospitals for NLP research tasks. This dataset comprises 1,000 clinical case studies, where 500 documents are used for training, 250 for validation, and 250 for testing. Specifically, the dataset contains 22,795 entities in 22 different categories, including the patient's name, hospital, and telephone number (for more details on the categories and the number of entities in each, see [153]). The distribution of named entities according to each type is unbalanced, with those referring to territory, dates, patient name, and age being the most represented. In contrast, entities referring to institutions, health centers, professions, patient identifiers, or phone numbers are the least represented. The annotated dataset is provided in BRAT format without any labeling system such as IOB or similar.

Datasets	Classes	Lenght (Avg)	Dataset size	Entities	Language	Sources
CoNLL03	4	15	22,137	35,089	English	[143]
SemEval-2017 (Task 10)	3	160	500	9,946	English	[145]
i2b2-2010	3	880	826	72,840	English	[42]
n2c2-2018 (track 2)	9	1,893	505	83,869	English	[44]
ehealthkd-2019 (scenario 2)	4	15	1,000	6,320	Spanish	[152]
MEDDOCAN 2019	22	417	1,000	22,795	Spanish	[153]

Table 3.2: Overview table of datasets for NER. The size of each dataset, the average length (number of tokens) of the texts, the number of entity classes, the number of entities, and the language of the texts are included.

3.2. Evaluation Metrics

In order to compare the different systems and draw a conclusion about them, several common evaluation metrics for the text classification and NER tasks need to be defined [154]:

- True Positives (TP): the number of true observations correctly identified as positive by the model.
- True Negatives (TN): the number of false observations that were correctly identified as negative by the model.
- False Positives (FP): the number of false observations that were incorrectly identified as positive by the model.
- False Negatives (FN): the number of true observations incorrectly identified as negative by the model.
- Accuracy (Acc): refers to the ratio of true observations among the total observations. It is calculated as Equation 3.1.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

- Precision (P): refers to the proportion of true positives among the total number of observations predicted as positives. It is calculated as Equation 3.2.

$$P = \frac{TP}{TP + FP} \tag{3.2}$$

- Recall (R): refers to the proportion of true positives among the total number of true observations. It is calculated as Equation 3.3.

$$R = \frac{TP}{TP + FN} \tag{3.3}$$

- F-measure (F1): refers to the harmonic mean of the model's precision and recall. It is calculated as Equation 3.4.

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{3.4}$$

The P, R and F1 metrics can be extended to multi-classification problems using the micro-averaged and macro-averaged versions [155], [156]. The macro-averaged version shows the mean of the values obtained for each class independently. In the micro-averaged version, the contributions of each class are summed and then the average metric is calculated. Since, in multi-class classification problems, the micro-F1 is calculated globally by counting the total of TP, FN, and FP, this metric will give the following results[5]: $F1 = P = R = Acc$. The use of both micro- and macro-averaged metrics can address complementary information in the evaluation of systems. While the micro-average provides information on overall performance independently of classes, the macro-average will give equal weight to all classes, providing information on the performance of systems on sets with an unbalanced class distribution.

3.3. Supervised approaches for Sentiment Analysis

In recent years, several researchers have made advances in SA. In contrast to traditional models, which require high cost to design features for each task, or high-dimensional BoW representations that do not consider word order, more recent approaches have used embedding models to represent texts using word vectors in which each word is associated with a fixed-length vector. These vectors are usually of sizes between 300 and 30 dimensions per word, with initialization of random values, which will be adjusted during training or pre-trained on other tasks (e.g., neural language model [157], word2vec [72], Glove [73], among others).

As mentioned in section 2.2, recursive networks can learn the semantic features of words and the syntactic structure of texts. However, they have a prior dependency on knowing the tree structure employed by the network. To deal with this problem, in [158], the authors propose to use the representation of the subtrees to add a reconstruction error in an autoencoder configuration. Specifically, they propose recursive autoencoders (RAE) for SA, which allow the tree structure to be obtained when it is not previously available.

[5]https://scikit-learn.org/stable/modules/model_evaluation.html#multiclass-and-multilabel-classification

For this purpose, an auxiliary task is carried out in which the reconstruction error is minimized based on the Euclidean distance between the original entries to a node and the reconstruction of these entries from a second shared W-weight matrix.

$$
\begin{aligned}
[u_l'; u_r'] &= W^{(2)}p + b^{(2)} \\
E_{rec}([u_l; u_r]) &= \frac{1}{2}\|[u_l; u_r] - [u_l'; u_r']\|^2
\end{aligned}
\tag{3.5}
$$

This is done recursively over all neighboring vectors selecting those with the lowest error. At the same time, RAE uses a weighted sum of the reconstruction error and the Cross-Entropy (CE) error for the SA task, using the sentence representation at the root node for sentence-level SA (see Figure 3.1).

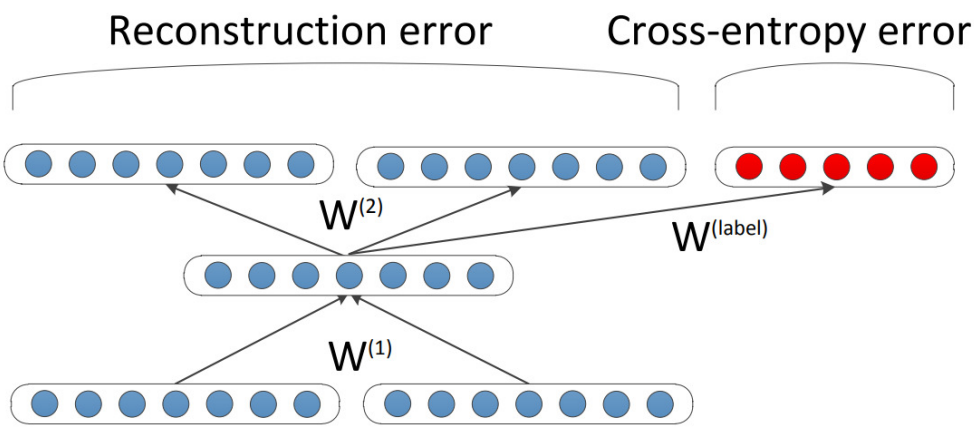


Figure 3.1: Illustration of a non-terminal RAE tree node. The red nodes consist of the softmax layer for label distribution prediction. Image retrieved from Socher et al. 2011 [158]

The authors of [158] compared RAE with other rule-based approaches and traditional algorithms such as BoW, DT and Conditional Random Fields (CRF), among others. The comparison was carried out on the MR and MPQA datasets (see section 3.1.1), obtaining a result of 77.7% and 86.4% accuracy, respectively. The improvement concerning the other approaches did not exceed one percent accuracy in some cases, giving the advantage of not requiring a predefined tree structure a priori.

On the other hand, the authors of [159] used RAE to conducted a comparative study between different deep learning models. The evaluation was carried out using the SST dataset, where RAE achieved 82.4% accuracy in binary classification but only 43.2% in the 5-class sentiment classification problem. The problem may be that the model only uses a shared weight matrix for the linear operation with the concatenation of the subtree representation, making it difficult to capture all the semantic-syntactic information at words or longer sentence level [160].

In [160], the authors present a ReNN model capable of learning the representation of compositional vectors that capture information for phrases and sentences of arbitrary length called Matrix-Vector Recursive Neural Network (MV-RNN). In contrast to the abovementioned approaches, the authors use a vector and a matrix per node to capture such information. Specifically, the vector focuses on capturing the inherent semantics, while the matrix captures how the meaning changes depending on the context of neighboring words. Therefore, the matrix of the first word is linearly combined with the vector of the second word and vice versa, thus capturing contextual information between the two (equation 3.6).

$$h_i = f\left(W\begin{bmatrix}Ba\\Ab\end{bmatrix}\right), H_i = f\left(W_M\begin{bmatrix}A\\B\end{bmatrix}\right) \tag{3.6}$$

Where A and B are single-word matrices, d is the dimension of the vectors and $W \in \mathbb{R}^{dx2d}$ is the shared matrix of weights that transforms the representation of both words back to the same dimensional space of the word vectors. The second part of the equation performs a linear combination between both matrices with $W_M \in \mathbb{R}^{dx2d}$ to obtain the matrix representation of the subtree.

The authors of [160] evaluated MV_RNN on the MR dataset, obtaining a result of 79% accuracy. The improvement over the RAE recursive network model was 2%. On the other hand, in [29], the authors evaluated MV_RNN on the SST dataset, obtaining results of 82.9% accuracy for the binary classification against 44.4% for the 5-class sentiment classification. However, the model uses one more matrix for each of the word representations or subtrees of the network, considerably increasing the number of parameters, which are also dependent on the vocabulary size.

Considering the above, Recursive Neural Tensor Networks (RNTN), proposed by [29], aim to help reduce the problem of MV-RNNs without renouncing the ability to capture contextual information between words or nodes of neighboring subtrees. For this, the authors replace individual matrices for each word or subtree and employ a tensor shared between all nodes. The equation used can be seen in Figure 3.2, where b and c are the word vectors, and $V \in \mathbb{R}^{2dx2dxd}$ is a tensor defining multiple bilinear forms.

The authors conducted a comparative study between different models of traditional approaches such as NB and SVM, both with BoW models with bigram characteristics. They also compared the approach with MV-RNN. The comparison was carried out on the SST dataset, obtaining an accuracy of 85.4% and 45.7% for the binary classification problem and the 5-class sentiment problem, respectively.

One of the main problems when dealing with ReNN is its reliance on parse trees that are generally known a priori and introduce bias into the models. Tree structures do not always capture the complexity of sentences, as sometimes a hierarchical structure cannot accurately represent relationships. Therefore, deep learning has employed other approaches to mitigate the bias induced by external parse trees. While all other approaches

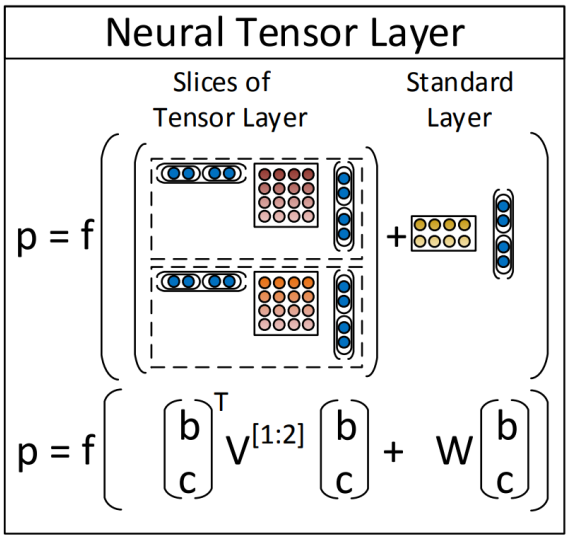


Figure 3.2: Overview of an RNTN layer. Each dashed box represents a slice of the last dimension of the tensor. Each captures the words' influence in their parent nodes. Image retrieved from Socher et al, 2013. [29]

can induce a hierarchical structure, these are internal to the neural networks [161].

In [161], the authors present Dynamic Convolutional Networks (DCNN). A type of network that performs 1-dimensional convolutional operations along the input sentence $s \in \mathbb{R}^s$. Specifically, they use vectors of weights $m \in \mathbb{R}^m$ as convolution filters that operate on the feature associated with the i-th word of s to obtain another "sentence" c (see equation 3.7). In this way, the network can capture local information from adjacent words.

$$c_i = m^T s_{i-m+1:i} \tag{3.7}$$

Once the resulting matrix is obtained, they reduce the features' dimensionality by keeping only the k maximum values of each of the features of f c_i (k-Max Pooling). The value of k is chosen depending on the number of convolutional layers that the model stacks and the specific layer to be pooled. The idea is to allow the model a smooth extraction of higher order and larger features.

After applying the pooling operation to the convolution result, a non-linear function is applied, and the process is repeated for each convolution layer of the model. Finally, the authors add a layer to the model to reduce the dimensionality of the features of the "sentences", consisting of the sum of every second row of the resulting features. The resulting matrix feeds a fully connected layer to obtain the probability distribution (see Figure 3.3).

The authors evaluated the DCNN model on the SST dataset. They used a random initialization of word embedding of dimension 48 and two convolutional layers with the hyperbolic tangent (tanh) as non-linear activation functions. They obtained 86.8% and

48.5% accuracy for the binary and 5-class sentiment classification problems, respectively.

However, the DCNN architecture performs one-dimensional convolutional operations overtime on the input sequence and finally over the resulting features, which makes it a computationally complex network. Therefore, the authors of [162] propose TextCNN, a convolutional network that employs a single convolutional layer with 2-dimensional filters, which slides over the time of the sentences s using the entire feature map $s_i \in \mathbb{R}^k$, where k is the number of features defining the i-th word vector of s. The TextCNN model uses a parameterisable m number of filters $w \in \mathbb{R}^{hk}$, where h indicates the filter window size, to obtain m vectors $v \in \mathbb{R}^{s-h+1}$, of which only the feature with the highest value is retained (Max-over-time pooling) over the time of the sequence. The resulting features are concatenated to obtain a vector of dimension m on which a fully connected layer is applied with a dropout regularization. The regularization method randomly sets a portion of the elements of the hidden layer to zero. Finally, a softmax layer is fed to obtain the probability distribution.

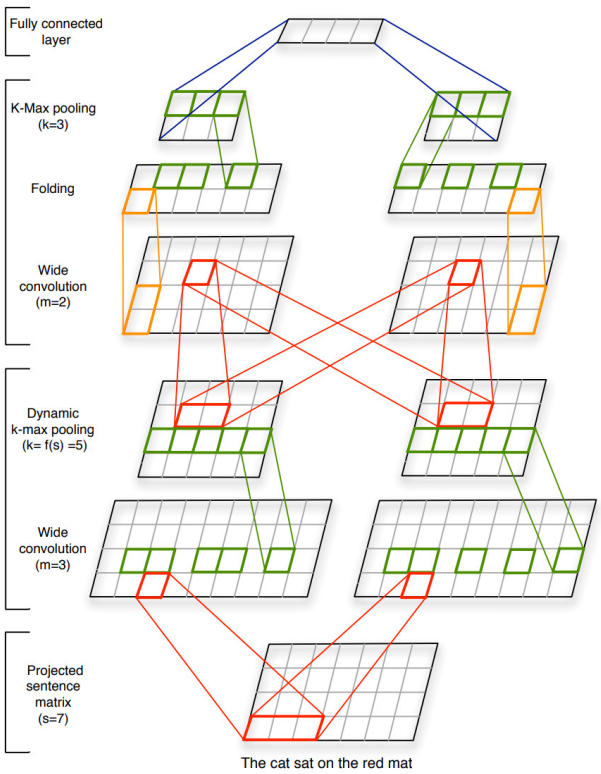


Figure 3.3: Overview of a DCNN network with a filter window size of 3 and 2, respectively. The (dynamic) layers have k-values of 5 and 3, respectively. Image retrieved from kalchbrenner et al, 2014. [161]

To evaluate the TextCNN model, the authors use a non-linear activation function of rectified linear units (Relu) [95], 100 h-window filters of sizes 3, 4, and 5, and a dropout of 0.5. In addition, combinations of different types of randomly initialized and pre-trained word embeddings were used separately and combined in a multi-channel configuration where each filter operates on multiple word embeddings and the results are summed to

give the resulting v-vector. Specifically, they employed 300-dimensional word embedding models, with word2vec [72] being the pre-trained model on Google News. In their setup, they evaluated the effect of transfer learning by the pre-trained word embeddings and proposed to use the multi-channel setup to train with static and non-static word embedding combinations during training.

The TextCNN authors evaluated the model's performance on various datasets such as MR, SST and MPQA, among others. In MR and SST-2, they obtained an accuracy of 81.5% and 88.1% in binary classification, respectively, the best result for MR being provided by the model with single-channel configuration with non-static word2vec. In contrast, for SST-2, the best result was provided by the model with multi-channel configuration. On the other hand, in SST-5, they obtained an accuracy of 48% for 5-class sentiment, a decrease of 0.7% in comparison to DCNN. Finally, in MPQA, they obtained a result of 89.6% accuracy using static word2vec.

One of the main advantages of the different CNN-based approaches is their ability to learn local features in a similar way to n-grams by considering the filters' sliding windows. All this is performed in parallel without the dependence on previous data in the input sequence. Despite this, CNNs alone have problems computing long dependencies within long sequences. Therefore, to learn long dependencies and decrease the bias of external parse trees, several works were performed with a type of recursive network called recurrent networks. Recurrent networks (RNN) [86] are a subtype of recursive networks where dependencies are not captured using a hierarchical structure but sequentially (see 2.2.3).

Within RNNs, vanishing or exploding gradient problems can be encountered when texts are too long. Thus networks such as [77], [98] are used to retain useful information and to discard irrelevant ones for a specific task, helping to mitigate the problem of forgetting information in long-term dependencies.

In [101], the authors compare different approaches based on LSTM [77] recurrent networks for the SA task, among others. To represent the words in their experiment, they used word2vec embeddings pre-trained on the Wikipedia corpus (1B words), which were adjustable during training. Also, the embedding size for each word was 64, and the output representation of each LSTM cell was of size 50 (see 2.2.3)

The first approach used as a baseline is using only one LSTM layer together with the word representation using word2vec. The approach was evaluated on different datasets, such as SST and IMDB. In SST-5, the model achieved an accuracy of 45.9% for the 5-class sentiment task, while in SST-2, it achieved 85.8%. On the other hand, on the IMDB dataset, composed of longer text sequences, the model achieved an accuracy of 88.5%. However, the authors claim that training the models by simultaneously sharing information acquired on different datasets can improve performance for specific tasks. Therefore, the authors propose a series of models that can leverage the information acquired for the similar task but on different datasets.

The best-performing model, which the authors call the Multi-task model, is based on an LSTM layer for each task in different datasets together with a Bidirectional Long Short-Term Memory (BiLSTM) layer shared by each of the other layers (see Figure 3.4). In the bidirectional layer, the output of the LSTM cells in a bidirectional configuration is denoted as $h_t^{(s)}$, being $h_t^{(s)} = \overrightarrow{h}_t^{(s)} \oplus \overleftarrow{h}_t^{(s)}$, while the output at the specific layer m is denoted as $h_t^{(m)}$. For the interaction between a specific layer and the bidirectional shared layer, a sigmoidal activation mechanism is used to allow the network to accept or reject the information that the shared layer passes to the specific layer. For this purpose, the calculation of the new state for the cells of the specific layer LSTM is calculated as follows (see equation 3.8):

$$\begin{aligned} \tilde{C}_t^m &= \tanh\left(W_c^{(m)} x_t + g^{(m)} U_c^{(m)} h_{t-1}^{(m)} + g^{(s\to m)} U_c^{(s)} h_t^{(s)}\right) \\ \text{where} \quad g^{(m)} &= \sigma\left(W_g^{(m)} x_t + U_g^{(m)} h_{t-1}^{(m)}\right) \\ \text{and} \quad g^{(s\to m)} &= \sigma\left(W_g^{(m)} x_t + U_g^{(s\to m)} h_t^{(s)}\right) \end{aligned} \tag{3.8}$$

Where W_c and U_c are weight matrices for the linear combination between the input x_t and the output of the previous cell $h_t - 1$ to calculate the current state of the cell, and g is a gate for calculating the extent to which the information of the specific layer cell m and the shared layer cell s is retained. Gate g uses its matrices to combine cell outputs and inputs W_g and U_g. While in $g^{(m)}$, the information it has to preserve comes from the output of the previous cell $h_t - 1$, $g^{(s\to m)}$ has to decide on the output of the current cell of the compared bidirectional layer.

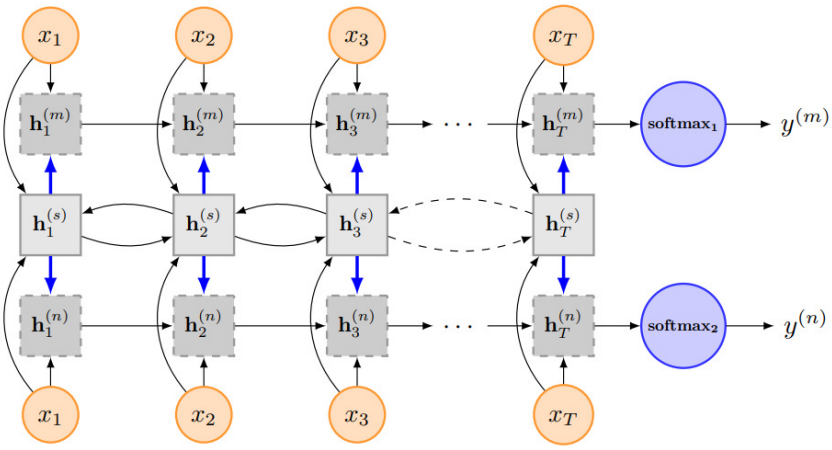


Figure 3.4: Overview of the Multi-task model with shared layer architecture. Image retrieved from Liu et al., 2016. [101]

Before the joint training, the authors pre-train the bidirectional shared layer to initialize the weights. Specifically, they train a language model to predict the next word in the tasks' texts. Once the change in the equations of the LSTM cells has been defined and the weights of the shared layer have been initialized, the model is trained for the different tasks as a whole, propagating the sum of all the individual errors for each task. Specifically, for each task, the cross-entropy is used as a loss function. Finally, the learned

weights are used to fine-tune a particular task using only the shared BiLSTM layer and the specific LSTM layer.

The new approach was evaluated on the same datasets as the previous LSTM model. In SST-5, the model achieved an accuracy of 49.6% for the 5-class sentiment task, while in SST-2, it achieved 87.9%. On the other hand, in the IMDB dataset, composed of longer text sequences, the model achieved an accuracy of 91.3%.

As we have seen up to this point, RNNs and CNNs are among the most popular architectures for NLP tasks. RNNs can learn from the context of sequences by considering the dependencies, while CNNs can learn local information in sequences. The two capabilities are not mutually exclusive, and several works have been done considering hybrid combinations of these two types of architecture.

In [163], the authors present the BLSTM-2DCNN model, a model capable of obtaining context information using a bidirectional LSTM network, which provides a feature matrix h, where h_t is the output of each of the LSTM cells formed as follows: $h_t = \overrightarrow{h}_t \oplus \overleftarrow{h}_t$. Due to a bidirectional configuration, each h vector contains context information for all the words in the sequence. These features can be treated as the input to convolutional networks to obtain contextual information in the form of n-grams. The authors also propose using 2-dimensional convolutions and max poolings, arguing that the features of the vectors are not mutually independent and that using 1-dimensional convolutions and pooling could destroy the structure of feature representations.

The authors employ as input a pre-trained word embedding model that provides 300-dimensional word representations. Specifically, the GloVe embeddings [73] trained on 6 billion words from Wikipedia and Gigaword 5. They employ a hidden LSTM unit size of 300, 100 convolutional filters with a window size of (3,3) and a max-pooling of (2,2). As a regularisation method, they use dropout operations after the word embeddings, after the BiLSTM layer and before the softmax layer. The model was evaluated using several datasets, including SST and MR. Concretely, the authors obtained a 52% accuracy on the SST-5 dataset for 5-sentiment classes, while for MR and SST-2, they obtained 82.3% and 89.5%, respectively, outperforming models such as Multi-task.

As can be seen, the different approaches described are tested on datasets in various domains, but the pharmaceutical has received much less attention. Early work on SA of drug reviews mainly employed rule-based [164] systems and sentiment lexicons (such as SentiWordNet [165]) to detect a given pharmaceutical review's overall polarity (positive or negative). Later approaches employed different machine learning algorithms applying feature engineering to the data [47], [80], [81], [166], [167]. However, deep learning approaches have been less investigated in this domain. As mentioned above, since using CNNs, RNNs and hybrid combinations thereof for SA has gained significant relevance, several works have started to apply it in the pharmacological domain.

In [168], the authors propose employing a combination of deep learning models for SA on drug reviews in positive, neutral, and negative classes. The 3W3DT model consists

of an assembly of three deep learning models and one traditional machine learning model, each trained on drug reviews. Each model performs classification individually, and subsequently, the most confident model is selected to classify the review. For each model, the greater the distance between the results of the three output neurons of the softmax layer output, the greater the confidence of the answer. Therefore, in order to obtain the confidence score, the authors use the following equation: $Cf(M) = |o_1, o_2, o_3|$ where M is a model and o_i represents the probability distribution of each model softmax layer. Once each model's score has been calculated, the selected model will be $\text{argmax}_{0<i<4} Cf(M)$.

For the 3W3DT model, the authors use three deep learning models, a GRU recurrent network (see 2.2.3), a CNN, and a hybrid model called 3CRNN. The 3CRNN model consists of concatenating the features of 3 convolutional layers with max-pooling together with the h representation provided by the output of a GRU layer. The concatenation of features with local and contextual information feeds a fully connected layer followed by a softmax. The authors do not specify the number of filters used nor the size of H in the GRU layers. However, they use filters of window sizes 2 and 3.

Once the model architecture is defined (see Figure 3.5), the model is trained and evaluated on the DRD dataset. Preprocessing of the data includes removing stop words and normalizing the drug name using a personalized product name corpus made by intersecting the Google 20K most common words. Moreover, the authors use the following thresholds, in the same way as [47], to group the drug review ratings into positive, neutral and negative classes: negative (class 0, rating<=4), neutral (class 1; rating at [5,6]) and positive (class 2; rating>=7).

In a comparative study of the 3W3DT model, the authors trained the model in conjunction with traditional algorithms such as NB, DT, Random Forest (RF) and K-Nearest Neighbours (KNN). The best performance was obtained using NB and the deep learning models mentioned above. Specifically, the best 3W3DT model obtains an F1 of 84.80%, 54.38% and 92.81% for the positive, neutral and negative classes, respectively. This result provides a macro-F1 of 77.33% and a weighted average of 87.35% due to the unbalanced data.

In other works, such as [94], the authors use a CNN network for SA of drug reviews in the DRD dataset based on the results obtained by CNN networks in text classification tasks such as [169]. In this work, a convolution layer with 64 filters of window sizes 2, 3 and 5 is proposed with a concatenation of the features obtained after max-pooling to classify the reviews into the initial 10-sentiment classes of the dataset. The authors obtained a macro- and micro-f1 of 66.72% and 61.18%, respectively. This work gave rise to a future research [23], which is part of this Thesis and is described in detail in the later sections (see 4.1). This work compared different combinations of hybrid models, which employ BiLSTM and CNN layers to extract local and contextual features in text sequences. Furthermore, the authors perform the training and evaluation of the different combinations of architectures, both on the 10-class sentiment problem and on the same grouping for

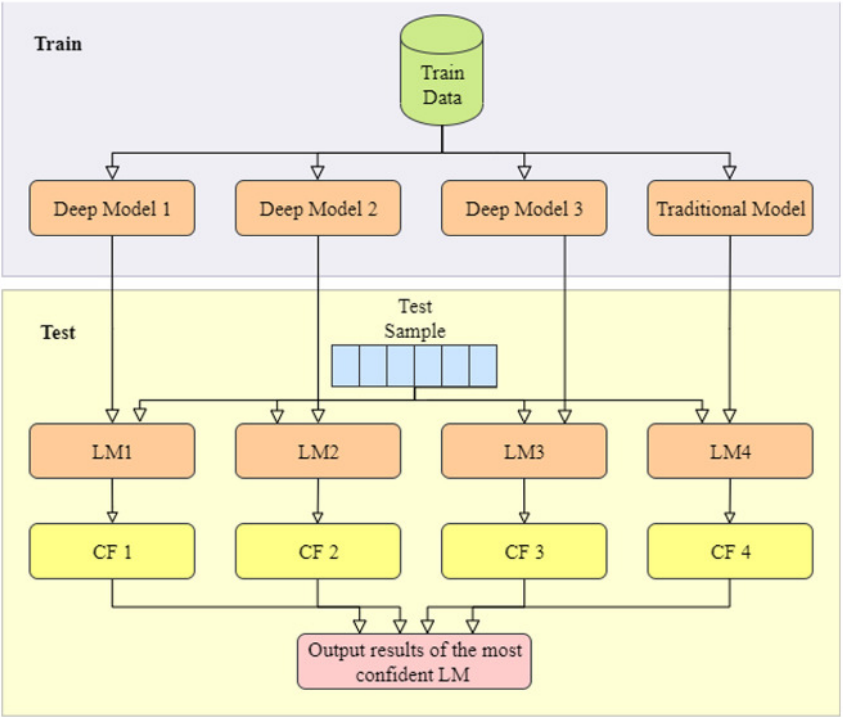


Figure 3.5: Overview of the 3W3DT model, where LM is a trained model and CF is the model confidence function. Image retrieved from Basiri et al., 2020. [168]

3-classes as originally proposed by [47]: negative (class 0, rating<=4), neutral (class 1; rating at [5,6]) and positive (class 2; rating>=7).

The best results for the 10-class sentiment problem were obtained by the LSTM+CNN model, with random initialization of embeddings. The model obtained a micro- and macro-f1 of 69.8% and 64.09%, respectively, outperforming the results of the CNN above model. In addition, for the 3-class problem, the model obtained results of 89.20% and 81.03% micro- and macro-f1, slightly outperformed by the CNN+LSTM model, which obtained 89.44% and 81.46%, respectively.

In recent years, attention mechanisms started to gain popularity despite the performance of models based on hybrid combinations of CNN and RNN architectures on their own. One of the problems with models based on the architectures described above is that they have poor interpretability, and attention mechanisms allow the creation of weights to indicate the relative importance of each word in the sequence for a given task. As a result, not only does the interpretability increase, but the model itself is able to focus its "attention" on the most relevant parts, ignoring the others and thus improving performance. Work such as [76] changed state of the art and various models based on Vaswani et al. transformers began to be widely studied in NLP tasks. Models such as BERT [54] (see 2.2.4), as well as successors of BERT such as XLNET [56] and Robustly Optimized BERT approach (RoBERTa) [55], have been studied within SA tasks obtaining state-of-the-art results.

All these language models consist of several layers of transformers. They require a

large amount of text to learn contextualized representations of words and the logic and semantic relationships between different words or sequence structures. For this reason, all of these models are pre-trained on large corpora in a non-supervised way and each employs different approaches or rescaling.

The authors of BERT employ two tasks, one based on word masking, called MLM, and the other on predicting the order of occurrence between two sentences, called NSP. Both tasks are explained in section 2.2.4 of this Thesis. The authors used a corpus based on the BooksCorpus (800M words) [170] and the English Wikipedia (2,500M words) for these tasks. Once the model has been pre-trained, it can be adjusted for other NLP tasks in a supervised way by taking advantage of the transfer of learning provided by the pre-training (see Figure 3.6).

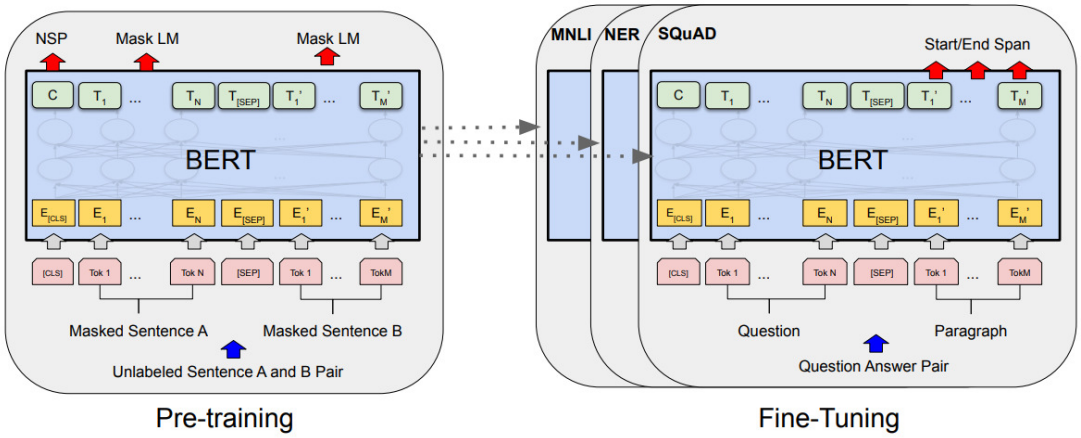


Figure 3.6: Overall pre-training and fine-tuning procedures for BERT. Image retrieved from Devlin et al., 2018. [54]

In [54], BERT is evaluated on several NLP tasks, one of which is SA. Among other datasets, it employs SST-2 to fine-tune two types of BERT models. The BERT-base model employs 12 transformer layers (L=12), a hidden size of dimension 768 (H=768) and 12 self-attention heads for the multi-head attention layers (A=12), giving a total of 110M parameters. On the other hand, a BERT-large model of 340M parameters (L=24, H=1024, A16) is fine-tuned. The BERT-base model achieved an accuracy of 93.5%, while the BERT-large model achieved 94.9%. Moreover, in [56], a BERT-large evaluation was performed on the IMDB dataset, obtaining an accuracy of 95.49%.

In [55], the authors of RoBERTa employ a BERT-large architecture (L=24, H=1024, A=16) but with several differences in the pre-training tasks on unannotated corpora. In BERT, the MSM task consists of predicting the tokens that have been masked in the pre-processing of the data, resulting in a pre-training where the masked sequences always retain the same masking position. This configuration can lead to problems where BERT assumes conditional independence between several masked tokens. Therefore, in RoBERTa, the authors perform dynamic masking, where each sequence is replicated several times with different masks. Furthermore, while [54] showed that dispensing with the

NSP task reduces corpus performance for QA tasks, the authors of RoBERTa dispense with the NSP task, showing that modifying the encoding of contiguous text segments to an encoding of complete contiguous sentences can provide similar or slightly better results in downstream tasks.

RoBERTa model was evaluated on multiple tasks and datasets, including SST-2. For this, the authors took into consideration different aspects of pre-training. Specifically, the model was pre-trained with the same corpus of unannotated data to make a fair comparison with BERT. However, they increased the batch size to 8K versus 256 in BERT-large and reduced the number of training steps to 100K versus 1M in BERT-large. For this pre-training configuration, the result obtained in the SST-2 validation dataset was 95.3% accuracy against 93.7% for BERT-large. The RoBERTa model, however, is pre-trained on an unannotated corpus larger than the one used by BERT, adding the CC-News [171], OpenWebText [172] and Stories [173] corpora resulting in a total of 160GB of text. With the latter pre-training, RoBERTa obtained an accuracy on the SST-2 test set of 96.7%, outperforming BERT-large by 1.8%.

The authors of XLNet carried out a further modification of the BERT architecture. In [56], the authors indicate that the MSM task during pre-training ignores the dependency between the masked tokens and produces discrepancies between pre-training and fine-tuning. To solve this problem, the authors propose replacing the MSM task with an autoregressive pre-training to learn bidirectional contexts, differentiating it from models such as GPT [57], the latter being a unidirectional autoregressive model (see 2.2.4).

The authors indicate that the problem with autoregressive models is that they are ineffective at modeling deeply bidirectional contexts, whereas downstream tasks may require bidirectional contextual information. Therefore, to capture such information, the authors propose to perform pre-training with all possible permutations of the factorization order of the training sequences, giving a sequence x of length T a total of $T!$ different orders. Furthermore, the authors emphasize that only the factorization order is permuted, but positional embeddings maintain the original order of the sequence. However, they also point out the need to re-parameterize the distribution of the following token to become aware of the position of the target to be predicted, e.g., $h_\theta(x_{z_{t-1}})$ denotes the hidden representation of x_z, where z is the original index in the permutation and $(t-1)$ is the time position of the permuted sequence. In contrast, $g_\theta(x_{z_{t-1}}, z_t)$ is another type of representation that additionally uses the positional index z as input. As a consequence of the above, XLNet adds two sets of hidden representations (see equation 3.9):

$$\begin{aligned} g_{z_t}^{(m)} &= Attention\left(Q = g_{z_t}^{(m-1)}, KV = h_{z_{<t}}^{(m-1)}; \theta\right) \\ h_{z_t}^{(m)} &= Attention\left(Q = h_{z_t}^{(m-1)}, KV = h_{z_{\leq t}}^{(m-1)}; \theta\right) \end{aligned} \tag{3.9}$$

Where Q, K and V are the query, key and value matrices detailed in section 2.2.4 and m refers to the m-th layer of attention. Note that in this equation, being the last attention

layer of an autoregressive model, Q becomes a vector of the dimension of the hidden state h of the last word.

Figure 3.7 shows an overview of how the different representations described above are obtained in the attention layers. In (a), it can be seen how the h representations are obtained in the same way as the standard hidden states in transformers. In (b), it can be seen how the g representations are obtained where only the contextual information of all words in $x_{z_{<t}}$ is accessed, but only the z_t position of x_{z_t}.

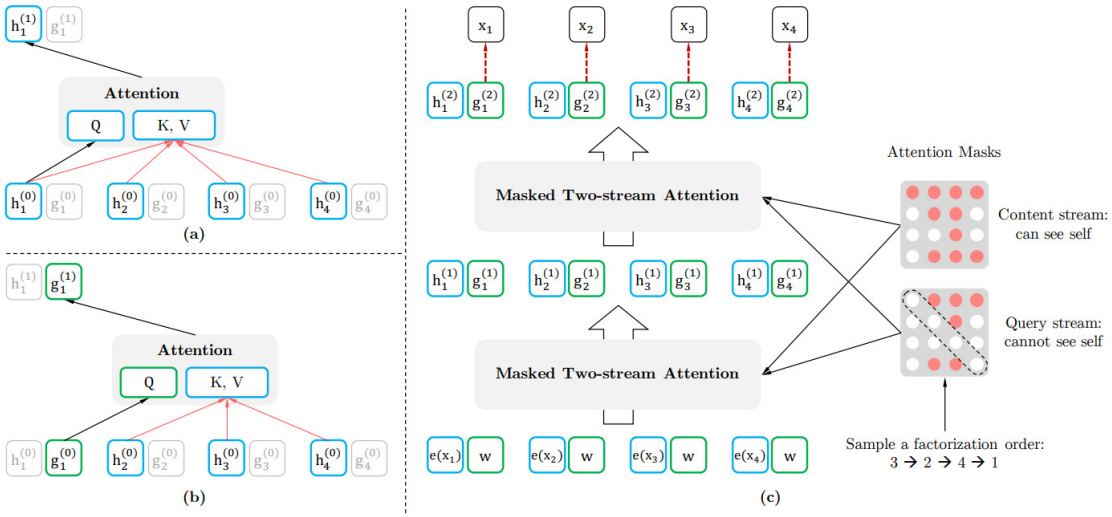


Figure 3.7: Overview of permutation language modelling training. Image retrieved from Yang et al., 2019. [56]

For pre-training, the authors employed the same architecture as BERT-large, increasing the batch size to 8192. In addition, they used the same corpora, adding Giga5 [174], ClueWeb 2012-B [175] and Common Crawl [176], resulting in a corpus of 158GB of text. During pre-training, the XLNet model aims to maximize the likelihood under the forward autoregressive factorization by replacing the context representation $h_\theta(x_{1:t-1})$ with $g_\theta(x_{z_{1:t-1}}, z_t)$. On the other hand, the authors do not perform the NSP task but employ a relative sequence membership encoding while encoding the input embeddings. The authors evaluated their pre-trained model for the downstream tasks on multiple datasets, including SST-2 and IMDB. Specifically, XLNet achieved an accuracy of 97.1% on the SST-2 test set and 96.80% on the IMDB set.

Once different pre-trained transformer-based models were seen, as well as their ability to be fine-tuned to downstream tasks providing state-of-the-art results, several research works were carried out for SA in drug reviews. Specifically, the description of several methods and their analyses are described in this Thesis in the subsequent sections, A being one of the first works to apply pre-trained transformer-based models to the SA task in drug reviews.

The authors of [23], employed a pre-trained BERT model with an architecture consisting of only four layers of transformers called BERT-small [177]. This model allows

fine-tuning to be carried out without the need to adjust many parameters and, therefore, requires less computational power. In addition, BERT-small provides the ability to obtain word representations with contextual information, unlike the static embeddings used in previous approaches. However, due to the model's dimensions, a BiLSTM layer is fed with these contextual embeddings to subsequently use a fully connected layer and a softmax for the probability distribution. The model (BERT-s+BLSTM) was trained and evaluated on the DRD dataset using the three sentiment classes used by [47]: negative (class 0, rating<=4), neutral (class 1; rating in [5,6]) and positive (class 2; rating>=7), and obtained a micro- and macro-f1 of 90.46% and 82.23% respectively. On the other hand, this Thesis also describes using a pre-trained BERT-base model on the MIMIC-III [151] corpus called Clinical BioBERT (CBBERT) [37]. The latter model was also fine-tuned and evaluated using the DBD dataset, obtaining a micro- and macro-f1 of 91.38% and 83.63%, respectively.

In [106], the authors evaluate different pre-trained transformer-based models and an ELMO [75] model with a dense layer. The authors train and validate the models using the DRD dataset, performing class clustering: negative (class 0, rating<=4), neutral (class 1; rating in [5,6]), and positive (class 2; rating>=7). However, they suppress neutral reviews due to their underrepresentation in the dataset, turning the problem into a binary sentiment classification. In addition, they lemmatize the words before feeding the different models. ELMO+NN achieved 86% accuracy on the test set, while BERT_base, XLNet and Clinical BioBERT achieved 91.9%, 94.08% and 93.71% accuracy, respectively.

On the other hand, in [64], another study was conducted on the performance of pre-trained transformer-based models on the SA task on the DRD dataset. However, the authors decided to tackle a binary problem, using the median to determine that ratings of 8 or higher are "above average" while the rest are "below average", intending to minimize the drug user's subjectivity. The results obtained for BERT were a macro-f1 and accuracy of 84% in both cases, while for RoBERTa, XLNet and Clinical BioBERT were 83%, 84% and 87%, respectively.

To conclude, a summary of the results obtained by the approaches described in this section is shown in Tables 3.3 and 3.4. The first table shows the performance gains of the different models trained and evaluated on the MR, SST and IMDB datasets. The second table shows the results obtained by the different works whose models were trained and evaluated on the DRD dataset.

3.4. Supervised approaches for Named Entity Recognition

Similar to the task of SA, in recent years, several researchers have made advances in NER with approaches similar to those described in the previous section. These approaches share common architectural components, starting with an input considering word and/or character level information that can incorporate additional features, such as part-of-speech

Model	MR	SST		IMDB
	2-classes	2-classes	5-classes	2-classes
	acc	acc		acc
RAE [158]	0.777	0.824	0.432	-
MV-RNN [160]	0.790	0.829	0.444	-
RNTN [29]	-	0.854	0.457	-
DCNN [161]	-	0.868	0.485	-
TextCNN [162]	0.815	0.881	0.480	-
Multi-Task [101]	-	0.879	0.496	0.913
BLSTM-2DCNN [163]	**0.823**	0.895	**0.520**	-
BERT [54]	-	0.935	-	-
BERT-large [54]	-	0.949	-	0.9549
RoBERTa [56]	-	0.967	-	-
XLNET [55]	-	**0.971**	-	**0.9680**

Table 3.3: Accuracy of text classification models on SST, IMDB and MR datasets. The best results are shown in bold.

Model	DRD				
	3-classes		2-classes	2-classes (median)*	
	micro-F1	macro-F1	acc	macro-F1	acc
3W3DT [168]	-	0.7733	-	-	-
ELMO-NN [106]	-	-	0.86	-	-
LSTM+CNN [23]	0.8920	0.8103	-	-	-
BERT-s+BLSTM [23]	0.9046	0.8223	-	-	-
CBBERT [23]	**0.9138**	**0.8363**	-	-	-
BERT [106]	-	-	0.919	-	-
BERT [64]	-	-	-	0.84	0.84
CBBERT [106]	-	-	0.9371	-	-
CBBERT [64]	-	-	-	**0.87**	**0.87**
RoBERTa [64]	-	-	-	0.83	0.83
XLNET [106]	-	-	**0.9408**	-	-
XLNET [64]	-	-	-	0.84	0.84

Table 3.4: Results of the model evaluation on the DRD dataset. The best results are shown in bold. (*) The dataset was divided into two sentiment classes (median), i.e., into classes above- and below-average.

(POS) tags and dictionaries. Such input feeds context encoders that capture dependencies in a latent representations, and label decoders are used to predict tags based on the latent representations of the input sequence [24]. However, in this type of problem, the label is not decoded from a text representation but at the token level to obtain a sequence of labels given a sequence of text. In NER, the most widely used architectures for label decoding are Multilayer Perceptron (MLP) + softmax layer at the token level, CRFs, and recurrent

networks RNNs.

These approaches have been carried out using traditional machine learning algorithms as in [178]–[180]. For example, in [180], the authors employ a system called BANNER [181] in combination with a dictionary based on the UMLS semantic network [182] for task 10 of SemEval-2017. Specifically, BANNER is based on second-order linear CRFs, which allow capturing the dependencies between a label y_t and its previous labels in succession. For feature extraction, text sequences are divided into tokens, where a set of features represents each token: lemmas, 2-, 3- and 4-character prefixes and suffixes, bigrams, and trigrams, as well as a set of regular expressions to normalize numeric values and convert them to "0". In addition, the word class feature normalizes the shapes of a token based on its letters, converting uppercase letters to "A", lowercase letters to "a", and numbers to "0".

Finally, the authors incorporate as a feature the occurrence of a given token within the dictionary defined based on UMLS semantic groups for each type of key phrase (TASK, MATERIAL, PROCESS) and label the tokens using different labeling schemes: IO, IOB, and IOBEW (O=outside, B=beginning of an entity, I=within an entity, E=end of an entity, W=only one entity). In the best of their configurations, they obtained a micro-F1 of 23% on the test dataset of the task.

However, deep learning approaches allow automatic learning of data representations at different levels of abstraction from raw text without the need for extensive feature engineering. For example, in the same task, the authors of [183] proposed to use a model in which inputs were represented from word embeddings of the pre-trained GloVe model [73]. Nevertheless, the word embedding models contain no representation for those tokens not included in their vocabularies, so the authors use a character-level word representation parameterized as a CNN network with 3-character size filters, resulting in $x_k = [c_k; w_k]$ where c_k is the character-level representation of the word k and w_k the word embedding obtained from GloVe.

Next, the authors employ a BiLSTM network to capture the context of the token embeddings, followed by a CRF layer to predict the entities of the text sequence. The authors use the CRF loss function to fine-tune the model to minimize the negative log-likelihood error, as shown in equation 3.10:

$$\begin{aligned} L(\theta) &= -\log\left(\frac{\exp\left(\sum_{k=1}^{K} f(y_k, y_{k-1}, x_k)\right)}{\sum_{y'} \exp\left(\sum_{k=1}^{K} f(y'_k, y'_{k-1}, x_k)\right)}\right) \\ &= -\log\left(\frac{\exp\left(\sum_{k=1}^{K} U(x_k, y_k) + \sum_{k=1}^{K} T(y_k, y_{k-1})\right)}{Z(\mathbf{X})}\right) \end{aligned} \tag{3.10}$$

Where $f(y_k, y_{k-1}, x_k)$ is a score feature function that assigns a score to each label assignment for a particular position k in the sequence, taking into account the current label

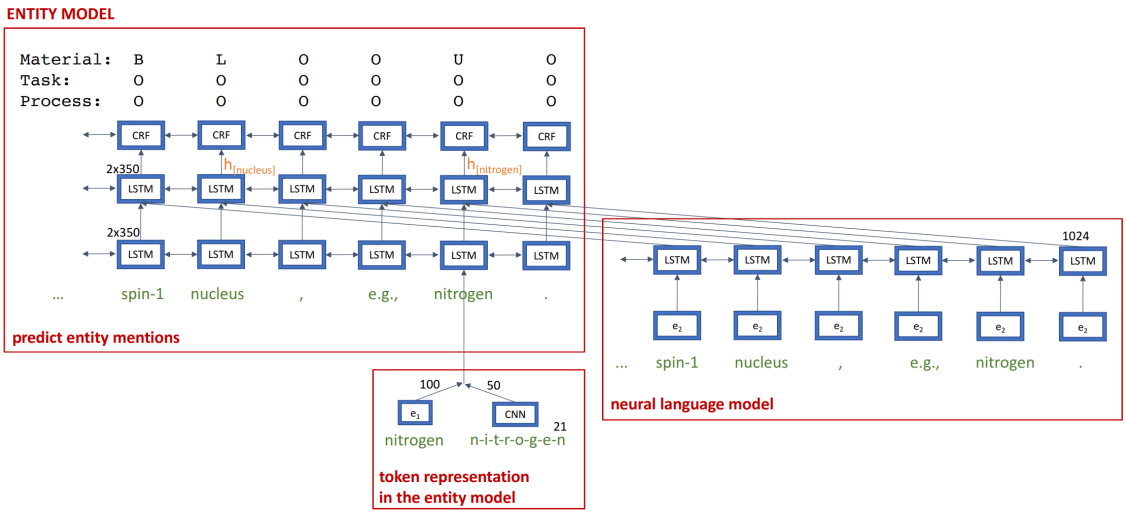


Figure 3.8: General architecture of the winning entity recognition model in task 10 of SemEval-2017 (AI2). Image retrieved from Ammar et al, 2017. [183]

y_k, the previous label y_{k-1}, and the input features x_k (in this case, x_k is the latent representation obtained at each step of the sequence by the BiLSTM network). The feature function is decomposed into $U(x_k, y_k)$ representing the score assigned to the observation features and $T(y_k, y_{k+1})$ representing the score assigned to the transition between consecutive labels y_k and y_{k-1}. $Z(X)$ in the equation represents the partition function, a normalizing sum of all possible feature scores for all possible label sequences.

However, before fine-tuning the model to the training dataset, the authors initialize the weights of the BiLSTM network by pre-training each LSTM as a language model where the networks are tuned to predict the probability of each token in an autoregressive way, predicting future tokens in one of the LSTMs and previous tokens in the other. In addition, they propose to use lists of scientific terms collected from the web and freebase as an extra feature added at the token level. For this, they employ a dense layer with a tanh activation function whose output is concatenated with the output of the BiLSTM layer. Finally, they used an IOBEW scheme for labeling the TASK, MATERIAL, and PROCESS entities (see Figure 3.8). In the best of their configurations, they obtained a micro-F1 of 44% on the test dataset of the task, winning the task.

Like in [183], the architectures based on recurrent networks as context encoders and CRFs as label sequence decoders have been widely used in NER tasks due to their ability to capture contextual information of words in text sequences. For example, in subtask A of ehealthkd-2019 [152], systems must be able to extract key phrases from sentences of EHRs written in Spanish whose semantics are identified with concepts, actions, predicates, and references to concepts. In this subtask, [184] proposes to adapt the NeuroNER system [185] based on three main layers: a character and token embedding layer, a BiLSTM contextual encoding layer, and a CRF layer for label sequence decoding.

The first layer (Character and Token Embedding Layer) aims to generate vector representations of the tokens that make up the input sequences, where a representation of the

tokens based on their characters (character embeddings) is incorporated. Each character of the token is represented by its vector, which allows the network to learn morphological information even from tokens not included in the vocabulary of the word embedding model [186]. The character embedding sequence of each token is passed as input to a BiLSTM to obtain character-based word embeddings as output, which are concatenated to the token-level embeddings.

The resulting representations form the input for the second BiLSTM layer, which aims to obtain the sequence of probabilities that each token belongs to a given label using the IOBEW encoding. Finally, the last layer (CRF optimization layer) consists of a CRF for label sequence decoding. This layer receives the sequence of probabilities from the previous layer as input to improve the predictions (see Figure 3.9).

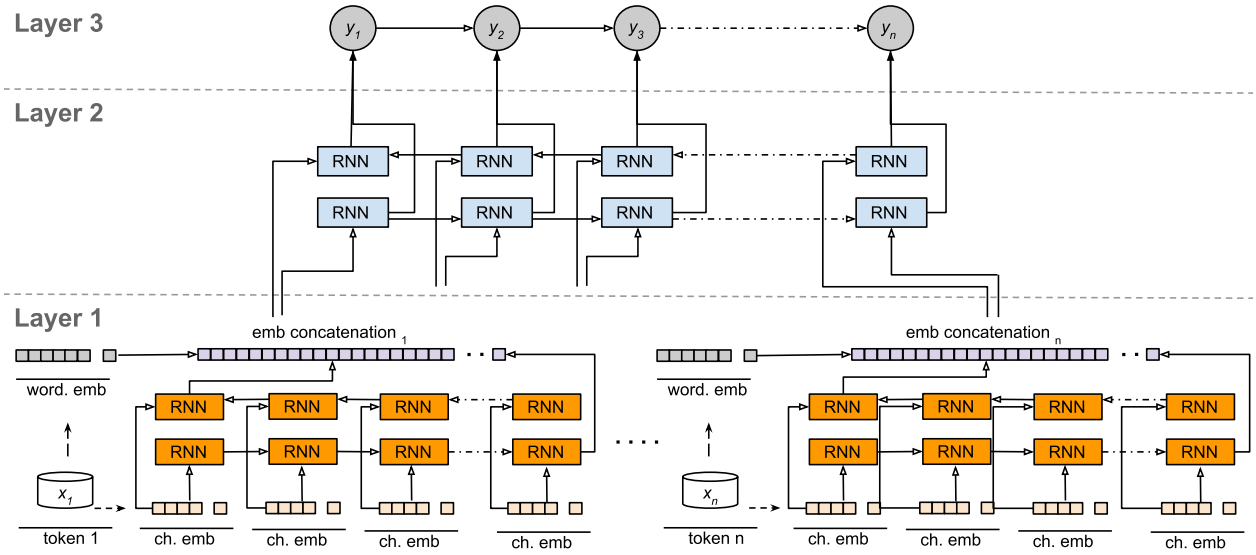


Figure 3.9: Overview architecture of the Hulat-TaskAB model in subtask A of ehealthkd-2019. Image retrieved from Colon et al, 2019. [184]

The authors trained the proposed architecture on the training dataset provided by subtask A, obtaining a result on the scenario 2 test dataset of 77.5% micro-F1, where the micro-F1 is calculated taking into account predictions with partial matches at the boundaries of the annotations as half of a correct prediction.

In the same task, [187] proposes the use of a model that uses a BiLSTM layer to obtain the contextual information at the token level, which in turn feeds a convolutional layer to obtain local information from the representations of each token, the latter being concatenated with each of the outputs of the BiLSTM network. Then, the final representations of each token feed a fully connected layer followed by a CRF layer to obtain the sequence of probabilities that each token belongs to a given label using the IOBEW encoding (the authors use two CRF layers since the system is also used to extract relations between entities). In addition, the authors use the word embeddings provided by the BERT-based multilingual cased model output as input to the proposed model.

To avoid overfitting problems due to the training dataset's size, the authors propose

to pre-train their model using the dataset provided by the same task in 2018 [188] but discarding the weights of the output layers due to discrepancies between the entities and relations in both datasets. Once the model is pre-trained, the authors fine-tune it to the training set of ehealthkd-2019, obtaining micro-F1 results in scenario 2 of subtask A of 82%, where micro-F1 is calculated by taking into account predictions with partial matches at the boundaries of the annotations as half of a correct prediction.

Similarly, in the MEDDOCAN-2019 task [153], systems must be able to automatically detect entities referring to PHI in clinical texts. To this end, [39] proposes to re-adapt NeuroNER (see Figure 3.9). However, clinical cases are provided at the document level, and the authors propose to perform a pre-processing where the texts take into account different steps. First, the documents are split using Spacy (Spacy.io), an open-source library that supports texts in several languages, including Spanish. In addition, due to the language used in this type of text, the authors use a set of rules to avoid detecting dots corresponding to acronyms or abbreviations as sentence separators. Finally, the text and its annotations are transformed into the CoNLL-2003 format[6] using the IOBEW schema. The authors trained the model on the training dataset of the task, obtaining a result on the test set of 93.22% micro-F1.

In the same task, the authors of [108] propose using another architecture based on BiLSTM+CRF with an IOB tagging scheme. However, the authors compare different combinations of word embedding methods: The first one consists of a character-level word representation, using a BiLSTM network and concatenating the last hidden states of the forward and backward steps. The second method uses fastText [74] to represent words with the normalized sum of all n-gram word embeddings. Finally, the third method obtains the word embeddings from the FLAIR framework [189], specifically by pre-training the model as a character-level language model to obtain the probability of the next character.

The proposed BiLSTM+CRF model was trained on the MEDDOCAN-2019 training dataset using different combinations of embeddings as inputs (not being adjusted during training). The best model used a concatenation of the embeddings obtained by FLAIR (4096 dimensions), together with fastText embeddings pre-trained on domain-independent standard narrative and news data from Common Crawl and Wikipedia [190] (300 dimensions) and fastText embeddings pre-trained on domain-specific Spanish SciELO and Wikipedia articles [191] (100 dimensions). The results obtained on the test dataset reached 96.94% micro-F1.

At this point, several systems have been described for NER tasks where recurrent networks with optimization layers based on linear CRFs are predominant, where different approaches focus on enriching token-level representations using combinations of different methods, and on transfer learning methods. The CoNLL-2003 dataset [143], widely used to evaluate advances in NER research, has also been used to evaluate such approaches.

[6]https://www.clips.uantwerpen.be/conll2003/ner/

In work such as [102], the authors propose the LM-LSTM-CRF framework, which is based on obtaining word-level representations with a character-level BiLSTM network and concatenation of the obtained word-level representations by the pre-trained GloVe model [73]. These word-level representations feed another BiLSTM+CRF network. During the training, the authors employ a joined learning where, at the same time, the error of the CRF loss function is reduced along with the error of predicting a word from the input sequence using the previous characters in the forward configuration and the subsequent characters in the backward configuration of the first BiLSTM layer (see equation 3.11):

$$\mathcal{J} = -\sum_{i} (log\ p(y_i|Z_i) + \lambda\,(log\ pf(x_i) + log\ pr(x_i))) \tag{3.11}$$

Where $p(y_i|Z_i)$ represents the probability of a label y_i given a word representation Z_i, $pf(x_i)$ represents the probability of a word x_i given a sequence of preceding characters, and $pr(x_i)$ represents the probability of a word x_i given a sequence of succeeding characters.

Finally, the authors trained the model on the CoNLL-2003 training set with randomly initialized parameters, using a hidden state size of the LSTMs of dimension 300 and with an IOBEW coding scheme. The model was evaluated on the CoNLL-2003 test dataset with a micro-F1 of 91.71%.

However, in recent years, with the advent of large models based on pre-trained transformers such as BERT [54], other work has evaluated different approaches using such models as context encoders, being fine-tuned in NER tasks and not only used to obtain context-aware word-level representations. For example, the authors of BERT fine-tuned their case-sensitive models using the CoNLL-2003 dataset for the NER task. Specifically, the authors employed a sequence decoder based on a dense+softmax layer at the word level, where the first subtoken representation of each word is used as input. As a result, the BERT-base model obtained a micro-F1 of 92.4%, while the BERT-large model obtained 92.8%, outperforming other approaches based on recurrent networks such as ELMo [75].

In addition to the approaches described above, the capability of large pre-trained transformer-based models has allowed NER tasks to be addressed as Machine Reading Comprehension (MRC) tasks where the goal is to extract spans of the input sequence as an answer to a given question, being formalized as two classification tasks to predict the initial and final position of the answer within the input sequence (context). For example, in [107], the authors use the pre-trained BERT-base model to model the NER task as an MRC task. Specifically, they propose to modify the labeling style in the form of sets of (Question, Answer, Context), where the context is the input sequence, X, the answer consists of a substring of X, and the questions encode the prior knowledge of the labels and are generated by the authors taking the notes from the annotation guidelines as a reference.

Specifically, to use BERT, the authors concatenate the question sequences q_y, related to the entity type y, with the original text sequences X to form the following chain: $\{[CLS], q_1, ..., q_m, [SEP], x_1, ..., x_n\}$. Subsequently, only the representation matrix $E \in \mathbb{R}^{nxd}$, where d is the dimension of the BERT output vectors, is used once the model is fed with the new sequences, ignoring the question representations. Finally, the authors employ three different binary classifiers based on a softmax layer, where one predicts whether each token belongs to a start index of the entity type, the second predicts the tokens that belong to the end indices of the entity type, and the third predicts the probability that two entity start and end indices coincide to form the entity boundaries. Considering the above, the overall training objective to be minimized is defined as follows (see equation 3.12):

$$\begin{aligned}
\mathcal{L}_{start} &= CE(P_{start}, Y_{start}) \\
\mathcal{L}_{end} &= CE(P_{end}, Y_{end}) \\
\mathcal{L}_{span} &= CE(P_{start,end}, Y_{start,end}) \\
\mathcal{L} &= \alpha\mathcal{L}_{start} + \beta\mathcal{L}_{end} + \gamma\mathcal{L}_{span}
\end{aligned} \tag{3.12}$$

Where each loss is calculated with the cross-entropy, and α, β, γ are hyperparameters to control the respective contributions to the training objective.

The authors evaluated their system on multiple datasets, CoNLL-2003 being one of them. Specifically, the authors obtained a micro-F1 of 93.04% on the test set, outperforming the approach proposed by the BERT authors.

	Input representation			**Context**	**Label**	**SemEval-2017**	**eHealthkd-2019**
	Character	**Word**	**Hybrid**	**encoder**	**decoder**	**(Task 10)**	**(scenario 2)**
[180]	BANNER		UMLS dict	-	CRF	0.23	-
[183]	CNN	GloVe	freebase+web dicts	BiLSTM	CRF	**0.44**	-
[184]	BiLSTM	randomly initialized	-	BiLSTM	CRF	-	0.775
[187]	-	BiLSTM	-	CNN	CRF	-	**0.82**
	Input representation			**Context**	**Label**	**CoNLL03**	**MEDDOCAN**
	Character	**Word**	**Hybrid**	**encoder**	**decoder**		**2019**
[39]	BiLSTM	randomly initialized	-	BiLSTM	CRF	-	0.9322
[108]	FLAIR+fastText	-	-	BiLSTM	CRF	-	**0.9694**
[102]	BiLSTM	GloVe	-	BiLSTM	CRF	0.9171	-
[54]	-	BERT-base	-	-	softmax	0.924	-
[54]	-	BERT-large	-	-	softmax	0.928	-
[107]	-	BERT-MRC	-	-	softmax	**0.9304**	-

Table 3.5: Results of evaluating NER approaches on test sets from different datasets in micro-F1. The best results are shown in bold.

To conclude, Table 3.5 summarizes the results obtained by the approaches described in this section. The table shows that BiLSTM recurrent networks are widely used in NER tasks due to their ability to capture contextual information in the text sequence, which is beneficial for word-level tagging tasks. Furthermore, there is a tendency to

use different complementary networks, such as LSTMs and CNNs, to enrich local information by producing character-level word representations jointly with word embeddings obtained from pre-trained embedding models. Some works, such as [102], [108], [183], also explore transfer learning by initializing recurrent networks with weights obtained by pre-training language models or fine-tuning with joined objectives. However, large pre-trained transformer-based models provide state-of-the-art results in NER since they allow obtaining contextualized representations from encoders based on self-attention layers and also since such models are pre-trained on large corpora.

3.5. Semi-Supervised approaches for Sentiment Analysis

As introduced in section 3.3 on supervised approaches, in the last decade, deep learning approaches have been shown to provide state-of-the-art results due to their ability to extract and learn relevant features automatically [3], [82]. However, manually assigning labels to texts can be costly and susceptible to human error [8]. Therefore, it is important to adopt such approaches in semi-supervised deep learning methods by studying how to effectively use labeled and unlabeled data by deep neural networks.

As a result of the above, several research works have been carried out in this context in recent years. In works such as [192], the authors propose to use doc2vec [193] to obtain document-level representations for a given dataset. However, learning feature representations on resource-poor datasets can lead to several problems due to limited vocabulary coverage. Therefore, they propose to train the doc2vec model on larger external corpora in a transfer learning approach. Specifically, they train the doc2vec model in an unsupervised way on the Wikipedia dataset with approximately 4M articles.

Subsequently, the authors propose pre-training the classifier model on an external dataset closely related to the target task and then retraining it on the limited target dataset. Once extracting the features for the target task has been performed using the described pre-trainings, the authors propose applying a Manifold regularization to add a penalty to the supervised error. For this purpose, the authors use the Reproducible Kernel Hilbert Space (RKHS) kernel to transform the doc2vec vector representations to a high-dimensional Hilbert space, include an L2 regularization in the neural network and add another term to the loss function in addition to the categorical cross-entropy (see equation 3.13).

$$
\begin{aligned}
Loss &= \frac{1}{l}\sum_{i=1}^{l} V\left(x_i, y_i, f\right) + \gamma_A \|f\|_k^2 + \gamma_I \|f\|_I^2 \\
\text{where} \quad \|f\|_I^2 &= \sum_{i=1}^{l} \sum_{\substack{x_i^u \in \\ \text{Neighborhood of } x_i}} \frac{\|f(x_i) - f(x_i^u)\|_2^2}{\|x_i - x_i^u\|}
\end{aligned}
\tag{3.13}
$$

Where f refers to the neural network function, V represents a supervised loss func-

tion, and the second term denotes L2 regularization in the network weights. The third term is the Manifold regularization, which penalizes the differences between the model outputs. The numerator calculates the Euclidean distance between the network output of the labeled data $f(x_i)$ and its unlabeled nearest neighbors $f(x_i^u)$, whereas the denominator provides an inverse weighting of the distance between the labeled and unlabeled input data. The authors use this term to highlight the model differences where x_i and x_i^u are closer in space. Thus, the model considers the examples closer in space to share similar labels. This concept aligns with the Manifold hypothesis, where examples of the same label tend to cluster in lower dimensional structures in Euclidean space.

The authors evaluate their approach using various datasets, including the MR dataset [130], which is divided into 1,000 examples for training and testing. They conduct several tests, some of which include the pre-training phase with the Sentiment140 dataset [194], consisting of 1.6M tweets with positive or negative sentiments. Additionally, the authors perform different splits of the training data by retaining only a certain percentage of annotated data and using the rest as unlabeled data. The best-performing model in the MR test set was trained solely on the MR dataset without pre-training in Sentiment140. The pre-trained model yielded superior results only when the training set was severely limited, primarily due to variations in the data distribution between the two datasets. The top-performing model achieved accuracies of 65%, 67.5%, 73%, and 74% using 10%, 16%, 40%, and 56% labeled data, respectively. In contrast, the model with a supervised approach achieved 56%, 57%, 72%, and 74% accuracy for the same proportions of labeled data.

Other authors, such as [195], employ wrapper-based approaches as self-training [196]. However, the authors propose to use the properties of reinforcement learning-based training in an adversarial network context [197]. In this way, they propose to mitigate the problems of a pure self-training approach, where inaccurate predictions could be reinforced. Hence, the authors use two different models based on recurrent LSTM networks: a predictive P model to predict the class of texts and a J model to distinguish between real labels and labels predicted by the P model. The aim is to treat the probability distribution of the J-model as a value for the action-value function in the case of unlabeled data. In this way, P can be trained via gradient policy where the objective function to be maximized is the following:

$$R(\theta) = \sum_{\hat{y} \in Y} P_\theta(\hat{y}|x) V(\hat{y}|x) \tag{3.14}$$

$$\text{where} \quad V(\hat{y}|x) = \begin{cases} 1 & \text{if } x \in D_L \ \& \ \hat{y} = y, \\ 0 & \text{if } x \in D_L \ \& \ \hat{y} \neq y, \\ J_\phi(x, \hat{y}) & \text{if } x \in D_U \end{cases} \tag{3.15}$$

Where θ are the parameters of the predictive model P, Y is the set of possible labels, x is the input data, D_L is the set of labeled data, D_U is the set of unlabeled data, and y is

the true label of x. J_ϕ is the discriminator model J with parameters ϕ that distinguishes between the true labels and the pseudo-labels generated by P for the unlabeled data. The objective function $R(\theta)$ in Equation 3.14 is a reinforcement learning objective based on the action-value function $V(\hat{y}|x)$ in Equation 3.15, which determines the expected reward for a given action (i.e., predicting a label $\hat{y}$) at a given state (i.e., input data x).

In the first stage, the authors pre-train P on the labeled dataset and use it to pseudo-label the unlabeled ones. Once pseudo-labeled, the J model is pre-trained to distinguish the real labels from the false ones. In the subsequent stage, iteratively on the dataset, the J weights and V function are updated, and the gradient of the expected reward is computed and used to update the P weights with gradient ascent.

The authors evaluate their proposed approach, Reinforcement Learning-based Adversarial Networks for Semi-supervised learning (RLANS), using various datasets, including the IMDB dataset, which consists of 25,000 examples for training and testing. They also perform different splits of the training data, retaining only a certain percentage of annotated data and using 50,000 unlabeled data. For training, the authors used a word embedding dimension of 256 with randomly initialized vectors, a hidden LSTM unit size of 1024, and a batch size of 64. The RLANS model achieved accuracies of 82.05%, 90.29%, and 91.63% using only 0.4%, 2%, and 4% labeled data, respectively. In contrast, the model with a supervised approach, based solely on an LSTM network, achieved accuracies of 53.87%, 61.52%, and 63.61% with the same proportion of labeled data.

Until now, approaches have been seen that use the properties of distributions to perform regularizations in Manifolds or wrapper methods. However, in other works, such as [198], [199], the authors propose using generative models to impute missing data (in this case, the labels of unlabeled data) and subsequently learn a classification function from the complete data. This approach allows taking advantage of latent structures in the data to obtain better imputation than simply assuming random labels or using heuristic methods.

In [198], a model called Semi-supervised Sequential Variational Autoencoder (SS-VAE) is specifically proposed, which consists of three main components: a $q_\phi(z|x, y)$ encoder, a $p_\theta(x|y, z)$ decoder, and a $q_\phi(y|x)$ classifier. In the encoder, each data pair (x, y) is encoded in a region of the latent space z, where a Gaussian distribution parametrizes the z distribution. In order to achieve this, the encoder attempts to maximize the expectation of the conditional log-likelihood of the z-representation through the reconstruction error at the decoder while minimizing the Kullback-Leibler (KL) divergence between the prior distribution $p(z)$, consisting of a diagonal Gaussian distribution, and the posterior distribution learned by the encoder.

The p-decoder is a conditional generative model that estimates the probability of generating x given its latent representation z, together with the categorical observable variable y. In the case of labeled data, the label can be directly employed to propagate the classification error with categorical cross-entropy. However, in the case of unlabeled data, the

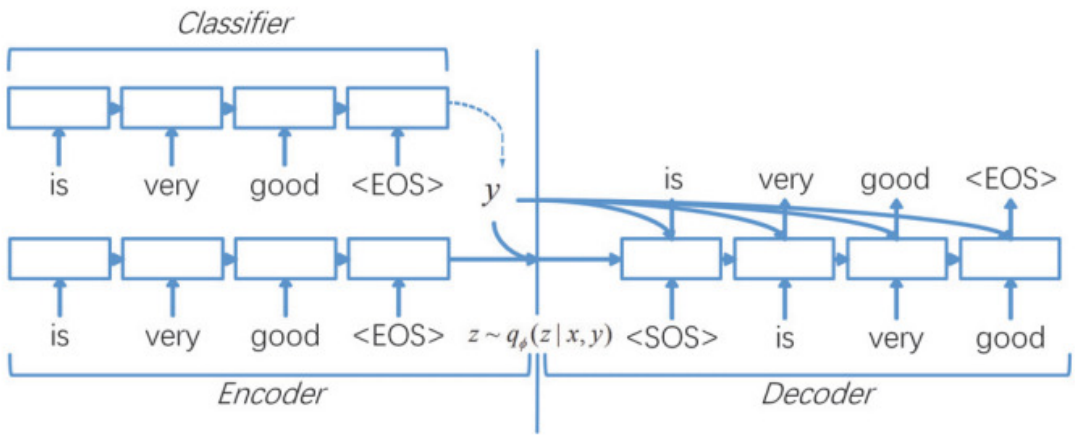


Figure 3.10: Overview of SSVAE architecture. Image retrieved from Xu et al., 2017. [198]

gradient expectation of $q_\phi(z|x, y)$ must be calculated since its labels are unknown. In this case, the authors propose to use a sampling of the likelihood provided by the classifier model on the unlabeled data, using Monte Carlo to capture the model uncertainty based on the mean of the gradients obtained for $y^{(k)}$, where k is the number of labels sampled.

As shown in Figure 3.10, the SSVAE decoder includes information about the y label in each step of the LSTM network. The authors propose two ways of including the information to prevent the label's influence from being lost. One is to concatenate the word embedding vectors of the words with the label vector, while the other is to add the label information directly to the LSTM cells through a hyperbolic tangent (tanh) activation gate for the calculation of the LSTM cell state $C_t = f_t * C_{t-1} + i_t * \hat{C}_t + tanh(W_{yc}y)$.

The authors evaluated the proposed SSVAE model with the different proposed combinations on several datasets, including the IMDB dataset. Specifically, they evaluated the result using 50,000 unlabeled data and varied the amount of labeling between 10%, 20%, 40% and 80% of the 25,000 labeled data available from the training dataset. The results on the IMDB test dataset showed that the best accuracy was obtained by the model whose LSTM classifier q had been pre-trained as an unsupervised language model, and the decoder had been fed with information about the label in each LSTM cell. Specifically, the SSVAE model obtained accuracies of 90.6%, 91%, 92% and 92.4%, respectively, while the supervised approach obtained 82.03%, 84.33%, 87.01% and 89.1%.

In other works, such as [16], the authors propose the CVAE-Attention model based on modifications of SSVAE. The main modifications consist of adding an attention layer to evaluate the "but" clauses within the texts and adding the information to the latent representation z. In addition, the authors employ the Gumbel-softmax [200] distribution to approximate the categorical labels y provided by the classifier to smooth, continuous distributions avoiding the need to sample k $y^{(k)}$ within the possible labels (see Figure 3.11).

The authors evaluate the models on another dataset, Cornell MR, which includes the

MR dataset and SST-2. Specifically, the authors decided to use SST-2 as the labeled dataset to train and evaluate the different approaches, while the MR dataset is the unlabeled dataset. Evaluations are carried out by measuring the impact of variation in the amount of annotated data, and the authors propose to vary between 1k, 2k, 4k, and 10k of unlabeled data. Moreover, the results are compared with other variants of VAE [201] where, instead of using an LSTM-based decoder, it uses a residual block consisting of three dilated convolutional layers. In this case, the generation in the decoder of x_t must only be conditioned by $x_{<t}$.

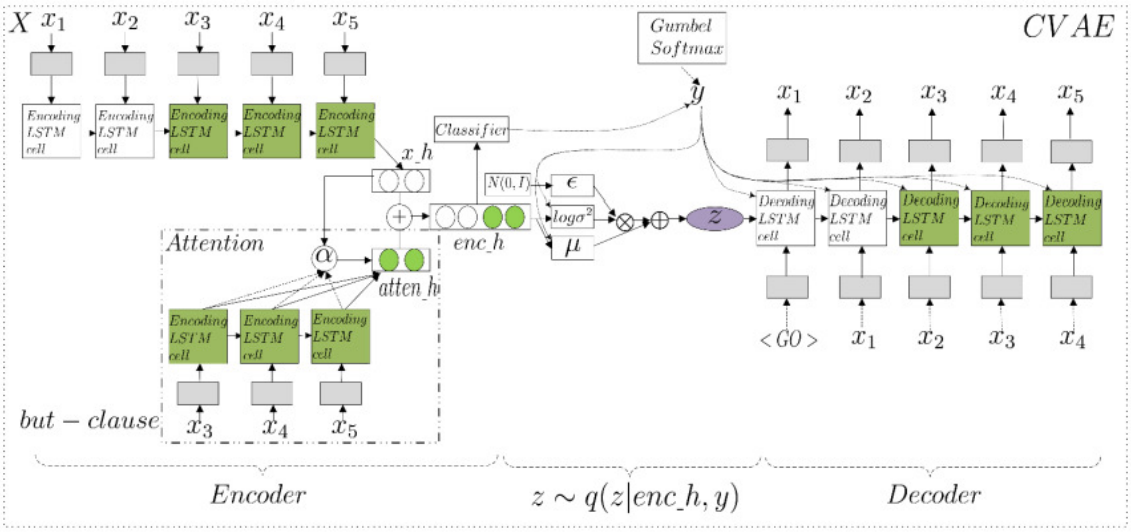


Figure 3.11: Overview of CVAE-Attention architecture. Image retrieved from Yu et al., 2019. [16]

The results show that the SSVAE model obtains an accuracy of 82.14%, 74.61%, 71.71% and 79.52%, being outperformed by the VAE version with dilated CNNs with an accuracy of 89.90%, 87.72%, 83.20% and 76.06%, in both cases for an unlabeled data portion of 1k, 2k, 4k and 10k respectively. On the other hand, the CVAE-attention model improves the performance compared to the previous examples, especially when the number of unlabeled examples increases, obtaining accuracies of 89.90%, 87.67%, 86.89% and 81.64%, respectively. However, it can be observed that as the number of unlabeled data increases, the overall performance hurts due to the influence of the auto-annotation of the data.

Alternatively, semi-supervised generative approaches can be used to improve the performance of models in classification tasks by using approaches based on Generative Adversarial Networks (GAN) [202]. This approach is based on game theory, where GANs aim to train a generative network $G(z)$ that provides a distribution similar to real data from noise vectors z. The training signal for G is obtained from a discriminative network $D(x)$ trained to distinguish real examples from the distribution provided by G. Therefore, the goal of G is to fool the discriminator D, while the goal of D is to improve its ability to identify realistic data distributions from increasingly similar distributions provided by G, both G and D maintaining a Nash equilibrium. As a result, competition between G and

D can lead to more informative latent representations for class separation and reduce the risk of overfitting in classification problems with few labeled data [203].

While Semi-Supervised Generative Adversarial Networks (SSGAN) have been extensively studied for images, these have been less employed in NLP problems due to the discrete nature of text representations, e.g., BoW representations. However, with the emergence of word embedding representation techniques in continuous spaces (e.g., word2vec and GloVe, among others), the linguistic information encoded in these representations can benefit this scenario. While in works such as [204], the authors employ pre-trained low-dimensional embeddings of sentences transformed to Hilbert space via Kernels; few works have explored the application of the GAN perspective in deep learning architectures.

In [65], the authors propose to employ the pre-trained BERT model to obtain text embeds and employ them in a generative adversarial setting called GAN-BERT. In this way, during fine-tuning, the text representations of the model are enriched with the unlabeled data from an SSGAN perspective. For this purpose, the authors propose to use a MLP as a generative network $\mathcal{G}$ and another MLP as a discriminative network $\mathcal{D}$, followed by a softmax layer for prediction. The model obtains the input data representation p_d through BERTs, which are used to feed $\mathcal{D}$ and obtain the probability distribution. In this case, the discriminator not only obtains the probability of the k-class label of each input data but also obtains the probability that the representation is real or generated by $\mathcal{G}$ $(k+1)$. At the same time, $\mathcal{G}$ generates a distribution p_g from $N(\mu, \sigma^2)$ noise vectors that resembles the distribution of the real representations p_d, feeding these to the discriminator and trying to fool it. To train $\mathcal{D}$, the authors define a loss function composed of several components, as can be seen in equation 3.16:

$$\begin{aligned} L_{D_{sup}} &= -\mathbb{E}_{x,y\sim p_d}\, log[p_m(\hat{y} = y|x, y \in (1, ..., k))] \\ L_{D_{unsup}} &= -\mathbb{E}_{x\sim p_d}\, log[1 - p_m(\hat{y} = y|x, y = k+1)] \\ &\quad - \mathbb{E}_{x\sim \mathcal{G}}\, log[p_m(\hat{y} = y|x, y = k+1)] \end{aligned} \tag{3.16}$$

Where $L_{D_{sup}}$ represents the loss of the discriminative network in the supervised task, where labeled data from the real data distribution p_d is used to compute the probability that the predicted label $\hat{y}$ equals the real label y. On the other hand, $L_{D_{unsup}}$ is the loss of the discriminative network in the unsupervised task, where a combination of labeled and unlabeled data is used. In particular, data from the real data distribution p_d and a random sample from the distribution generated by the generative model $\mathcal{G}$ are used to calculate the probability that the predicted label $\hat{y}$ is class $k+1$. The loss is minimized when the discriminative network can distinguish with high certainty between the real and generated data and classify the labeled data appropriately. On the other hand, to train $\mathcal{G}$, the authors define a loss function composed of two components, as can be seen in equation 3.17:

$$
\begin{aligned}
L_{G_{\text{feature matching}}} &= \|\mathbb{E}_{x\sim p_d}\, f(x) - \mathbb{E}_{x\sim \mathcal{G}}\, f(x)\|_2^2 \\
L_{G_{unsup}} &= -\mathbb{E}_{x\sim \mathcal{G}}\, log[1 - p_m(\hat{y} = y|x, y = k+1)]
\end{aligned}
\tag{3.17}
$$

Where $L_{G_{\text{feature matching}}}$ is the feature matching loss function, which measures the distance of the mean of the features extracted from the real data x and those generated by the $\mathcal{G}$ generator. $L_{G_{unsup}}$ is the unsupervised loss function for the $\mathcal{G}$ generator, which guides the generation of samples that fool the discriminator into classifying them as real while maintaining the diversity of the generated samples.

The authors evaluated GAN-BERT on different datasets, including the SST-5 set of 5 sentiment classes for the SA task. For this purpose, partitioning was performed on labeled data subsets of 1%, 2%, 5%, 10%, 20%, 30%, 40% and 50% of the training dataset, using 100 unlabeled data examples for each labeled example. The results showed that GAN-BERT is beneficial, especially when the number of labeled examples is more limited, obtaining an accuracy of 30.4% compared to 22.2% obtained by BERT in a supervised approach when the available labeled data was 1%. However, from 20% of labeled data, BERT's results are comparable to GAN-BERT's.

Other types of semi-supervised approaches, however, focus on introducing perturbations into the data, using both labeled and unlabeled data to explore different regions in the feature space and, in this way, increase the diversity in the data by improving the robustness of the models and their ability to generalize [205]. However, similar to generative approaches, while these methods have been widely applied in imaging tasks, they have been less explored in NLP, given the discrete nature of texts. Employing continuous representations based on embeddings helps to mitigate this problem. However, each word in a sentence has a specific meaning, and when noise is added independently to each word in the sentence, the coherence and meaning of the original sentence may be lost.

In works such as [205], the authors propose a framework where the predictive model is split into two different components, F and U, with F being a set of layers frozen during training and U being the remaining set of unfrozen layers. Thus, the predictive model M is defined as $M = U(F(x))$, where F is a pre-trained language model treated as a domain-general feature extractor, while U focuses on the specific task. The authors use dropout on the sentence-level representations to add perturbations to the input texts. Each time $F(x)$ is fed, the output will contain different perturbations concerning the previous one. Once these points are defined, the authors propose to study the framework in algorithms dependent on variations in continuous space inputs. Specifically, they use the proposed framework with the Π-model [206] and temporal ensemble (TE) models [207].

For experimentation, the authors employ a 16-layer Transformer encoder [76] pre-trained as a language model on the WikiText-103 corpus [208] and use the output [CLS] (see section 2.2.4) to predict the class. During training, the set of layers consisting of the word embeddings and the first layer of the transformer form F, while the rest of the transformer forms U.

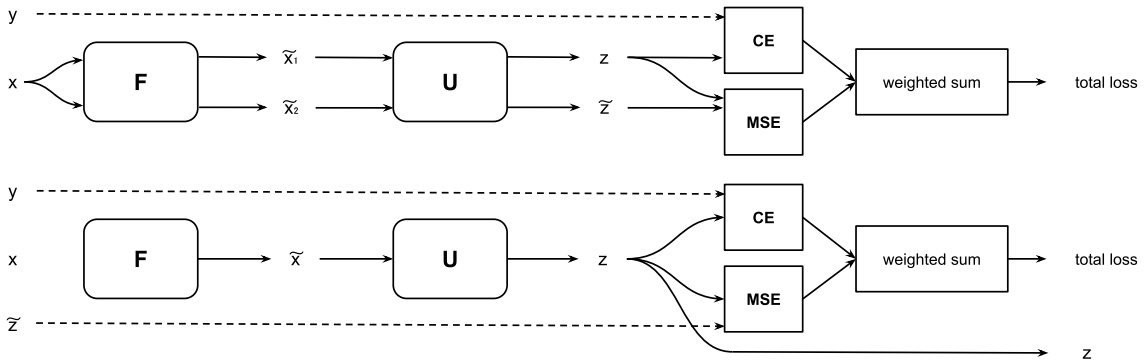


Figure 3.12: Overview of M with Π-model (top) and TE (bottom).

In the defined framework, when using Π-model, each input, labeled and unlabeled data, feeds M twice, obtaining two probability distributions of z and $\tilde{z}$. The loss function is calculated from these distributions, which consists of the weighted sum of two components: a consistency loss consisting of the Mean Squared Error (MSE) between z and $\tilde{z}$ and the categorical cross-entropy loss when $z, \tilde{z} \in D_L$ belong to the labeled dataset. In contrast, when employing TE, the input data is only fed once to M. However, the probability distribution z obtained for that input in that step will be reused to compute the consistency loss in subsequent training steps (see Figure 3.12).

The authors evaluated various models using multiple datasets, including the IMDB dataset. They evaluated the models using 50k unlabeled data and different portions of the training set. The batches were sampled to include 25% of labeled data. When using a supervised approach, the accuracy achieved with only 0.4% and 2% labeled data was 64.7% and 81.5%, respectively. The accuracy was 90.5% when using the entire training dataset. In contrast, when using the semi-supervised framework, the Π-model achieved accuracies of 69.3%, 83.2%, and 91.4%, while the TE model achieved accuracies of 75.9%, 85.1%, and 91.8%. However, the authors suggest that pre-training language models on task-specific text could yield comparable results in a supervised approach, particularly in scenarios where texts are lengthy, such as those found in IMDB.

Other works, such as [209], employ transductive graph-based approaches, where each node consists of word and sentence representations of labeled and unlabeled data. Specifically, the authors propose the Multi-head-Pooling-based Graph Convolutional Network (MP-GCN) model as a semi-supervised approach for text classification. To construct the graph, the authors create different adjacency matrices A, differentiating into adjacency matrices with axes only between word nodes, only axes between word and sentence nodes, and a combination of both. Each of these can be obtained by masking A:

$$A = \begin{cases} PMI(i,j) & \text{i,j are words} \\ TF - IDF & \text{i is word, j is sentence} \\ 1 & \text{i = j} \\ 0 & \text{otherwise} \end{cases} \tag{3.18}$$

$$\text{where} \quad PMI(i,j) = log2(P(i,j)/P(i) * P(j))$$

Where i and j are two words, $P(i, j)$ is the joint probability that they appear together in the corpus, $P(i)$ and $P(j)$ are the marginal probabilities of i and j, respectively. Point-wise Mutual Information (PMI) measures how likely two words occur together in the corpus compared to if they were independent. If the PMI is high, the two words are more likely to be related or used in the same context. If it is low or negative, then the two words are unrelated or even opposite.

The idea is to perform convolutional operations between the features of nearby nodes to obtain the maximum likelihood estimate on the probability distribution of the labeled data, thus fitting the representations of the nodes to propagate information from the label to the neighboring nodes. Moreover, the authors use spectral convolutions, considering the representations as signals multiplied by a g filter parameterized in the Fourier domain:

$$g * x \approx \theta\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\right)x \tag{3.19}$$

Where $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ is the normalised form of the adjacency matrix, D is the degree matrix of A and θ is a scalar. In this way, it is possible to process graphs of variable size without defining a local convolution operation at each node. Additionally, the authors propose using multi-attention heads instead of pooling operations to select important nodes to improve their representation learning.

The authors evaluated their approach on several datasets, the MR dataset being one of them. The authors selected different portions of the training data as the labeled dataset, concretely 2.5%, 5%, 10%, 30% and 50%. The accuracy obtained on the test dataset was approximately 63.5%, 64%, 68%, 74% and 76%, respectively, while the results obtained with a supervised approach with convolutional networks were approximately 53%, 61%, 63%, 69% and 71.5%.

To conclude, Tables 3.6, 3.7 and 3.8 show a summary of the results obtained, by the approaches described in this section, based on the amount of labeled data used. The tables show results obtained with supervised learning approaches compared to semi-supervised learning approaches. In addition, the performance increase with Δ is indicated.

As can be seen from the tables above, the improvement in the performance of the semi-supervised approach models has a decreasing trend as the amount of labeled data used for training increases. While they may provide a slight improvement due to the

		IMDB							
	Annotated %	**0.4**	**2**	**4**	**10**	**20**	**40**	**80**	**100**
Self-training [195]	LSTM (SL)	0.538	0.615	0.636	-	-	-	-	-
	RLANS (SSL)	**0.82 (Δ0.282)**	**0.902 (Δ0.287)**	**0.916 (Δ0.28)**	-	-	-	-	-
Generative [198]	LSTM (SL)	-	-	-	0.82	0.843	0.87	0.891	-
	SSVAE (SSL)	-	-	-	**0.9 (Δ0.08)**	**0.91 (Δ0.067)**	**0.92 (Δ0.05)**	**0.924 (Δ0.033)**	-
Perturbation [205]	Transformer (SL)	0.647	0.815	-	-	-	-	-	0.905
	Π-model (SSL)	0.693 (Δ0.046)	0.832 (Δ0.017)	-	-	-	-	-	0.914 (Δ0.009)
	TE model (SSL)	**0.759 (Δ0.112)**	**0.851 (Δ0.036)**	-	-	-	-	-	**0.918 (Δ0.013)**

Table 3.6: Accuracy of semi-supervised text classification models on the IMDB dataset. The best results are shown in bold.

		MR							
	Annotated %	**2.5**	**5**	**10**	**16**	**30**	**40**	**50**	**56**
Manifold [193]	NN (SL)	-	-	0.56	0.57	-	0.72	-	0.74
	NN (SSL)	-	-	**0.65 (Δ0.09)**	**0.675 (Δ0.105)**	-	**0.73 (Δ0.01)**	-	0.74 (Δ0)
Graph [209]	CNN (SL)	0.53	0.61	0.63	-	0.69	-	0.715	-
	MP-GCN (SSL)	**0.635 (Δ0.105)**	**0.64 (Δ0.03)**	**0.68 (Δ0.05)**	-	**0.74 (Δ0.05)**	-	**0.76 (Δ0.045)**	-

Table 3.7: Accuracy of semi-supervised text classification models on the MR dataset. The best results are shown in bold.

		SST-5					
	Annotated %	**0.1**	**1**	**2**	**5**	**10**	**20**
Generative [65]	BERT (SL)	-	0.222	0.25	0.35	0.435	**0.48**
	GAN-BERT (SSL)	-	**0.304 (Δ0.082)**	**0.36 (Δ0.11)**	**0.405 (Δ0.055)**	**0.46 (Δ0.025)**	0.479 (Δ-0.001)

Table 3.8: Results of the SSGAN text classification model on the SST-5 dataset. Results are shown in terms of accuracy. The best results are shown in bold.

regularisation effect when the number of labeled data used is high, this effect decreases compared to supervised approaches where pre-training has been carried out on similar large corpora.

On the other hand, Table 3.9 shows the effect of different unlabeled data sizes in generative approaches. Increasing the number of unlabeled data can negatively affect the models' performance. This may be due to the increased noise or to dilute the information the labeled data provides.

SST-2	Annotated %	100 (≈ 7K)			
MR	**Unannotated %**	**9.40 (1K)**	**18.80 (2K)**	**37.60 (4K)**	**94 (10K)**
Generative [16]	SSVAE (SSL)	0.8214	0.7461	0.7171	0.7952
	VAE-dilated-CNNs (SSL)	**0.8990**	**0.8772**	0.8320	0.7606
	CVAE-Attention (SSL)	**0.8990**	0.8767	**0.8689**	**0.8164**
Generative [16]	SSVAE (SSL)	0.8234	0.7448	0.7399	0.7994
	VAE-dilated-CNNs (SSL)	**0.8958**	0.8709	0.8145	0.7487
	CVAE-Attention (SSL)	0.8952	**0.8713**	**0.8568**	**0.8191**

Table 3.9: Results of VAE-based text classification models according to the amount of unlabeled data. The first part of the table is in terms of accuracy and the second is in terms of F1. The best results are shown in bold.

3.6. Semi-Supervised approaches for Named Entity Recognition

As mentioned in the section 3.4 on supervised approaches, the task of NER focuses on automatically identifying and classifying named entities in texts. In NER, supervised approaches involve training models on annotated datasets to generalize and make inferences on unseen data. Over the past decade, deep learning has demonstrated exceptional results in this field by effectively extracting and learning relevant features automatically. However, these approaches heavily rely on data, and it is common for real-world scenarios to have limited annotated data available. Therefore, incorporating these approaches into a semi-supervised learning setting can be advantageous in improving model performance when unannotated data is accessible.

While the techniques used for NER in a semi-supervised setting may share similarities with those discussed in section 3.5, which focuses on instance-level classification tasks such as text classification and SA, NER itself is a token-level classification task that involves recognizing and labeling specific entities within a text, while also considering dependencies between different labels in the text sequence. As a result, numerous research studies have been conducted to tackle NER's unique characteristics and challenges in semi-supervised environments.

In [210], the authors address the issue using Variational Sequential Labelers (VSL). This approach, similar to Variational Autoencoders (VAE) [211], employs a generative model with latent variables but also incorporates a discriminative model for sequence labeling. The training of VSL focuses on maximizing the conditional probability of a word given its context $p\theta(x_t|x_{-t})$ while simultaneously minimizing the classification error within the latent space. The authors explore various configurations of latent variables and utilize hidden vectors h_t generated by a bidirectional GRU network (BiGRU). This BiGRU network is fed with a matrix consisting of concatenated word embeddings and final hidden states from another BiGRU network operating at the character level.

Among the different configurations of latent variables, the hierarchical latent variable

yielded the most favorable outcomes as it captures precise latent information derived from the entity label information y_t. In this setup, two distinct Gaussian latent variables, namely z_t and y_t, are extracted for each time step t from the hidden variable h_t. The authors define the objective as maximizing the lower bound of the marginal log-likelihood (refer to Equation 3.20):.

$$\begin{aligned} log\, p_\theta(x_t|x_{-t}) \geq & \mathop{\mathbb{E}}_{q_\phi(z_t, y_t|h_t)} [log\, p_\theta(x_t|z_t)] \\ & - KL\Big(q_\phi(z_t|y_t, h_t) \,||\, p_\theta(z_t|y_t, x_{-t})\Big) \\ & - KL\Big(q_\phi(y_t|h_t) \,||\, p_\theta(y_t|x_{-t})\Big) \\ & = U(h_t) \end{aligned} \tag{3.20}$$

Where the encoder is denoted by $q_\phi(z_t, y_t|h_t)$, while the decoder is represented by $p_\theta(x_t|z_t)$. The first term of the equation captures the probability of a word x_t given its latent representation z_t. Following this, the second term measures the KL divergence between the posterior distribution of z_t given y_t and h_t, and a prior distribution $p_\theta(z_t|y_t, x_{-t})$. It is important to note that the posterior distribution is parameterized by concatenating the hidden vector h_t and y_t. The third term in the equation quantifies the KL divergence between the posterior distribution of y_t given h_t and a prior distribution $p_\theta(y_t|x_{-t})$. These last two terms act as regularization terms, encouraging the encoder to produce posterior distributions that closely resemble a standard normal distribution (the prior distribution). While the first term can be interpreted as the reconstruction term, capturing the likelihood of generating the observed word, the last two ensure that the encoder generates posterior distributions that align with the desired prior distribution.

The rationale behind incorporating regularization with a standard Gaussian distribution in VAE-based networks is to generate distributions with continuity and completeness properties. This ensures that decodings from nearby points in the latent space produce similar data while obtaining meaningful information. However, since this approach does not primarily focus on generating new data using the learned models, the prior standard multivariate Gaussian distributions are updated iteratively with the latest optimized posterior distribution.

Additionally, the authors employ a feed-forward neural network layer to classify x_t into their corresponding entity label l_t. The classification error is defined in equation 3.21, contributing to the final loss function described in equation 3.22. Here, the hyperparameter α represents a trade-off

$$C(h_t, l_t) = \mathop{\mathbb{E}}_{q_\phi(y_t|h_t)} [-log\, f(l_t|y_t)] \tag{3.21}$$

$$\mathcal{L} = \sum_{t=0}^{T} [C(h_t, l_t) - \alpha U(h_t)] \tag{3.22}$$

During the training process, the authors utilize $C(h_t, l_t)$ on labeled data and $U(h_t)$ on both labeled and unlabeled data. Figure 3.13 illustrates the inference and learning process, where dashed lines represent the encoder, the decoder is depicted as solid lines (a two-layer feed-forward neural network with ReLU activation), and dotted lines denote the classifier.

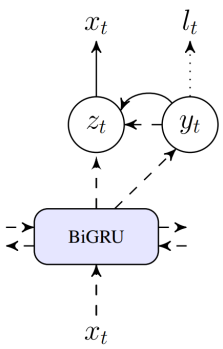


Figure 3.13: Overview of Variational Sequential Labelers with Hierarchical Latent Variables (VSL-GG-Hier). Image retrieved from Chen et al., 2018. [210]

The authors evaluated the performance of their model using the CoNLL-2003 dataset and utilized an IOBEW labeling scheme. Before training, they performed text preprocessing by normalizing the digits to 0 and selected the pre-trained GloVe model for word embedding representation. For training, they utilized only 10% of the original labeled data and allocated 50% of the remaining data as unlabeled. The VSL-GG-Hier model achieved a micro-F1 score of 88.4% on the test set, while the supervised approach utilizing a BiGRU model yielded a slightly lower score of 87.6%

On the other hand, in [212], the authors also use regularisation terms, in this case, the KL divergence, to tackle NER tasks semi-supervised. For this purpose, the authors propose to use Virtual adversarial training for sequences (SeqVAT). Virtual adversarial training (VAT) approaches are based on adding an adversarial loss term for the model to focus on those examples that produce the most errors. This type of approach results in models that are more robust to adversarial examples or attacks, which can also increase the performance of the models on the target task.

The main problem with VAT is that when dealing with a token-level labeling problem within sequences, it is assumed that all labels have the same probability of transitioning between them within the sequence. To solve this problem, SeqVAT incorporates a CRF optimizing top layer.

Specifically, the basis of SeqVAT consists of a model where the representations of the input sequences are formed by the concatenation of a randomly initialized embedding model, together with another embedding resulting from a character-level CNN network with morphological information of unigrams, bigrams, and trigrams. These representations x_t feed a BiLSTM layer, whose hidden variables h_t, in turn, feed another BiLSTM+CRF network.

In SeqVAT, noise is introduced at the character level d_c and token level d_w. This noise is used to calculate the adversarial loss as follows (see equation 3.23):

$$
\begin{aligned}
P(S\,; w, c, \theta) &= (p'_1, ..., p'_k, 1 - \sum_{i=1}^{k} p'_i) \\
P_{original} &= P(S\,; w, c, \theta) \\
P_{adv} &= P(S\,; w + d_w, c + d_c, \theta) \\
\mathcal{L}_{adv} &= KL(P_{original} \,||\, P_{adv})
\end{aligned}
\tag{3.23}
$$

where S represents the k-best label sequences (obtained with Viterbi decoding [213]), p' represents the probability of a label sequence for a given input sequence, w represents the word embeddings, and c represents the character-level embeddings.

Once the authors have defined the adversarial error, the total loss function of SeqVAT consists of the sum of the supervised loss of the negative log-likelihood of the CRF $\mathcal{L}_{label}$ plus the adversarial loss multiplied by a weight λ as a hyperparameter trade-off between the two terms $\mathcal{L}_{total} = \mathcal{L}_{label} + \lambda\mathcal{L}_{adv}$.

The authors evaluated SeqVAT on several datasets, including the CoNLL-2003 task dataset. Specifically, for the semi-supervised approach, the authors use the labeled data to propagate the error $\mathcal{L}_{total}$, while the unlabeled data only propagates $\mathcal{L}_{adv}$. As unlabeled data, the authors use in their evaluation a 1% random sample of the One Billion Word Language Model Benchmark dataset [214], which is 20 times more unlabeled data than the labeled data available in CoNLL-2003.

Finally, the model was trained with λ set to 0.6, k set to 3, a batch size of 64, and a learning rate set to 6e-4 with a decay ratio of 0.992. The model results on the test dataset obtained a micro-F1 of 92.27%, while the base model supervised approach (CNN-LSTM-CRF) obtained a 91.20% micro-F1.

The SeqVAT approach was also compared against one of the state-of-the-art systems called Cross-View-Training (CVT) [215]. In CVT, the authors also propose an adaptation of the same CNN-LSTM-CRF baseline. Specifically, CVT uses the hidden representations of the first BiLSTM context encoding layer to train auxiliary classifiers (auxiliary modules) with different views than the original approach (primary module).

As shown in Figure 3.14, within the BiLSTM layer, the authors use the outputs of the forward LSTM network to predict labels from preceding words and, simultaneously, use the outputs of the backward LSTM network to predict labels from following words. The semi-supervised configuration employs the unlabeled data to improve the distribution of the hidden representations by using KL divergence to force the probability distributions of the auxiliary modules to resemble the distribution provided by the primary module. The authors also evaluate CVT on the CoNLL-2003 dataset, although they do not specify the amount of unlabeled data employed (obtained from [214]). However, in [212], the

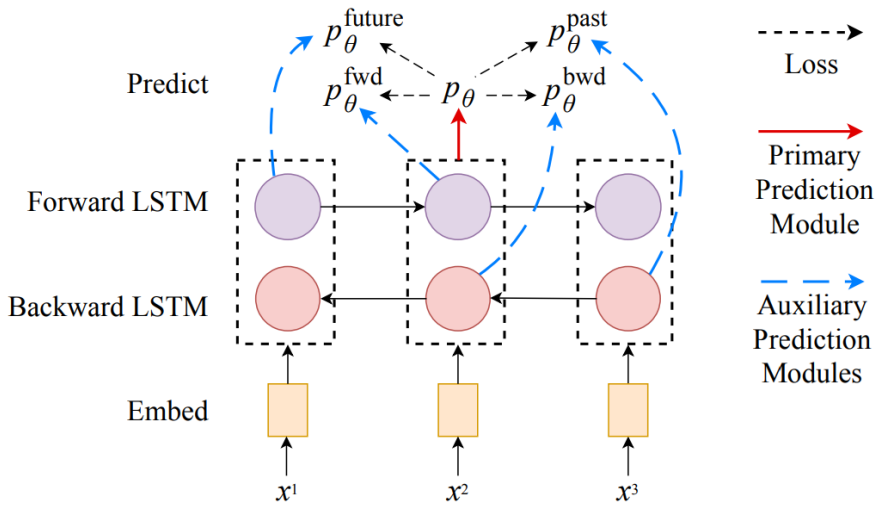


Figure 3.14: Overview of prediction auxiliary modules for sequence labeling of the CVT model. Image retrieved from Clark et al., 2018. [215]

SeqVAT authors use the same 1% of unlabeled data to compare the two approaches, with CVT obtaining a micro-F1 of 92.08% compared to 92.27% from SeqVAT.

Up to this point, different related works have been described where the basic idea is to use regularization and reconstruction terms in the loss functions in order to be able to use semi-supervised configurations. However, other approaches, such as [68], adopt self-training in a NER scenario instead of using regularization and reconstruction terms. The authors utilize the multilingual BERT model (mBERT) [54] as the pre-trained base model. They incorporate a softmax layer at the token level to derive the probability distribution of each token's entity type, employing the IOB labeling scheme. The model is fine-tuned on the labeled dataset and subsequently utilized for predicting pseudo-labels from unlabeled data during training iterations, employing a self-training approach.

The authors deviate from using a one-hot encoding scheme for the pseudo-labeled data in the iterative training process. Instead, they directly utilize the probability distribution, which offers information about the model's uncertainty. Consequently, during training, they minimize the mean square error (MSE) for the pseudo-labeled data instead of the cross-entropy.

Furthermore, the authors propose a method to assess the confidence of the pseudo-labeled data and select those of higher quality. This is done to avoid incorporating noise generated by poor-quality pseudo-labeled examples into the model during re-training. Specifically, they suggest employing a linear classifier (in the following referred to as Judge) that takes the hidden state h_t provided by mBERT and concatenates it with the probability distribution, resulting in $x_t^j = [h_t; f(x_t, \theta)]$. This classifier produces a binary vector that determines the correctness of the NER prediction for each token. However, there is a drawback to this approach. Even with filtering out pseudo-labels from unlabeled examples with the previous method, selecting the best examples using the mean of the

model confidence reduces the probability of providing diverse information. To address this, the authors introduce another confidence measure based on entropy. This measure is used to rank pseudo-labeled data that have been filtered using the first confidence measure (see equation 3.24):

$$
\begin{aligned}
H(\tilde{y}_t) &= -\frac{1}{C}\sum_{c=1}^{C}\tilde{y}_{t,c}\, log\, \tilde{y}_{t,c} \\
\omega_{easy} &= 1 - \frac{1}{|\tilde{y}_t \neq \text{'O'}|}\sum_{\tilde{y}_t \neq \text{'O'}} H(\tilde{y}_t) \\
\omega_{hard} &= \frac{1}{|\tilde{y}_t \neq \text{'O'}|}\sum_{\tilde{y}_t \neq \text{'O'}} H(\tilde{y}_t)
\end{aligned}
\tag{3.24}
$$

Where $\tilde{y}_t$ is the probability distribution $f(x_t, \theta)$, C is the dimension of the distribution, and $\tilde{y}_t \neq$ 'O' is used to avoid taking into account entities labeled O in the IOB scheme. In this way, each pseudo-labeled example gets a ω_{easy} ranking where the value will be higher if the probability distributions of the sequence have a high confidence level. In contrast, the ω_{hard} ranking will be higher if the confidence in the probability distributions is low.

Once the ω-rankings are obtained for each pseudo-labeled example, the number of easy and difficult examples used to retrain the model in the next epoch will be based on a cumulative function of curricular learning (see equation 3.25):

$$
\begin{aligned}
\alpha &= 1 - \left(\frac{E}{E_{max}}\right)^2 \\
T_{easy} &= \alpha T \\
T_{hard} &= (1-\alpha)T
\end{aligned}
\tag{3.25}
$$

Where T represents the number of selected examples, and E refers to the training epoch. T_{easy} defines the number of pseudo-labeled examples ranked with ω_{easy} for utilization, while T_{hard} defines the number of elements ranked with ω_{hard}.

Consequently, the initial fine-tuning step entails training the mBERT model on the labeled data. The second step involves selecting a specific amount of unlabeled data, predicting their pseudo-labels using mBERT, filtering them through the Judge classifier, and ranking them to choose T number of examples. Lastly, in the third step, the selected pseudo-labeled data are incorporated into the training set, and mBERT is retrained using this new dataset. The second and third steps are repeated iteratively until the model converges. Ecuation 3.26 is the loss function for the training in the first step, while loss function 3.27 is used for the subsequent steps.

$$\mathcal{L}_1 = \frac{1}{|D_L|} \sum_{X \in D_L} \frac{1}{n} \sum_{t=1}^{n} \Big(CE(f(x_t, \theta), y_t) + \lambda_j CE(x_t^j, y_t^j) \Big) \tag{3.26}$$

$$\begin{aligned} \mathcal{L}_2 &= \frac{1}{|D_U|} \sum_{\tilde{X} \in D_U} \frac{1}{n} \sum_{t=1}^{n} H(\tilde{y}_t)\, MSE(f(\tilde{x}_t, \theta), \tilde{y}_t) \\ \mathcal{L} &= \mathcal{L}_1 + \lambda_U \mathcal{L}_2 \end{aligned} \tag{3.27}$$

Where, in the first equation, $|D_L|$ represents the number of labeled examples, n represents the number of tokens in each example, $CE(f(x_t, \theta), y_t)$ denotes the cross-entropy of the NER model, $CE(x_t^j, y_t^j)$ signifies the cross-entropy of the Judge classifier, and λ_j is a hyperparameter that determines the balance between these two terms. On the other hand, the second equation involves $|D_U|$, which stands for the number of unlabeled examples utilized. $H(\tilde{y}_t)$ (defined in ecuation 3.24) serves as a weight to control the contribution of each token to the loss function, $MSE(f(\tilde{x}_t, \theta)$ represents the mean squared error of the NER model on the pseudo-labeled data, and λ_U is a hyperparameter that determines the trade-off between $\mathcal{L}_1$ and $\mathcal{L}_2$.

The authors evaluated their model (TS Double Selectors) on several datasets, including CoNLL-2003. For training, they utilized subsets of the labeled data, representing 3%, 5%, and 10% of the training dataset, while the remaining data was left unlabeled. They set the hyperparameters λ_j and λ_U to 0.5 and 1, respectively, and employed the Adam optimizer with a learning rate of 1e-5. In the initial phase, the authors trained the model on the labeled dataset for 5 epochs, followed by an additional 25 epochs using a semi-supervised learning configuration. The results obtained on the test dataset were 85.01, 87.32, and 89.63 micro-F1, respectively, while the baseline supervised approach (mBERT+softmax) obtained 81.51, 83.45, and 85.22 micro-F1.

Furthermore, the authors assessed various methods for comparative analysis. These include the VSL-GG-Hier model [210] (previously described in this section), a conventional self-training model without confidence measures (ST), and BOND [216], a self-training framework for sequence labeling that selects examples based on their confidence. In this study, the authors substitute BOND's initial fine-tuning step, which involves distant supervised data, with fine-tuning on labeled data. The results obtained in such comparisons can be seen in Table 3.10.

To conclude, Table 3.10 also summarizes the results obtained by the approaches described in this section as a function of the amount of labeled data used. The Table shows the micro-F1 obtained with supervised approaches compared to semi-supervised approaches. In addition, the increase in performance with Δ over the baseline supervised approaches are indicated. Note that double lines separate the different papers described. Comparisons between them must be made with caution since the baseline approaches and the amount of unlabeled data employed vary between them.

As can be seen from the Table 3.10, the performance improvement of the semi-

		CoNLL03			
	Annotated %	**3%**	**5%**	**10%**	**100%**
Baseline [210]	BiGRU (SL)	-	-	0.837	-
Generative [210]	VSL-GG-Hier (SSL)	-	-	**0.847 (Δ0.01)**	-
Baseline [212], [215]	CNN-LSTM-CRF (SL)	-	-	-	0.912
Self-training / Perturbation [215]	CVT (SSL)	-	-	-	0.9208 (Δ0.0088)
Perturbation [212]	SeqVAT (SSL)	-	-	-	**0.9227 (Δ0.0107)**
Baseline [68]	mBERT (SL)	0.8151	0.8345	0.8522	-
Generative [68]	VSL-GG-Hier (SSL)	0.8155 (Δ0.0004)	0.8321 (Δ-0.0024)	0.8502 (Δ-0.002)	-
Self-training [68]	ST (SSL)	0.8247 (Δ0.0096)	0.8417 (Δ0.0072)	0.876 (Δ0.0238)	-
	BOND (SSL)	0.8321 (Δ0.017)	0.8602 (Δ0.0257)	0.8851 (Δ0.0329)	-
	TS Double Selectors (SSL)	**0.8501 (Δ0.035)**	**0.8732 (Δ0.0387)**	**0.8963 (Δ0.0441)**	-

Table 3.10: Micro-F1 of the semi-supervised NER models on the CoNLL-2003 dataset. The best results are shown in bold.

supervised approach models based on regularization terms has a decreasing trend as the amount of labeled data used for training increases. Although they may provide a slight improvement due to the regularization effect when the number of labeled data used is high, this effect decreases compared to supervised approaches where pre-training has been performed with similar large corpora. For example, VSL-GG-Hier is able to improve its performance compared to a supervised BiGRU approach but is not able to outperform the supervised approach based on mBERT since mBERT is a large pre-trained model based on transformers and VSL-GG-Hier is based on recurrent networks.

This effect does not seem to occur in wrapper-based approaches such as self-training, as the more labeled data used, the better the model performs in predicting pseudo-labels for subsequent training. However, such approaches are subject to propagating model biases throughout training and may lead to unstable training depending on the nature of the task or data employed [217].

3.7. Data Augmentation approaches for Sentiment Analysis

As described in section 2.3.2, data augmentation techniques focus on adding new examples to the training set, increasing its diversity without needing more labeled data. However, a challenge in this approach is to ensure that the generated data maintains a distribution that strikes a balance between being not too similar and not too different from

the original data distribution. By achieving this balance, the generated data can provide new information that helps the generalization of classification models.

Techniques applied to the data space modify the raw data by transforming the text. One proposed method, discussed in [126], involves adding noise to the training data at the character level to reduce the impact of adversarial examples on model performance, where adversarial examples refer to inputs designed to cause poor performance by the predictive model. The authors suggest introducing natural noise, such as common misspellings in the relevant language, and artificial noise, consisting of character-level transformations like swapping or substituting with neighboring characters on the keyboard. Although this approach has been applied successfully in text classification, as demonstrated in [218], it has not been widely studied in the context of SA. Nonetheless, some researchers have explored using random permutations, insertions, or deletions at the word level to modify texts.

In [122], Wei and Zou propose to use a series of word-level transformations consisting of deletion, insertion, and substitution. They propose a method called Easy Data Augmentation (EDA) in which the use of four specific transformations can be specified (see Table 3.11): Synonym Replacement (SR), where chosen synonyms randomly replace n words in the text (not including stop words); Random Insertion (RI), where, n times, the random synonym of a random word is inserted into a random position in the text; Random Swap (RS), where, n times, two words are randomly chosen and their positions are swapped in the text; and Random Deletion (RD), where each word in the text is randomly deleted with probability p. Since texts can absorb more or less noise based on length while maintaining the original text label, the authors propose to change the number n of words based on the length of the texts l as follows: $n = \alpha l$, where α indicates the percentage of words to be changed. In the case of the RD transformation, the authors use $p = \alpha$.

Operation	**Sentence**
None	The quick brown fox jumps over the lazy dog.
SR	The **speedy** brown fox jumps over the lazy dog.
RI	The quick brown fox jumps over the **indolent** lazy dog.
RS	The **lazy** brown fox jumps over the **quick** dog.
RD	The brown fox jumps the lazy.

Table 3.11: Sentences generated using EDA.

To conduct their experiments, the authors utilized WordNet as a thesaurus, along with various combinations of transformations employing different values of α for each. Their empirical study, encompassing a range of text classification datasets, including SST-2, demonstrated that all transformations contributed to improved performance, with a "sweet spot" of $\alpha = 0.1$. The quantity of augmented data varied based on the fraction of training data employed, resulting in 16 to 4 for each original data.

Regarding the SST-2 dataset, the authors trained the CNN model introduced by [162].

Without utilizing data augmentation for fractions (%): 1, 5, 10, 20, the model achieved an accuracy of roughly 55%, 57%, 80%, and 81%, respectively, on the test dataset. By contrast, training the model with augmented data yielded an accuracy of approximately 80%, 80.5%, 81%, and 81.5%.

In the discussion of the results, the authors describe that their method provides large performance gains when the number of training data is more limited as it provides more diversity of information and avoids early overfitting. In contrast, the models generalize more appropriately for larger training datasets without data augmentation. In their results, the decreasing trend of performance gain can be observed as the number of available training data increases.

Nevertheless, using thesaurus dictionaries can be difficult when robust dictionaries are unavailable for a given language. Other works, such as [127], address this problem by replacing words in the embedding space with similar words. Specifically, using the neighborhood of words in their continuous representations within the embedding space is proposed. In this way, the most similar words can be selected by considering cosine similarity in a pre-trained model (e.g., word2vec or GloVe). However, this type of approach bases word similarity on the co-occurrence of words in large corpora without considering the context in which they appear in the text to be modified.

To overcome this limitation, the authors of [104] propose using pre-trained BERT models to obtain word substitutes for augmented data. Thus, the authors employ the "[MASK]" token used by the MLM task to mask the words to be substituted by letting the model predict the most probable new words (for details of the MLM task, see section 2.2.4). However, employing the BERT pre-training MLM task may result in substitutions where the predicted words are incompatible with their labels since the word prediction depends solely on the context. Consequently, the authors propose Conditional BERT (cBERT), a BERT model capable of predicting the words masked with the token "[MASK]", taking into account the context and the label.

The cBERT model is a pre-trained BERT-base model that is retrained on the MLM task using the dataset for the downstream task. However, it uses the segment embeddings from the pre-training task NSP to encode the label information instead of the segment encoding (see section 2.2.4). In this way, the model is able to predict the masked word also taking into account the label information of the texts (see Figure 3.15).

The authors trained and evaluated their method on several datasets, including SST-5 and SST-2. After re-training cBERT on the training datasets, the model generated new examples by selecting random masking in each original example. The resulting dataset, along with the augmented data, was used to train CNN and LSTM classifiers. The accuracy of these classifiers was compared with and without the augmented data.

In the SST-5 dataset, the CNN model achieved an accuracy of 41.3% without augmented data. However, with augmented data using BERT and cBERT, the accuracy improved to 41.5% and 42.3%, respectively. The LSTM model had an accuracy of 40.2%

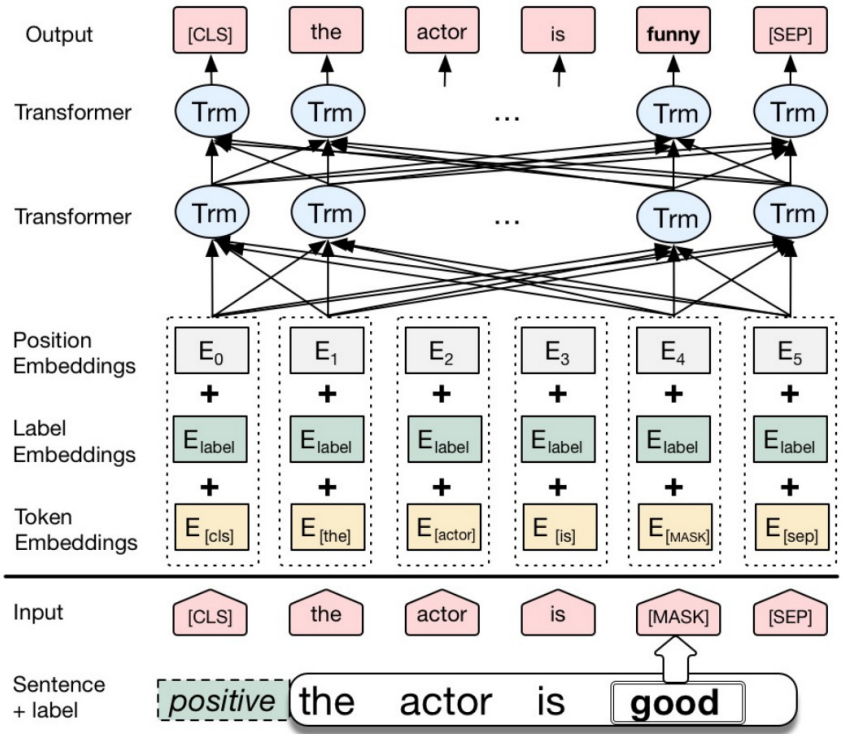


Figure 3.15: Overview of the architecture of the conditional BERT model. The tag embeddings correspond to the BERT segment embeddings. Image retrieved from Wu et al., 2018. [104]

without augmented data, while with augmented data using BERT and cBERT, the accuracy increased to 41.3% and 42.6%, respectively. Furthermore, in the SST-2 dataset, the CNN model achieved an accuracy of 79.5% without augmented data, which increased to 81.9% and 82.1% with augmented data using BERT and cBERT, respectively. The LSTM model had an accuracy of 80.3% without augmented data, while with augmented data using BERT and cBERT, the accuracy improved to 81.4% and 81.9%, respectively.

Other works, however, focus on modifying texts at the sentence level, using changes in their structure based on Parts of Speech (POS) tags, constituency parsing, dependency parsing or text spaces as substructures. In [15], the authors carry out a study on substructure substitution methods (SUB^2) to generate new data by interpolating between different examples, defining a series of restrictions to be applied to the texts to maintain consistency of the texts generated with their labels.

Specifically, the authors make a comparison between the joint use of the following restrictions: Text classification label, where only text space substitutions can be made between texts annotated with the same class label; Constituency, where only substitutions between constituents can be made; Annotated text span label, where substituted constituents must belong to text spans with the same label (e.g., constituent label); and Number of words, where only substitutions of spans containing the same number of words are allowed. The authors also evaluate balanced binary trees analogous to constituency

parse trees. To do so, they generate binary trees by recursively splitting and treating non-terminal nodes as substructures of the tree.

The authors evaluate their approach using the SST-5 dataset with only 10% of the training dataset for the SA task. For data augmentation, they use the training dataset to generate $k = 20$ new augmented data examples for each original example. However, the authors replicate the original data k times to ensure they provide the model with sufficient information about the original distributions. Specifically, the authors train the XLM-R model [219], a multilingual version of RoBERTa [55], using only the original data as a baseline against the augmented data using different combinations of constraints. In addition, the authors compare with other approaches, one of them being contextualized word substitution using a multilingual BERT model [54] (mBERT) with the MLM task.

The optimal results for the SST-5 test dataset were achieved by training with augmented data, utilizing Text Classification label, Constituency, and Annotated Text Span label constraints. This approach employed the Penn Treebank-style formatted sentence parse tree provided for SST, which assigns a positive sentiment degree ranging from 0 to 4 to each component of the tree in order to control the substitution constraints for generating augmented data. The XLM-R model achieved an accuracy of 23.3% without augmented data, which increased to 44.9% using the contextual substitution data augmentation approach with BERT. Nevertheless, both approaches were surpassed by the method proposed by the authors, which achieved an accuracy of 45.8%.

On the other hand, in [220], the authors compare the performance of different data augmentation approaches in the data space for low annotated data scenarios. In their study, they use approaches already described in this section with word and sentence level transformations to obtain augmented data, but, in addition, also with document level transformation approaches. As a base classifier, the authors use the BERT-base model and the token "[CLS]" as a full-text representation.

At the word level, the authors employ data augmentation approaches such as EDA and cBERT. In contrast, they employ a full transformer model (including both encoder and decoder) called Bidirectional and Auto-Regressive Transformers (BART) [221] at the sentence level. BART is a seq2seq model whose architecture is based on [76] (see section 2.2.4) but uses six encoder and decoder layers in its base version, compared to 12 layers in its large version. The BART decoder uses contextual information provided by the last encoder layer in the last multi-head attention of each decoder layer. In this way, BART can introduce a wide variety of noise into the encoder while the decoder generates the target texts in an autoregressive way using the corrupted information provided by the encoder.

In the specific case of BART, as with cBERT, the authors train the model on the target datasets by introducing information from the label. Specifically by prefixing y_i to each sequence x_i in the training data. While cBERT has trained on the target dataset in the MLM task, BART is trained using a similar masking task that masks a continuous set of words (Text Infilling task), allowing BART to predict new words in the masks to provide

augmented data. Specifically, the authors trained BART by masking 40% of the words.

Furthermore, the authors employ two different approaches to augment the training data at the document level. The first is the Backtrainslation (BT) approach [222], which consists of translating the input data into another language and back-translating it into the initial language to obtain substantial changes at the level of the whole text. Specifically, the authors use the pre-trained English-to-German (EN-DE) and German-to-English (DE-EN) machine translation models for BT [128]. The second approach consists of autoregressively fine-tuning the pre-trained GPT-2 model [223] on the target dataset (see section 2.2.4). However, the authors train the model conditionally by adding label information to avoid generating new texts unrelated to the original label. Specifically, by concatenating the input data sequences as follows: y_1SEPx_1EOS...y_nSEPx_nEOS, where SEP denotes a separation between the label and the sentence, and EOS denotes the end of the sentence. To generate the augmented data, the authors provide the following sequence or prompt as input to the model: y_1SEP$w_1, ..., w_k; w_i \in x_i$, to obtain a new sequence $\hat{x}_i$ where $k = 3$ to provide a small context.

The authors evaluated the approaches described above on several datasets, including SST-2. In the case of SST-2, only 10 examples per class were used for training to simulate a few-data scenario. The experiments were repeated 15 times by randomly sampling examples from the training and dev dataset. For each original data, the authors generated a new example for each approach described. The results can be seen in Table 3.12, where the mean and standard deviation of the accuracy on the SST-2 test dataset are given.

	SST-2	
Model (BERT-base)	**Mean**	**STD**
No Augmentation	0.5293	0.050
EDA	0.5382	0.044
BT	0.5745	0.056
cBERT	0.5736	0.067
GPT-2	0.5540	0.067
BART_large (word)	**0.5797**	0.068
BART_large (span)	0.5768	0.071

Table 3.12: Accuracy of the BERT-base model on the SST-2 test dataset from [220] paper. The classifier has been trained with different data augmentation approaches on 10 examples of each class. The best results are shown in bold.

The results show that EDA is the approach that provides the smallest increase in accuracy. It should be noted that the base classifier model is a pre-trained model on large corpora, which may have implicit knowledge that makes the diversity provided by EDA not strongly impact the SST-2 dataset. Furthermore, because EDA performs random substitutions, it is more likely to not preserve the semantics of the input data than other approaches in general. Moreover, GPT-2 shows problems in this regard. Specifically, the

authors note the difficulty of preserving the desired semantics given the label's context, finding that providing a few initial words along with the label may help. On the other hand, the BT approach provides a noticeable increase in accuracy, although this increase is closely related to the quality of the translation models. In comparison, BART stands out due to its ability to generate masked tokens autoregressively, differentiating it from approaches such as cBERT, where different masked words are assumed to be mutually independent at the time of inference.

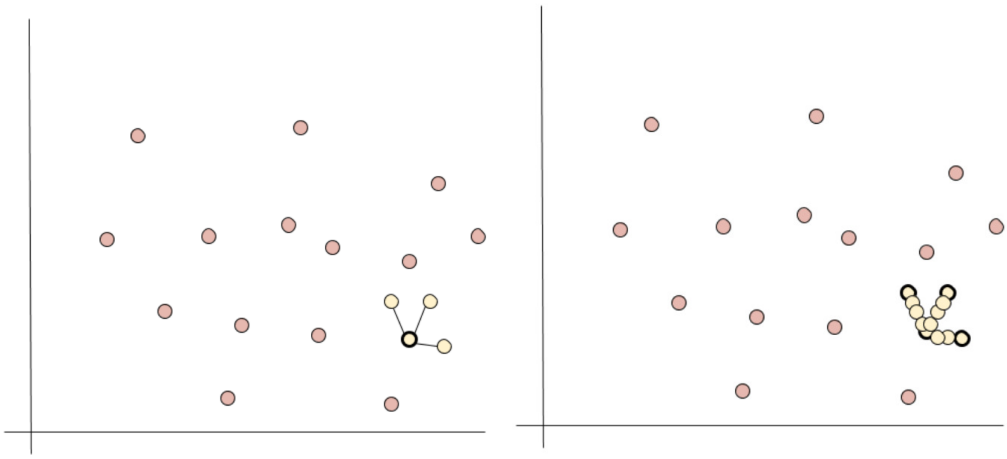
GPT-2 faces difficulties in maintaining the intended semantics when the label is provided at the beginning of the sequence, leading to a dilution of the label information during token generation if a context window is not included. To address this, this Thesis proposes modifying the GPT fine-tuning (GPT-J [224], [225]). The proposed approach is to adjust GPT-J by adding the label at the end of the sequence. In doing so, the GPT-J model considers the previous tokens when generating the probability distribution of the label. This method does not rely on directly adding the label at the prompt, thus eliminating the possibility of generating new examples based on the label alone. However, generating it at the end of the augmented data allows for greater label consistency. This approach avoids extreme smoothing in the probability distribution by lowering the temperature when generating the label and allows for a more precise correspondence between the generated texts and their respective labels (see section 4.3).

Up to this point, several data augmentation approaches related to SA tasks have been described in the data space. However, several data augmentation works have been carried out exploring techniques from the point of view of feature space for computer vision. These approaches have been less employed in NLP tasks since there is no intuitive way to interpret the effect on feature representations of text in continuous spaces. In this context, few works, such as [124], have used oversampling methods such as Synthetic Minority Oversampling Technique (SMOTE) [226], which interpolates feature space representations of the input data. With SMOTE, several nearest neighbors of a specific instance within the feature space are searched for and interpolated with the following equation 3.28:

$$x' = x_i + \lambda * dist(x_i, x_j) \tag{3.28}$$

Where x_i and x_j are instances of the same class, $dist(x_i, x_j)$ is a distance measure, e.g., euclidean and $\lambda \in [0, 1]$. In Figure 3.16, it can be seen how the SMOTE interpolation method produces new synthetic examples

A similar method proposed by [125], called Mixup, generates synthetic samples by linearly interpolating the features of two training samples. It combines the features of two input data points to create a new sample according to the following equation 3.29:



(a) Example of close neighbors of minority class instance (b) Example of synthetic instances generation

Figure 3.16: Illustration of interpolation using SMOTE method.

$$
\begin{aligned}
x' &= \lambda x_i + (1-\lambda)x_j \\
y' &= \lambda y_i + (1-\lambda)y_j
\end{aligned}
\tag{3.29}
$$

Where (x_i, y_i) and (x_j, y_j) are examples from the training dataset and λ is employed similarly to 3.28 or can be sampled from a Beta distribution $Beta(\alpha, \alpha); \alpha \in (0, \infty)$ by adjusting the values of α during training. However, while in computer vision the different examples used for Mixup interpolation meet the requirement of being examples of equal dimension, texts cannot trivially always be represented in the same dimension. Therefore, works such as [227] use representations of the model's internal features to perform the interpolation.

Specifically, in [227], the authors introduce the TMix method. This approach involves randomly selecting two examples from each training batch, obtaining their feature representation from a specific layer m in the model, and then performing an interpolation of these representations. For this purpose, the authors use a BERT-base model and randomly select the m model layer for each training batch to perform the interpolation. Once the interpolation is obtained, the result is propagated forward in the coding model, using the mean of the BERT output features as input to a dense layer to obtain the probability distribution.

The authors minimize the KL divergence between the labels and the classifier probability distribution for training. Specifically, the loss function is defined in equation 3.30:

$$
L_{TMix} = KL(mix(y_i, y_j) \| p(TMix(x_i, x_j); \phi) \tag{3.30}
$$

Where p is the classifier that takes the interpolated representations $TMix(x_i, x_j)$.

The authors evaluate their approach on different datasets, including IMDB. For the simulation of a low-data scenario, the authors train the model using 10, 200 and 2500 examples per class. Specifically, the authors train their models by using 0.08%, 1.6% and 20% of data from the training set on the IMDB dataset. The results are evaluated on the test dataset, and the BERT-base model with TMix obtains an accuracy of 69.3%, 87.4% and 90.3%, respectively. In contrast, without using the TMix method, the BERT model obtains an accuracy of 67.5%, 86.9% and 89.8%.

To conclude, Tables 3.13 and 3.14 show a summary of the results obtained, by the approaches described in this section, based on the amount of labeled data used. The tables show the results obtained with and without data-augmented training. In addition, the increase in performance with Δ is indicated.

		SST-2 (Annotated %)					
Classifier	**DA Method**	**0.3**	**1**	**5**	**10**	**20**	**100**
CNN [122]	No Aug	-	0.55	0.57	0.8	0.81	-
	EDA (x4)	-	**0.8 (Δ0.25)**	**0.805 (Δ0.235)**	**0.81 (Δ0.01)**	**0.815 (Δ0.005)**	-
CNN [104]	No Aug	-	-	-	-	-	0.795
	cBERT (x1)	-	-	-	-	-	**0.821 (Δ0.026)**
LSTM [104]	No Aug	-	-	-	-	-	0.803
	cBERT (x1)	-	-	-	-	-	**0.819 (Δ0.016)**
BERT-base [220]	No Aug (x1)	0.5293	-	-	-	-	-
	EDA (x1)	0.5382 (Δ0.0089)	-	-	-	-	-
	BT (x1)	0.5745 (Δ0.0452)	-	-	-	-	-
	cBERT (x1)	0.5736 (Δ0.0443)	-	-	-	-	-
	GPT-2 (x1)	0.554 (Δ0.0247)	-	-	-	-	-
	BART_large (x1) word-level	**0.5779 (Δ0.0486)**	-	-	-	-	-
	BART_large (x1) span-level	0.5768 (Δ0.0475)	-	-	-	-	-

Table 3.13: Accuracy obtained by models trained with data augmentation methods based on the amount of labeled data available. Results are obtained on the SST-2 test dataset. The best results are shown in bold.

Table 3.13 shows how the results obtained by the different data augmentation approaches provide a greater increase in performance when the number of original data available for training is smaller. Moreover, it specifically benefits classifier models based on convolutional and recurrent networks. However, the margin for improvement decreases when using large pre-trained models such as BERT, where prior knowledge of the lan-

guage is available due to transfer learning mechanisms.

Furthermore, Table 3.14 illustrates that, even when employing large pre-trained models as classifiers, it may exhibit subpar performance for the downstream task when confronted with limited data. In such scenarios, using data augmentation techniques can greatly enhance its performance.

		SST-5 (Annotated %)		**IMDB (Annotated %)**		
Classifier	**DA Method**	**10**	**100**	**0.08**	**1.6**	**20**
CNN [104]	No Aug	-	0.413	-	-	-
	cBERT (x1)	-	**0.423 (Δ0.01)**	-	-	-
LSTM [104]	No Aug (x1)	-	0.402	-	-	-
	cBERT (x1)	-	**0.426 (Δ0.024)**	-	-	-
XLM-R [15]	No Aug	0.233	-	-	-	-
	mBERT (x20)	0.449 (Δ0.216)	-	-	-	-
	SUB^2 (x20)	**0.458 (Δ0.225)**	-	-	-	-
BERT-base [227]	No Aug	-	-	0.675	0.869	0.898
	TMix (x1)	-	-	**0.693 (Δ0.018)**	**0.874 (Δ0.005)**	**0.903 (Δ0.005)**

Table 3.14: Accuracy achieved by models trained using data augmentation methods based on the amount of labeled data available from the SST-5 and IMDB datasets. The evaluation is conducted on the respective test datasets. The most outstanding results are highlighted in bold.

3.8. Data Augmentation approaches for Named Entity Recognition

As mentioned in section 3.7 regarding sentence/document level data augmentation, a common challenge in this approach is to ensure that the generated data maintains a distribution that strikes a balance between being not too similar and not too different from the original data distribution. By achieving this equilibrium, the generated data can contribute new information to aid in the generalization of classification models.

This problem is also found in data augmentation techniques employed for label sequence prediction tasks, including NER. Furthermore, while the approaches explained in the preceding section effectively maintain consistency between new texts and their labels at the sentence/document level, they must tackle the issue of label misalignment at the token level in a NER scenario. Specifically, modifying a word may result in the substitute not aligning with its original labels.

In [228], the authors employ various methods to adapt existing techniques for the NER task. These methods involve making word-level changes to the data space, similar to EDA [122] (explained in section 3.7). Specifically, they utilize four types of transformations along with an IOB labeling scheme (refer to Table 3.15). The different transformations are described below:

First, there is the Synonym Replacement (SR) transformation. The authors use a binomial distribution to randomly select tokens for replacement with synonyms from WordNet. When a token to be replaced corresponds to a "B-EntityType" labeling, the first token of the synonym is labeled as "B-EntityType," and subsequent tokens are labeled as "I-EntityType." If the token corresponds to an "I-EntityType" labeling, all tokens of the synonym receive the "I-EntityType" label.

Next, the Mention Replacement (MR) transformation, similar to SR, randomly decides whether a mention in the sequence should be replaced. If a replacement is chosen, another mention from the training set with the same entity type is selected to ensure consistency between the replaced mention and its label.

The Label-wise token replacement (LwTR) transformation involves replacing tokens with the same label from the dataset. This means tokens labeled with the IOB schema are substituted with others with the same label.

Lastly, the Shuffle within Segments (SiS) transformation divides the sequence into segments based on the IOB labeling. Then, using a binomial distribution, a segment is chosen to be shuffled while maintaining the same label order.

	Sentence										
None	He	reported	no	mention	of	migraine	or	any	additional	neurologic	conditions
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem
SR	He	reported	**non**	mention	of	migraine	or	any	**further**	**neurological**	**ailments**
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem
MR	He	reported	no	mention	of	**stomach**	**ulcers**	or	**any**	**gastrointestinal**	**problems.**
	O	O	O	O	O	**B-problem**	**I-problem**	O	**B-problem**	**I-problem**	**I-problem**
LwTR	**or**	**he**	no	mention	**and**	migraine	**of**	any	**ulcers**	**arrhythmia**	**issues**
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem
SiS	**no**	**reported**	**mention**	**He**	**of**	migraine	or	**additional**	**conditions**	**any**	**neurologic**
	O	O	O	O	O	B-problem	O	B-problem	I-problem	I-problem	I-problem

Table 3.15: Original example and different transformations of augmented examples. The changes are highlighted in bold.

The authors evaluate these transformations on the i2b2-2010 dataset [42]. Specifically, to simulate a scenario of few annotated data, only the first 50, 150, and 500 sentences are selected, where the number of augmented sentences per original is tuned with the validation set from a list of numbers between 1 and 10 augmented sentences per original sentence. Moreover, the authors conducted a study where they trained two models for comparison purposes. The first model was based on a BiLSTM context encoding model, followed by a token-level softmax layer, and utilized GloVe [73] word embeddings. Additionally, they trained a BERT model, specifically pre-trained on scholar articles (SciBERT

[229]), followed by a token-level softmax layer, to assess the impact of data augmentation on both models.

The results obtained on the test set show that the joint use of all the described transformations increases the performance of both models. In the case of the recurrent model, the results were as follows: when using 50, 150, and 500, and the total amount of data, the micro-F1 scores were 17.1%, 43.3%, 54.1%, and 81.1%, respectively. However, when the augmented data was added to the training set, the micro-F1 scores improved to 26.9%, 45.4%, 54.6%, and 81.5%, respectively. As for the SciBERT model, the micro-F1 scores were 35.1%, 62.7%, 70.2%, and 87.8% when using only the original data, but when augmented data was incorporated, the scores improved to 41.5%, 65.2%, 72.3%, and 87.2%.

These results demonstrate the approach's effectiveness, particularly when the number of annotated examples is limited. However, the degree of improvement diminishes as the amount of labeled data increases, both for the recurrent network-based and pre-trained transformer models. This effect is more pronounced in the latter model due to the inherent prior knowledge of the pre-trained model itself.

The diversity of augmented data in approaches like the one mentioned above has certain limitations since other elements present in the training dataset perform the substitutions at the token level or entity mentions. Furthermore, using external resources like WordNet for synonym substitutions may have limited coverage. In order to address these limitations, the authors of [230] propose the Data Augmentation with a Generation (DAGA) approach, which uses generative language models to generate new synthetic data for data augmentation. Specifically, the authors suggest training a language model based on randomly initialized word embeddings with an LSTM network + softmax layer at the token level (see Figure 3.17).

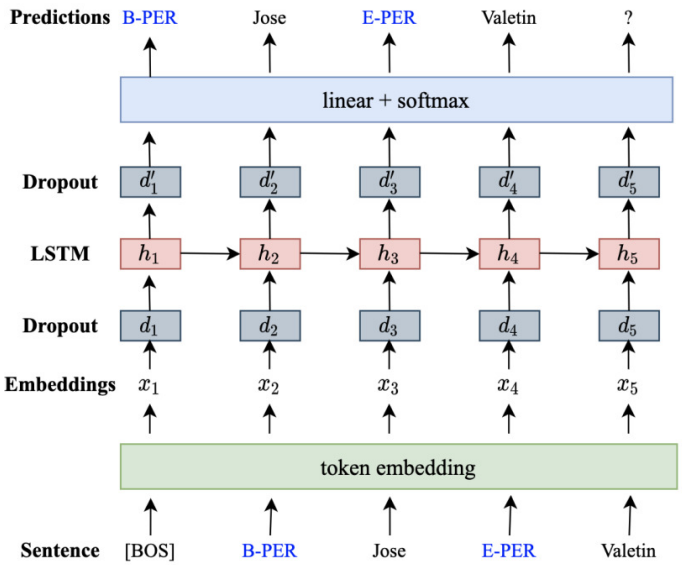


Figure 3.17: Overview of the language model architecture with LSTM Image retrieved from Ding et al., 2020. [230]

The LM-LSTM model proposed by the authors aims to maximize the probability of a word given the prior words, denoted as $p_\theta(w_t|w_{<t})$. However, the authors train the generative model using training data where the labeled sentences are linearized. This scheme incorporates NER labels within the vocabulary, following an IOBES format, e.i., the label of an entity is represented as its previous token. For example, "*John Smith works in London*" is transformed to "*B-PER John E-PER Smith works in S-LOC London*". In this way, the trained model will be able to generate new synthetic data, including the labels within it.

However, the new texts generated will be based on probability distributions within the vocabulary acquired and limited to the training set. Therefore, the authors propose to adopt their approach by adding unlabeled data from external sources such as Wikipedia to generate new texts based on knowledge acquired from more diverse sources. To this end, the authors propose to use conditional generation in their LM-LSTM model so that the autoregressive training of the model takes into account whether the texts come from gold-standard, unlabeled, or labeled data from another external knowledge base, such as dictionaries. Thus, the probability of a word in the autoregressive model is given by $p_\theta(w_t|w_{<t}, c)$, where the context c is considered when generating new data. Specifically, during training, the token c is defined as "[labeled]" for texts with gold labels, as "[unlabeled]" for unlabeled texts, and as "[KB]" for texts that have been labeled with an external source.

The authors evaluate their system on several datasets, including the CoNLL-2003 dataset. For that dataset, the authors train the LM-LSTM model using the above format, employing fractions of the training dataset of 1k, 2k, 4k, 6k, and 8k sentences and adding 10k randomly sampled English Wikipedia as unlabeled data. In addition, the authors perform a comparative evaluation between using unlabeled or labeled data with a dictionary extracted from the training set and Geonames[7].

Once the LM-LSTM model is trained, the authors generate 20k synthetic examples combined with the fractions of the training data described above to train the NER model. The NER model consists of a BiLSTM-CRF model of the Flair [231] framework with pre-trained fastText word embeddings [74]. The results obtained (micro-F1) by the NER model on the CoNLL-2003 test dataset can be seen in Table 3.16, where gen_{ud} refers to the augmented data being generated by the LM-LSTM model trained with unlabeled data, and gen_{kb} refers to it being trained with the dictionary-labeled data.

The results show that the approach is effective, especially when the number of labeled data is limited, and similar to other approaches, the improvement has a decreasing trend as the number of labeled data increases. In addition, employing unlabeled data, where dictionaries have annotated some entities to train the generative LM-LSTM model, has a greater impact in scenarios of few labeled data. Therefore, providing extra information on how to learn token distributions in similar texts and as a function of their pseudo labels

[7]https://datahub.io/core/world-cities

	CoNLL-2003 (number of annotated data)					
DA Method	**1k**	**2k**	**4k**	**6k**	**8k**	**all**
No Aug	0.5806	0.6785	0.7455	0.7716	0.803	0.8304
$+gen_{ud}$	0.6619 (Δ0.0813)	0.73 (Δ0.0515)	0.7808 (Δ0.0353)	0.7975 (Δ0.0259)	**0.8198 (Δ0.0168)**	**0.8433 (Δ0.0129)**
$+gen_{kb}$	**0.6667 (Δ0.0861)**	**0.7354 (Δ0.0569)**	**0.7832 (Δ0.0377)**	**0.7998 (Δ0.0282)**	0.8193 (Δ0.0163)	0.8403 (Δ0.0099)

Table 3.16: Micro-F1 on the CoNLL-2003 test dataset from the work [230]. The classifier has been trained with different data augmentation approaches with 20k synthetic examples. The best results are shown in bold.

helps improve the performance of the NER model.

Since the approach described above relies on the ability of the language model to generate new texts, errors made by the model are propagated throughout the system. Therefore, other work relies on using the prior knowledge acquired by large models based on pre-trained transformers, such as BERT, to perform word substitution like BERT's pre-training MLM task (see section 2.2.4). Specifically, in [232], the authors propose to use the case-sensitive BERT-base model for this purpose.

Such approaches have already been described in the section on augmenting data for SA, e.g., cBERT [104]. However, this approach is based on performing word substitutions focusing on maintaining consistency with sentence-level labels. Therefore, in [232], the authors propose to use an approach called PromptDA, which is similar to a combination between cBERT and BERT-MRC [107] (also described in section 3.4) based on Machine Readable Comprehension (MRC).

Specifically, the authors design two different data augmentation methods. The first, called Label-Conditioned Word Replacement (LC), focuses on augmenting the training data by performing entity substitutions in the labeled data. In contrast, the second one, called Prompting with QA, is based on augmenting the training data by pseudo-labeling unlabeled data from other similar corpora. For more detail, in the first case, the authors pre-train the BERT model by performing random masking of one token per sentence, with a text encoding: $\{[CLS]\, pmt_{l_i}\, [SEP]\, w_{i-1}, [MASK], w_{i+1}, ..., w_T\}$. where pmt_{l_i} consists of a substring defining the entity type of the masked token w_i and '[MASK]' consists of the token to be predicted by BERT based on the context. On the other hand, in the second case, the authors use an approach similar to BERT-MRC to obtain pseudo-labels from unlabeled data.

Furthermore, the authors propose using self-training as a noise reduction strategy in the generated synthetic data. To do so, they first train a NER model on the labeled data and then predict K times the probability distributions of the augmented data maintaining the dropout of the NER model. Once the set of predicted pseudo-labels $\hat{L}_i$ for each token i is obtained, the degree of confidence is obtained as follows:

$$C_{token}(\hat{w}_i, \hat{l}_i) = n(\hat{l}_i)/K \tag{3.31}$$

Where $n(\hat{l}_i)$ represents the frequency of the labels $\hat{l}_i$ of the augmented data $\hat{W} = \{\hat{w}_1, ..., \hat{w}_T\}$ that match $\hat{L}_i$. The degree of certainty at the sentence level consists of the mean of $C_{token}(\hat{w}_i, \hat{l}_i)$. This ranking method is referred to by the authors as Uncertainty-guided Confidence Ranking (UC).

Next, in the self-training approach, the authors use the degree of certainty of the augmented data to compute the trade-off value in the loss function between the labeled data and the augmented data. Specifically, the trade-off value is calculated by normalizing the certainty of the augmented data by the sum of the certainties of all the augmented data.

Once the compensation parameter is defined, the authors sample N examples to incorporate into the training set at each training iteration within the self-training framework.

The authors evaluate their approach on different datasets, including CoNLL-2003 with an IOB labeling scheme.. To simulate a scenario with few-annotated data, they define three fractions of the training dataset. Specifically, they employ 50, 150, and 500, in addition to the entire dataset. For data augmentation, the authors generate 10 new synthetic data for each original training data in the first method. In contrast, they use 100,000 unlabeled sentences from English Wikipedia for the second method. In the self-training framework, the authors set K and N to 10 and 200, respectively, for the noise reduction strategy. Furthermore, the authors evaluate the impact of data augmentation methods on two NER models, namely a BiLSTM-CRF model with word embeddings from the pre-trained GloVe model [73] and a BERT-base model + softmax layer at the token level.

The micro-F1 obtained by the different approaches on the CoNLL-2003 test dataset can be seen in Table 3.17, where, in addition, the authors compare the performance of their approaches with [228] (first paper in this section). These results are effective in few-annotated data scenarios following the trend described in previous works, even slightly improving the results obtained when the full training set is used.

It should be noted that the LC data augmentation method alone provides better results than the QA method. Although the QA method may introduce more noise than LC, the QA method always adds 100,000 augmented data while LC adds augmented data proportional to the amount of original data used (x10). On the other hand, combining both methods provides better results than each of them separately.

Contrasting with the previous work, word substitution-based approaches utilizing pre-trained transformational models can be adapted for the NER task without requiring QA or MRC approaches. A notable example is [233], where the authors propose Masked Entity Language Modeling (MELM), an approach based on word substitution that leverages XLM-RoBERTa MLM pre-training task for NER tasks. However, this approach randomly masks only entity tokens with a masking ratio η.

In this approach, the authors train a model for generating new synthetic data using

		CoNLL-2003 (number of annotated data)			
NER model	**DA Method**	**50**	**150**	**500**	**all**
BiLSTM+CRF	No Aug	-	0.133	0.272	**0.917**
	+EDA [228]	0.063 (Δ0.063)	0.455 (Δ0.322)	0.624 (Δ0.352)	0.901 (Δ-0.016)
	+PromptDA (LC)	0.126 (Δ0.126)	0.514 (Δ0.381)	0.558 (Δ0.286)	0.91 (Δ-0.007)
	+PromptDA (QA)	0.113 (Δ0.113)	0.41 (Δ0.277)	0.531 (Δ0.259)	0.908 (Δ-0.009)
	+PromptDA (LC+QA)	0.149 (Δ0.149)	0.6 (Δ0.467)	0.667 (Δ0.395)	0.915 (Δ-0.002)
	+PromptDA (LC+QA) + UC	**0.152 (Δ0.152)**	**0.612 (Δ0.479)**	**0.672 (Δ0.4)**	0.912 (Δ-0.005)
BERT	No Aug	-	0.550	0.588	0.924
	+EDA [228]	0.011 (Δ0.011)	0.614 (Δ0.064)	0.642 (Δ0.054)	0.913 (Δ-0.011)
	+PromptDA (LC)	0.331 (Δ0.331)	0.582 (Δ0.032)	0.67 (Δ0.082)	0.916 (Δ-0.008)
	+PromptDA (QA)	0.235 (Δ0.235)	0.569 (Δ0.019)	0.669 (Δ0.081)	0.92 (Δ-0.004)
	+PromptDA (LC+QA)	0.434 (Δ0.434)	0.669 (Δ0.119)	0.697 (Δ0.109)	0.919 (Δ-0.005)
	+PromptDA (LC+QA) + UC	**0.441 (Δ0.441)**	**0.672 (Δ0.122)**	**0.701 (Δ0.113)**	**0.925 (Δ0.001)**

Table 3.17: Micro-F1 on the CoNLL-2003 test dataset from the [232] work. The NER models have been trained with different data augmentation approaches: LC adds 10 augmented data per original data, and QA adds 100,000 augmented data in each configuration. The best results are shown in bold.

training data where labeled sentences are linearized. NER labels are incorporated within the vocabulary using this scheme. However, unlike DAGA [230] (previously described in this section), the authors adopt an IOB format, representing the label of an entity as its preceding and following token. This distinction arises because DAGA generates new examples through an autoregressive approach, whereas MELM employs an autoencoder approach to generate new surrogates based on the bidirectional context of the texts. An illustration of how MELM generates new synthetic data can be seen in Figure 3.18.

After training MELM, a number of R synthetic data are generated for each original data in the training dataset. This process involves randomly masking each entity, employing dynamic masking for entities composed of n tokens. The masking ratio is determined by a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, where the value of the variance σ^2 is set as $1/n^2$. This step ensures diverse masking for the augmented data associated with each original data point. Furthermore, the authors randomly sample surrogate tokens from the first k

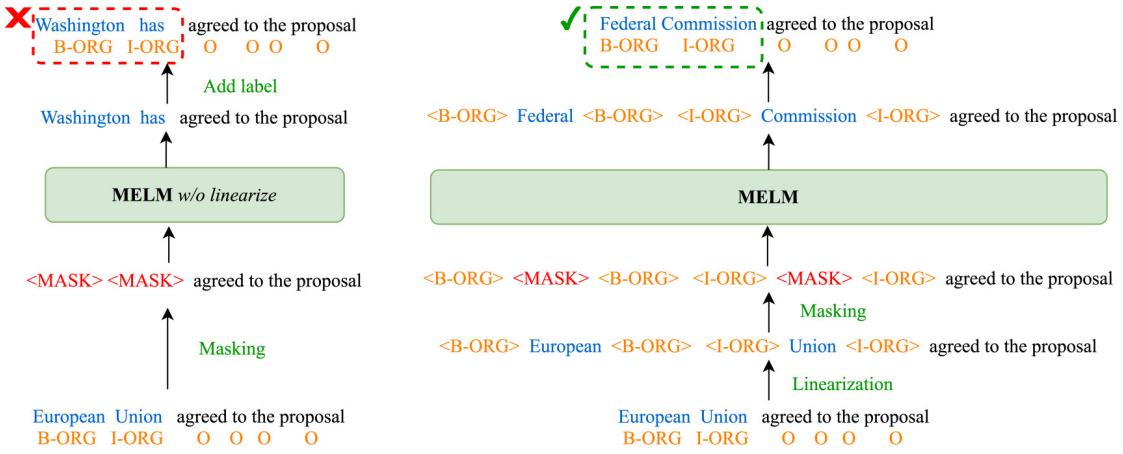


Figure 3.18: Overview of Masked Entity Linguistic Modeling (MELM) for data augmentation. On the left without sequence linearization and on the right with the proposed sequence linearization. Image retrieved from Zhou et al., 2022. [233]

tokens of the probability distribution across the vocabulary to address the potential repetition of the original tokens in the predicted entity tokens. Finally, the synthetic data are added to the training set to fine-tune another XLM-RoBERTa-Large model, followed by a CRF decoder layer, as the NER model.

The authors assessed the performance of their MELM approach using various datasets. Specifically, they conducted experiments on the CoNLL-2003 dataset, where subsets of 100, 200, 400, and 800 sentences were selected from the training set to simulate a scenario with limited annotated data. Additionally, the authors assigned the hyperparameters η, μ, and R values of 0.7, 0.5, and 3, respectively. The results in terms of micro-F1 are presented in Table 3.18, along with a comparison to previously described approaches in this section, such as DAGA [230] and EDA (label-wise) [228].

		CoNLL-2003 (number of annotated data)			
NER model	**DA Method**	**100**	**200**	**400**	**800**
XLM-R-large +CRF	No Aug	0.5057	0.7464	0.8185	**0.8635**
	+EDA [228]	0.6134 (Δ0.1077)	0.7682 (Δ0.0218)	0.8462 (Δ0.0277)	0.8672 (Δ0.0037)
	+DAGA [230]	0.6806 (Δ0.1749)	0.7911 (Δ0.0447)	0.8436 (Δ0.0251)	0.8661 (Δ0.0026)
	+MELM w/o linearize	0.7001 (Δ0.1944)	0.8177 (Δ0.0713)	0.8516 (Δ0.0331)	0.8735 (Δ0.01)
	+MELM	**0.7521 (Δ0.2464)**	**0.8291 (Δ0.0827)**	**0.8573 (Δ0.0388)**	**0.8759 (Δ0.0124)**

Table 3.18: Micro-F1 on the CoNLL-2003 test dataset from the [233] work. The NER models have been trained with different data augmentation approaches. The best results are shown in bold.

As shown in Table 3.18, the results provided by training with augmented data improves the performance of the NER model following the common trend in all the described approaches. However, it is worth noting that word substitution approaches using transformers can outperform even DAGA when linearized sentences are not used to train the MELM model. Although the non-linearized MELM approach can provide new context-aware tokens, these tokens could be unrelated to the original entity label. This is why MELM with linearized sentence training performs best in work proposed by [233].

Several studies have been conducted on data augmentation approaches in the data space. However, there has also been research focusing on the feature space. One such example is [69], where the authors propose an approach based on the TMix method, which is explained in the augmented data for SA section (section 3.7).

In contrast to TMix, which performs label interpolation at the sentence/document level, in the context of NER, label interpolation must consider a sequence of labels instead of a single label per example. To address this, the TMix method can be easily adapted to the NER context. However, directly interpolating examples would introduce significant noise due to potential variations in data and label sequence distributions. To tackle this issue, the authors introduce a method called Local Additivity based Data Augmentation (LADA), along with two different interpolation techniques: (i) Intra-LADA, which performs interpolation between permutations of the input and label sequences within the same example, and (ii) Inter-LADA, which performs interpolation between examples with similar distributions sampled using a KNN strategy.

While Intra-LADA limits the context switching between the samples to be interpolated, it also limits the ability to generate diverse examples. Therefore, in order to define a trade-off between noise and regularization, in Inter-LADA, the authors sample the examples to permute (x', y') with each example (x, y) by defining a combination weight between two different distributions:

$$P_{inter}(x'|x) = \begin{cases} \frac{\mu}{k} & x' \in \text{Neighbor}_k(x) \\ \frac{1-\mu}{|S|} & (x', y') \in S \end{cases} \tag{3.32}$$

Where μ is the weight of combining two distributions treated as a hyperparameter, k is the number of neighbors and S is the whole training dataset. The sampling is performed on the KNNs with a probability of μ or the complete training dataset with a probability of $1 - \mu$.

The authors used the CoNLL-2003 dataset, employing an IOB labeling scheme to assess their system's performance. Specifically, they trained on various fractions of the training set, 5%, 10%, and 30%, to simulate few-annotated data scenarios. Regarding NER models, a comparison was conducted between a BiLSTM-CRF model utilizing Flair pre-trained word embeddings and a BERT-based multilingual case-sensitive model. Due to the structural modifications introduced by Intra-LADA, the authors did not evaluate Flair's performance based on an LSTM-CRF architecture. In the case of BERT, Intra-

LADA employed layer 12 to extract the interpolation representations. In the case of Inter-LADA, layers 8, 9, and 10 were employed for BERT, while Flair embeddings were used for the BiLSTM-CRF model. The authors set the number of KNN to 3 and μ to 0.5 for all cases.

The results for Micro-F1 obtained through different approaches are presented in Table 3.19, showcasing the effectiveness of the methods. Notably, Inter-LADA outperforms Intra-LADA. However, the authors evaluate both methods simultaneously by generating interpolated data using either approach based on a π-ratio, indicating the probability of selecting one method over the other. In this study, the authors set π to 0.3, representing the probability of employing the Intra-LADA method, while $1 - \pi$ represents the probability of employing the Inter-LADA method.

		CoNLL-2003 (% Annotated data)		
NER model	**DA Method**	**5**	**10**	**30**
Flair	No Aug	0.7932	0.8631	0.8996
	+Intra-LADA	-	-	-
	+Inter-LADA	**0.8084 (Δ0.0152)**	**0.8633 (Δ0.0002)**	**0.9061 (Δ0.0065)**
BERT	No Aug	0.8328	0.8685	0.8928
	+Intra-LADA	0.8352 (Δ0.0024)	0.8754 (Δ0.0069)	0.8931 (Δ0.0003)
	+Inter-LADA	0.846 (Δ0.0132)	0.8781 (Δ0.0096)	0.8968 (Δ0.004)
	+Intra&Inter-LADA	**0.8485 (Δ0.0157)**	**0.8785 (Δ0.01)**	**0.8987 (Δ0.0059)**

Table 3.19: Micro-F1 on the CoNLL-2003 test dataset from the [69] work. The NER models have been trained with different data augmentation approaches. The best results are shown in bold.

In conclusion, the works discussed in this section demonstrate significant overlap in the methods employed for comparison, making it appropriate to refer to the tables summarizing the results as Tables 3.19, 3.18, and 3.17. Upon reviewing all the results, it becomes apparent that most substantial performance improvement over the CoNLL-2003 dataset is achieved by [233], which focuses on generating new synthetic data through token-level substitution using pre-trained transformer models. However, it is worth noting that all approaches enhance the performance of NER models in scenarios with limited annotated data. As the amount of labeled data increases, these improvements tend to diminish, except in the case of self-training frameworks, where the issue lies in training instability caused by the propagation of model biases.

3.9. Conclusions

Throughout this Chapter, different deep learning approaches used for NLP tasks have been described, with emphasis on the SA and NER tasks. Within each task, approaches based on recurrent networks (RNN), convolutional networks (CNN), and models based on transformers and attention mechanisms are widely used approaches that provide state-of-the-art results in those tasks.

Each approach described has different qualities. Those based on CNN networks are able to automatically extract local features within texts in the form of n-grams with parallel execution, allowing fast training and inference times [23], [161], [162]. Recurrent networks, on the other hand, are able to extract contextual information by capturing dependencies within the sequence of texts at the cost of higher computational cost, making them very effective when working in NLP due to the sequential dependencies characteristic of texts [23], [39], [101], [102], [108], [163]. However, these types of networks suffer from retaining long dependency information inherent in longer texts [77], [98].

While deep learning approaches are able to learn task-relevant features from the data themselves, they rely on having a large amount of annotated data. For this reason, and since large annotated corpora are not usually available in real-world applications, research has been conducted on transfer learning mechanisms. Such mechanisms are based on pre-training models on different tasks where annotated data is available or using unsupervised learning such as autoregression or autoencoding, allowing models to obtain prior knowledge to help in the target task [23], [54]–[56], [102], [108], [183].

In recent years, transformer-based approaches have gained dominance, capturing dependencies within texts through attention mechanisms and mitigating problems characteristic of recurrent networks such as vanishing gradient. However, due to the complexity of this type of model, large amounts of data are required to fit them into a given task. For the latter reason, research has been extended in applying pre-trained models on large corpora to use the implicit knowledge acquired as a starting point to fit such models to downstream tasks.

All the approaches described so far have provided state-of-the-art results using learning transfer mechanisms to mitigate the dependence on large annotated corpora. However, transfer learning is particularly useful when the transferred knowledge comes from similar tasks or corpora [234]. For this reason, this Thesis has detailed different works focused on weakly supervised learning and, more specifically, on semi-supervised and data-augmented approaches.

In this Chapter, several semi-supervised learning approaches for SA and NER tasks have been described, showing how unlabeled data can improve the performance of deep learning models. However, this requires that the distribution of the data $p(x)$ over the input space contains information about the posterior distribution $p(y|x)$ [111]. semi-supervised learning should therefore be treated as another approach within the set of algorithms con-

sidered for further evaluation in a particular scenario.

The described works demonstrate how semi-supervised learning approaches utilize reconstruction or regularization terms between distributions derived from unlabeled data to enhance the model's knowledge. Some methods explicitly incorporate these terms into their loss functions [16], [65], [193], [198], [205], [210], [212], while others propagate labels to unlabeled data, incorporating them into iterative training processes based on confidence metrics [68], [195], [215]. Although these approaches significantly improve performance in scenarios with limited annotated data, their effectiveness diminishes as the number of available annotated data increases. This is because deep learning algorithms become increasingly proficient at learning linguistic and semantic patterns with a larger annotated dataset. However, wrapper methods like self-training enhance the confidence of pseudo-labeling as the number of labeled data increases, being their main problem the propagation of model biases and training instability due to pseudo-label noise.

On the other hand, data augmentation shares several similarities with that described above. While it does not require unlabeled data for a particular task, it uses prior knowledge to transfer learning through new data whose distributions can provide information to the models. This approach employs word [15], [104], [122], [220], [228], [232], [233], phrase [15], [220], [232], [233], or document-level [69], [220], [227], [230] transformation methods to generate new data not previously included in the training set. However, mechanisms must be employed to constrain such data to be related and consistent with the original data while increasing the diversity of the training dataset .

As with semi-supervised learning, employing data augmentation approaches helps to improve model performance in few-annotated data scenarios. However, the gain also has a decreasing trend as the number of available annotated data increases. Hence, the more training data available and the more implicit knowledge of the pre-trained models, the more difficult it becomes to provide diversity to the training set.

To summarize, this Chapter has described various related works for SA and NER tasks utilizing deep learning approaches. These approaches have been discussed in ideal scenarios with ample annotated data and with weakly annotated data, where deep learning methods have been adapted to semi-supervised and data-augmented learning techniques. Specifically, these techniques have been applied in scenarios involving subjective labels, such as SA, and situations with a scarcity of labeled data for training.

After discussing the different works, it has been demonstrated that the proposed approaches are effective. However, while deep learning approaches using large transformer-based pre-trained models are currently prevalent due to their state-of-the-art performance [54]–[56] (see Tables 3.3, 3.4 and 3.5), it remains unclear which semi-supervised deep learning approach is superior to others. This is not the case for data augmentation, as there is a widespread inclination to leverage the implicit knowledge of large pre-trained models to generate a greater diversity of augmented data (see Tables 3.19, 3.18, 3.17 and 3.13). Furthermore, it is worth noting that in recent years, these models have been in-

creasingly used in zero or few-shot approaches to generate new data [235], [236]. This trend is attributed to the capability of pre-trained models trained on extensive corpora.

Chapter 4

Sentiment Analysis in Drug Reviews

Since the beginning of the century, millions and millions of users' reviews are available on the Internet. In fact, online users are generating a huge amount of reviews on fresh topics every second, making it more difficult to massively label this amount of data in a short time [237].

As a result, SA (a computational process that evaluates the opinion or emotion expressed about something in a text [18]) has become one of the most fruitful research fields in NLP. This task covers a wide range of domains and could contribute to a variety of applications like financial forecasting [19], political strategy planning [20], public opinion analyzing[21], online decision making [22], and pharmacovigilance [23], among many others. In the pharmaceutical field, the automatic analysis of online user reviews allows us to analyze large amounts of users' opinions and obtain relevant information about the effectiveness of drugs and their side effects, which could be used to improve the pharmacovigilance systems [35].

Pharmacovigilance, also known as drug safety, tries to prevent drug side effects [34]. The early detection of side effects depends on clinical trials and specific testing protocols, which are usually conducted under given conditions with a limited number of test subjects and time. As a consequence of the design of clinical trials, discrepancies in patient selection and treatment conditions can have a significant impact on drug efficacy and detection of potential adverse effects [238]. Additional information sources, such as user reviews, can offer great potential in this regard [239]. Automatic detection of negative user reviews could alert to unexpected drug behaviors and obtain relevant information to improve pharmacovigilance systems [33]. Moreover, the identification of positive user reviews could help to assess the drug efficacy and even discover new uses for a drug [240].

Throughout the years, approaches in SA have evolved from deterministic rule-based systems [241] to machine-learning [242] and hybrid methods [243]. Over the last years, the emergence of deep learning techniques has brought remarkable breakthroughs in NLP, providing state-of-the-art results in many tasks [244]. SA is not oblivious to this success, and a wide range of deep learning-based systems have also achieved state-of-the-art re-

sults on standard SA datasets [56], [245]–[249]. However, patient SA, specifically of their experience with drugs, has received less research attention [47], [59], [60]. This is a challenging task because user opinions can have different degrees of subjectivity and could be focused on other different aspects (such as the quality of healthcare service or healthcare professionals), introducing noise in the classification task [250].

This chapter describes and compares several deep learning-based approaches for SA of drug reviews. First, supervised approaches are explored to evaluate their performance on a weakly labeled dataset. Subsequently, semi-supervised and data augmentation approaches are employed to mitigate the potential lack of labeled data [251].

4.1. Supervised Approaches

4.1.1. Introduction

In supervised approaches, different algorithms fit functions to the training examples. These examples consist of a pair of elements, including an input vector and the desired output. The fitted functions must be able to infer the outputs from previously unseen examples, which requires the learning algorithm to generalize using the training data. SA at the document (or sentence) level can be considered as a text classification task where the fitted function must be able to assign a polarity to each text [2], [18].

The ideal supervised learning scenario is based on sufficient annotated data with the same distribution as the test sets. However, obtaining sufficient quantity and quality of data in real scenarios is an expensive and time-consuming task [8]. Transfer learning focuses on using knowledge from related domains to mitigate the abovementioned problem. This is achieved by using previously fitted models in other tasks related to the target task, helping to generalize when training new models for similar tasks [234].

Due to the above, this section evaluates and compares the performance of several models using transfer learning against randomly initialized models. Specifically, we use word embeddings and language models previously trained on similar tasks in this experimentation. Deep learning approaches such as CNN, LSTM and BERT, three state-of-the-art deep learning models that have been used in different NLP tasks, including SA, are compared. In addition, various combinations of these models are compared to exploit their advantages.

Most previous studies only deal with two or three polarities (positive, negative and neutral) [140]. A finer-grained polarity classification is more challenging due to the larger number of classes [252]. To the best of the knowledge of the researcher, this is the first attempt to conduct a finer-grained polarity classification of drug reviews.

4.1.2. Dataset

The dataset used, proposed in [47], is a collection of drug reviews taken from the Drugs.com website, a widely used online resource for detailed information on drugs and pharmaceuticals, as well as for access to reviews and user experiences on different drugs. Each drug review includes an integer number (rating) from 1 (most negative) to 10 (most positive), which reflects the patient's degree of satisfaction with the drug. For example, a patient with *atrial fibrillation* posted the following comment: "*Only on it for 8 days. After 5 days started having shortness of breath, muscle spasms in upper back, pounding heart rate, fatigue, stiffness in neck and face*". This comment refers to the drug *Flecainide* and describes a series of adverse effects it has produced within a few days. The score of the drug provided by the patient is negative with a value of 1.

The corpus contains a total of 215,063 drugs reviews. The corpus is split into training and test datasets in the ratio 75:25, maintaining in both the proportion of the classes by stratified random sampling. Additionally, 15% of the total training dataset is used as development dataset, which allows us to learn the best hyperparameters of the different models.

Moreover, the creators of this corpus also grouped the drug reviews according three levels of polarity based on the review's rating: negative (class 0, rating<=4), neutral (class 1; rating in [5,6]) and positive (class 2; rating>=7).

In general, SA is seen as a text classification task [56], [134], [253]. For this reason, the task is approached using the three polarities proposed in [47], but also as a more challenging classification task using 10 classes. As mentioned before, each drug review is classified by an integer number (rating) from 1 to 10, which refers to the overall patient satisfaction. Thus, these ratings are considered as the classes for the problem.

For 10-class dataset, Figure 4.1 shows the distribution of the different drug reviews according to their class (degree of satisfaction). There is a strong unbalanced distribution of the classes, predominating those with more polarized scores (such as 1, 8, 9 and 10).

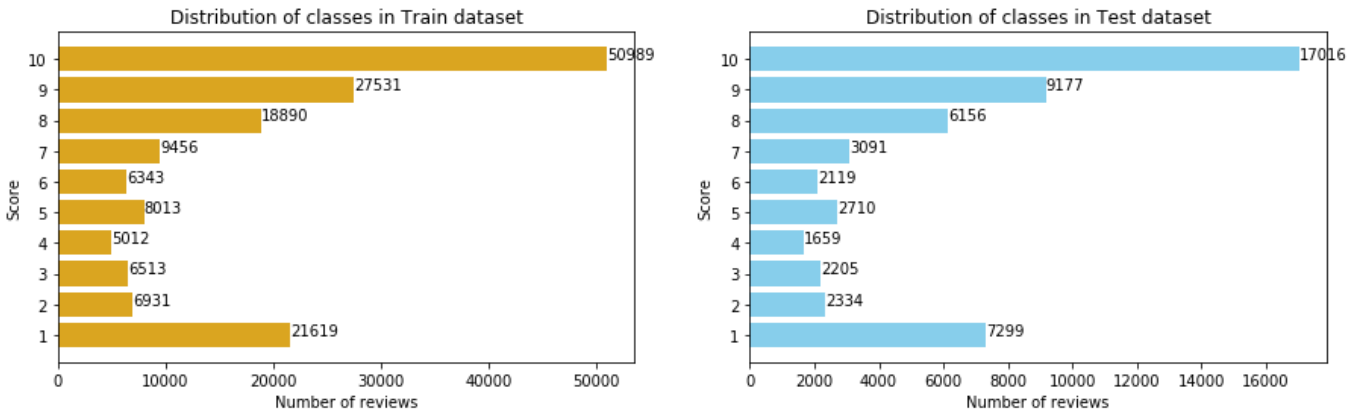


Figure 4.1: Class distribution in training and test datasets (10 classes)

The 3-class dataset also has an unbalanced distribution of the classes (see Figure 4.2). This figure shows that the distribution of the three classes in test and training sets is similar. The reviews with positive polarity (class 2) represent approximately 66%, while the reviews with negative polarity are around 25% and only 9% for the reviews with neutral polarity (class 1).

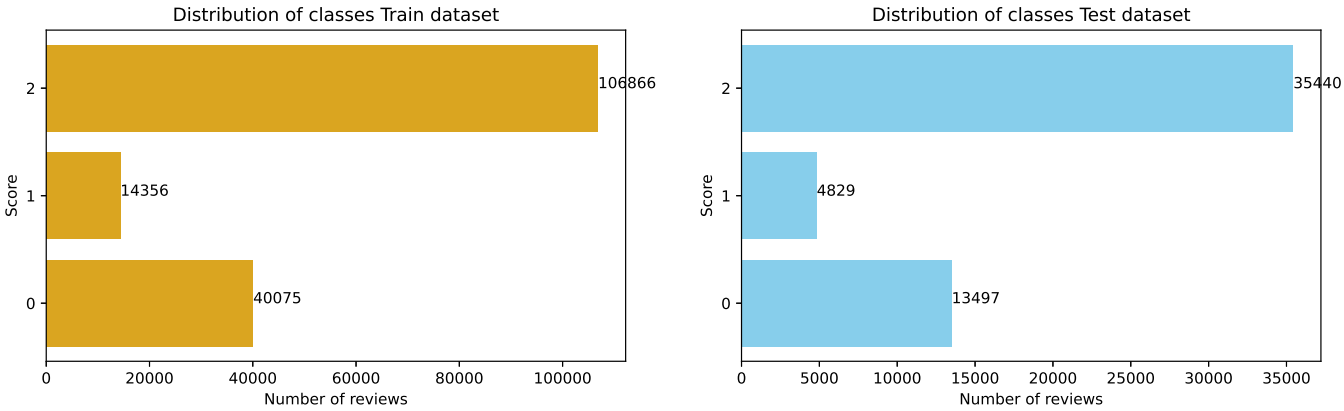


Figure 4.2: Class distribution in training and test datasets (3 classes)

4.1.3. Data Preprocessing

This section describes the preprocessing task applied to transform the input texts (reviews) into a suitable representation for the models. In addition, the different pre-trained Word2vec [72] word embedding models (see section 2.1.1) used to initialize the networks are described.

- The word embeddings model described in [254], which from now on will be called "WE model A", was trained on a collection of texts from PubMed, PMC, and Wikipedia in English (2013 version). This model contains a vocabulary of 5,443,656 words and the dimension of the word embeddings is 200.
- The word embeddings model presented in [255], which from now on will be called "WE model B", was trained using more than a million sentences in English from tweets about drugs. It contains a vocabulary of 26,278 lemmatized words and the dimension of the embeddings is 150.
- The word embeddings model described in [256], which from now on will be called "WE model C", was trained on a collection of twitter and Wikipedia (2017 version) texts in English, Reuters' news articles, the UMBC web-based corpus, and The "One Billion Word Language Modeling Benchmark". This model contains a vocabulary of 2,156,970 words and the dimension of the embeddings is 300.

To represent drug reviews using these pre-trained word embedding models, it is necessary to preprocess the reviews. Specifically, texts are tokenized and numerical expressions

are blinded to normalize them and avoid biases arising from specific numerical values that may be irrelevant to the sentiment of the text. Moreover, when the "WE model B" is employed, tokens are also lemmatized using NLTK[8] lemmatizer to obtain as much vocabulary coverage as possible. The pre-processed drug reviews with lemmatization resulted in a vocabulary of 45,278 tokens. On the other hand, when the lemmatization step was omitted, this resulted in a vocabulary of 49,339 tokens.

Due to the different lengths of the drug reviews, it was necessary to pad and truncate the texts to equal their lengths, as the topology of neural networks is designed to work with fixed-dimensional input tensors. Based on the cumulative distribution function over the length of reviews (see Figure 4.3), almost 100% of processed texts have a length of less than or equal to 250 tokens (2,006 is the maximum length). Thus, the length of the processed texts was set to 250 tokens.

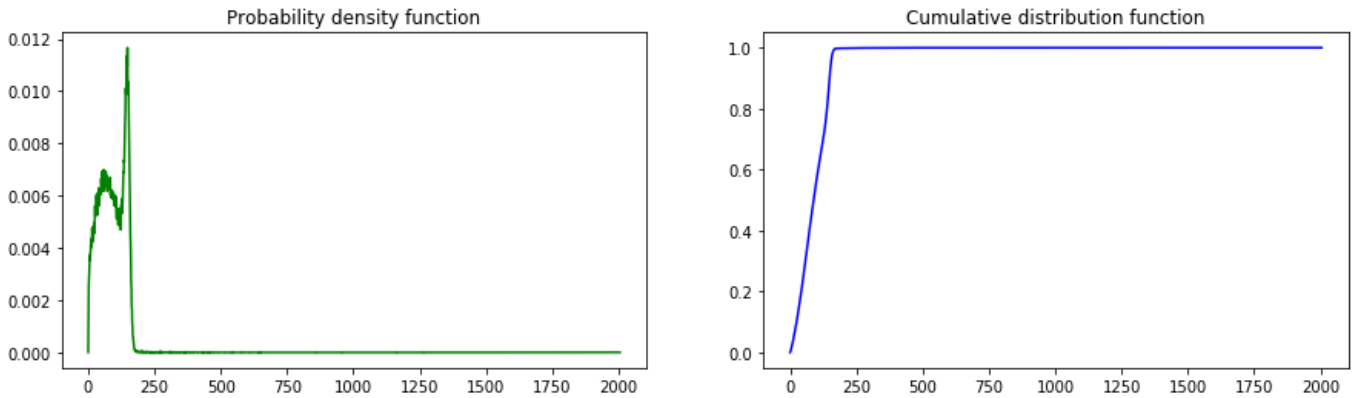


Figure 4.3: Probability density function and Cumulative distribution function over the length of reviews

4.1.4. Methods

This section describes several deep learning architectures for SA of drug reviews. In particular, different deep learning architectures are proposed: 1) a simple CNN, 2) a bidirectional LSTM, 3) a hybrid model comprising a CNN followed by a bidirectional LSTM, 4) a hybrid model including a CNN concatenated with a bidirectional LSTM, and 5) a hybrid model consisting of a bidirectional LSTM followed by a CNN. In addition, a sixth model based on BERT-small with an LSTM as a classifier and a seventh model based on BERT-base are proposed.

CNN and LSTM

The first of the deep learning approaches consists of a simple CNN architecture. This type of network (detailed in section 2.2.1) can extract representative patterns that describe the

[8]http://www.nltk.org/

text in the form of n-grams [257]. The CNN architecture has a convolution layer, where different filters operate sliding along the matrix of word embeddings of each drug review, producing a mapping of features of the reviews as output.

This architecture consists of 64 filters with a vector window size of 2, 3 and 5 tokens. A linear rectification unit (ReLU) [95] has been employed as an activation function to avoid gradient disappearance problems [258]. Once the feature mapping is obtained, a pooling based on the maximum of the values of each convolution is employed. This method is used to prevent the padding operation from negatively affecting the representation of the texts [259].

The second approach is based on a bidirectional LSTM, which processes the input text storing the semantics of the previous and future tokens. This type of recurrent network (RNN) is able of capturing contextual information and long-term dependencies[260]. LSTM layers (detailed in section 2.2.3) are composed of recurrently connected memory blocks where each of the memory cells contains three multiplicative gates. These gates are able to use, store, and forget information for long periods, solving the vanishing gradient problem. This bidirectional LSTM has a hidden state dimension of 250 for the forward and backward layers and hyperbolic tangent (tanh) as activation function.

Hybrid models combining CNN and LSTM

The following approaches consist of a combination of those described above. These combinations aim to capture both the local features of the texts and their global and temporal semantics [261]–[263]. The combination of these features may provide more discriminative information to perform the inference of the different classes.

For the third approach, the same CNN architecture described above is employed to extract a sequence of representations of higher-level texts. These representations are conformed by the concatenation of the pooling layers and are used to feed a bidirectional LSTM to capture the contextual information of the local features. For the convolutional layer, 64 filters with a window size of 2, 3 and 5-word vectors with ReLU activation and a maximum clustering layer are used. For the bidirectional LSTM layer, a hidden state dimension of 200 is used for the front and back layers and the hyperbolic tangent (tanh) as activation function.

For the fourth approach, a parallel combination of the bidirectional CNN and LSTM architectures described above is employed to provide both contextual and local features from the text. With this parallel combination, it is possible to extract and concatenate both types of features directly from the word embedding representation of the reviews. As can be seen in Figure 4.4, this architecture is composed of 64 filters with a window size of 2, 3 and 5-word embeddings and a max-pooling layer for obtaining the local features. To obtain the contextual features, the architecture is also composed of a bidirectional LSTM layer with a hidden state dimension of 250 for the forward and backward layers. A linear

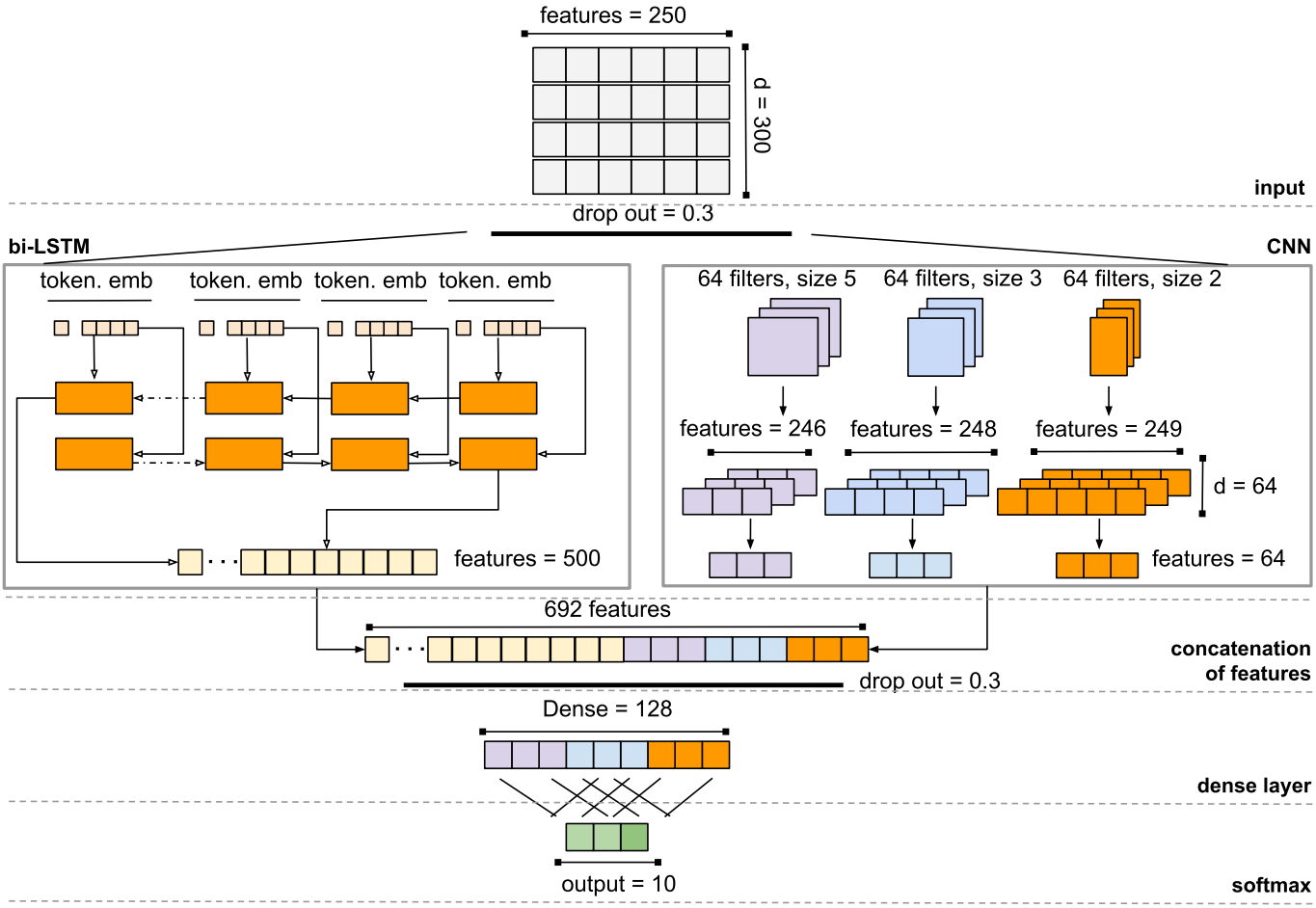


Figure 4.4: Overview architecture of the concatenation of the CNN and LSTM models

rectification unit (ReLU) and a hyperbolic tangent (tanh) are used as activation functions respectively.

For the last approach, an architecture is used with the objective of extracting local features from the abstract representation provided by a bidirectional LSTM layer. This abstract representation contains contextual information from the reviews. As can be seen in Figure 4.5, the word embedding of each review is used as input to a bidirectional-LSTM layer with a hidden state dimension of 250 for the forward and backward layers. This layer does not return only the sequence obtained in the last time step, but it returns the complete sequence of features with the information of each time step. Once the complete sequence of hidden states is obtained, a convolutional layer is used to extract the local information between the different time steps. To obtain the local features, 64 filters are employed with a window size of 2, 3 and 5 steps. A max-pooling layer is used to provide the most relevant features. These features, obtained by each type of convolutional filter, are finally concatenated. For the LSTM+CNN architecture, a linear rectification unit (ReLU) and a hyperbolic tangent (tanh) are employed as activation functions for the convolutional and bidirectional-LSTM layers respectively.

Finally, the last layer of all the architectures described above is composed of fully

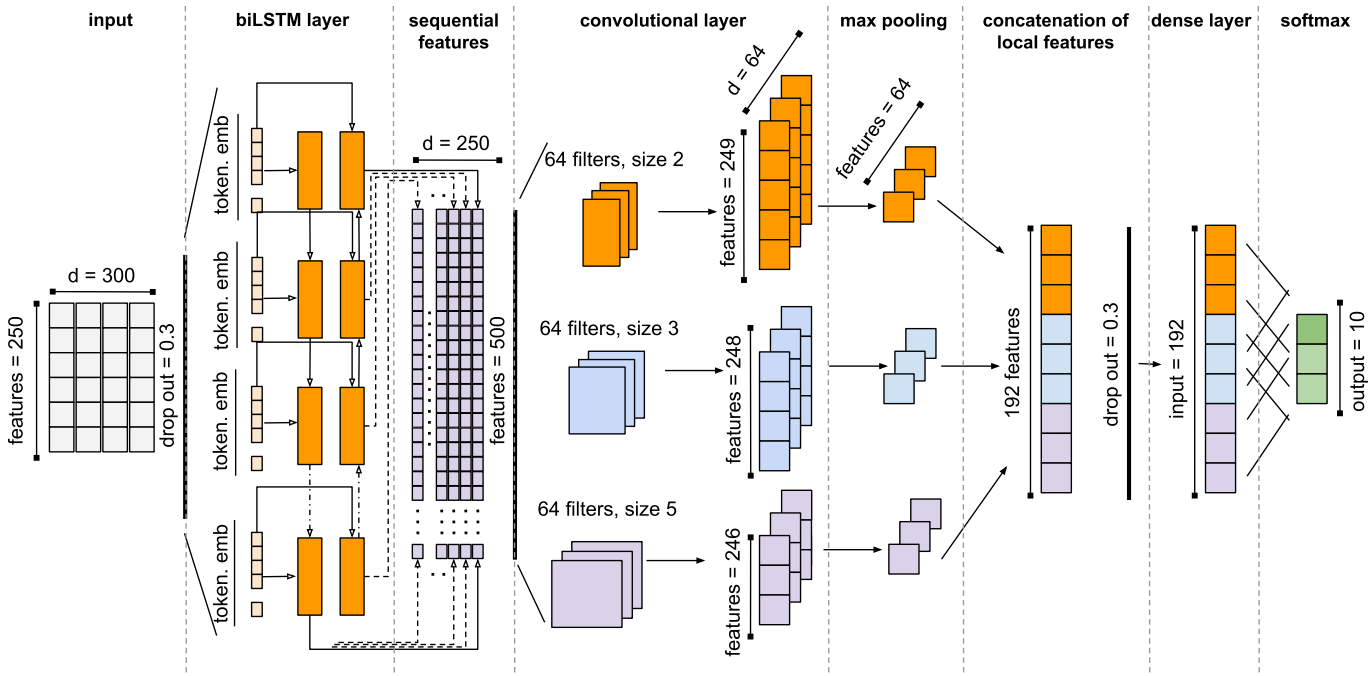


Figure 4.5: Overview architecture of the LSTM + CNN system

connected perceptrons. In this layer, the ReLU activation function is used in order to improve class prediction. At this point, the different vectors representing each of the texts are connected to a Softmax layer, which predicts the score for each review.

For the training of the different models, it was used the ADAM optimizer [264], with a learning rate of 0.001, a batch size of 200 reviews during 200 epochs and categorical cross-entropy as a loss function.

BERT-small and LSTM model

The use of BERT as a linguistic model to represent drug reviews was also explored. The main advantage of BERT compared to Word2Vec models is its ability to generate contextualized word embeddings. Thus, in a word2vec model, the word "*sentence*" will have the same representation for different sentences (for example, "*The judge gave a sentence*" and "*A sentence is a linguistic structure*"). However, with BERT, the representation of the word "*sentence*" will be different for those sentences. This is due to BERT generates contextualized word embeddings taking into account the context to provide the most accurate embedding. On the other hand, fine-tuning BERT for a specific task is inefficient from a parameter point of view. Due to the computational cost, it was used a pre-trained BERT model with four encoder layers [177], which is known as the BERT-small model.

Moreover, as a transfer mechanism, the adaptor modules proposed by Houlsby et al. [265] were used to considerably reduce the number of trainable parameters. The use of adapter modules consists of injecting new layers into the original BERT model. Adapter modules are a bottleneck architecture that projects the input features to a smaller dimen-

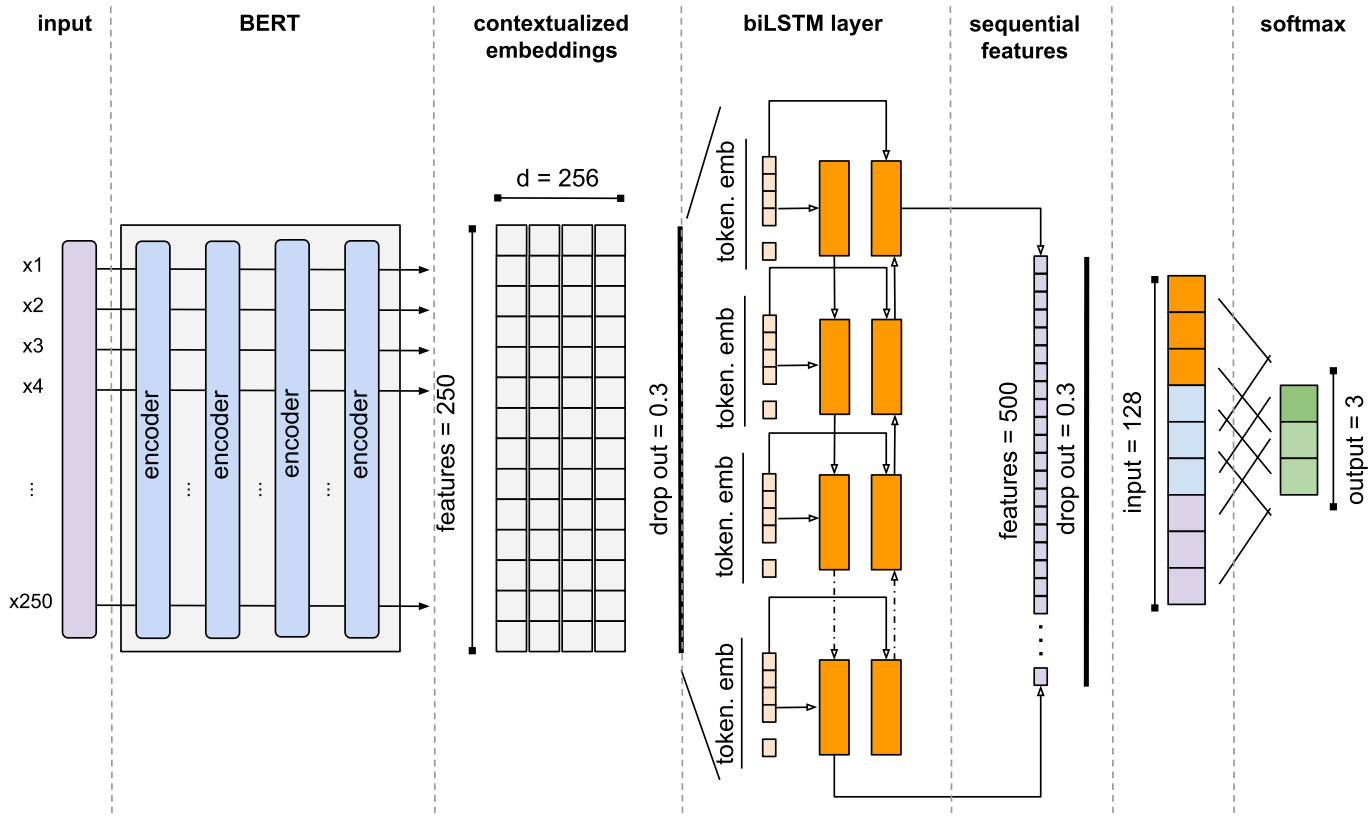


Figure 4.6: Overview architecture of BERT + LSTM model

sion m (m was set to 4 for this experimentation), applies non-linearity and projects the features to the original dimension. The adapter module also incorporates an internal skip-connection to avoid problems with near-zero initialization. The original BERT model weights remain unchanged during fine tuning for the specific task.

To represent the drug reviews based on BERT, the texts were tokenized using the vocabulary of BERT's pre-trained model and the tokenizer provided by bert-for-tf2[9]. The pre-processed texts resulted in a vocabulary of 30,522 tokens. As a last preprocessing step, based on the cumulative distribution function over the length of the reviews (see Figure 4.3), the length of the processed texts was set to 250 tokens.

Since the output of the BERT model consists of a representation of the texts using contextualized word embeddings, top layers can be added for the classification task. In particular, as can be seen in Figure 4.6, it was used a bidirectional LSTM layer followed by a fully connected perceptron and softmax layers. Bi-LSTM was chosen because it is simpler than the hybrid architectures and achieves similar results, as will be seen in Section 4.1.5.

For the training of the model, it was used the ADAM optimizer [264], with a learning rate of 0.001, a batch size of 200 during 200 epoch and categorical cross-entropy as loss function.

[9]https://pypi.org/project/bert-for-tf2/

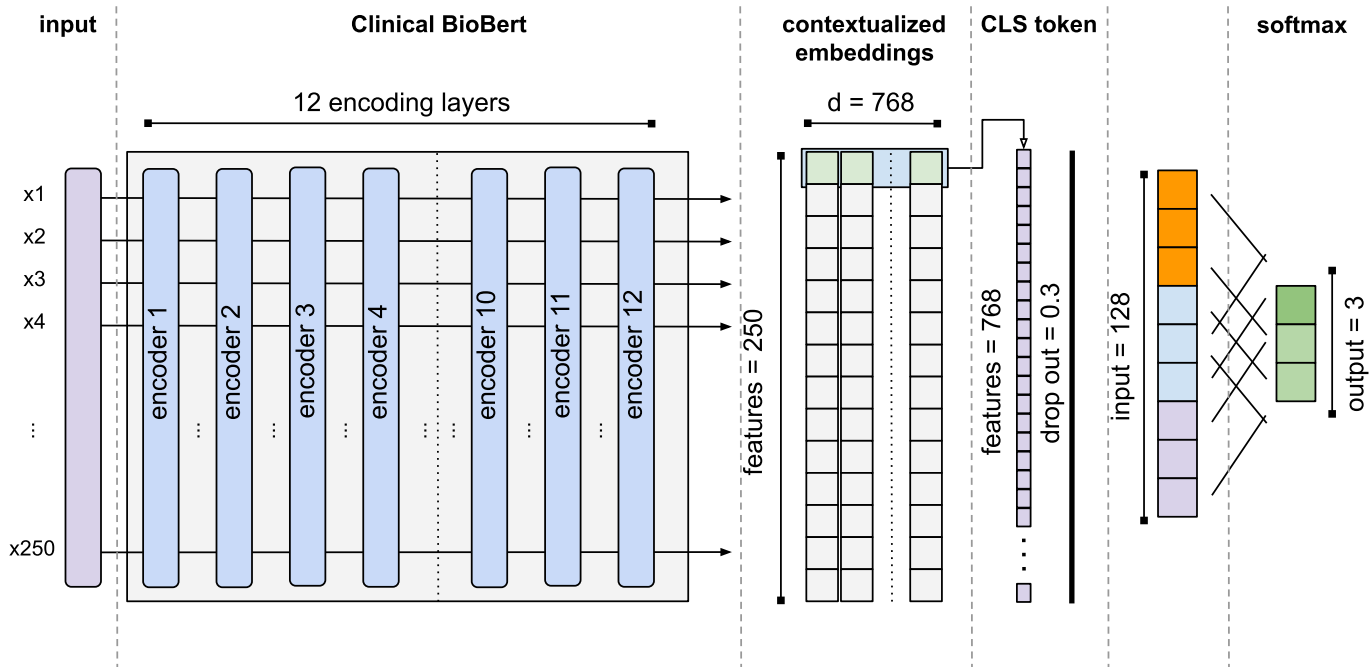


Figure 4.7: Overview architecture of Clinical BioBERT model

BERT-base model

The seventh approach employed is a BERT-based model called Clinical BioBert [37], which uses 12 encoding layers with a hidden layer's dimension of 768, resulting in a value of approximately 110M trainable parameters. Specifically, the model was pre-trained by Alsentzer, Emily, et al. by adding biomedical and clinical texts from the MIMIC-III v1.4 database [151] to improve its performance on clinical text tasks. Thus, the Clinical BioBert model can help mitigate the differences between the linguistic features of generic texts and drug review texts.

In order to represent BERT-based drug reviews, the texts were tokenized using the vocabulary of the pre-trained Clinical BioBERT model and the tokenizer provided by the transformers library[10]. The pre-processed texts resulted in a vocabulary of 28,996 tokens, leaving the length of the processed texts set at 250 tokens.

For this particular case, due to the significant increase in the number of encoding layers concerning the previous 4-encoding layer approach, the top bidirectional LSTM layer was skipped and only a perceptron layer fully connected to the output token "CLS" was added, followed by a softmax layer, as can be seen in Figure 4.7. As described in [54], the "CLS" token can be used for classification tasks such as SA since its value is deduced from the other tokens in the text.

In addition, in this approach, it was allowed to modify all weights during fine-tuning for the specific task increasing the computational cost due to the number of trainable parameters. Therefore, fine-tuning a BERT-base model requires a larger amount of hardware

[10]https://huggingface.co/docs/transformers/index

resources in order to train with sufficient batch size [177].

For the training of the model, it was used the ADAM optimizer [264], with a learning rate of 5e-5, a batch size of 128 during 10 epochs and categorical cross-entropy as loss function.

The source code is publicly available to enable the reproducibility of the experiments[11].

4.1.5. Evaluation

To evaluate the different systems, it has been used the standard metrics for the text classification tasks: precision, recall and F1. These metrics can be extended to multi-classification problems using the micro-averaged and macro-averaged versions. In the macro-average, the mean of the values obtained for each class is shown independently. In the micro-average, the contributions of each class are summed and then the average metric is calculated. In general, micro-average is preferable for class imbalance problems.

Results obtained for 10 classes

As a baseline, it has been used a classical classifier SVM [266] with a tf-idf representation of the reviews, which has been proven very effective for text classification [267]. The parameter tuning was performed using grid search. This classifier obtains a macro-F1 of 41.68% and a micro-F1 of 51.97%. As expected, classes with a larger number of examples (0 and 9) tend to provide better results since the classifier has more examples to learn.

Table 4.2 shows the results of the CNN models. Models using random initialization or non-static word embeddings provide very similar performance with micro F1 around 66%-67% and macro F1 around 60%-62%. However, models using static word embeddings show low results, around 46%-48% of micro-F1 and 26%-30% of macro-F1. Comparing with the baseline system, while the static CNN models perform worse, the non-static CNN models provide significantly better results than the baseline system, with an improvement of almost 15 percentage points in micro-F1 and 20 points in macro-F1. Therefore, a CNN model trained with non-static word embeddings overcomes the SVM baseline.

With regard to the effect of the different word embedding models on the results, the non-static CNN model trained with the "WE model C" model (which was trained with a large collection of tweets and open-domain texts) achieves slightly better micro-F1 (67.4%) and macro-F1 (62.8%) than the other CNN model. On the other hand, random initialization also yields very similar results to those provided by the non-static CNN models. Therefore, it can be concluded that the word embedding models studied do not guarantee a significant improvement on the random initialization.

[11] https://github.com/ccolonruiz/DrugSentimentAnalysis/

Linear SVM		
Label	**F1-score**	**Support**
0	0.6571	7299
1	0.3832	2334
2	0.3608	2205
3	0.3570	1659
4	0.3434	2710
5	0.3249	2119
6	0.3176	3091
7	0.3543	6156
8	0.3952	9177
9	0.6744	17016
Micro	0.5197	53766
Macro	0.4168	53766

Table 4.1: Results of Linear SVM (baseline)

	CNN models (F1-score)							
	Random	**WE model A**		**WE model B**		**WE model C**		
Label	**Non-Static**	**Static**	**Non-Static**	**Static**	**Non-Static**	**Static**	**Non-Static**	**Support**
0	0.7288	0.6474	0.74	0.6248	0.7355	0.6572	0.745	7299
1	0.6042	0.1655	0.5983	0.1485	0.5926	0.1598	0.6158	2334
2	0.6118	0.1684	0.5764	0.1801	0.5712	0.2287	0.608	2205
3	0.588	0.1224	0.6062	0.0996	0.5771	0.1448	0.6185	1659
4	0.5743	0.1886	0.5947	0.1886	0.5577	0.2331	0.5881	2710
5	0.5613	0.1247	0.5462	0.1112	0.5517	0.1452	0.5777	2119
6	0.5647	0.1431	0.562	0.1096	0.5518	0.1952	0.5626	3091
7	0.5869	0.2754	0.5843	0.2387	0.575	0.2424	0.597	6156
8	0.6194	0.3016	0.6098	0.3098	0.605	0.3508	0.6129	9177
9	0.7585	0.6807	0.7591	0.6701	0.7589	0.6908	0.7636	17016
micro avg	0.6681	0.475	0.6685	0.4618	0.6616	0.4871	**0.6745**	53766
macro avg	0.6198	0.2818	0.6177	0.2681	0.6077	0.3048	**0.6289**	53766

Table 4.2: Results of the CNN models using different initializations. The best results are shown in bold

LSTM far exceeds the baseline results (see Table 4.3). Unlike the CNN models where there is a strong difference (around 20 percentage points) between the performance of the static and non-static models, the non-static LSTM models show better results than the static LSTM ones, but the improvement is more modest (from 3 to 7 points). However, as happened in CNN models, there is no significant difference between the performance using random initialization and non-static word embeddings. In this case, the word em-

bedding model that provides better results is the 'WE model A', which was trained with biomedical and general texts. This LSTM model achieves a micro-F1 of 69.1% and a macro-F1 of 63.9%. However, the differences between CNN and LSTM are not statistically significant.

	LSTM models (F1-score)							
	Random	**WE model A**		**WE model B**		**WE model C**		
Label	**Non-Static**	**Static**	**Non-Static**	**Static**	**Non-Static**	**Static**	**Non-Static**	**Support**
0	0.7575	0.7799	0.7732	0.7499	0.7665	0.7825	0.7687	7299
1	0.6077	0.5333	0.6249	0.4859	0.6161	0.532	0.6374	2334
2	0.6181	0.5048	0.6058	0.4511	0.6057	0.5239	0.5943	2205
3	0.5953	0.5019	0.5836	0.4066	0.605	0.4839	0.6142	1659
4	0.5848	0.5108	0.6017	0.4592	0.5829	0.5052	0.5764	2710
5	0.5743	0.463	0.5909	0.3863	0.5897	0.4171	0.57	2119
6	0.5786	0.4941	0.5802	0.4252	0.5734	0.4755	0.5749	3091
7	0.5939	0.5675	0.6074	0.4957	0.5997	0.5461	0.5968	6156
8	0.635	0.5988	0.6376	0.5489	0.6398	0.5881	0.6409	9177
9	0.7783	0.7904	0.7858	0.7688	0.7853	0.7905	0.782	17016
micro avg	0.6835	0.6583	**0.6918**	0.616	0.6886	0.6518	0.6871	53766
macro avg	0.6323	0.5744	**0.6391**	0.5178	0.6364	0.5645	0.6356	53766

Table 4.3: Results of the LSTM models using different initializations. The best results are shown in bold

Table 4.4 shows the results for the three hybrid architectures combining CNN and LSTM. Random initialization for the three models was used since none of the word embedding models offer significantly better results than random initialization. The results show that LSTM followed by a CNN obtains the highest micro-F1 and macro-F1, however, the differences with respect to the other hybrid architectures are not statistically significant.

As previously described, there is a strong unbalanced distribution of the classes, predominating those with more polarized ratings. This results in a challenge when trying to classify those reviews with less representation. The analysis of the results reveals that the performance of each class is remarkably dependent of the number of instances of each in the training set, regardless of the model used. So the classes with a higher number of training instances usually show better results than those with less number of instances in the training set.

Approaches based on CNN architectures with pre-trained static embeddings provide the lowest results of all the experimentation, especially in the underrepresented classes. CNN networks are good at extracting local and location-independent features, but they are not able to extract information from long-range semantic dependencies [260]. The

	Hybrid models (F1-score)			
Label	**CNN concat LSTM**	**CNN + LSTM**	**LSTM + CNN**	**Support**
0	0.7625	0.761	0.7646	7299
1	0.6187	0.648	0.6268	2334
2	0.6161	0.4784	0.6068	2205
3	0.5916	0.6444	0.613	1659
4	0.5926	0.5591	0.587	2710
5	0.5653	0.5968	0.5966	2119
6	0.5745	0.6112	0.5739	3091
7	0.6005	0.6071	0.6092	6156
8	0.6266	0.6381	0.6443	9177
9	0.7756	0.7707	0.7865	17016
micro avg	0.6825	0.68	**0.6928**	53766
macro avg	0.6324	0.6315	**0.6409**	53766

Table 4.4: Results of the hybrid architectures (using random initialization). The best results are shown in bold

reason that LSTM models with pre-trained static embeddings perform better in the experimentation might be due to long dependencies contained in the texts of the reviews (see Table 4.3). For example, this review: "*I was taking* Aviane *before and transferred my prescription to a different pharmacy. There they gave me* lutera *since they did not have* aviane... *Worst experience of my life. Constant headaches and acne were the main symptoms right before my period. Extremely painful periods as well. If you are experiencing these symptoms while on* lutera*, I would highly recommend trying* aviane*. I do not have any headaches or acne and my periods are pain free.*". It refers to the side effects produced by the drug *Lutera*, but it also recommended using *Aviane* as a substitute for *Lutera*. The rating predicted by the CNN network is highly positive (maximum value), while the LSTM network predicts a negative rating of 2 (correct rating). However, when the part of the text that indicates the good performance of *Aviane* is removed, the CNN network scores the review with a negative value of 1 (minimum value). In this case, the LSTM network is able to maintain information from the context of the full review, where the assessment is negative towards the drug *Lutera*.

On the other hand, allowing the adjustment of the word embeddings during the training time, provides a significant improvement, especially in the case of CNN classifiers (see Table 4.2). This improvement might be due, among other reasons, to the vocabulary coverage provided by the different pre-trained embedding models. For example, the "WE model A" vocabulary contains 69.92% of the dataset vocabulary, the "WE model B" model contains only 30.54% and the "WE model C" model covers 78.23%. Terms not represented by these embedding models do not provide semantic information to the classifiers. The fine-tuning of the embeddings allows adjusting their values to the context

of the problem by modifying the previously learned associations (under the risk of overfitting). In this way, those terms that did not provide information are differentiated from each other, fitting to the problem.

Finally, hybrid approaches that combine different CNN and LSTM network architectures (allowing the adjustment of the word embeddings) are used to obtain both local and sequential features. The resulting classifiers provide slightly higher results than those provided by the non-static CNN models, but similar to the provided by the non-static LSTM ones (the difference in micro-F1 and macro-F1 averages ranges from 1% as shown in Tables 4.3 and 4.4). As discussed above, this might be due to providing the new models with the ability to maintain long dependency information contained in the texts.

Despite this, these approaches are not always able to accurately classify those reviews where the text contains sarcasm, irony or humor. For example, in the following review, all of these elements can be observed: "*My Dr thought this would be a good idea for me because of my knee condition I had had for years. I figured I'd give it a go and big surprise it made it worse a lot worse - it was like he stirred up a hornets nest after those shots. Now I'm having more trouble with it than ever before with walking pain, sleeping pain, knee resting pain, it's all worse I'm 40 and walk like I'm 80. Physical therapy was a joke considering my condition can't be fixed without replacing the knee. All physical therapy did was get me to shift my weight to one side and now my other knee is giving me problems. I guess the Dr won't be happy until I'm in a wheelchair*". The classifiers LSTM and CNN concatenated with LSTM correctly score the example (rating of 0), but the rest of the classifiers score with very different values (CNN: 7, CNN+LSTM: 6, LSTM+CNN: 5). In this case, all the classifiers correctly score the example by removing expressions such as: "*it was like he stirred up a hornets nest after those shots*" and "*I guess the Dr won't be happy until I'm in a wheelchair.*"

Overall, the top performance is a micro-F1 of 69.28% and a macro-F1 of 64.09% provided by the LSTM+CNN hybrid classifier (see Table 4.4). This classifier aims to extract local features with the convolutional layer from the contextual information provided by the bidirectional LSTM layer. This might be since the sequential features obtained directly from the representation of embeddings (by the bi-directional LSTM layer) are the most discriminating ones in the experimentation. Providing local information on these features might improve the performance compared to classifiers based only on LSTM networks.

However, noise in datasets based on subjectivity in the degree of satisfaction should be taken into account. This task is extremely difficult especially considering the number of possible choices (10) and the subjectivity differences especially those between classes very close to one another. This results in models that can accurately distinguish between positive and negative polarities, but having difficulty in providing the accurate rating. Thus, when classifying a review (whose rating is 10), different types of errors should be distinguished, because a predicted rating of 9 or 8 should be considered more accurate

than 5 or a lower value. For example, the following review is only correctly labeled by the CNN+LSTM classifier (with a value of 7): "*I started taking gabapentin experimentally to treat chronic depression and am now prescribed 1800mg a day. At doses of 2700mg I experienced a significant improvement in impulse control and depression in general. Went from being a shut in, to going for walks and enjoying my time in the company of others. At the lower dose of 1800mg a day, I don't experience much improvement, save for a decrease in depressive symptoms. I'm looking forward to a larger dose if my doctor okays it.*" The remaining classifiers, however, label it with very similar degrees of satisfaction, for example, 6 or 8.

	Training time (minutes)						
	WE model A		**WE model B**		**WE model C**		**Random**
Model	**Non-Static**	**Static**	**Non-Static**	**Static**	**Non-Static**	**Static**	**Non-Static**
CNN	43'	30'	36'	23'	60'	36'	43'
LSTM	1383'	800'	870'	796'	906'	833'	886'
CNN+LSTM	-	-	-	-	-	-	910'
LSTM+CNN	-	-	-	-	-	-	950'
CNNconcatLSTM	-	-	-	-	-	-	916'

Table 4.5: Training time in minutes for 200 epochs using the Keras 2.0 library (10-class dataset)

Another factor to take into consideration is the time needed for the training of each of the models (see Table 4.5). This experimentation was done with an Nvidia Titan XP graphics card, an Intel Core i7-6700K and 32GB of RAM at 1600 MHz. The training time using the Keras 2.0 library for LSTM models (200 epochs) far exceeds the time needed to train CNN models (the training time for LSTM models is approximately 20 times longer).

Results obtained for 3 classes (positive, negative and neutral)

This subsection focuses on the experiments performed using the dataset with three polarities: positive, neutral and negative. As seen Tables 4.6, 4.7, 4.8 and 4.9, all models show much better performance compared to the models evaluated on the dataset with 10 classes. Thus, the highest micro-F1 (0.6928) and macro-F1 (0.6409) (see Table 4.4), which were provided by the CNN+LSTM model, are almost 20% lower than those provided by the same model evaluated on the dataset with three polarities. This may be due to the number of instances for each class is increased when working only with three classes.

Table 4.6 shows the results of the baseline system (Linear SVM) on the dataset with three polarities with a tf-idf representation of the reviews. Table 4.7 shows the results of the simple models (CNN and LSTM) on the dataset with three polarities. CNN and LSTM provide very similar performance, as happened on the dataset with 10 classes (see Tables

4.2 and 4.3). The simple models CNN and LSTM provide significantly better results than the baseline system, with an improvement of almost 7 percentage points in micro-F1 and 14 points in macro-F1

	Linear SVM	
Label	**F1-score**	**Support**
Negative (Class 0)	0.7430	13497
Neutral (Class 1)	0.3579	4829
Positive (Class 2)	0.8892	35440
micro avg	0.8175	53766
macro avg	0.6634	53766

Table 4.6: Results of Linear SVM with tf-idf representation (baseline)

	CNN and LSTM models (F1-score)		
Label	**CNN**	**LSTM**	**Support**
Negative (Class 0)	0.8364	0.8482	13497
Neutral (Class 1)	0.6457	0.6361	4829
Positive (Class 2)	0.9302	0.9342	35440
micro avg	0.8844	**0.8882**	53766
macro avg	0.8041	**0.8062**	53766

Table 4.7: Results of the CNN/LSTM architectures (using random initialization). The best results are shown in bold

The hybrid models combining CNN and LSTM (see Table 4.8) show slightly better performance than the simple models, with an increase of 0.62% on the micro-F1 score and 0.84% on the macro-F1 score.

	Hybrid models (F1-score)			
Label	**CNN concat LSTM**	**CNN + LSTM**	**LSTM + CNN**	**Support**
Negative (Class 0)	0.8498	0.8560	0.8545	13497
Neutral (Class 1)	0.6398	0.6507	0.6393	4829
Positive (Class 2)	0.9359	0.9371	0.9372	35440
micro avg	0.8903	**0.8944**	0.8920	53766
macro avg	0.8085	**0.8146**	0.8103	53766

Table 4.8: Results of the hybrid architectures (using random initialization). The best results are shown in bold

	BERT models (F1-score)		
Label	**BERT-small BiLSTM**	**Clinical BioBERT**	**Support**
Negative (Class 0)	0.8720	0.8939	13497
Neutral (Class 1)	0.6514	0.6597	4829
Positive (Class 2)	0.9477	0.9555	35440
micro avg	0.9046	**0.9138**	53766
macro avg	0.8237	**0.8363**	53766

Table 4.9: Results of BERT architectures. The best results are shown in bold

Recently, BERT models have shown huge improvements in a wide range of NLP tasks. However, as seen in Table 4.9, BERT-small embeddings with BiLSTM provide slightly better results than the previous models. The model achieves a 1.64 improvement on micro-F1 and a 1.75 macro-F1 score over the simple LSTM model with random initialization. Compared to the hybrid models combining CNN and LSTM, the improvement is more modest, with only 1.02% in the micro-F1 score and 0.91% in the macro-F1 score. On the other hand, the Clinical BioBERT model further increases this difference, achieving an improvement of 1.94% in micro-F1 and 2.17% in macro-F1 compared to the hybrid models combining CNN and LSTM but obtaining a more modest improvement over the BERT-small model with BiLSTM.

However, it is important to clarify that the results obtained by the Clinical BioBERT model were obtained using a higher computational capacity. Specifically, the BERT-small BiLSTM model was trained with an Nvidia Titan XP graphics card, while Clinical BioBERT was trained with an Nvidia A100 graphics card.

To train a BERT-base model, a minimum of 12GB of VRAM is required regardless of the batch size or the input data size. Thus, it should be considered that a modest difference in metric-F1 results between the two BERT-based models may not justify the increased computational capacity in some cases, the latter being an advantage of the BERT-small model with BiLSTM over a larger model such as Clinical BioBERT.

Moreover, the BERT based models have a similar accuracy (0.9046) to that of the previous work [60], which also exploited different BERT models. However, these approaches cannot surpass the accuracy of the logistic regression model proposed in [47], whose accuracy was 0.9224.

Clearly, all models show similar behavior for the three classes (see Figure 4.8). Thus, all models classify drug reviews with positive polarity (class 2) with high performance. Specifically, using more limited resources, such as the Nvidia Titan XP video card, the highest F1 score (0.9477) is achieved with the BERT-small embedding with BiLSTM. This model also achieves the highest F1 score (0.8720) for reviews with negative polarity (class 0). For the reviews with neutral polarity (class 1), all models show lower results

than in the other classes, with the highest F1 score around 0.65. This may be because this class only represents 10 percent of all instances.

In the confusion matrix (see Figure 4.8), it can be seen there are a large number of cases, for all models, where neutral class examples are classified as positive or negative. Moreover, also a large number of positive or negative class examples are classified as neutral class ones. This is conditioned by the small representation of neutral class examples. On the other hand, it is observed that the BERT-based models classify fewer examples in classes opposite to the correct ones. Specifically, the Clinical BioBERT model stands out in this respect, significantly reducing this class of errors concerning the rest of the models, while the CNN model is the one that commits the most errors. This may be because the CNN model cannot store information about long dependencies and does not have contextualized information at the word level.

Training time (minutes)	
Model	**Time**
CNN	217'
LSTM	353'
CNN+LSTM	733'
LSTM+CNN	953'
CNNconcatLSTM	953'
BERT-small+BiLSTM	1,343'

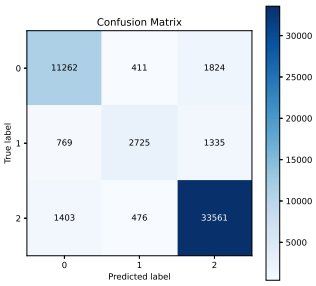
Table 4.10: Training time in minutes for 200 epochs using the Tensorflow 2.3.0-tf library (3-class dataset)

Training time (minutes)	
Model	**Time**
Clinical BioBERT	123'

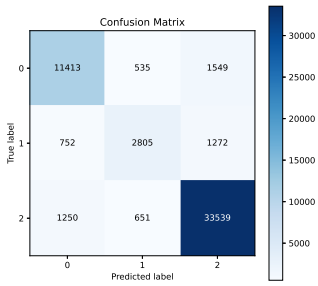
Table 4.11: Training time in minutes for 10 epochs using the Tensorflow 2.7.0-tf library (3-class dataset)

Regarding the training times of the models that classify three classes, it can be seen in Table 4.10) that the simple CNN model is the model that requires the least training time, while the BERT-small BiLSTM has the longest training time.

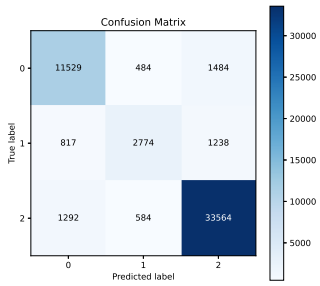
However, it can be seen in Table 4.11) that the Clinical BioBERT model takes only 123 minutes to train. This is due to several factors, the main one being access to an Nvidia A100 card and the ability to reduce the number of epochs significantly by using a model with so many adjustable parameters.



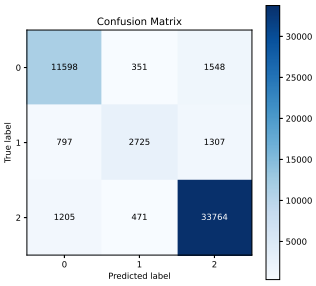
(a) CNN confusion matrix



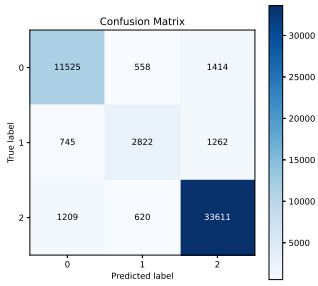
(b) LSTM confusion matrix



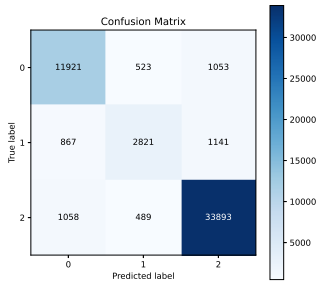
(c) CNN concat LSTM confusion matrix



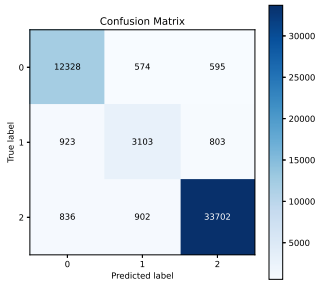
(d) CNN + LSTM confusion matrix



(e) LSTM + CNN confusion matrix



(f) BERT-small + LSTM confusion matrix



(g) Clinical BioBERT confusion matrix

Figure 4.8: Model confusion matrices (3 classes)

4.1.6. Conclusions

To the best of the researcher's knowledge, this is the first work to compare deep learning architectures for the SA task of drug reviews. The effects of different word embedding models on the performance of the different models were also studied; however, none of them seems to offer significantly better results than the rest of word embedding models or random initialization.

SA is usually considered as a text classification task. In this case, the task was addressed using the three polarities (positive, negative and neutral) proposed in [47], but

also as a more challenging classification task using 10 classes, which are the ratings defined by the users to show their overall satisfaction with the drug of the review. As is to be expected, the results for the 3-class dataset are much higher than those obtained with the 10-class dataset. In the 3-class dataset, there is not only less classes to classify, but also each class has more examples to learn.

The results on the 10-class dataset show that the hybrid model composed of a bidirectional LSTM followed by a CNN provides the best results. CNN models initialized with static word embedding show very low performance for classes with fewer instances in the training set. On the other hand, the CNN model requires less training time than the bidirectional LSTM and hybrid models.

In recent years, the advent of BERT has revolutionized the field of NLP, achieving state-of-the-art results for many NLP tasks. Therefore, two different approaches based on BERT are used in this Thesis: a BERT-small model followed by a BiLSTM layer and a computationally more demanding BERT-base model (Clinical BioBERT).

Focusing on the results of the 3-class dataset, it can be observed that the BERT-based models provide slightly better results than the other models. However, using BERT considerably increases the computational cost, and the BERT-small+BiLSTM model may be an alternative to Clinical BioBERT when there is little computational capacity but at the expense of high training times. On the other hand, the CNN model provides acceptable results requiring little computational capacity and less training time.

Due to the increasing use of language representation models for classification tasks, future work plans to apply them in VAEs and adversarial networks for semi-supervised approaches. Thus, new methods to reduce the dependency on annotated corpora will be explored. Moreover, the use of semantic features together with contextual ones will be explored to improve the fit of these approaches.

4.2. Semi-Supervised Approaches: Generative Adversarial Network

4.2.1. Introduction

In the previous section 4.1, a comparison was conducted between different supervised approaches to SA in drug reviews. For this purpose, the dataset provided by [47] was used to train and evaluate the models, having enough annotated data to obtain results around 90% micro-F1 and 82-83% macro-F1. However, in real situations, the shortage of annotated data, which is essential for training machine learning algorithms, has become one of the greatest bottlenecks in NLP [268], [269], and, in particular also for the task of SA [251].

To overcome the lack of labeled examples, the use of unsupervised and semi-supervised approaches have been applied to SA [270]–[272]. However, these approaches do not outperform the traditional supervised machine learning methods.

More recently, transfer learning and cross-domain approaches have been applied to mitigate the lack of labeled data [47], [273]–[275]. Transfer learning is a machine learning approach where a first model, which is trained for a specific task, is then reused as input for training a new model for another task. A successful example of transfer learning in NLP is BERT [54], whose main advantage is that it can accurately represent the different meanings of a word. Then, a fine-tuned phase is performed for a specific task, such as text classification, text summarization or NER, among many others. These BERT-based models are achieving the current state-of-the-art results for many NLP tasks[54], [253]. However, the fine-tuning phase still requires thousands of annotated examples for the target task. Therefore, the quality of the results drops significantly when the number of labeled examples is insufficient [65].

To address the challenge caused by the shortage of labeled drug reviews, employing a semi-supervised method by implementing a SSGAN [203] is proposed. Generally, in GANs [202], a generator is trained to produce synthetic examples from different data distributions. During training, the generator tries to mislead a discriminator, which is trained to distinguish between synthetic and real examples.

To the best of the researcher's knowledge, only one previous work [65] has used SS-GAN for the SA task. Therefore, this study is the first to apply SSGAN to the sentiment classification of drug reviews.

In order to evaluate the actual contribution of this semi-supervised approach, a comparison between the supervised approaches described in sub-section 4.1.4 was also performed. Specifically, BERT-small [177] model tuned with a BiLSTM layer and BERT-base model (Clinical BioBERT). In this comparison, BERT-small BiLSTM exploits a fine-tuned architecture for the SA task with a significant decrease in trainable parameters compared to Clinical BioBERT and the one proposed in [65].

In addition, different splits of the training dataset from sub-section 4.1.2 are used to train the models. The purpose of these splits is to show the performance of the semi-supervised approach compared to the supervised ones that are trained with only a small annotated percentage of the training dataset. Thus, these splits will show how much data must be annotated to obtain similar results to the supervised approaches.

4.2.2. Dataset

For this work, the dataset of drug reviews created by [47] was employed. In this dataset, the drug reviews written by patients were collected from Drugs.com and Druglib.com, two of the most visited pharmaceutical information websites offering information to consumers and health care professionals. Each drug review includes a rating from 1 to 10, indicating the degree of patient satisfaction with the drug. For example, a patient with *epilepsy* posted the following comment: "*I've had nothing but problems with the Keppera : constant shaking in my arms & legs, pins needles feeling in my arms & legs, severe*

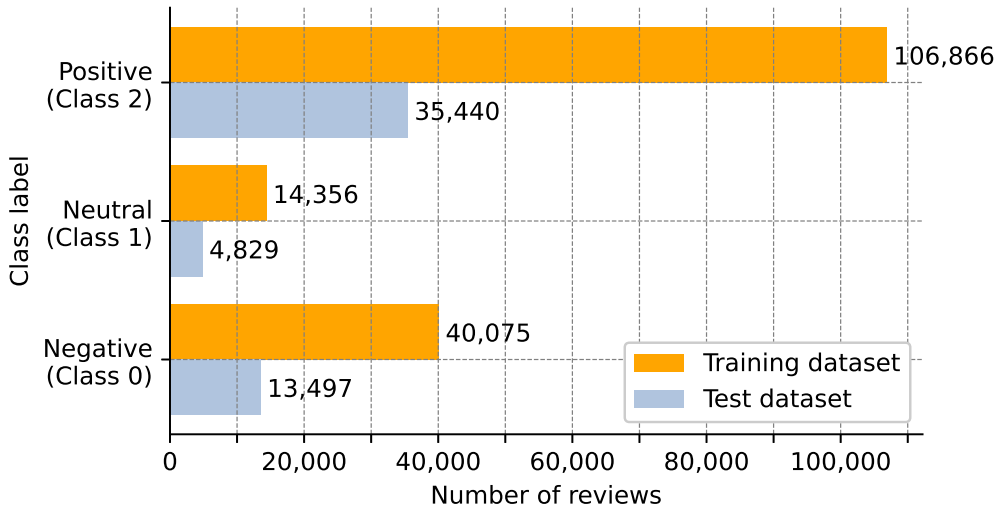


Figure 4.9: Class distribution in training and test datasets

light headedness, no appetite, etc.". The patient refers to drug *Keppera* and describes the adverse effects it has caused. The rating for the drug provided by the patient is negative with a value of 2. The reviews were grouped into three levels of polarity according to their rating: positive ($rating \geq 7$), negative ($rating \leq 4$) and neutral (rating [5,6]), such was proposed in [47]. The dataset, which contains a total of 215,063 drug reviews, is split into training and test sets with a ratio of 75:25 with stratified random sampling to maintain the same proportion of all classes. Additionally, as in [23], 15% of the total training set is used as the validation set, which is used to fit the hyperparameters of the models.

Figure 4.9 shows that the distribution of the different reviews according to their classes (degree of satisfaction) is strongly unbalanced in the training and test sets. Reviews with positive polarity (class 2) represent about 66% of the data, reviews with negative polarity (class 0) represent about 25% and, finally, reviews with neutral polarity (class 1) represent only 9%.

In order to prepare the training dataset for the comparison between the semi-supervised approach and the supervised one, it was performed a series of splits (see Table 4.12) to obtain several ratios between labeled and unlabeled reviews. By employing different ratios between labeled and unlabeled data, it will be able to observe how much annotated data is needed to use with each approach to obtain similar results.

For example, in the split "20:80", only 20% of the training dataset is randomly selected to be used as labeled reviews, while the remaining 80% will be used as unlabeled reviews. In this particular split of 20%, there are 27,420 labeled reviews, where 6,820 belong to class 0 (negative), 2,427 belong to class 1 (neutral) and 18,173 belong to class 2 (positive). The splits are created using a stratified random sampling to keep the same class distribution in both partitions.

The semi-supervised approach will use labeled examples for training the classification model, but also the unlabeled examples to improve the distinction between automatically

	Number of labeled reviews			
Split ratio	**Total**	**Negative (0)**	**Neutral (1)**	**Positive (2)**
1:99	1,371	341	121	909
2:98	2,742	682	243	1817
5:95	6,855	1,705	607	4,543
10:90	13,710	3,410	1,213	9,087
20:80	27,420	6,820	2,427	18,173

Table 4.12: Ratios between labeled and unlabeled reviews. For example, the ratio 1:99 indicates that only 1% of the dataset is labeled.

generated examples and real ones. On the other hand, the supervised approach will only use the labeled reviews to train its model.

4.2.3. Method

In order to assess the real contribution of the semi-supervised approach, the best two supervised approach models described in section 4.1 are proposed: the BERT-small model followed by a BiLSTM, as well as the Clinical BioBERT model.

As mentioned above, one of the main advantages of BERT is its ability to provide contextualized word embeddings with a more accurate representation of words. However, training BERT for a specific task is costly from the point of view of the [265] parameters. Due to the computational cost, the BERT-small [177] model supplemented with adapter modules [265] and followed by a BiLSTM layer was used.

BERT-small requires much less memory than the original model (BERT-base), which allows less computational resources to be used at the cost of worse results in the final task [177]. For this reason, a comparison is made between BERT-small BiLSTM and the BERT-base model, Clinical BioBERT, using both in a semi-supervised approach.

To format the drug reviews in a BERT-based format, the texts were processed using the tokenizers described in section 4.1. In the case of BERT-small, the tokenizer provided by bert-for-tf2[12] and the pre-trained BERT-small model vocabulary were used, while for Clinical BioBERT, the tokenizer provided by the transformers library[13] was used. In addition, due to the difference in the length of the texts, it was necessary to fill in and truncate the reviews based on the cumulative distribution function (Figure 4.10) and the available computational capacity. This figure shows that almost 100% of the reviews have a length of fewer than 250 tokens. Therefore, the length of the processed texts was set to

[12]https://pypi.org/project/bert-for-tf2/
[13]https://huggingface.co/docs/transformers/index

250 tokens.

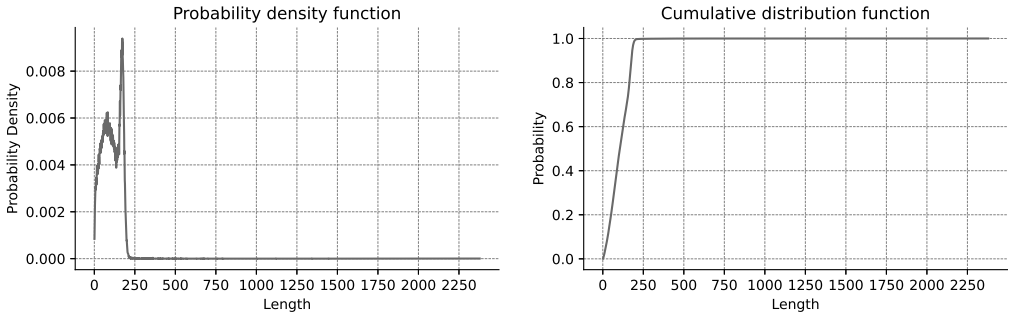


Figure 4.10: Probability density function and Cumulative distribution function over the length of reviews

This semi-supervised approach uses the same architecture as the supervised one, but also exploits an adversarial network to improve the internal representation of the examples for the classification task. In a GAN architecture [202], two models are trained: a generative model G and a discriminative model D. G learns the distribution of the data and generates synthetic examples. D predicts the probability that an example is synthetic (generated by G) or real (from the training dataset). The training of G consists of maximizing the probability that D makes a mistake. This scenario corresponds to a minimax game of two players [276], G and D, competing against each other. The generative model is confronted with an adversary, a discriminative model that learns to differentiate whether an example comes from G or belongs to the training data. This competition leads both models to improve their performances until the distributions generated by G are practically indistinguishable from those belonging to the real training data.

In this way, it is used a SSGAN [203] that adversarially improves the distributions generated by G and D, and thereby, provides a better representations for the classification task. To achieve this, the generator must produces a batch of new synthetic examples to try to mislead the discriminator. The generator uses as input (x_g) an uniform noise distribution and a MLP to learn the synthetic examples. Then, the discriminator, which is also a MLP, is trained on c classes (in this case, three classes: positive, negative and neutral) plus an additional binary class *(c+1)* to identify whether the input example comes from the generator or the real training data. The last layer of D is a softmax layer for the *(c+1)* classes. Figure 4.11 shows the complete architecture of the SSGAN-BL approach, which consists of the SSGAN model, which aims to improve the internal representations of examples, and the BERT-small BiLSTM model, which learns the representation of drug reviews according to the sentiment classification task. Both models are trained simultaneously.

Likewise, Figure 4.12 shows the architecture of the SSGAN-CBB approach, which, unlike SSGAN-BL, employs the Clinical BioBERT model to represent drug reviews.

In a first stage, while G and D try to adversarally improve the internal representations of both models, D take as input the output from G ($h_{fake} \in R^d$), but also the output from

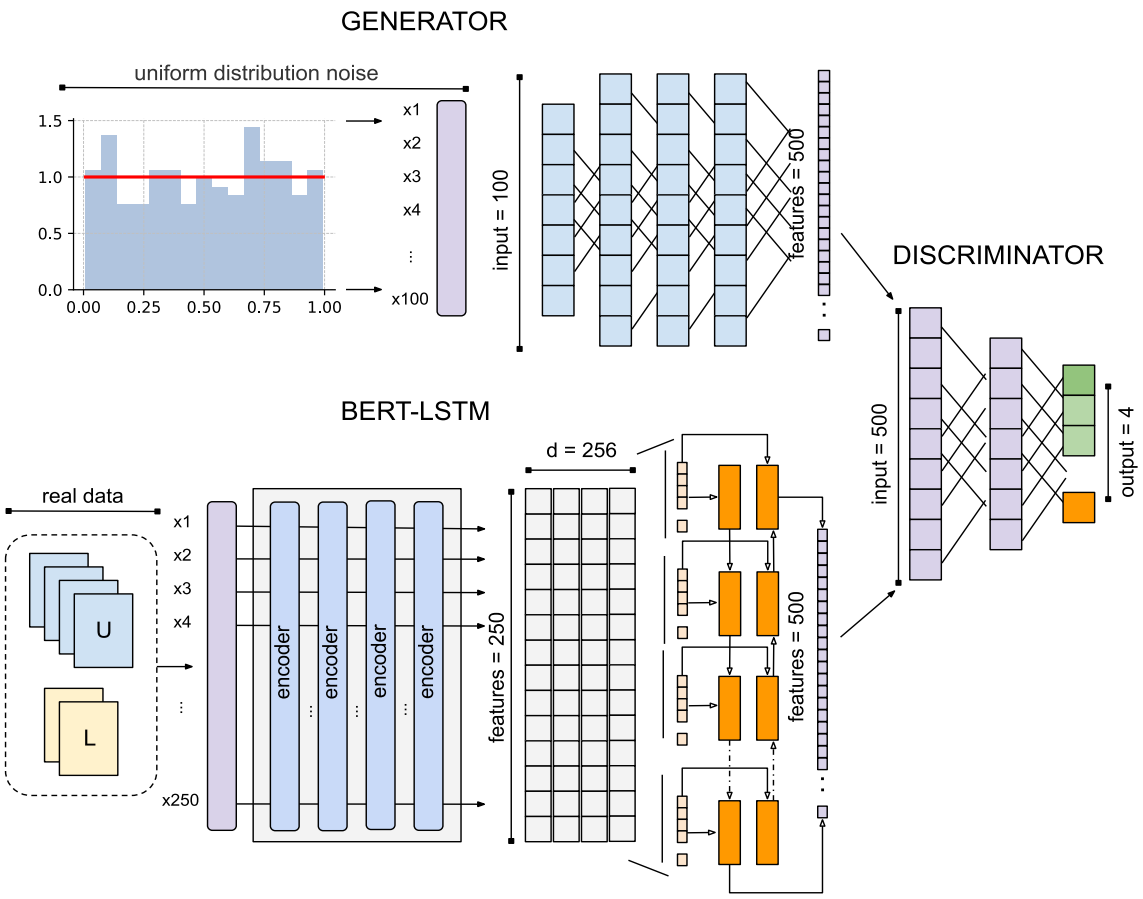


Figure 4.11: Overview architecture of SSGAN-BL

the BERT-small BiLSTM or Clinical BioBERT models ($h_{real} \in R^d$). Ideally, D should distinguish synthetic examples from real examples. In other words, D should classify the synthetic examples coming from G with the binary class *(c+1)*, but not the real ones. Moreover, as the training dataset was split into a percentage of labeled examples and the remaining ones to be considered as unlabeled, D should classify the labeled samples with one of the three classes (positive, negative or neutral).

During the training of the SSGAN-BL model, the weights of G and D are updated and those of the BERT-small BiLSTM model. Once the SSGAN-BL model is trained, G is discarded, keeping the rest of the model for the sentiment classification task with the three classes (positive, negative and neutral) and ignoring the binary class *(c+1)* in the softmax layer of D. Similarly, during SSGAN-CBB training, the Clinical BioBERT weights are also updated.

The loss functions used to train the models are described in detail below. From now on, x_b is a vector of a original review (input for BERT models) and x_g is a vector generated from a uniform random distribution (input for G). Once this is defined, the probability P, provided by the discriminator D, can be expressed as the probability of a given input, x_b or x_g, to be associated with one of the *(1, ..., c+1)* classes. The error of each of the models can be measured using the probability distributions P. In order to train D, it is necessary

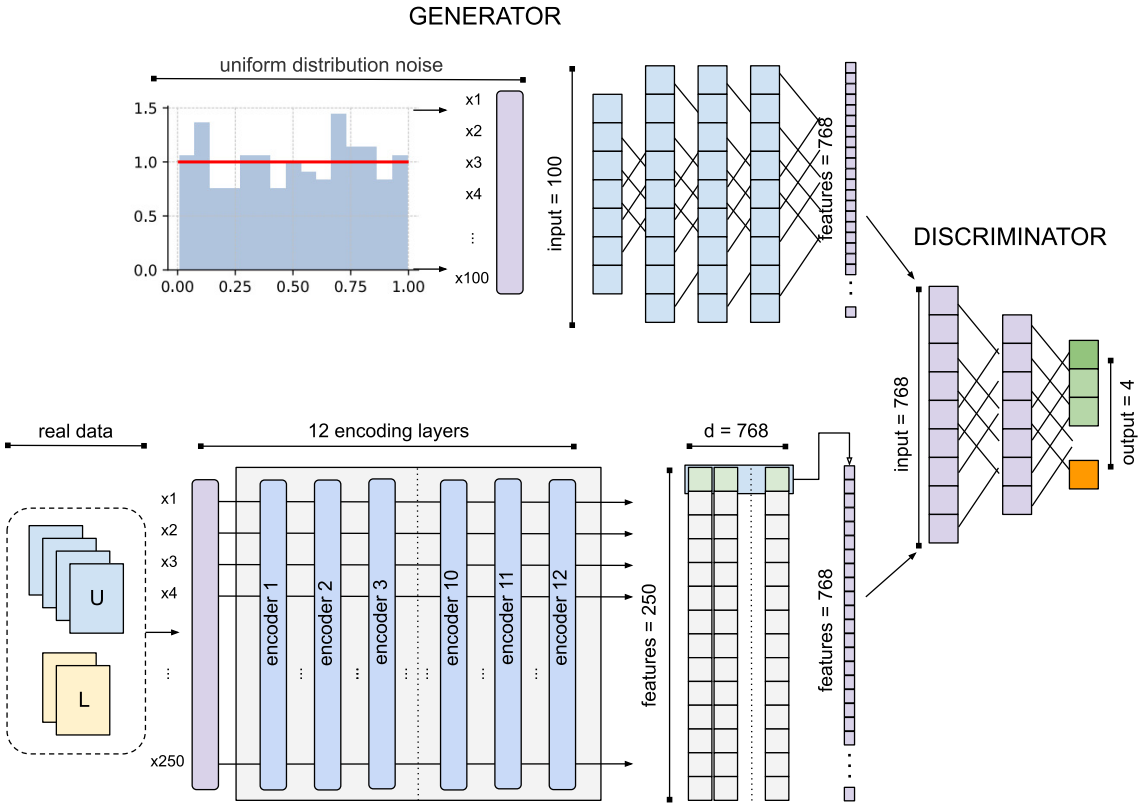


Figure 4.12: Overview architecture of SSGAN-CBB

to minimize the sum of errors produced by misclassifying labeled reviews with wrong classes in *(1, ..., c)*, by misidentifying labeled and unlabeled reviews as "fake" reviews (that is, with the *(c+1)* class) and by misidentifying synthetic examples generated by G as "real" reviews. On the other hand, to train G, it is necessary to minimize the sum of errors produced by D in identifying an example generated by G as a synthetic one and by G in generating examples with distributions significantly different from the real reviews.

Therefore, the loss function of D is defined as $L_{D_{sup}} + L_{D_{uns}}$, where (4.1) provides the error in assigning a labeled review x_b in the original *c-classes*, (4.2) defines the error of recognizing a labeled or unlabeled review x_b as a synthetic example *(c+1)*, and (4.3) defines the error of recognizing a synthetic example x_g as a real example *(c+1)*. Finally, the error is back-propagated and the weights are updated for the D and BERT models (adjusting the inner representations of the reviews).

In the meantime, the loss function of G is defined as $L_{G_{uns}} + L_{G_{match}}$. G must be able to generate synthetic examples similar to the representation of the real ones. For this purpose, the average of the examples generated by G in a batch must be similar to the average of the real examples' representation. In (4.5), $f(x)$ is defined as the activation on an intermediate layer of D, where x represents either the input data of the BERT models or the input data of G. The loss function of G, in (4.4), also measures the error of recognizing a synthetic example x_g as a synthetic example, as G tries to mislead D with examples similar to the real ones.

$$L_{D_{sup}} = -\sum_{i=1}^{c} y_i log[P(\hat{y} = y|x_b, y \in (1, ..., c))_i] \tag{4.1}$$

$$L_{D_{uns}} = -log[1 - P(\hat{y} = y|x_b, y = c + 1)] \tag{4.2}$$

$$- log[P(\hat{y} = y|x_g, y = c + 1)] \tag{4.3}$$

$$L_{G_{uns}} = -log[1 - P(\hat{y} = y|x_g, y = c + 1)] \tag{4.4}$$

$$L_{G_{match}} = \frac{1}{n} \sum_{i=1}^{n} \left| f(x_b)_i - f(x_g)_i \right|^2 \tag{4.5}$$

For the training of the SSGAN-BL model, it was used 400 epochs with the ADAM optimizer (learning ratio of 4e-5). The batch sizes can differ (64-200) depending on the number of labeled reviews used for training. This is due to the need to preserve batches with labeled reviews representation to avoid divergence produced by unsupervised components in adversarial training. On the other hand, for both G and D, Leaky Relu activation functions were used except in the last layer of the generator, where a hyperbolic tangent function (tanh) was used as the activation function. The latter is to ensure that the h_{fake} representations produced by the generator are similar to the h_{real} representations provided by BERT-small BiLSTM, where the last layer is composed of a BiLSTM recurrent network with hyperbolic tangent activation functions.

On the other hand, for the training of the SSGAN-CBB model, 10 epochs were used with the ADAM optimizer (learning ratio of 2e-5) with a batch size of 128 instances. In contrast to the SSGAN-BL model, a hyperbolic tangent activation function was not used in G as it was dispensed with a BiLSTM top layer. Instead, the outputs of G from the previous layer were normalized to provide h_{fake} representations similar to the h_{real} ones provided by Clinical BioBERT

The pseudo-code of the training algorithm can be found in Algorithm 1.

The source code is publicly available to enable the reproducibility of the experiments[14].

4.2.4. Evaluation

This section focuses on the results obtained with the approaches on the test dataset described in Section 4.2.3.

One of the standardized metrics for text classification tasks is used to evaluate both approaches: F1. This metric can be extended for multi-classification problems with the micro-averaged and macro-averaged versions. With micro-averaged F1, the contributions of each class are shown, while the average of the values obtained by each class is shown independently with macro-averaged F1.

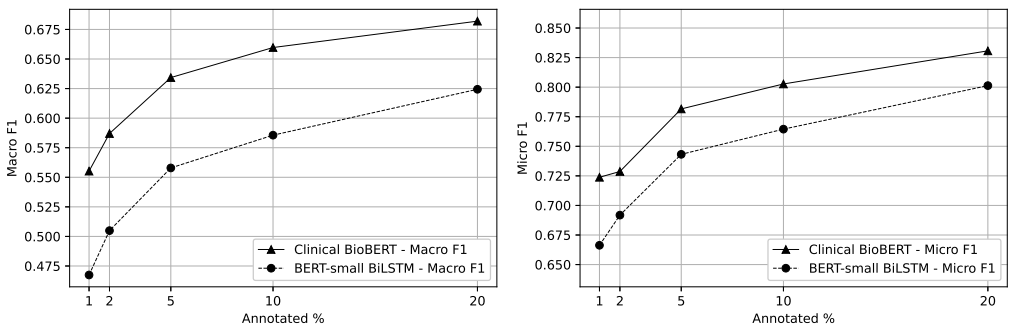
[14]https://github.com/ccolonruiz/SSGAN-DrugSentimentAnalysis/

Algorithm 1 SSGAN-BL/SSGAN-CBB training pseudo-code

Inputs: epochs, batch size n, annotated ratio nr $(L|U)$
for $epoch$ in $epochs$ **do**
  for $batch$ in $epoch$ **do**
    $D \rightarrow trainable = True$
    Sample $l_1, ..., l_{nr \cdot n}, u_1, ..., u_{(1-nr) \cdot n}$ from $x_{b_1}, ..., x_{b_n}$
    Sample $z_1, ..., z_n$ from $N(\mu, \sigma^2)$
    while $D \leftarrow train$ **do** ▷ B is also updated
      $L_{D_{sup}} \leftarrow D(B(l))$
      $L_{D_{uns}} \leftarrow D(B(l+u))$ and $D(G(z))$
      $L_{D_{total}} = L_{D_{sup}} + L_{D_{uns}}$
    end while
    $D \rightarrow trainable = False$
    while $G \leftarrow train$ **do**
      $L_{G_{uns}} \leftarrow D(G(z))$
      $L_{G_{match}} \leftarrow f(l+u)$ and $f(z)$
      $L_{G_{total}} = L_{G_{uns}} + L_{G_{match}}$
    end while
  end for
end for

Results obtained by BERT-small LSTM and Clinical BioBERT

Figure 4.13 shows the performance of the supervised approaches using different proportions of labeled (Annotated %) and unlabeled reviews. This will show the difference in results provided by BERT-small LSTM and Clinical BioBERT when only limited subsets of labeled reviews are available.



(a) Macro F1 - Annotated data (%)

(b) Micro F1 - Annotated data (%)

Figure 4.13: Macro-F1 and Micro-F1 curves of BERT-small BiLSTM and Clinical BioBERT classifiers

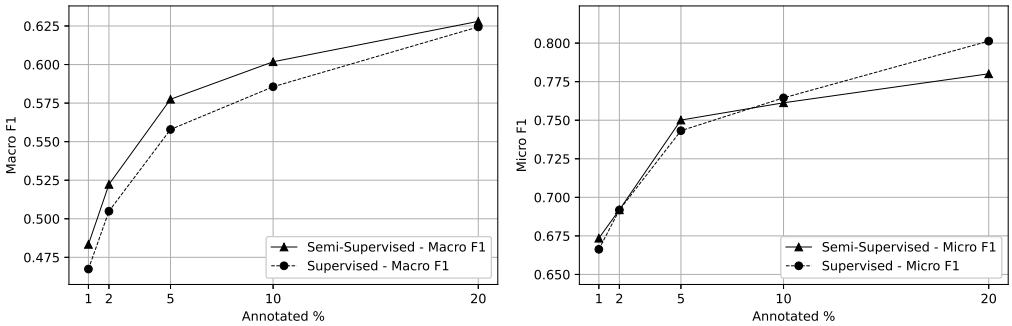
As can be seen, the Clinical BioBERT model outperforms the BERT-small BiLSTM model by far. This is mainly due to several factors: firstly, due to the difference in com-

plexity of the models, with Clinical BioBERT using 12 coding layers instead of 4, as is the case with BERT-small LSTM; secondly, the BERT-small model only modifies the weights of the adaptor modules, leaving the rest intact with no ability to adjust during training; thirdly, Clinical BioBERT is a pre-trained text model with linguistic characteristics of clinical and biomedical texts that are more in line with the type of language that drug reviews may contain; and finally, BERT-small uses a BiLSTM layer with randomly initialized weights. All these factors imply that the small model requires a larger number of training steps, even at the cost of lower performance, to fit the particular task when little training data is available.

At the same time, we can observe that Clinical BioBERT can provide similar results to those provided by BERT-small LSTM using significantly less training data, in some cases using five times less annotated data than required by BERT-small LSTM in terms of macro-F1. For example, when Clinical BioBERT uses 2% annotated data, 58.70% of macro-F1 is obtained compared to 58.56% for BERT-small LSTM using 10 percent of annotated training data.

Results obtained by BERT-small LSTM and SSGAN-BL

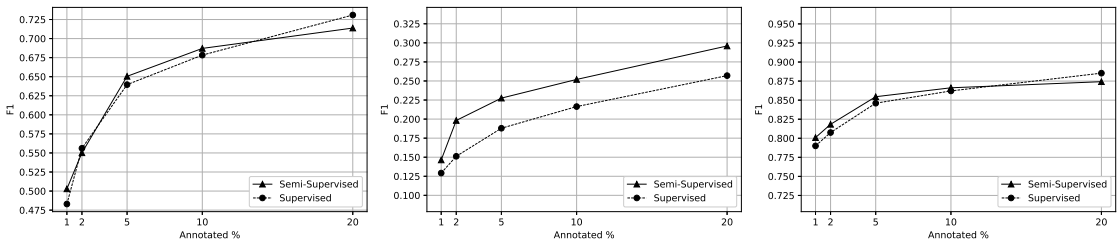
Figures 4.14 and 4.15 show the performance of the supervised and semi-supervised approaches using different ratios of labeled (Annotated %) and unlabeled reviews. In this way, the difference in the results provided by the semi-supervised and supervised approaches will be shown when only limited subsets of labeled reviews are available.



(a) Macro F1 - Annotated data (%) (b) Micro F1 - Annotated data (%)

Figure 4.14: Macro-F1 and Micro-F1 curves of BERT-mini LSTM and SSGAN-BL classifiers

Figures 4.14 compare both approaches when different percentages of labeled examples are used to train both models. It can be observed that, with less than 10% of labeled examples, the semi-supervised approach performs better in terms of micro-F1 than the supervised approach (see Fig. 4.14b). Similarly, the semi-supervised approach overcomes the supervised one in terms of macro-F1 when 20%, or less, of the training examples (see Fig. 4.14a) are used. Therefore, when only 10% or 20% of the labeled examples are used



(a) Negative rating (Class 0) (b) Neutral rating (Class 1) (c) Positive rating (Class 2)

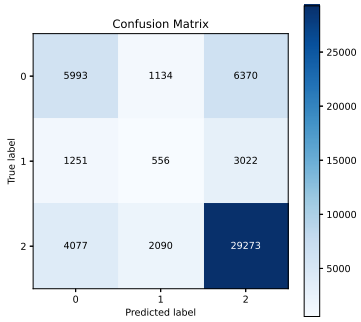
Figure 4.15: F1 curves of BERT-mini LSTM and SSGAN-BL classifiers for each class labels

to train the models, the semi-supervised approach obtains worse results than the supervised one. This may be due to the fact that, from 10% onwards, the supervised approach has enough data to obtain good enough internal representations of the reviews on its own compared to the semi-supervised approach

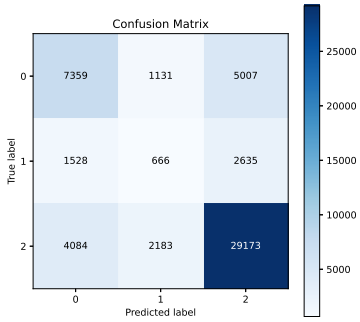
Moreover, it can be observed in Figure 4.15b that the semi-supervised model obtains better results than the supervised model for the class with the least instances (neutral class), regardless of the percentage of labeled data used. Thus, the model benefits more from adversarially enhanced data representations by the SSGAN-BL model.

The difference in the performance of the two models for the underrepresented class can be seen in Figure 4.14a. The Macro-F1 metric, described above, demonstrates a greater differentiation in the performance of both approaches regardless of the percentage of labeled data. Considering the performance of SSGAN-BL in class 1 (Neutral class), the mean of the values obtained for each class individually is higher than in the case of BERT-BiLSTM. However, Figure 4.14b shows that there is no statistically significant difference in terms of Micro-F1 (below 10% of labeled data). This may be due to a slight increase of failures in SSGAN-BL when classifying instances of the other classes (positive or negative) at the expense of improving performance with the under-represented class (neutral). As seen in the confusion matrices (see Fig. 4.16), there is a similar behavior pattern when comparing both approaches. It can be seen how the number of reviews correctly classified as class 1 (neutral) is higher with SSGAN-BL (see Fig. 4.16f) than with BERT-mini BiLSTM (see Fig. 4.16c). However, the number of reviews correctly classified as class 0 (negative) is slightly lower with SSGAN-BL. This is repeated for each of the comparisons up to Figures 4.16g and 4.16i onwards. From this setup, there is also a decrease in the number of reviews correctly classified as class 2 (positive) by the SSGAN-BL model. The latter fact can be seen in Figures 4.14 and 4.15 from the 10% of labeled data.

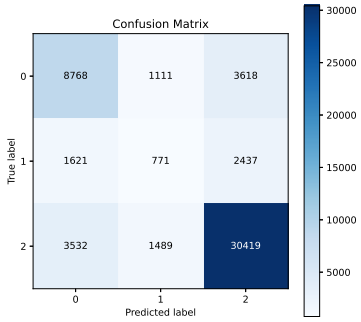
Moreover, a pattern of behavior can be seen in the confusion matrices, which is not reflected by the F1 metrics. When observing Figures 4.16b and 4.16e, it can be noticed that the SSGAN-BL model classifies reviews less often in the opposite polarity than BERT-BiLSTM. In contrast, BERT-BiLSTM is more inclined to classify a positive example as a



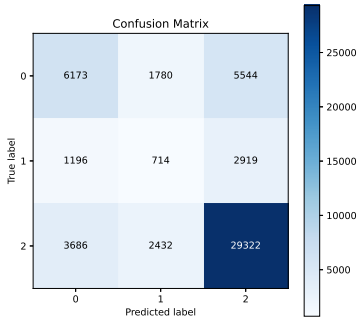
(a) BERT-LSTM
(1% of annotated data)



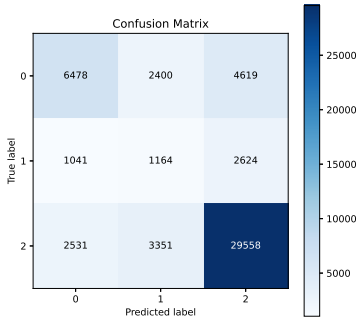
(b) BERT-LSTM
(2% of annotated data)



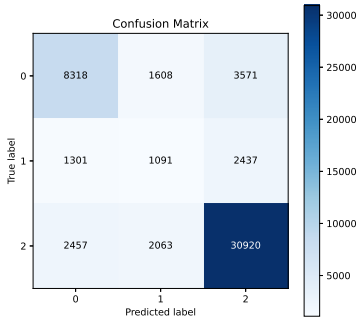
(c) BERT-LSTM
(5% of annotated data)



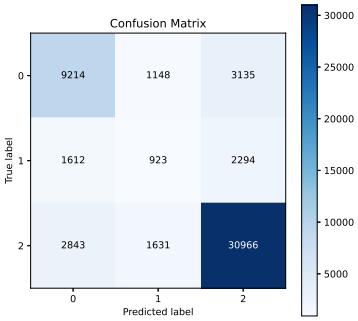
(d) SSGAN-BL
(1% of annotated data)



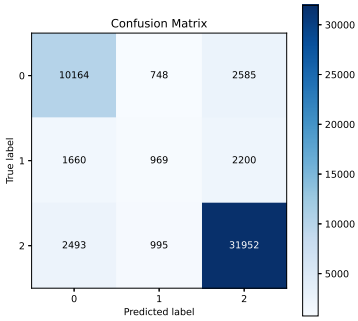
(e) SSGAN-BL
(2% of annotated data)



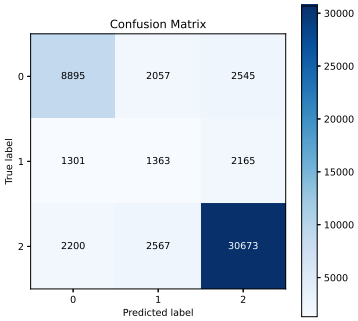
(f) SSGAN-BL
(5% of annotated data)



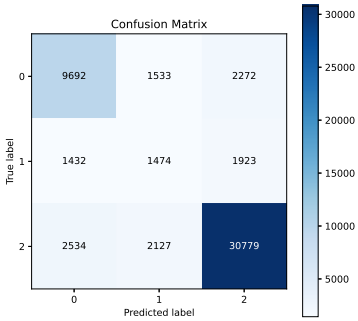
(g) BERT-LSTM
(10% of annotated data)



(h) BERT-LSTM
(20% of annotated data)



(i) SSGAN-BL
(10% of annotated data)

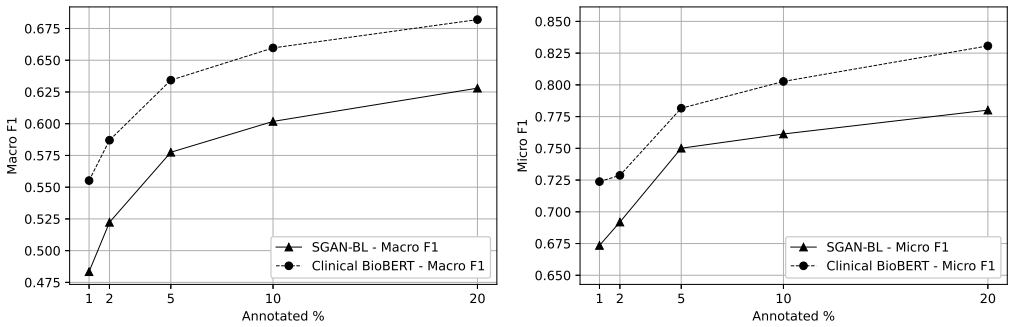


(j) SSGAN-BL
(20% of annotated data)

Figure 4.16: BERT-LSTM and SSGAN-BL classifiers confusion matrices

negative one or a negative example as a positive one. It can be argued that misclassifying a polarized review as neutral is preferable to misclassifying such a review in the opposite polarity (e.g., positive, as negative). For example, the following review is labeled in class 0 (negative): "*Have been on Actos for almost a year, gained 24 pounds and have swelling in hands and feet and are retaining a lot of water in my thighs. My sugar levels are good. My doctor lowered my dosage from 30 mg to 15 mg but refused to take me off. Will get a second opinion because the side effects are too much.*". In this case, the BERT-LSTM model classifies it as positive, while the SSGAN-BL model classifies it as neutral.

Finally, although the SSGAN-BL model is able to slightly outperform the BERT-small BiLSTM results when less than 10% of the annotated data is available, it is completely outperformed by the supervised approach of Clinical BioBERT, as can be seen in Figure 4.17 below.



(a) Macro F1 - Annotated data (%)

(b) Micro F1 - Annotated data (%)

Figure 4.17: Macro-F1 and Micro-F1 curves of SSGAN-BL and Clinical BioBERT classifiers

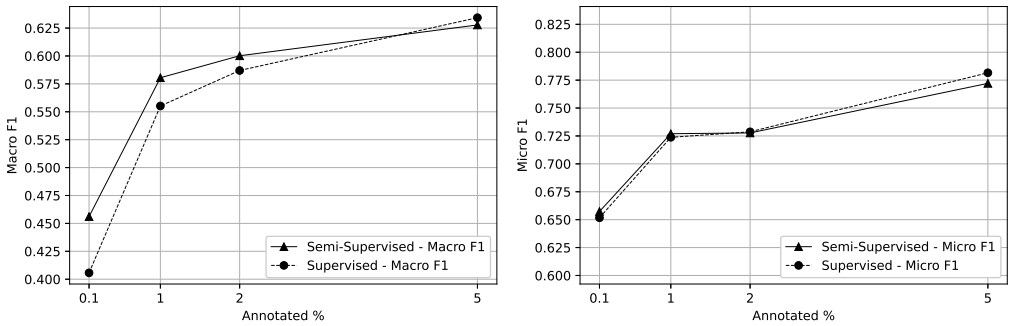
Results obtained by Clinical BioBERT and SSGAN-CBB

Figures 4.18 and 4.19 show the performance of the supervised and semi-supervised approaches using different ratios of labeled (Annotated %) and unlabeled reviews. In this way, the difference in the results provided by the semi-supervised and supervised approaches will be shown when only limited subsets of labeled reviews are available.

In this case, it can be seen in Figure 4.18 how the Clinical BioBERT model is able to improve the performance of SSGAN-CBB on its own once the number of annotated data exceeds 2% of the total. As mentioned above, in the case of BERT-small, this may be because Clinical BioBERT has the ability to obtain good enough internal representations from that percentage of annotated data. Therefore, it was proposed to analyze the capacity of Clinical BioBERT and SSGAN-CBB when only 0.1% of the annotated data is available, obtaining an improvement at this point of 5.04% in terms of macro-F1. However, as shown in figure 4.18b, no significant improvement in micro-F1 is obtained at any time.

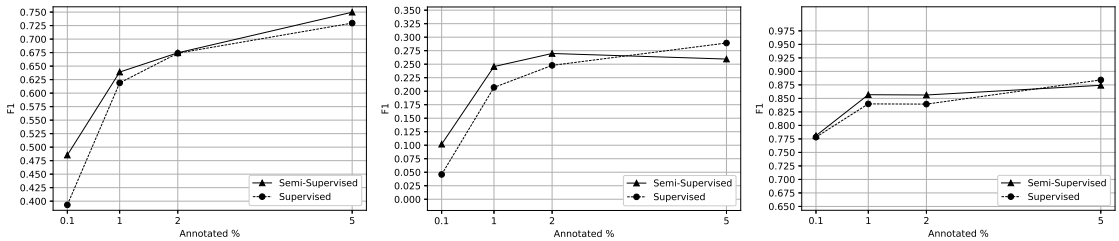
In a more detailed view at the class level (Figure 4.19), improved results are observed

for negative and neutral classes when the amount of annotated data is below 2%. These increases are slightly higher than those obtained with a semi-supervised approach with BERT-small LSTM and are much more pronounced when less annotated data is available. However, from 2% onwards, performance in the neutral class is reduced in favor of the negative class.



(a) Macro F1 - Annotated data (%) (b) Micro F1 - Annotated data (%)

Figure 4.18: Macro-F1 and Micro-F1 curves of Clinical BioBERT and SSGAN-CBB classifiers



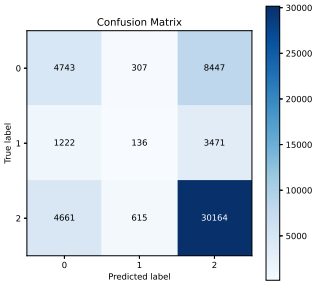
(a) Negative rating (Class 0) (b) Neutral rating (Class 1) (c) Positive rating (Class 2)

Figure 4.19: F1 curves of Clinical BioBERT and SSGAN-CBB classifiers for each class labels

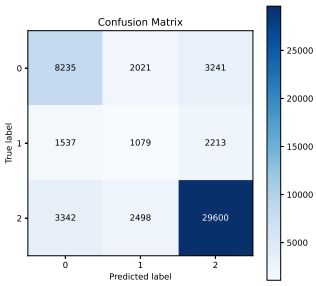
Moreover, as with the SSGAN-BL model, a certain pattern of behavior can be observed when looking at the confusion matrices between the supervised and unsupervised approaches (Figure 4.20). When less than 2% of annotated data is available, the SSGAN-CBB model is able to increase the number of hits in class-neutral reviews concerning the Clinical BioBERT model while at the same time reducing the number of reviews classified in the opposite polarity to the real one. However, as the number of reviews labeled as neutral class increases, there is also an increase in the number of examples misclassified as neutral.

4.2.5. Conclusions

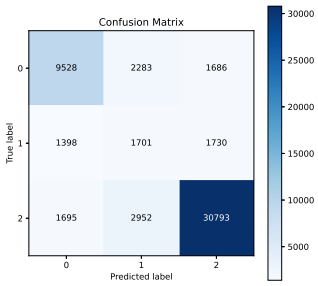
In recent years, the emergence of transformer-based approaches has revolutionized the field of NLP. Models such as BERT have achieved state-of-the-art results for many NLP



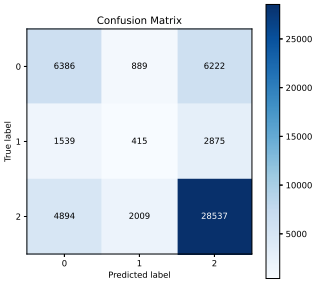
(a) Clinical BioBERT (0.1% of annotated data)



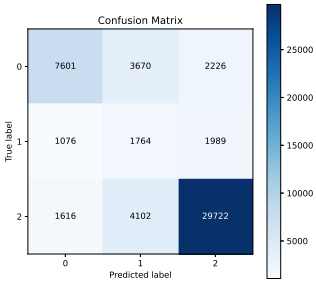
(b) Clinical BioBERT (1% of annotated data)



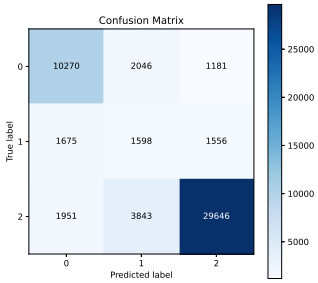
(c) Clinical BioBERT (5% of annotated data)



(d) SSGAN-CBB (0.1% of annotated data)



(e) SSGAN-CBB (1% of annotated data)



(f) SSGAN-CBB (5% of annotated data)

Figure 4.20: Clinical BioBERT and SSGAN-CBB classifiers confusion matrices

tasks. However, to fit these models to specific tasks, a large amount of annotated corpora is still necessary and not always available.

This work proposes a semi-supervised adversarial network (SSGAN) approach capable of addressing the SA of drug reviews when the size of the labeled dataset is limited. In order to evaluate the real contribution of this approach, a comparison between two BERT-based models with a supervised and a semi-supervised approach has been performed.

Since the training data used are highly unbalanced, the experiment results show that the semi-supervised approach can help in terms of Macro-F1 when the amount of such annotated data is reduced. Therefore, the contribution of the SSGAN adversarial approach can help to identify better examples of those classes that are under-represented. However, in terms of Micro-F1, the differences are not statistically significant.

The amount of annotated data required for the semi-supervised approach to fail to improve results over the supervised approach depends on the complexity and capacity of the BERT model used to represent the texts for the specific task. Specifically, the BERT-small model requires more than 10% of annotated data to outperform the semi-supervised approach, while Clinical BioBERT starts to outperform the semi-supervised approach from 2% of the total annotated data.

The semi-supervised approach provides promising results when very small amounts of labeled data are available and will be explored for other NLP tasks such as NER, RE or text summarisation, among others; however, there is still much room for improvement. Therefore, it plans to propose new deep learning methods capable of automatically generating instances for training, reducing the dependency on annotated and unannotated datasets. Furthermore, other transformer-based approaches, such as GPT-2[223] and XML[277], will be explored to address these tasks within the environments that allow semi-supervised approaches and few-shot transfer learning. The basic idea of this technique is to reuse a model, which was trained to distinguish between different classes, to learn to identify examples of classes unknown to the model, using only a few seed examples.

4.3. Data Augmentation approaches: Generative Pre-trained Transformers

4.3.1. Introduction

As mentioned in the conclusions of section 4.2.5, adding unlabeled data in SSGANs provides promising results when the available training data is limited; however, there is still sample room for improvement. Therefore, this section explores the use of data augmentation approaches.

Deep learning approaches have achieved state-of-the-art results [244], but such results are highly dependent on the quantity and quality of training data for a specific task. Therefore, data augmentation comprises a large number of techniques commonly employed in computer vision tasks, such as generating new images from translations, rotations, and alterations in the RGB channels of the original images [278]. However, when focusing on NLP tasks, such as SA, the techniques employed generally consist of replacing words with synonyms and randomly performing word insertions, swaps, and deletions [122].

Data augmentation approaches are not specifically based on the selection and modeling of classifiers but on the development and processing of training data for the models. Thus, the data augmentation approach consists of artificially augmenting training data and might be more relevant than classifier selection when dealing with very limited datasets [279]. Moreover, with the artificial generation of training data, data imbalance problems can be mitigated and, at the same time, provide robustness to classifier performance due to the regularisation effect of this type of approach [280].

However, data augmentation in NLP tasks faces an added difficulty in contrast to other tasks: maintaining the quality of annotations in the generated texts. For example, in the sentence “the movie was good” synthetically adding the word “not” changes its polarity in SA. Similarly, when using generative models where the selection of a new token depends on the probability distribution given the predecessor words (4.6), using a context like “the movie was” the model would assign a high probability to words like "good", "slow" and

"horrible" among several, completely modifying the label of the text (positive, negative) [281].

$$P(w_t|w_1, ..., w_{t-1})\forall t \tag{4.6}$$

In recent years, generative approaches are gaining interest in data augmentation research due to the increasing capabilities of generative language models. Such models are increasingly capable of creating texts incorporating new information that can help in SA. Models based on VAEs [282], [283], RNN [284], and GANs[285], [286] have been extensively studied for this purpose. GPT2 and GPT3 [223] are no exception to this increase in capabilities, obtaining state-of-the-art results in text generation tasks [124], [287] at the cost of being very complex and computationally expensive models to be fine-tuned for specific tasks [235].

In order to increase the performance of classifiers when the number of drug reviews is limited, this work proposes using a GPT-J [224], [225] model to create new synthetic reviews that can provide new information to classifiers without the need to use unlabeled data.

To mitigate the effect of generating new synthetic drug reviews with an erroneous label, some work has explored the use of other classifiers[288] or clustering using cosine distance on the latent representations of the texts[289] in order to filter synthetics. In this work, we propose to perform conditional fine-tuning with the available drug reviews by including label information at the end of each revision during training. Finally, it is proposed to modify the temperature parameter (scaling the logits before softmax normalization) of the GPT-J model when obtaining the label of a new synthetic text to avoid adding possible biases that other classifiers or Siamese networks could provide. Once the synthetic revisions are obtained, it is proposed to include them in the training set to refine the Clinical BioBERT model instead of the BERT-small due to its better performance in a limited training data scenario.

4.3.2. Dataset

In this section, the dataset of drug reviews used will be the same as the one described in the previous sections (4.1.2 and 4.2.2. A dataset created by [47] from collecting reviews from Drugs.com and Druglib.com where each review has been grouped into three levels of polarity based on its rating: positive ($rating \geq 7$), negative ($rating \leq 4$) and neutral (rating [5,6]).

In order to make a comparison between the supervised non-data augmented and semi-supervised approaches described above in sections 4.1 and 4.2, a series of splits (see Table 4.13) were performed to obtain subsets of annotated data and simulate data sparsity. By employing these splits, it will be possible to compare the results obtained by different approaches when the same limited amount of annotated data is available.

	Number of labeled reviews			
Annotated %	**Total**	**Negative (0)**	**Neutral (1)**	**Positive (2)**
0.1	137	34	12	91
1	1,371	341	121	909
2	2,742	682	243	1817
5	6,855	1,705	607	4,543

Table 4.13: Percentage of labeled reviews out of the total dataset used for the training phase.

4.3.3. Methods

The GPT models are based on transformers [76]; however, GPT differs from BERT mainly in that the latter is composed of encoders, while GPT employs decoders. These decoders modify the masking of the multi-head attention layers to prevent a particular position in the token sequence from attending subsequent positions. In this way, GPT behaves as an autoregressive model where new tokens are generated iteratively.

Since the most current versions of GPT3 are only accessible through the OpenAI API, the open-source GPT-J[224], [225] model was used in this experimentation. This model consists of a total of 6 billion parameters and is composed of a total of 28 layers with a model dimension of 4,096, as opposed to the GPT2 model, where the longest model is composed of 1.5 billion parameters. In addition, GPT-J was trained by Wang, Ben, and Aran Komatsuzaki employing a large-scale dataset called the Pile, an open-source dataset of 825GiB in size derived from academic and professional sources [290].

The tokenizer and the pre-trained GPT-J model used in the experimentation were provided by the transformers library[15], the tokenizer being the same as the one used by GPT-2/3 with a vocabulary of 50257 words. Due to the model's size, fine-tuning was carried out on an Nvidia A100 graphics card with 80GB of VRAM and 200GB of RAM. For this purpose, the software package deepspeed[16] was also employed to provide memory-efficient data parallelism to increase scalability and train models with a large number of parameters while consuming fewer resources.

In order to use the language model for data augmentation, it must be ensured that the model is able to generate texts similar to the training set while maintaining consistency with the label. In the following, the method of this work is described, composed of two steps: fine-tuning the GPT-J model for synthetic drug review generation and fine-tuning the Clinical BioBERT model for drug review classification.

[15]https://huggingface.co/docs/transformers/index
[16]https://www.deepspeed.ai/

Fine-tuning the GPT-J model for synthetic drug review generation

In this first step, the GPT-J model is fine-tuned with the available training set, i.e., with the different splits of annotated data. This allows the model to learn the different language features contained within the texts of the training set. Moreover, GPT is a language model that allows us to obtain the probability of the current word given a set of predecessor words. Therefore, when training in a downstream task, using a prefix or a sequence of words (prompt) as input to the model can serve as a context for generating a certain type of text. Although the remaining unlabeled data can provide further information on the language features of the reviews, due to the amount of data and computational complexity, only the data from the annotated splits are used in this work.

A special "*start of text*" token is added at the beginning of each training data to enforce task-specific text generation. This token behaves as a prefix indicating to the model that it has to generate texts similar to those in the training set, in this case, drug reviews. On the other hand, the model must also be able to relate the distribution and patterns of text sequences to their class (positive, neutral, negative). For this purpose, per-class models can be trained where each model learns the distribution and linguistic features of each class[289], but also a conditional model where the training data explicitly contain information about the label of each example: $P(w_t|w_1, ..., w_{t-1}, c)\forall t$.

In cases such as [291], contextual information is added as a prefix to the language model prompt to provide control when generating new text. However, for the task of SA in drug reviews, adding polarity to the prompt can cause problems when generating conditional reviews, causing the resulting texts to be unrelated to the specific class. This may be due to the autoregressive feature of the model, as the generation of tokens takes into account their predecessors (where the label information is located), as well as the non-deterministic selection of the most likely tokens. For example, during training, long positive texts could contain negative sentiment words that refer to a specific topic and not to the drug's performance. This feature will increase the probability that negative sentiment words will be generated independently of the conditioned context, completely changing the meaning of the new text:

"*I was on the patch for 7 months. When I switched to this, I didn't have any issues at all. I did experience some headaches, which went away after 3-4 months. I did notice that I gained a lot of weight. I had to stop using the patch in December.*"

The beginning of the generated text matches the positive sentiment: "*I was on the patch for 7 months. When I switched to this, I didn't have any issues at all*". However, as it continues to generate, several adverse effects are described: "*I did experience some headaches, which went away after 3-4 months. I did notice that I gained a lot of weight.*" which increases the probability of the model generating: "*I had to stop using the patch in December.*", indicating a bad experience and therefore a negative opinion towards the drug.

Therefore, the label information is introduced at the end of each text, where class probability depends on the preceding words $P(c|w_1, ..., w_{t-1})\forall t$. However, GPT-J aims to predict the next token in a non-deterministic way from the vocabulary. Thus a sequence of text could be erroneously labeled in a less probable class or another token in the vocabulary. In order to prevent this problem without fine-tuning by adding a classification layer to the model, it is proposed to modify the temperature parameter explained below.

Finally, all instances of drug reviews are composed of a prefix token "<|*startiftext*|>" and two suffix tokens "|*{sentiment}*|" and "<|*endoftext*|>. For all cases, *{sentiment}* is replaced by the positive, neutral or negative class (see Table 4.14).

The GPT-J model was trained using the AdamW[292] optimizer, with a learning rate of 5e-5, 100 warmup steps, a batch size of 16 for four epochs, and cross-entropy as a loss function.

Original Text
I have been on this medicine on and off for eight months now. I weighed 210lbs when I started and now weigh 163lbs. I have not changed eating habits however food portion and exercise is the key.
I had a spot of gastritis. First had to take 150mg twice a day, but still had issues so I went up to 300mg twice a day, and sometimes still having issues...
Almost halfway through my second week I had to stop taking Contrave because it jacked up my blood pressure. BP probably would've been a lot higher than the 156 93 had I not been talking blood pressure medication already.
Processed Text
<|startoftext|>*I have been on this medicine on and off for eight months now. I weighed 210lbs when I started and now weigh 163lbs. I have not changed eating habits however food portion and exercise is the key.* ***(Sentiment: Positive)<|endoftext|>***
<|startoftext|>*I had a spot of gastritis. First had to take 150mg twice a day, but still had issues so I went up to 300mg twice a day, and sometimes still having issues...* ***(Sentiment: Neutral)<|endoftext|>***
<|startoftext|>*Almost halfway through my second week I had to stop taking Contrave because it jacked up my blood pressure. BP probably would've been a lot higher than the 156 93 had I not been talking blood pressure medication already.* ***(Sentiment: Negative)<|endoftext|>***

Table 4.14: Example of text pre-processing. The examples shown are short reviews due to a lack of space.

In order to evaluate the performance of generative models, a wide variety of metrics are used; for example, for MT and summarisation tasks, the metrics BLEU and ROUGE scores are widely used [293], [294]. However, they are parameterized metrics used with a target text in mind by making comparisons based on n-grams, where the result may vary depending on the parameters used [295]. In data augmentation tasks, something similar happens, and there is no specific target text to compare, as is the case in MT and summarization tasks. Metrics such as affinity and diversity proposed by [296] aim to help

in this direction; however, they are based on results obtained by other models that use data augmentation to make relative comparisons.

To evaluate the performance of the GPT-J model, this work employs the Perplexity (PPL) metric, a metric defined as the exponentiated mean negative log-likelihood of a sequence (4.7) used in data augmentation tasks [297], [298]. While entropy can be seen as the amount of information needed to encode the outcome of a random variable, PPL provides information about the number of options a random variable has. Thus, the lower the PPL value, the better the model will be at predicting a given previously unseen test dataset.

$$PPL(X) = \exp\left\{-\frac{1}{t}\sum_{i=1}^{t} logP(x_i|x_{\leq i})\right\} \tag{4.7}$$

As can be seen in Table 4.15, GPT-J was fine-tuned to a PPL between 5 and 6 for each of the percentages of data used. Unlike the vanilla model, the fitted models are able to provide texts similar to the test dataset. In the worst-case scenario, a generative model would provide a PPL value equal to the size of its vocabulary, being 50,257 tokens in the case of GPT-J.

Annotated data (%)	**PPL**
0%	1,725
0.1%	5.02
1%	5.63
2%	5.99
5%	5.44

Table 4.15: PPL of the GPT-J model on the test set. GPT-J was fine-tuned on different percentages of annotated data from the training set.

Moreover, a visual approach similar to [122] is taken to evaluate that the generated synthetic texts maintain a similar distribution to the original texts, where the Clinical BioBERT model is trained without data augmentation. Then, since a larger number of examples requires more computational time, 10 new synthetic texts are generated for each review of a subset of the test dataset, and both the original drug reviews of the test set and the synthetic ones are fed into the classifier model by extracting the latent representation of the last layer. Subsequently, t-SNE [299] is applied to the latent representations obtaining a spatial representation where it is observed that most of the synthetic texts are closely surrounding the original reviews, suggesting that the generated synthetic texts maintain distributions inherent to each of the classes (see Fig. 4.21).

GPT models incorporate a parameter called "*temperature* τ" which consists of scaling the logits before normalizing them with the softmax function, as can be seen in (4.8). Thus, if the temperature has a low value, tokens with the highest probability are more

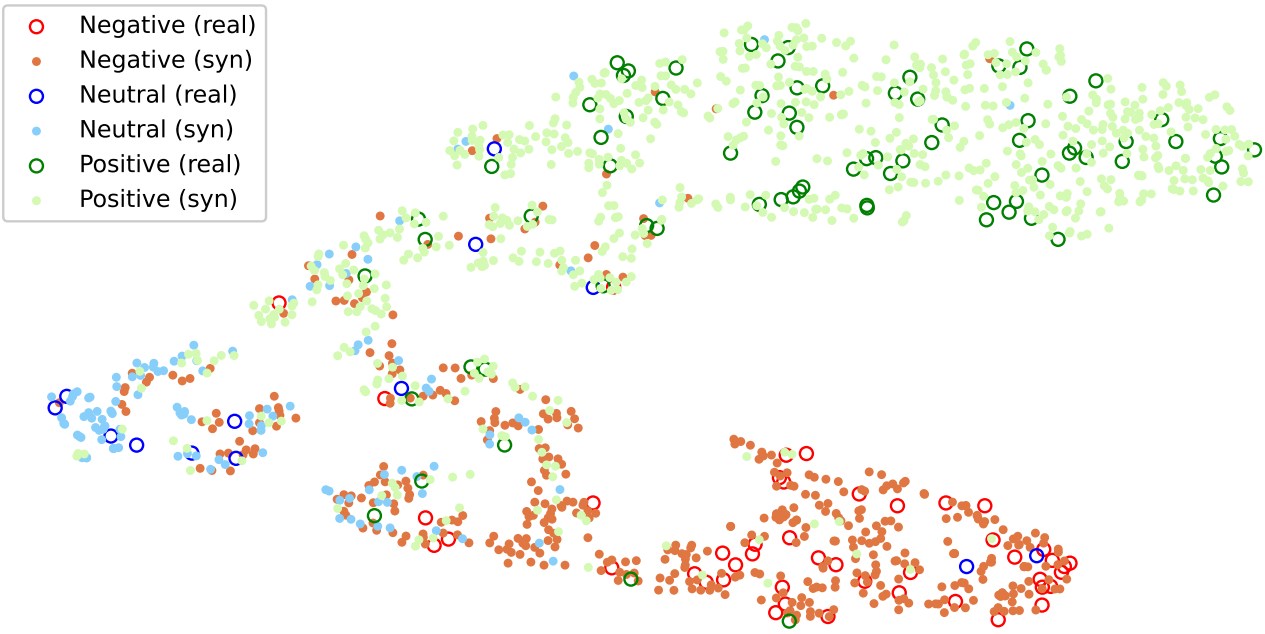


Figure 4.21: Latent space visualisation of real and synthetic reviews.

probable to be sampled, resulting in small text variations. On the other hand, a high value for the temperature increases the possibility of sampling tokens with smaller probability, obtaining more diverse texts but with a higher possibility of grammatical, syntactic, or semantic errors.

$$\sigma(z)_i = \frac{e^{z_i/\tau}}{\sum_{j=1}^{K} e^{z_j/\tau}}, i \in (1, ..., k) \tag{4.8}$$

Our process for data augmentation can be divided into two parts: the first part consists of generating texts with a certain degree of diversity that can contribute new information to the SA task. The second part is to "generate" the label of the new texts leaving little freedom of choice in the resulting probability distribution (see Figure 4.22).

In order to generate new drug reviews, the model prompt must include the special token "<|*start of text*|>", as it provides context to the model to start iteratively generating new tokens prioritizing those in the training set. This prevents the model from starting to generate texts that are outside the context of the drug reviews. The temperature is set to 0.8, with values between 0.7 and 0.9 recommended [289]. This provides greater freedom when generating new tokens, giving more "creativity" to the process. Specifically, a high temperature modulates the logits prior to the softmax function, shortening the differences in the values of the probability distribution of the different tokens in the vocabulary. Finally, our model should be able to provide us with synthetic drug reviews, as shown in Table 4.16.

Once the new reviews have been generated, all tokens except the generated label (Positive, Neutral, Negative) are kept, letting each text end with the sequence "*(Sentiment:* ". This increases the probability that the next token to be predicted will be one of the desired

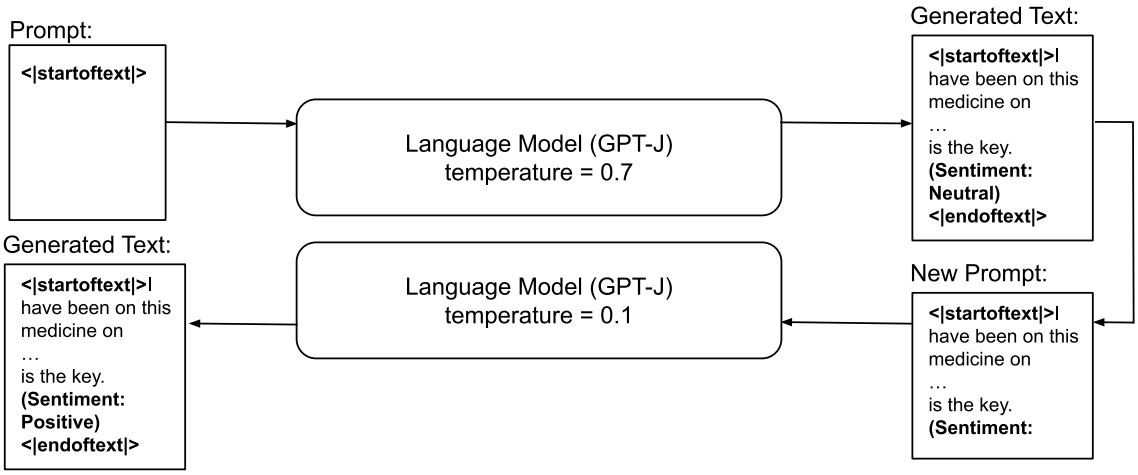


Figure 4.22: Overview of data augmentation method

classes. Subsequently, the resulting text is used as input to GPT-J with a low temperature, preventing the model from being free to select lower probability tokens from the probability distribution. A temperature close to zero brings the model closer to a deterministic generation.

Finally, examples of synthetic drug screens are shown in Table 4.16 below. While synthetic reviews show consistent generation of language and alignment with the label, it should be noted that such reviews may not accurately reflect real-world experiences with these drugs.

For evaluating the effect of data augmentation on the SA task, the amount of synthetic data was limited to a maximum of 10 augmented examples per review since a large number of examples requires more computational time.

Fine-tuning the Clinical BioBERT for drug review classification.

Once the synthetic reviews are generated, Clinical BioBERT is trained using both real and synthetic data for the SA task. In the previous sections of Chapter 4, it was possible to observe the ability of Clinical BioBERT to outperform the rest of the proposed models. Furthermore, it was shown that Clinical BioBERT outperforms BERT-small LSTM when using a limited amount of annotated data. A more detailed description of Clinical BioBERT can be found in Section 4.1.4.

For the training of the model, it was used the ADAM optimizer [264], with a learning rate of 5e-5, a batch size of 128 during 10 epochs and categorical cross-entropy as loss function

Label	Synthetic Review
Negative	This birth control is very good if enjoy bleeding 26 of 28 days. I was on this to control my bleeding. Now I just bleed every day.
Negative	This drug made me extremely nauseous for weeks at a time, even after taking it with food. It also gave me migraines, which I have never had. I know many people have had good results, and I am glad it works for them.
Negative	I started taking this for restless leg syndrome and I have horrible involuntary muscle spasms in my legs. I have had to go to the ER when it gets really bad. It is a terrible drug.
Neutral	I have been using Taclonex for 3 days for my back pain. It seems to subside the pain somewhat, but not 100 percent. I also take percocets when required as well. The only side effect I have had is the drowsiness.
Neutral	I was prescribed this drug recently for my back pain and headaches. It seems to subside the pain somewhat, but not completely. I find its strength somewhat limiting, and I have to take it at night.
Neutral	I had oral surgery and my doctor prescribed me me this and it didn't touch the pain at all.
Positive	I started this medication a month ago. I take it at night. I'm a lot happier and my attacks are gone. The first night I had terrible pains in my stomach but after that night no more problems. I highly recommend this medication.
Positive	Seeks to be working well, had 3 teeth extracted, and my dentist prescribed this medication due to the soft tissue swelling, tissue death, and bone loss caused by the extraction, along with the general fatigue, ache, and pain that I have had.
Positive	I was given Trulicity 0.75mg when I was 218lbs and I lost 22 lbs in 3 months. The only side effect I had was the dry mouth, but I can live with that. I highly recommend this medication.

Table 4.16: Examples of synthetic drug reviews. The examples shown are short reviews due to a lack of space.

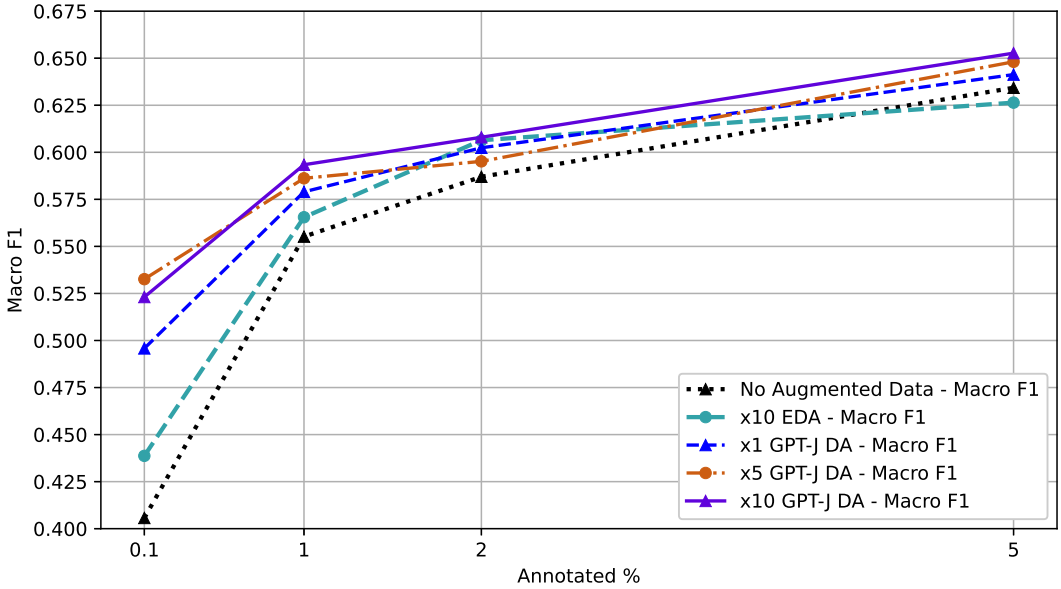
4.3.4. Evaluation

This section focuses on the results obtained with the data augmentation approach on the test dataset described in section 4.1.2, as well as describing the different scenarios in which the approach was tested.

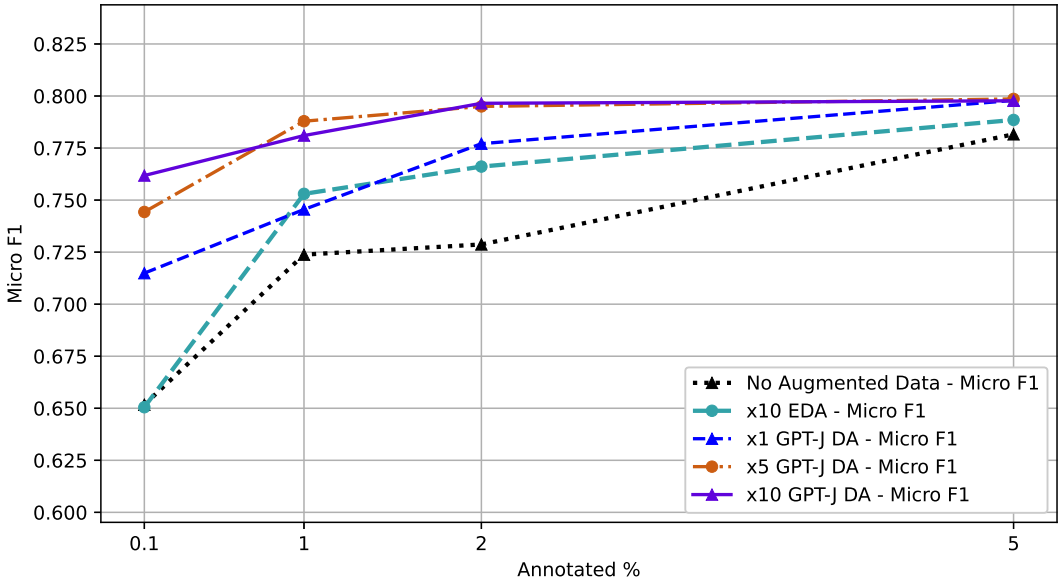
As can be seen in Table 4.13, the evaluations were conducted on different percentages of annotated data. For the different scenarios, the GPT-J model was fine-tuned using only the percentage of available data, while Clinical BioBERT was trained using, in addition, the new synthetic data. Furthermore, the EDA method [122] is used as a baseline for the data augmentation approach. This allows us to compare the results obtained by our approach based on GPT-J with the results provided by this baseline.

As shown in Figure 4.23, the higher the number of augmented data, the better the performance. However, although it would be logical to think this is a linear behavior, the

improvement decreases as the number of augmented data increases. This might be due to the increase in repetitions produced when we increase the number of augmented data, avoiding providing new patterns or linguistic features.



(a) Macro F1 - Annotated data (%)



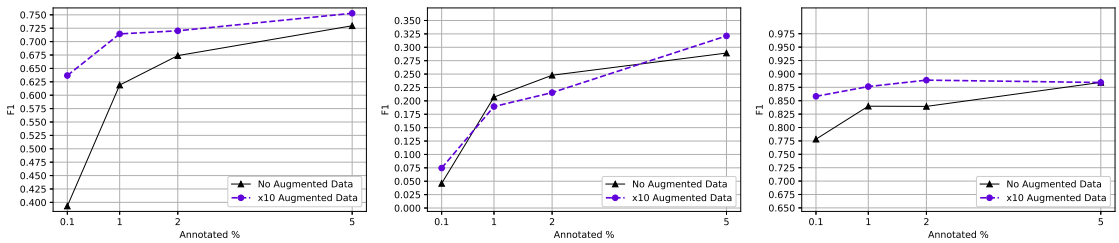
(b) Micro F1 - Annotated data (%)

Figure 4.23: Macro-F1 and Micro-F1 curves of Clinical BioBERT with Data Augmentation

In both Figures 4.23a and 4.23b, a significant increase in results can be observed, especially in those scenarios where the amount of annotated data is more limited than the others. Specifically, the largest increase can be seen when only 0.1% of the annotated data is used. In this case, Clinical BioBERT benefits from the contribution of augmented data because the implicit knowledge of the generative model itself contributes new information to the dataset used by Clinical BioBERT. However, the results seem to converge

when the number of available annotated data increases. As we can see in 4.23b, there seems to be a flattening in micro-F1 when the amount of data is 5%. Similarly, in terms of macro-F1, the benefits are shortened when sufficient annotated data is available, regardless of the amount of augmented data included. At the same time, it can be seen how the data augmentation approach with GPT-J improves the results provided with EDA (10 augmented data per review). This could be attributed to GPT-J's greater capacity to generate coherent texts, whereas EDA relies on internal substitutions and permutations within existing texts. Furthermore, while EDA can incorporate synonyms during word substitution, it lacks control over contextual relevance. On the other hand, while EDA can augment text variability through its permutations and substitutions, GPT-J enriches the diversity of generated texts by transferring learning from its pre-training. The difference is most noticeable when we use only 0.1% and 1% of the annotated reviews, while from 2% and above, the benefit of data augmentation starts to shorten.

In more detail, Figure 4.24 shows the results at the class level when 10 synthetic GPT-J reviews per review are added, where, following the same pattern as described above, the positive and negative classes benefit the most. However, this is not the case for the neutral class, where the results obtained are even lower in some scenarios. This might indicate that being the minority class, the generative model does not have enough information to generate new quality texts for the neutral class until the 5% of annotated data is reached.

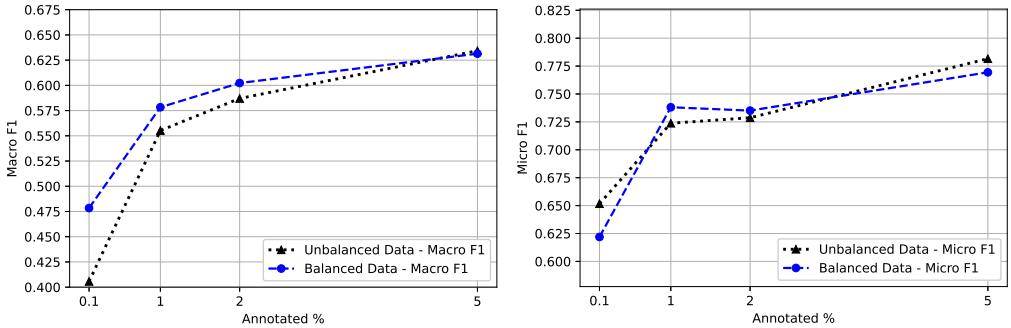


(a) Negative rating (Class 0) (b) Neutral rating (Class 1) (c) Positive rating (Class 2)

Figure 4.24: F1 curves of Clinical BioBERT with Data Augmentation for each class labels

In addition, in Figure 4.25, we can see the performance obtained when balancing the data, in this case, equating the number of revisions of each class to the most represented of them. The results obtained are similar when compared to the unbalanced data. However, an increase in macro-F1 is observed when the number of real revisions available is lower.

The overall results show an improvement when using data augmentation, especially benefiting those scenarios where the amount of data is limited. This is not the case when increasing the amount of real data available from the training set, as the classifiers start to learn the different patterns and linguistic features inherent to each class. For example, when using 100% of the training data (Table 4.17), it can be seen that there is no statistically significant difference. In [300], it is suggested that using large pre-trained transformer models and transfer learning currently meets the goal of data augmentation since they only perturb the input and do not add any new linguistic patterns. Nevertheless,



(a) Macro F1 - Annotated data (%)

(b) Micro F1 - Annotated data (%)

Figure 4.25: Clinical BioBERT Macro-F1 and Micro-F1 curves with balanced data augmentation.

with large models such as GPT-J and their potential to provide unseen patterns relevant to the task, the increase in data seems beneficial for downstream tasks where the amount of training data is limited.

Label	x1 Data Augmentation	No Data Augmentation
Negative (Class 0)	0.8944	0.8939
Neutral (Class 1)	0.6539	0.6597
Positive (Class 2)	0.9560	0.9555
micro avg	**0.9147**	0.9138
macro avg	0.8348	**0.8363**

Table 4.17: Clinical BioBERT results using data augmentation (100% of annotated data). Due to computational cost, the comparison is performed using only one synthetic sample for each instance.

4.3.5. Conclusions

In recent years, transformer-based approaches have revolutionized the field of NLP. Moreover, transfer learning and large pre-trained models are capable of reducing the quantity of training data required for downstream tasks compared to upstream tasks. However, the quantity and quality of training data still play a crucial role in obtaining specialized models for specific tasks.

This work proposes a data-augmentation approach with the GPT-J [224], [225] language model, capable of addressing the SA task of drug reviews when the training data is limited. To evaluate the approach contribution, a comparison was made where the Clinical BioBERT [37] classifier was trained only on limited subsets of annotated data, as well as with the EDA [122] data augmentation approach with 10 samples per review.

The results show that GPT-J is able to generate synthetic texts that contribute new information to the SA task. Pre-trained generative models based on transformers contain implicit knowledge that can provide linguistic patterns not seen in the training set. Moreover, the results show similar performance to that obtained without data augmentation using less than half the number of real drug reviews. At best, the micro-F1 obtained with 5% of drug reviews is achieved using only 1% of the reviews and augmented data. However, this method is only effective when the number of available drug revisions is limited, as the new patterns and linguistic features provided by the GPT-J model will tend to be repeated the more augmented data is generated, or additional drug revisions are obtained.

Data augmentation approaches with generative transformer-based models provide beneficial results in low-resource data scenarios for SA of drug reviews. Therefore, their performance is planned to be explored for other NLP tasks such as text summarization, NER, or RE. At the same time, to maintain consistency among the class associated with a synthetic sample, it is planned to compare the performance of the temperature parameter against other filtering approaches based on comparing similarities between latent representations of texts and synthetic examples.

However, the emergence of large language models pre-trained on massive corpora such as GPT3 [301] or T5[302] may provide a language understanding capable of addressing few-shot learning approaches [235] with promising results based on contextual information. Future work plans to use such models to develop approaches to NLP tasks where a limited amount of training data is available.

4.4. Conclusions

Since the beginning of the century, millions and millions of user reviews have been available on the internet; however, the massive labeling of this data is becoming more and more complicated due to the high consumption of time and resources. As a result, the task of SA has become one of the most relevant fields within NLP.

In the pharmacological field, the automatic analysis of user reviews offers a rich amount of information about the efficacy of drugs, as well as their side effects, and can detect those effects that escaped clinical trials. Positive aspects detected in drug reviews could also discover new drug uses, while negative reviews can help detect unknown side effects.

Since analyzing patients' sentiments about their drug experiences has received less research attention, this chapter describes and compares several automated approaches to SA in drug reviews. This task is challenging due to several factors, patient subjectivity being one of the main ones. Moreover, in real situations, the amount of labeled information is a problem when designing automatic systems based on deep learning.

In order to address the challenges described above, an exploration of several deep learning approaches has been performed, showing that models based on pre-trained trans-

formers are able to address this task effectively, outperforming the results provided by classifiers based on convolutional or recursive networks. Specifically, Clinical BioBERT obtains 91.38% micro-F1 and 83.63% macro-F1, outperforming the best hybrid classifier of BiLSTM and CNN by approximately 2% in both metrics. These results are achieved when 100% of the drug reviews in the training set are employed (over 137,000 revisions). However, pre-trained models on large corpora such as Clinical BioBERT are not sufficient on their own when dealing with a low-data scenario for the downstream task. Different approaches, such as semi-supervised and data augmentation, have been proposed and evaluated to address this challenge.

The results show that using GANs in a semi-supervised approach can provide information on the latent representation of drug reviews using unlabeled reviews. However, it only seems to benefit the underrepresented classes as the classifier does not yet have enough information inherent to these classes. Furthermore, the results show that the number of labeled reviews required for a supervised approach of pre-trained BERT-based classifiers to match the performance of the semi-supervised approach depends on the implicit knowledge of the language of each classifier. Specifically, the SSGAN approach with Clinical BioBERT (SSGAN CBB) obtains 5% more macro-F1 than Clinical BioBERT when we use only 0.1% of annotated reviews and 100 unlabeled examples per review but does not improve the results when more than 2% of annotated reviews are available.

On the other hand, data-augmented approaches are able to provide information on features and linguistic patterns in few-data scenarios, outperforming the semi-supervised approach for the SA of drug reviews. Specifically, a transformer-based generator model, pre-trained on large corpora such as GPT-J, is able to generate synthetic drug reviews that improve the performance of Clinical BioBERT even when 5% of the annotated data is available, outperforming approaches such as EDA. At best, Clinical BioBERT obtains 11.74% more macro-F1 and 11.01% more micro-F1 when using only 0.1% of the annotated reviews, with micro-F1 being only 2% lower than when using 5% of the annotated data (78.16%).

The results have shown that generating synthetic texts using transformer models such as GPT-J can improve the performance of classifiers when limited data is available. However, although the results are promising, there is still room for improvement. Currently, due to the availability of increasingly complex transformer models pre-trained on large amounts of texts, it is starting to evaluate their performance with few-shot approaches for specific tasks, obtaining results similar, in some cases, to state-of-the-art with full training texts [235]. In future work, we plan to use models such as GPT3[301], T5[302] or BART [303] to address the task of SA on drug reviews in few-data scenarios.

Chapter 5

Named Entity Recognition in Clinical Texts

As mentioned in Chapter 4, NLP has become one of the most fruitful research areas due to the massive increase of texts available on the Internet. Within the different fields that comprise NLP tasks, this Thesis has explored the task of SA in drug reviews. However, NER task, focused on automatically identifying concepts or designators belonging to predefined semantic types within texts, such as places, people, organizations, quantities, etc.[142], can provide valuable information for a wide range of tasks such as IE [25], [26], text summarisation [28], MT [27], and others.

Over the years, approaches in NER have evolved into different techniques: rule- and dictionary-based approaches where no prior annotations are required as they are based on occurrences of hand-generated dictionaries and rules [304], [305]; unsupervised approaches based on clustering of groups with contextual similarity [306]; supervised machine learning approaches focusing on feature engineering of the training data [307], [308]; and deep learning approaches where representations of different entities are automatically inferred for the specific task [309]–[311].

In recent years, the emergence of deep learning techniques have provided state-of-the-art results [142]. However, large amounts of annotated data are required for these techniques to infer the necessary latent representations. Transfer learning helps mitigate this problem, and transformer-based models such as BERT, Roberta, XML, etc., are pre-trained on large corpora for this purpose. Nevertheless, since most existing approaches focus on broadly labeled and generalist datasets, a problem arises when working with EHRs [312] due to their intrinsic text features and exceptional conditions, such as privacy and PHI.

In the clinical field, automatic recognition of entities or concepts within a text can help reduce the resources needed for pharmacovigilance and de-identification of data, among other tasks[39], [153], [313]–[315]. Pharmacogenetics studies how genetic variation leads to variable responses to drugs. Although guidelines for selecting the correct drug and

dose based on genetics are clinically effective, they are not yet widespread [316]. For some drugs, these guidelines may lead to the selection of a treatment that minimizes side effects and maximizes efficacy. Thus, entity recognition from unstructured data stored in biomedical databases and clinical research resources could provide valuable information in other areas, such as phenotype definition [317]–[319].

The UK Biobank is a large-scale biomedical database, a resource for research aimed at improving disease prevention and diagnosis. The database contains information from over half a million volunteers, including health information and biological samples. These records contain structured information recorded by general practitioners or nurses and other records, clinical events, and drug prescriptions.

Although some prescription databases contain structured prescription information (Danish National Prescription Register [320]), others only store prescriptions without preprocessing. Such registries may provide drug codes but are usually national or in-house coding systems. Specifically, the UK Biobank prescription dataset contains unstructured information on drug name, quantity prescribed, prescription date, and various codes used in the UK NHS primary care for administrative and prescribing purposes.

For pharmacogenomic and pharmacoepidemiological studies, it is important to identify the active ingredients of the different drugs, considering synonyms and brand names. Once the active ingredients have been identified, they can be categorized and used for different studies, with the Anatomical Therapeutic and Chemical Classification (ATC) system, an international standard of the World Health Organization (WHO), being one of the most widely used systems.

To assist in mapping between prescription drugs and the ATC codes, identifying the pharmaceutical form and route of administration can provide information to distinguish drugs that reach the circulatory system (systemic effects), while strength and dosage can assist in quantitative studies. Therefore, identifying these entities can help characterize an individual's prescription trajectory, which can be used in genetic or pharmacoepidemiological analysis [321].

However, training supervised approach models on this task require large amounts of annotated data [65]. Due to the nature of medical prescriptions, the availability of annotated data is limited, resulting in a few-data scenario where transfer learning methods with similar annotated datasets can improve the performance of the models [47], [273]–[275].

In order to address the challenge of scarce or non-existent annotated medical prescriptions, dataset annotation is proposed following a pipeline consisting of two parts: 1) Silver annotation (i.e. automated or semi-automated annotation) using dictionaries obtained from the ChEMBL[17] database as well as predictive models pre-trained on similar tasks on EHR; 2) Manual curation performed by experts on the domain.

After annotating the dataset, our plan involves using it for fine-tuning the clinical

[17] https://www.ebi.ac.uk/chembl/

BioBERT model [37] on the NER task related to medical prescriptions. Moreover, we propose to evaluate the performance of the same model with an initial fine-tuning on another dataset from a similar task (2018 n2c2 dataset [44]), followed by fine-tuning on the target task dataset to compare the effect of transfer of learning from one task to another.

This chapter describes the approach used for entity recognition in medical prescriptions to help define phenotypes for Genome Wide Association Studies (GWAS) of differences in drug response in depression from anonymized EHR prescription data from the UK Biobank[18]. First, the datasets used, and the annotation process of the UK Biobank dataset are described. Subsequently, the models employed, their training, and the evaluation of the results and conclusions.

This process was carried out thanks to the Birney Research Group[19] of the European Bioinformatics Institute (EMBL-EBI) focused on inter-individual genomic variation.

5.1. Datasets

This subsection describes the datasets used for the NER task in clinical texts. Specifically, medical prescriptions and EHRs. These texts contain relevant information in an unstructured form about patients' treatments. The medication information detailed in the texts may include the drugs, their dosages and strengths, the form, and the route of administration. However, not necessarily all these details are present in all cases.

5.1.1. 2018 n2c2 dataset

As mentioned above, in a few-annotated data scenario, a prior fine-tuning to a similarly labeled dataset containing drug entities, pharmacological form, potency, dose, and route of administration can improve the predictive model's performance for the target task through transfer learning. Therefore, using the dataset employed in track 2 of the 2018 task n2c2 [44] on drug-related concept extraction is proposed.

The annotated corpus consists of a collection of Intensive Care Unit (ICU) discharge notes with detailed information on the drugs used in the treatments. The dataset consists of a training and a test set with 303 and 202 documents, respectively. However, it also includes annotations on 1,584 adverse events (ADEs), 10,293 on the frequency of drug administration (Frequency), 970 on the duration of treatment (Duration), and 6,400 on the reason for drug administration (Reason). Since the proposed NER task on UK BioBank prescriptions does not require the identification of these entities, they are all filtered out.

To obtain the length of the texts based on their tokens, the documents were first divided into sentences. For this purpose, the *en_core_web_sm* model available in the Spacy

[18]https://www.ukbiobank.ac.uk/
[19]https://www.ebi.ac.uk/research-beta/birney/

library[20] was used. Subsequently, the sentences were tokenized using the vocabulary of the pre-trained Clinical BioBERT model (28,996 tokens) and the tokenizer provided by the transformer library[21]. As can be seen from the probability density function 5.1, most sentence texts tend to vary between lengths of a few tokens up to 50. However, the maximum sentence length of the 2018 n2c2 dataset is almost 225 tokens.

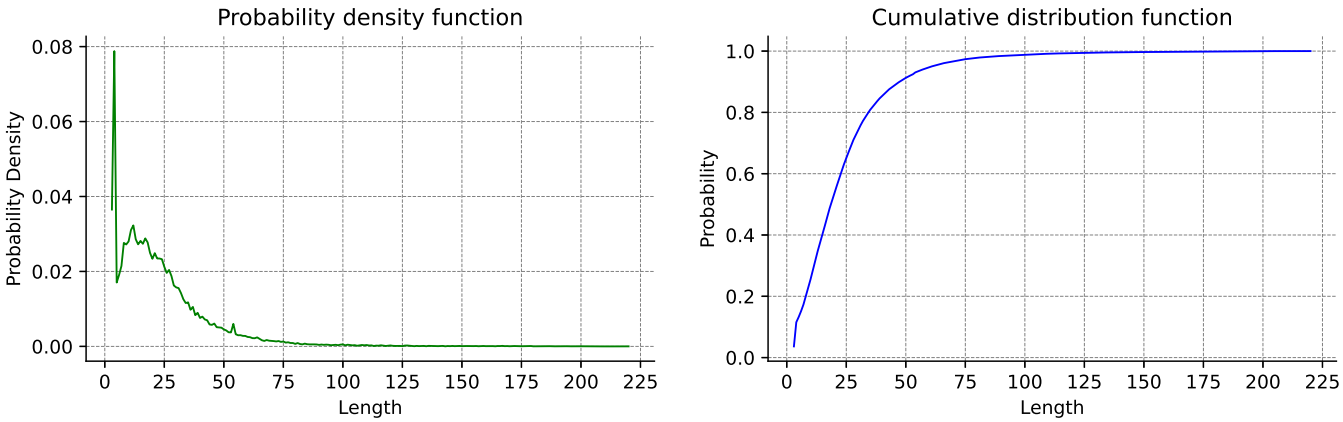


Figure 5.1: Probability density function and Cumulative distribution function over the length of EHR sentences (2018 n2c2 dataset).

As a result, the statistics on the filtered annotated dataset of the 2018 n2c2 task, from now on referred to as the n2c2 dataset, can be seen in Table 5.1. It can be observed that the distribution of named entities within these sets is unbalanced, with an overrepresentation of Drug entities, whereas all other entities are present in roughly equal proportions, with doses and routes being the least represented.

Entities	Train	Test
Dosage	4,227	2,681
Drug	16,257	10,551
Form	6,657	4,359
Route	5,460	3,513
Strength	6,694	4,230
Number of documents	303	202
Number of sentences	54,777	36,446

Table 5.1: Statistics on the filtered annotated dataset from the n2c2 dataset.

5.1.2. UK Biobank dataset

As detailed at the beginning of this chapter, the target task consists of entity recognition in medical prescriptions from anonymized EHR prescribing data from the UK Biobank. Ac-

[20]https://spacy.io/api/doc
[21]https://huggingface.co/docs/transformers/index

cess to the dataset has been made possible through collaboration with the Birney Research Group[22] of the European Bioinformatics Institute (EMBL-EBI).

The dataset consists of unstructured NHS primary care drug prescription data linked to approximately 200,000 participants in the UK Biobank (a large-scale biomedical database and research resource containing comprehensive genetic and health information on half a million UK participants), and it comprises approximately 700,000 prescription text combinations separated into two columns. The first column presents information on the drug, and the second column shows information on the quantity of the drug. However, this initial format of the texts presents ambiguity and redundancy of information; data such as drug strength and dosage may be included simultaneously in either or both columns. For example, in the following prescription, "*Erythromycin 250mg gastroresistant tablets*" and "*56 tablets - 250 mg*", both columns refer to the strength and form of the drug (250 mg and tablets). In addition, the first column refers to the dosage of the drug (56 tablets), while the second column, relating to the amount of the drug, does not.

To obtain the length of the texts based on their tokens, these were tokenized using the vocabulary of the pre-trained Clinical BioBERT model (28,996 tokens) and the tokenizer provided by library transformers[23]. As can be seen from the probability density function, the texts in the dataset vary in length from 10 to 60 tokens. In particular, the length of the prescription texts does not exceed 53 tokens (Figure 5.2).

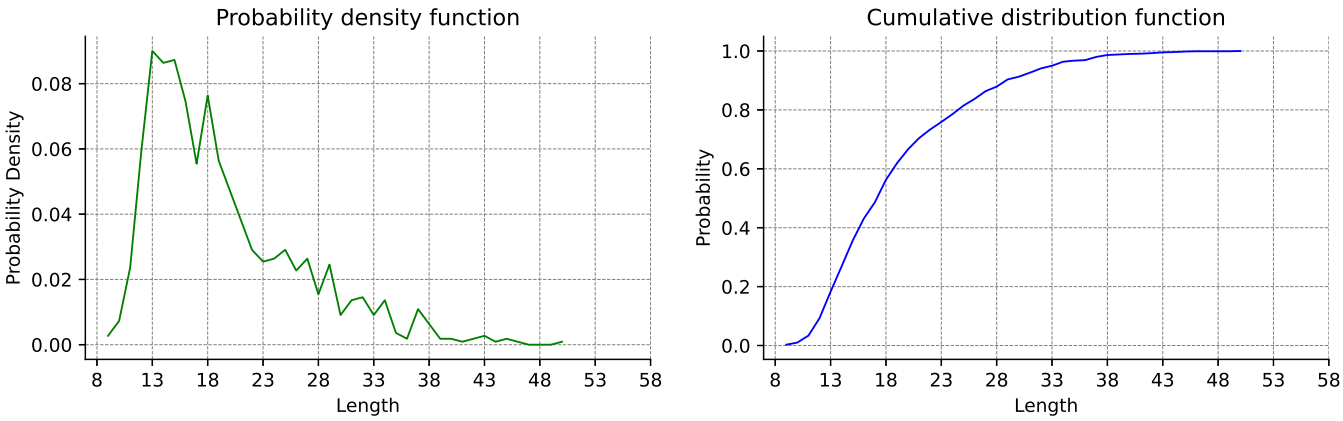


Figure 5.2: Probability density function and Cumulative distribution function over the length of prescriptions (UK Biobank dataset).

Having described the dataset, one of the challenges with this work is that, initially, no previous annotation is available in the UK Biobank texts. Therefore, an annotation process (detailed in the following section 5.2) must be performed to obtain a labeled dataset.

[22]https://www.ebi.ac.uk/research-beta/birney/
[23]https://huggingface.co/docs/transformers/index

5.2. Annotation of the UK Biobank dataset: UKB dataset

To address the lack of prior annotation in the UK Biobank dataset, the annotation process described at the beginning of the chapter is carried out, with the first phase involving automatic annotation of the texts and the second phase involving manual curation of the automatically annotated texts.

As a preliminary step to the automatic annotation, the data is preprocessed by transforming the texts into lowercase and separating the numerical values from the units of measurement to homogenize their format, as well as joining the two columns present in the texts into a single column. For example, preprocessing the medical prescription consisting of "*Erythromycin 250mg gastroresistant tablets*" and "*56 tablets - 250mg*" results in "*erythromycin 250 mg gastroresistant tablets, 56 tablets - 250 mg*".

Once the texts have been preprocessed, it is proposed to use existing dictionaries and predictive models as the first phase of the annotation pipeline. Specifically, Birney Research Group uses a baseline consisting of identifying drugs through updated dictionaries from the ChEMBL database. Moreover, the research group uses two pre-trained models (Med7_lg and Med7_trf) for entity recognition in EHR [51]:

- Med7_lg employs a network trained on the entire MIMIC-III [151] dataset. The model comprises several residual convolutional layers fed with Bloom embeddings [322] (compact embeddings of sparse instances of binary encoding) of the input texts' prefixes, suffixes, and lemmas. The convolutional layers provide contextual information about words, used in a transition-based approach to identify entities.
- Alternatively, the Med7_trf model employs a transformer-based language model (RoBERTa), trained on the entire MIMIC-III dataset, to provide word embeddings with contextual information for the transitions-based approach.

Once the result sets from each of the methods described above are obtained, they are merged into a single result set: $(ChEMBL(x) \cup Med7_lg(x) \cup Med7_trf(x), x \in dataset)$ where x refers to samples from the dataset.

Subsequently, as the second phase of the annotation pipeline aimed at ensuring the quality of the automatically annotated dataset, a manual curation process is conducted by an expert pharmacist with previous experience in annotating pharmacological texts[24]. During this phase, the annotated data is transformed into the BRAT format and subjected to manual curation using the BRAT Quick Annotation Tool[25], where each medical prescription has an *ann* file extension.

In the BRAT format, the *ann* files are composed of three different columns separated by a tab between them. The first column is an index listing the entities contained in the

[24]https://www.ebi.ac.uk/people/person/maria-herrero-zazo/
[25]https://brat.nlplab.org/index.html

Term	Entity	Text
T1	DRUG 0 11	sumatriptan
T2	STRENGTH 12 18	100 mg
T3	FORM 19 26	tablets
T4	DOSAGE 29 30	6
T5	FORM 31 38	tablets
T6	STRENGTH 41 47	100 mg

(a)

Token	Tag
sumatriptan	B-DRUG
100	B-STRENGTH
mg	I-STRENGTH
tablets	B-FORM
,	O
6	B-DOSAGE
tablets	B-FORM
-	O
100	B-STRENGTH
mg	I-STRENGTH

(b)

Figure 5.3: Example of the contents of a file with ann extension and its IOB tagging.
(a) Example of the content of a file with the ann extension. Tabs separate the columns.
(b) Example of IOB tagging for prescription drugs.

text; the second column indicates the entity type and offset spans it is located between; and finally, the third column shows the text contained within the described offset span. For example, the following prescription: "*Sumatriptan 100mg tablets*" and "*6 tablets - 100 mg*" once preprocessed, will be composed of the text "*sumatriptan 100 mg tablets , 6 tablets - 100 mg*" and an ann file (see Figure 5.3a).

Having manually curated the annotated texts, they are formatted following an IOB tagging scheme (I = inside an entity, O = outside an entity, B = beginning of an entity) to encode the boundaries between the different entities composed of one or multiple tokens. For example, in Figure 5.3b, “100 mg” constitutes a Strength entity, with “100” as the beginning token and “mg” within the entity. Subsequently, the “tablets” string stands alone and serves as the start of a Form entity. Meanwhile, the following token, “,”, is labeled as O to signify that it does not belong to any entity, and so forth.

For the training and evaluation of the proposed system, a total of 1,100 sentences were annotated from the original dataset. The resulting set, from now on called UKB dataset, was divided into 80% for the training dataset and the remaining 20% for the test dataset. In Table 5.2, it can be observed that the distribution of the named entities is unbalanced, with an over-representation of Form entities and an under-representation of Route ones. In contrast, the rest of the entities maintain a similar representation. However, the UKB dataset has significantly fewer labeled entities than the n2c2 dataset, so the n2c2 dataset could help mitigate the lack of annotated entities in the UKB dataset.

Entities	Train	Test
Dosage	822	206
Drug	817	206
Form	1,168	302
Route	92	27
Strength	884	215
Number of sentences	880	220

Table 5.2: Statistics on the annotated dataset from the UKB dataset.

5.3. NER model

Once the UKB annotated dataset is obtained, the proposed approach for entity recognition is the BERT-based model called Clinical BioBert [37], which uses 12 coding layers with a hidden layer dimension of 768, resulting in a value of approximately 110M trainable parameters. Specifically, the model was pre-trained by Alsentzer, Emily, et al. by adding biomedical and clinical texts from the MIMIC-III v1.4 database to improve its performance on clinical text tasks. Thus, the Clinical BioBert model can help mitigate the differences between the linguistic features of generic texts and medical prescription texts.

BERT models provide word embeddings with contextual information, allowing the capture of dependencies within tokens. Tag sequences can be obtained by calculating probability distributions by fine-tuning the BERT model with a softmax layer on top of the model. However, using a CRF layer [323], specifically the linear-chain CRF, allows capturing the dependencies between a label y_k and the previous label y_{k-1} in succession, as seen in 5.1. Where $\bar{x}$ is a sequence of words, $\bar{y}$ is its corresponding sequence of labels, and f is a feature function that decomposes into $U(x_k, y_k)$ representing the score assigned to the observation features, and $T(y_k, y_{k+1})$ representing the score assigned to the transition between consecutive labels y_k and y_{k-1}. In the equation, $Z(\bar{x})$ represents the partition function, a normalizing sum of all possible feature scores for all possible label sequences.

$$\begin{aligned} P(\bar{y}|\bar{x}) &= \frac{\prod_{k=1}^{K} f(y_k, y_{k-1}, x_k)}{\sum_{y'} \prod_{k=1}^{K} f(y'_k, y'_{k-1}, x_k)} \\ &= \frac{\exp\left(\sum_{k=1}^{K} U(x_k, y_k) + \sum_{k=1}^{K} T(y_k, y_{k-1})\right)}{Z(\bar{x})} \end{aligned} \tag{5.1}$$

As an example, in Figure 5.3a, it can be seen that "100 mg" is an entity that refers to the strength of the drug. However, "100 mg" could refer to the dosage of the drug. This ambiguity can be resolved by considering the dependence on the previous entity's tag, with the drug strength being more likely to be indicated following the drug.

Considering the above, the overview of the architecture of the Clinical BioBERT

model followed by a CRF optimization layer can be seen in figure 5.4. From now on, this model will be referred to as CBB-CRF.

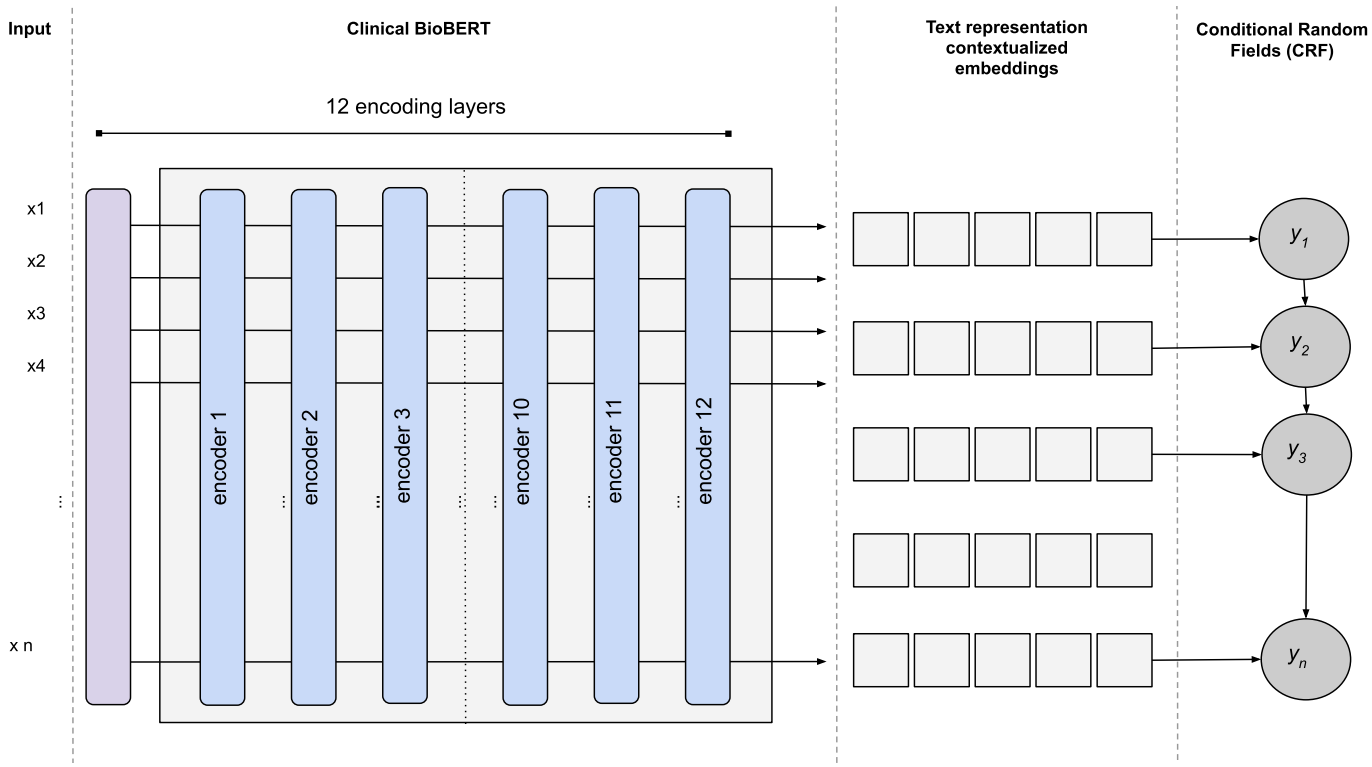


Figure 5.4: Overview architecture of CBB-CRF model

The CBB-CRF model must be able to predict a sequence of labels given a sequence of text in which each token has an associated label. For this purpose, sentences from both datasets are preprocessed using the vocabulary of Clinical BioBERT (28,996 tokens) with the tokenizer provided by the transformer library[26] and its annotations using the IOB tagging scheme (described in section 5.2) to encode the boundaries between different contiguous entities. Furthermore, to avoid truncation of the data and cover all available examples, the size of the texts was adjusted to 225 tokens, considering the maximum size in both datasets.

Once the CBB-CRF model has been described, to assess the benefits of transfer learning between similar task datasets, we compare the performance between 1) the model fine-tuned exclusively on the UKB dataset, 2) the model previously fine-tuned on the n2c2 dataset followed by further fine-tuning on the UKB dataset (see Figure 5.5). Considering the above, describing how the model is trained in both scenarios is necessary.

In the first model, where only the UKB dataset is used, due to the scarcity of annotated data, 5-fold cross-validation was used during training. In the second model, when training on the n2c2 dataset, 20% of the sentences in the training set were used as the validation set. Then, after adjusting the weights on the n2c2 dataset, the model is fine-tuned on the UKB dataset using 5-fold cross-validation. For both training cases, a dropout of 0.3, an

[26]https://huggingface.co/docs/transformers/index

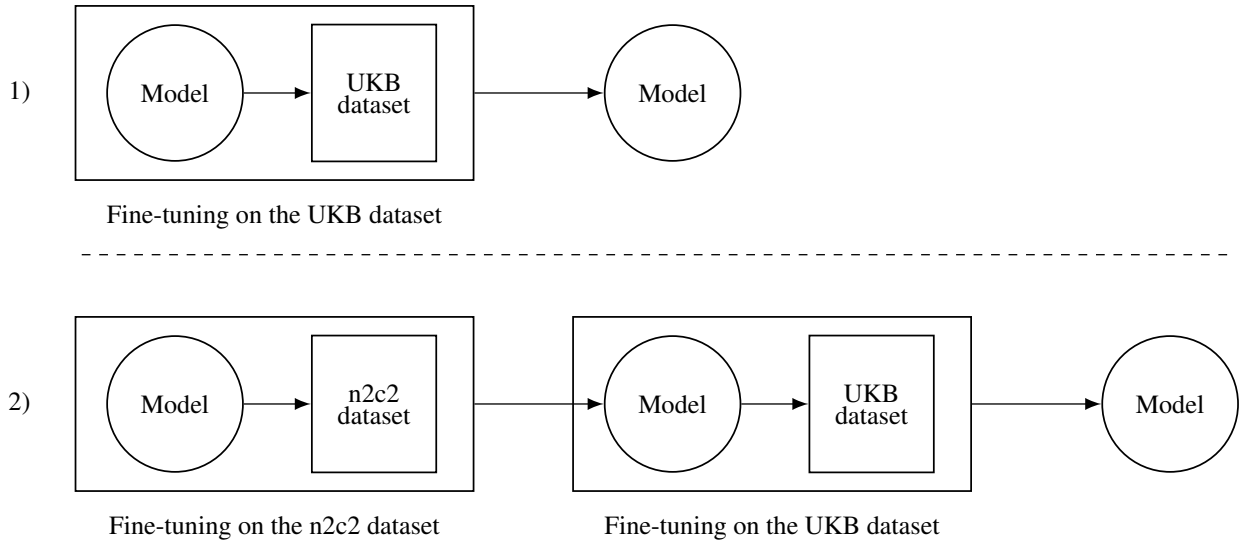


Figure 5.5: Overview of the CBB-CRF model fine-tuning

RMSprop optimizer with a learning rate of 3e-5, and a batch size of 128 were used for five epochs. Due to the model's size, fine-tuning was performed on an 80 GB Nvidia A100 graphics card.

5.4. Evaluation

Precision, recall and F1 metrics have been used following the scheme proposed in SemEval'13 (cf. Eq. 5.2) with strict and lenient matching to evaluate NER models. COR (correct) represents the number of entities with complete match, both in annotation boundaries and entity type; PAR (partial match) represents the number of entities where the predicted annotation boundaries overlap with the real ones without matching exactly; INC (incorrect) corresponds to the number of predicted entities where the boundaries or entity type do not match the gold annotation; MIS (missing) corresponds to the number of gold annotations not captured by the model; SPU (spurious) represents the number of predicted entities that do not exist in the gold-annotated corpus. The strict and lenient metrics are obtained with values for α of 0 and 1, respectively.

$$\begin{aligned}
POSSIBLE(POS) &= COR + PAR + INC + MIS = TP + FN \\
ACTUAL(ACT) &= COR + PAR + INC + SPU = TP + FP \\
Precision &= (COR + \alpha PAR)/ACT \\
Recall &= (COR + \alpha PAR)/POS \\
F1 &= 2 \cdot \frac{precision \cdot recall}{precision + recall}
\end{aligned} \tag{5.2}$$

Thus, in the lenient evaluation scheme, the partial match is counted as an exact match to avoid ambiguity in examples where different annotations correspond to the same con-

cept. E.g., in the text sequence "*duofilm paint (glaxosmithkline uk ltd) , 15 ml + appl*", the sequences "*duofilm*" and "*duofilm paint*" could be labeled as a DRUG entity.

Furthermore, precision, recall and F1 can be extended to multi-classification problems using the micro- and macro-averaged versions. In the macro-average, the mean of the values obtained for each class is shown independently. In the micro-average, the contributions of each class are summed and then the average metric is calculated. Since, in multi-class classification problems, the micro-F1 is calculated globally by counting the total of TP, FN, and FP, this metric will give the following results[27]: $F1 = P = R = Acc$. The use of both micro- and macro-averaged metrics can address complementary information in the evaluation of systems. While the micro-average provides information on overall performance independently of classes, the macro-average will give equal weight to all classes, providing information on the performance of systems on sets with an unbalanced class distribution.

Once the evaluation metrics have been described, the following sections will compare the three proposed approaches: the baseline approach, the CBB-CRF model fine-tuned only on the UKB dataset, and the CBB-CRF model fine-tuned first on the n2c2 dataset and subsequently on UKB (see Figure 5.5).

5.4.1. Baseline results

As described in the previous sections, a combination of the Med7 models and dictionaries extracted from the ChEMBL database is initially used as a baseline.

Table 5.3 shows that the baseline approach cannot identify a high number of dosage-type entities for the strict and lenient evaluations, obtaining a recall of 24% and 24.8%, respectively. Meanwhile, the precision obtained is 66.4% and 68.6%, denoting that the baseline approach misidentifies entities as dosage-type entities. As can be seen from the confusion matrices in Figures 5.6a and 5.6b, more than 150 dosage entities are not recognized due to difficulties differentiating strength-type entities. Due to this, the approach obtains a low F1 of 35.3% and 36.5% for the strict and lenient evaluation, respectively.

Regarding identifying strength-type entities, the approach achieves a recall of 85.4% in the strict evaluation and 99% in the lenient one. However, it penalizes precision when identifying dosage entities as strength-type entities, obtaining 50% and 58.2% precision, respectively, which penalizes F1, receiving 63.1% in the strict evaluation and 73.3% in the lenient one.

Furthermore, the approach achieves a recall of 88% and 99.2% in the strict and lenient evaluations for drug-type entities, respectively. These values may be due to the Med7 models and the coverage of the dictionaries generated from the ChEMBL database. However, the precision is 77.2% and 87.4%, respectively, showing that the baseline approach identifies drug-type entities that do not exist in the test dataset or confuses strength-type

[27] https://scikit-learn.org/stable/modules/model_evaluation.html#multiclass-and-multilabel-classification

entities with drugs due to differences in the distribution of the texts with which the Med7 models were trained (see Figures 5.6a and 5.6b). As a result, despite having high recall values, the precision values penalize F1, with 82.3% and 92.9% in the strict and lenient evaluations, respectively.

	Strict ($\alpha = 0$)			**Lenient** ($\alpha = 1$)			
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Support**
Dosage	0.664	0.240	0.353	0.686	0.248	0.365	206
Drug	0.772	0.880	0.823	0.874	0.992	0.929	206
Form	0.903	0.860	0.881	0.951	0.892	0.921	302
Route	0.703	0.730	0.717	0.703	0.730	0.717	27
Strength	0.500	0.854	0.631	0.582	0.99	0.733	215
micro avg	0.693	0.725	0.709	0.763	0.793	0.778	956
macro avg	0.687	0.718	0.702	0.757	0.787	0.772	956

Table 5.3: The evaluation results of the baseline approach on the UKB test dataset. The baseline approach consists of the union of the ChEMBL dictionaries, Med7_lg,and Med7_trf results ($ChEMBL(x) \cup Med7_lg(x) \cup Med7_trf(x), x \in test$).

While the baseline approach achieves its least favorable result in identifying dosage-type entities (35.3% and 36.5% F1 in the strict and lenient assessments, respectively), its better results are achieved in recognizing entities categorized as form and drug type. In strict evaluation, the baseline approach achieves the highest F1 score for form-type entities (88.1%). Contrarily, in lenient evaluation, the highest F1 score is reached for drug-type entities (92.9%) due to the baseline approach's predictive models identifying more drug-type entities with partial matches.

Consequently, the approach achieves micro- and macro-F1 scores of 70.9% and 70.2% in strict evaluation, while it records 77.8% and 77.2%, respectively, in lenient evaluation.

5.4.2. Results of CBB-CRF fine-tuned only on the UKB dataset

The results from the UKB test dataset show that the CBB-CRF model outperforms the baseline approach (see Table 5.4). The transferability in the clinical domain of models trained on medical center data presents a challenge when applied in similar contexts at other medical centers. In [52], it is discussed in detail how an accurate predictive model fails to generalize to the same level on new data when the data distribution used for training differs. Med7 models have been trained on several sets of EHRs from the MIMIC III database for the NER task. Although the texts share features of clinical texts, they have a different distribution than the prescriptions in the UKB dataset.

Table 5.4 shows the results obtained by the CBB-CRF model on the UKB test dataset. Unlike the baseline approach, this model has been directly fine-tuned on the UKB train-

	Strict (α = 0)			Lenient (α = 1)			
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Support**
Dosage	0.937	0.937	0.937	0.961	0.961	0.961	206
Drug	0.872	0.893	0.882	0.953	0.976	0.964	206
Form	0.873	0.960	0.915	0.908	0.983	0.944	302
Route	0.875	0.778	0.824	0.875	0.778	0.824	27
Strength	0.904	0.916	0.910	0.963	0.977	0.970	215
micro avg	0.893	0.926	0.909	0.940	0.970	0.955	956
macro avg	0.893	0.922	0.902	0.943	0.970	0.950	956

Table 5.4: The evaluation results of the CBB-CRF model on the UKB test dataset. The CBB-CRF model has not been previously fine-tuned on the n2c2 dataset.

ing set, improving the performance in recognition of dosage- and strength-type entities by learning the distribution of the prescriptions and the context in which each of the mentioned entities appears. Similarly, despite the route-type entities being under-represented in the UKB dataset, the CBB-CRF model slightly outperforms the baseline approach in recognizing this entity type.

However, spurious recognition and recall increase is observed for form-type entities (see Figures 5.6c and 5.6d). In this case, both approaches tend to identify words unrelated to the drug form concept. The most frequently identified words are "*pack*", "*vanilla*", "*lasagna*", and "*blister*", among others. Such words may appear in contexts similar to form-type entities. In the example "*erythromycin 250 mg gastro-resistant tablets , 10 tablets - 250 mg*" the word "*tablets*" refers to the concept of drug form. However, in the example "*erythromycin 250 mg gastro-resistant pack, 10 blisters - 250 mg*" the words "*pack*" and "*blisters*" do not refer to the form concept but do appear in similar contexts and distributions.

While the CBB-CRF model, only fine-tuned with the UKB dataset, obtains its least favorable result in the identification of route-type entities (82.4% F1 in both strict and lenient evaluations) due to its under-representation in the UKB dataset, the best results obtained vary depending on the type of evaluation. Specifically, in the strict assessment, the best result is obtained for dose-type entities with 93.7% F1. However, in the lenient evaluation, the best result is obtained for strength-type entities with 97% F1 due to identifications with partial matches.

Consequently, the approach achieves micro and macro-F1 scores of 90.9% and 90.2% in the strict evaluation, while it registers 95.5% and 95% in the lenient evaluation.

5.4.3. Results of CBB-CRF fine-tuned only on the n2c2 dataset

As shown in Tables 5.2 and 5.1, the n2c2 training set has a significantly larger number of annotated entities. A previous fine-tuning of the model on the n2c2 dataset can provide information about the features of different types of entities and knowledge about the context in which they appear, helping to differentiate between entities that appear in similar distributions or contexts.

For this reason, as previously mentioned, we initially fine-tuned the CBB-CRF model using the n2c2 training dataset and subsequently extended its fine-tuning to the UKB dataset. Nevertheless, before fine-tuning the model on the UKB dataset, we evaluated the model's performance, which had only been fine-tuned on the n2c2 training set, using the n2c2 test set. This evaluation allows us to assess the model's performance on the n2c2 dataset and the potential impact of transfer learning when extending the fine-tuning to the UKB dataset. This ensures that it doesn't introduce noise or propagate significant errors from one task to another.

	Strict (α = 0)			**Lenient** (α = 1)			
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Support**
Dosage	0.896	0.914	0.905	0.932	0.950	0.941	2681
Drug	0.915	0.892	0.903	0.964	0.938	0.951	10551
Form	0.937	0.904	0.920	0.975	0.937	0.956	4359
Route	0.946	0.922	0.934	0.964	0.937	0.951	3513
Strength	0.956	0.941	0.948	0.987	0.973	0.980	4230
micro avg	0.928	0.909	0.918	0.966	0.945	0.955	25334
macro avg	0.924	0.899	0.910	0.967	0.935	0.949	25334

Table 5.5: Evaluation results of the CBB-CRF model on the n2c2 test dataset. The CBB-CRF model has only been fine-tuned on the n2c2 dataset.

Table 5.5 shows that the model obtains micro- and macro-F1 results of over 91% in the strict evaluation and about 95% in lenient evaluation on the n2c2 test dataset. At the entity level, the worst result is obtained for drug-type entities (90.3% F1) in the strict evaluation, while in the lenient evaluation, the worst result is obtained for dosage-type entities (94.1% F1). In contrast, the best results are obtained for the strength-type entities, both for the strict (94.8% of F1) and the lenient (98% of F1).

While the model only fine-tuned on the n2c2 dataset achieved results with micro- and macro-F1 scores above 91% on the n2c2 test set, its performance on the UKB test set yielded a micro F1 score of 64.7% for the strict evaluation and 83% for the lenient one. Regarding macro F1, the model scored 61.8% for the strict evaluation and 80.7% for the lenient one (see Table 5.6).

On this point, the model obtains the best F1 scores for the drug and form entities

	Strict (α = 0)			Lenient (α = 1)			
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Support**
Dosage	0.704	0.393	0.505	0.774	0.432	0.555	206
Drug	0.817	0.864	0.840	0.912	0.961	0.936	206
Form	0.696	0.735	0.715	0.887	0.937	0.911	302
Route	0.579	0.407	0.478	0.842	0.593	0.696	27
Strength	0.462	0.535	0.496	0.771	0.893	0.828	215
micro avg	0.660	0.635	0.647	0.847	0.814	0.830	956
macro avg	0.630	0.613	0.618	0.835	0.794	0.807	956

Table 5.6: Evaluation results of the CBB-CRF model on the UKB test dataset. The CBB-CRF model has only been fine-tuned on the n2c2 dataset.

on the UKB test set (93.6% and 91.1% in the lenient evaluation). However, due to the difference between the text distributions in the UKB and n2c2 datasets, the model has problems identifying the context in which dose, route, and strength type entities occur, with the route entity type being the most disadvantaged with 47.8% F1 score in the strict evaluation.

5.4.4. Results of CBB-CRF fine-tuned on the n2c2 and UKB datasets

Once the model has been fine-tuned on the n2c2 dataset, it is subsequently fine-tuned on the UKB training set to provide the model with information about the distribution of texts in UKB dataset. Next, the model is evaluated on the UKB test set, where Table 5.7 shows a generalized increase in F1 concerning previous approaches.

The best results obtained by the CBB-CRF model, fine-tuned on both datasets, are for the form-type entities, with 96.9% F1 for the strict evaluation and 97.9% F1 for the lenient one. While the least favorable result is for identifying route-type entities (87.7% of F1 in both strict and lenient evaluation) due to their under-representation in the UKB dataset, the recognition of this entity type has benefited from the transfer of learning. This is due to an almost 15% recall increase concerning the CBB-CRF model only fine-tuned to the UKB training set. However, there is also an increase in false positives, decreasing precision by 4.2%.

Overall, the CBB-CRF model fine-tuned on both datasets achieves micro and macro-F1 scores of 95.1% and 94.3% in the strict evaluation, while it scores 97.2% and 96.9% in the lenient evaluation.

Further manual evaluation of the CBB-CRF model performance, fine-tuned using both datasets, has been carried out by an expert pharmacist[28] on all available UK Biobank

[28]https://www.ebi.ac.uk/people/person/maria-herrero-zazo/

	Strict (α = 0)			Lenient (α = 1)			
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Support**
Dosage	0.961	0.966	0.964	0.966	0.971	0.969	206
Drug	0.917	0.917	0.917	0.976	0.976	0.976	206
Form	0.967	0.970	0.969	0.977	0.980	0.979	302
Route	0.833	0.926	0.877	0.833	0.926	0.877	27
Strength	0.962	0.949	0.956	0.981	0.967	0.974	215
micro avg	0.950	0.952	0.951	0.971	0.973	0.972	956
macro avg	0.945	0.947	0.943	0.972	0.972	0.969	956

Table 5.7: The evaluation results of the CBB-CRF model on the UKB test dataset. The CBB-CRF model has been previously fine-tuned on the n2c2 dataset and subsequently on the UKB dataset.

prescriptions (approximately 700,000 prescription combinations). This evaluation has revealed that the model can retrieve correct terms from route-type entities such as "*eye*", "*ear*", "*vaginal*" or "*oromucosal*" more frequently than the baseline approach. On the other hand, all approaches identify incorrect route type entities such as "*injection*", "*dispersible*", "*gastro*" and "*conjugate*", being identified more frequently by the baseline approach.

5.4.5. Overview comparison between different approaches

As described above, the baseline approach employs Med7 models for entity recognition. These models, previously trained on EHRs from MIMIC-III, obtained micro-F1 scores of around 95% and macro-F1 scores of about 94% in the lenient evaluation on the n2c2 test set. However, these models have problems generalizing to UK Biobank prescription texts (77.8% micro-F1 and 77.2 macro-F1 in lenient evaluation). Similarly, this problem occurs when evaluating the performance of Med7 on UK-CRIS secondary care mental health EHRs [51]. In particular, the baseline approach has more problems correctly identifying dosage, strength, and route entities. Therefore, as shown in Tables 5.8 and 5.9, the two CBB-CRF approaches significantly outperform the baseline approach on the UKB dataset.

Fine-tuning on a limited annotated UKB dataset provides sufficient information to the CBB-CRF model to significantly improve the problem described with Med7 (delta 20% of micro- and macro-F1). However, employing the n2c2 dataset to learn the features specific to the entities improves the transfer learning mechanism for the NER task on the UKB dataset.

In Table 5.8, the CBB-CRF model pre-finetuned on the n2c2 dataset obtains 95.1% micro-F1 and 94.3% macro-F1 results in the strict evaluation with a delta above 4% com-

	Baseline Strict	**CBB-CRF** Fine-tuned on the UKB dataset	**CBB-CRF** Fine-tuned on the n2c2/UKB datasets	**Support**
Dosage	0.353	0.937	**0.964**	206
Drug	0.823	0.882	**0.917**	206
Form	0.881	0.915	**0.969**	302
Route	0.717	0.824	**0.877**	27
Strength	0.631	0.910	**0.956**	215
micro avg	0.709	0.909	**0.951**	956
macro avg	0.702	0.902	**0.943**	956

Table 5.8: Strict evaluation results of the different approaches on the UKB test dataset (F1-score). The best results are shown in bold.

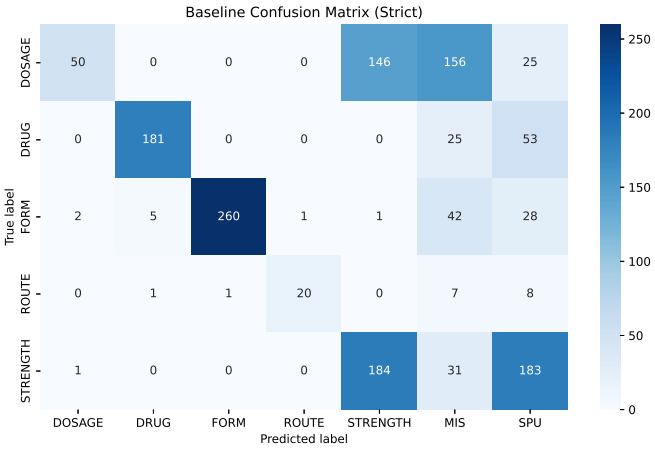
pared to the model without pre-finetuning. However, this difference is reduced with a lenient evaluation. In Table 5.9, the CBB-CRF model scores 97.2% micro-F1 and 96.9% macro-F1 in the lenient evaluation with a delta of almost 2% compared to the CBB-CRF model without pre-finetuning on the n2c2 dataset.

	Baseline Lenient	**CBB-CRF** Fine-tuned on the UKB dataset	**CBB-CRF** Fine-tuned on the n2c2/UKB datasets	**Support**
Dosage	0.365	0.961	**0.969**	206
Drug	0.929	0.964	**0.976**	206
Form	0.921	0.944	**0.979**	302
Route	0.717	0.824	**0.877**	27
Strength	0.733	0.970	**0.974**	215
micro avg	0.778	0.955	**0.972**	956
macro avg	0.772	0.950	**0.969**	956

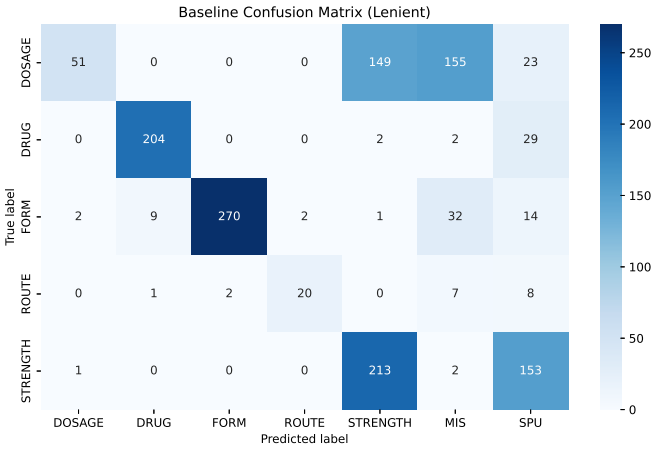
Table 5.9: Lenient evaluation results of the different approaches on the UKB test dataset (F1-score). The best results are shown in bold.

5.5. Conclusions

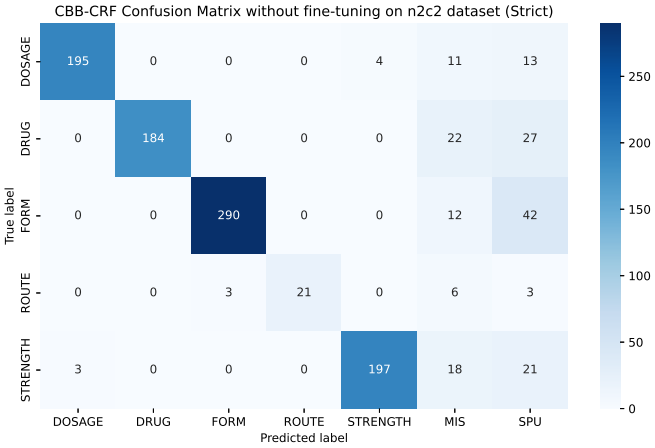
In recent years, the growing availability of anonymized EHRs has increased the scale of pharmacoepidemiological and pharmacogenomic studies when the data are linked to biobank registries. However, some of the information contained in the records is in unstructured form.



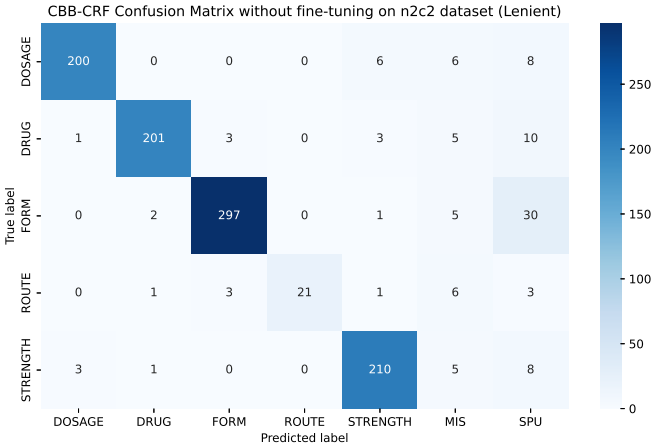
(a)



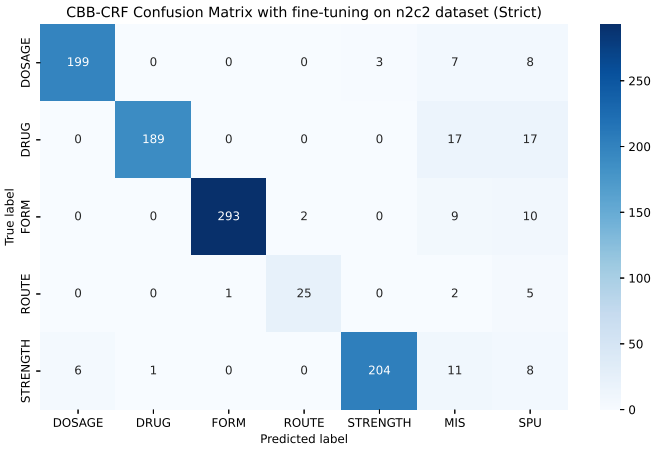
(b)



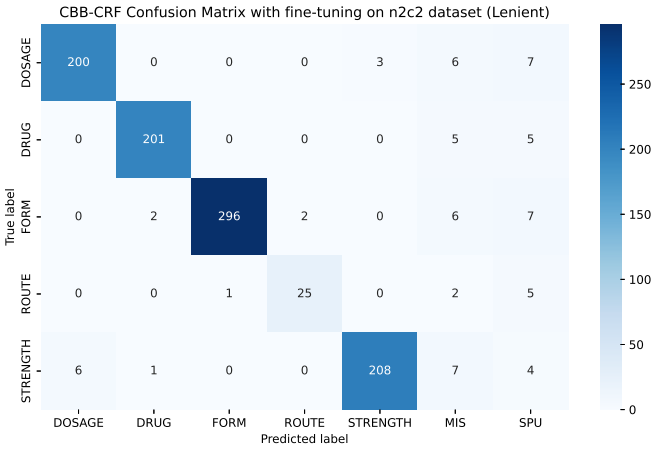
(c)



(d)



(e)



(f)

Figure 5.6: Baseline and CBB-CRF models confusion matrices on the UKB test dataset

In NLP, deep learning techniques have been shown to provide state-of-the-art results in extracting information from unstructured sources [142]. However, large amounts of annotated data are required for these techniques to achieve high performance, and obtaining a sufficient quantity and quality of data in real scenarios is costly and time-consuming.

In this chapter, approaches employed for NER in prescriptions in a few-data scenario have been described to help define phenotypes for GWAS of differences in drug response in depression from anonymized EHR prescription data from UK Biobank[29], specifically in the extraction of drug-type entities, as well as the drug form, strength, dosage, and route of administration to facilitate the mapping of drugs to their ATC code.

The results show that employing a limited set of medical prescriptions with annotated entities to fine-tune pre-trained models on clinical texts provides a better result than the baseline approach, where the pre-trained models were only fine-tuned for the NER task on EHRs from MIMIC-III. Specifically, the baseline approach obtains a micro-F1 of 70.9% and 77.8% in the strict and lenient evaluations, respectively, while the CBB-CRF model obtains a micro-F1 of 90.9% and 95.5%. However, due to the limited amount of annotated data, there is room for improvement in identifying under-represented entity types. Pre-fine-tuning the CBB-CRF model on entity recognition in EHRs benefits transfer learning, outperforming the results of the fine-tuned model only on medical prescriptions. Specifically, the fine-tuned approach on both datasets obtains a micro-F1 of 95.1% and 97.2% in the strict and lenient evaluations, respectively. Therefore, the best approach is the CBB-CRF model approach fine-tuned on both datasets, first on the n2c2 dataset and subsequently on the UKB dataset.

As explored in the previous chapter for SA tasks (Chapter 4), it is planned to use models based on semi-supervised approaches and generative models to extend synthetically annotated corpora for NER tasks. Currently, due to the availability of increasingly complex transformer models pre-trained on large amounts of texts, it is starting to evaluate their performance with few-shot approaches for specific tasks, obtaining results similar, in some cases, to state-of-the-art with full training texts [235]

Furthermore, the PRESNER pipeline, a NER and classification pipeline for EHR prescription data, is planned to be presented [324]. This system is tailored to the needs of researchers using the UK Biobank prescription dataset or other English prescription datasets to provide a system capable of extracting all the information detailed in this chapter. Moreover, PRESNER also maps drugs to the ATC classification system and distinguishes between systemic and non-systemic drug products. PRESNER uses the CBB-CRF model as the main model for the NER task but incorporates the possibility to employ dictionary-based approaches separately or jointly (union of result sets provided by different systems) from the ChEMBL database, as well as Med7 models. The PRESNER pipeline is publicly available to allow its implementation and use[30].

[29] https://www.ukbiobank.ac.uk/
[30] https://github.com/ccolonruiz/PRESNER

Chapter 6

Conclusions

This thesis aims to improve the performance of deep learning models applied to NLP tasks when few annotated data is available. Specifically, the thesis studies different approaches based on deep learning architectures to tackle SA and NER tasks. This chapter presents the conclusions of the work carried out using transfer learning techniques through semi-supervised and data-augmented approaches, as well as fine-tuning on annotated corpora from similar tasks. First, the chapter outlines the conclusions from the thesis concerning the SA task, subsequently addressing the conclusions related to the NER task. Later, the contributions of this thesis are described, followed by the proposition of future work.

6.1. Conclusions on sentiment analysis in drug reviews

The SA task has become one of the most relevant fields of NLP. Millions of user reviews are available on the Internet, and their mass labeling becomes increasingly complicated due to high time and resource consumption. SA in drug user reviews provides a wealth of information on drug efficacy and safety, which can help to detect effects that were not detected in clinical trials. For this reason, positive aspects detected in drug reviews could uncover new drug uses, while negative reviews can help detect unknown side effects.

First, this research compares the performance of various supervised deep learning algorithms applied to drug reviews for the SA task. For this purpose, we employ dataset proposed by [47], which contains labeled drug reviews with an unbalanced class distribution. The experiments reveal that models based on CNN and BiLSTM obtain satisfactory global (macro- and micro-F1 scores) and specific F1 scores, with the underrepresented class (Neutral) achieving the lowest performance. The performance of BiLSTM and CNN-based models is due to their ability to learn contextual and local text features, respectively, with convolutional networks standing out for their computational efficiency. However, it is important to note that models using static word embeddings show room for improvement. State-of-the-art model-based approaches, such as BERT, pre-trained on large corpora to learn prior language knowledge, rely on encoding layers with attention

mechanisms that allow providing contextualized word embeddings. These features result in higher performance compared to previous models. Specifically, a BERT model consisting of only four encoding layers (to reduce the computational requirements), followed by a BiLSTM layer, reveals an increase in micro- and macro-F1 metrics. Furthermore, a more complex BERT model with 12 encoding layers (Clinical BioBERT) achieves even higher performance in global and specific evaluation metrics. These results confirm the effectiveness of BERT-based models in addressing the SA task in the context of drug reviews. However, it is important to note that using these more complex models leads to an increase in computational requirements.

Given the effectiveness of BERT-based models, their performance is evaluated when only a limited number of drug reviews are available. For this purpose, the original training set is divided into smaller subsets to simulate few-annotated data scenarios. The results show that, although BERT is a pre-trained model with prior knowledge of the language, its performance is reduced due to the difficulty of generalizing from the patterns and features observed during fine-tuning when the training set is limited. Therefore, we also evaluate the performance of such pre-trained models with semi-supervised and data-augmented approaches in few-annotated data scenarios.

First, the effect of employing such pre-trained models with a semi-supervised generative adversarial network (SSGAN) approach is proposed. For this purpose, we compare the BERT models in the SSGAN approach fine-tuned with a few labeled data and the rest as unlabeled data, with a baseline of fine-tuning the BERT-based models on only the labeled data. The SSGAN approach employs labeled and unlabeled data to learn to generate synthetic representations similar to those of the original texts provided by BERT. Such synthetic representations try to fool the model's discriminator, forcing BERT to improve on the original text representations, thus aiding the SA task. Since the labeled data have an unbalanced class distribution, the experiment results show that the semi-supervised approach helps in terms of macro-F1 by identifying more examples of the underrepresented class (Neutral). However, regarding micro-F1, the differences concerning the supervised approach are not significant. In conclusion, although the SSGAN approach can help identify underrepresented examples, it is far from mitigating the lack of annotated data. Despite the promising results, more research is needed on generating synthetic text representations in an SSGAN approach.

Next, we evaluate the effect of automatically increasing the number of reviews from the few available annotated data. We conditionally fine-tuned the GPT-J language model pre-trained on the labeled data to generate new synthetic reviews, i.e., in addition to the text of the reviews, we include information from their respective labels. In this way, GPT-J is able to learn the distribution of texts conditional on their class, helping to maintain the correlation between the classes and the generated synthetic reviews. Once the new synthetic data is obtained, the performance of the Clinical BioBERT model, fine-tuned with the original reviews available (baseline approach), is compared with the Clinical BioBERT model, now fine-tuned with the synthetic reviews provided by GPT-J plus the

original reviews in the dataset. Moreover, Clinical BioBERT is also fine-tuned with synthetic revisions provided by the EDA method plus the original revisions for comparison purposes. Experimental results indicate that GPT-J is able to generate synthetic texts that contribute novel information to Clinical BioBERT, thus outperforming the EDA data augmentation technique. GPT-J incorporates implicit knowledge that generates coherent texts with linguistic patterns not present in the original training set. However, it is important to note that the adjustment of GPT-J to an unbalanced dataset negatively affects the generation of synthetic reviews labeled with the minority class (Neutral). Furthermore, the new linguistic patterns and features provided by the GPT-J model will tend to be repeated as more augmented data is generated. The results show that this method outperforms the baseline approach regarding micro- and macro-F1 in few-annotated data scenarios requiring fewer labeled reviews to achieve it. Therefore, this method proves to be effective when the available drug reviews are limited.

6.2. Conclusions on the named entities recognition in medical prescriptions

The second application of NLP addressed in this thesis is NER in the clinical domain. The increasing availability of anonymized EHRs has raised the scale of pharmacoepidemiological and pharmacogenomic studies when data are linked to biobank registries. However, some of the information in the registries is unstructured. This thesis addresses the task of NER in clinical texts in order to provide information that helps identify the active ingredients of different drugs to support research in the clinical domain.

In particular, the use of the pre-trained Clinical BioBERT model extended with a CRF layer for NER on medical prescription data is proposed (CBB-CRF). The results show that for extracting information from medical prescriptions, using a limited set of such prescriptions with annotated entities to fine-tune the Clinical BioBERT model provides better results than using models, such as Med7, fine-tuned in the NER task on multiple EHR datasets of clinical notes and discharge notes. This may be due to the difference in the language features used in prescriptions compared to clinical notes and the heterogeneity among the different datasets. Thus, fine-tuning Clinical BioBERT only to the 2018 n2c2 dataset provides satisfactory performance on the n2c2 test set, while its performance on the UKB prescription test set is low.

Despite the performance of CBB-CRF fine-tuned only to the UKB dataset, there is room for improvement in identifying under-represented entity types, such as route-type entities, due to the limited number of annotations in the prescription texts. Pre-fitting the CBB-CRF model for entity recognition in discharge note EHRs benefits transfer learning during fine-tuning to medical prescriptions. Specifically, prior fine-tuning on the 2018 n2c2 dataset, followed by fine-tuning on the UKB dataset of medical prescriptions, outperforms the results of the model fitted only for prescriptions.

This NER task can also be carried out by applying semi-supervised and data aug-

mentation approaches. However, due to the scope of this thesis, the exploration of the application of such approaches for few-annotated data scenarios is included as a proposal for future work.

6.3. Overall conclusions

To sum up, this thesis has studied and evaluated several approaches for SA in drug reviews and NER in medical prescriptions, both in supervised and weakly supervised scenarios. The results show that pre-trained language models outperform traditional deep learning models such as CNN and RNN. Pre-training language models on texts from similar domains can help improve their performance on specific tasks through transfer learning when limited annotated data is available. However, this depends on the model capabilities of language understanding and the amount of annotated data available. Furthermore, there are limitations to generalizing when the linguistic features between pre-training and task-specific corpora are heterogeneous.

The main hypothesis of this Ph.D. thesis is that using transfer learning mechanisms through semi-supervised and data-augmentation approaches and annotated corpus on similar tasks can improve the performance of deep learning models in scenarios with limited annotated data. The results show that, while there is room for improvement, such approaches outperform the supervised ones when the number of available annotated data is limited for SA and NER tasks.

6.4. Main Contribution

This thesis contributes to exploring different approaches to mitigate the weak performance of deep learning models due to the lack of annotated data for SA and NER tasks. It provides a detailed review of related work in the literature, an in-depth description of the approaches employed, and an exhaustive discussion and comparison of the results obtained. Furthermore, the study of the proposed systems constitutes a baseline for future research work on NLP tasks in the clinical setting and can be implemented as a medical resource.

As a medical resource, this thesis has contributed to developing PRESNER[31], a pipeline designed to assist researchers working with raw prescription data from EHRs. This pipeline includes the CBB-CRF model described in Chapter 5 as a core part of the NER module, plus additional functions that allow potential users to automatically identify systemic drugs and apply filters to prescriptions based on drug name or therapeutic group by their ATC codes.

The main contributions of the Thesis meet the defined specific objectives and are as follows:

[31] https://github.com/ccolonruiz/PRESNER

- A detailed review of deep learning approaches and models relevant to NLP, focusing on systems for SA and NER in full-supervision and few-annotation data scenarios, described in Chapter 3.
- One of the first comprehensive explorations and comparisons of different deep learning algorithms in the SA task on patient drug reviews, along with an analysis of the effect of transfer learning mechanisms on classifiers using pre-trained language models, which is described in Chapter 4 (SO1).
- One of the first studies on the effect of applying GANs in a semi-supervised approach together with BERT models on the SA task for patient drug reviews in few-annotated data scenarios, described in Chapter 4 (SO2).
- The use of pre-trained generative models based on transformers GPT, together with BERT models in the task of SA in patient drug reviews with limited labeled reviews and synthetic reviews as training dataset, which is described in Chapter 4 (SO3).
- The application of the Clinical BioBERT model and a CRF optimization layer (CBB-CRF) for the NER task on drug prescriptions and the performance of transfer learning mechanisms in a few annotated data scenarios, which is described in Chapter 5 (SO4).
- The proposal for an EHR prescription data processing system[32] capable of extracting and mapping information for pharmacogenetic studies, which is described in Chapter 5.

6.5. Publications

Throughout this Thesis, the work carried out has resulted in several publications. The publications respond to the specific objectives defined and are as follows:

6.5.1. Journals

These publications contribute to the research context of the SA task in drug reviews, are linked to the specific objective SO1, and their methods are described in Chapter 4.

- **C. Colón-Ruiz** and I. Segura-Bedmar. Comparing deep learning architectures for sentiment analysis on drug reviews. Journal of Biomedical Informatics, 2020, vol. 110, p. 103539. [Journal Impact Factor: 6.317, Q1]. https://doi.org/10.1016/j.jbi.2020.103539.

[32] https://github.com/ccolonruiz/PRESNER

- **C. Colón-Ruiz**, I. Segura-Bedmar, and P. Martínez (2019). Análisis de Sentimiento en el dominio salud: analizando comentarios sobre fármacos. Procesamiento del Lenguaje Natural, 63, 15-22. ISSN: 1135-5948. 2019, Septiembre. [SCOPUS, Computer Science Applications, 0,235, Q3]. http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/6090

- I. Segura-Bedmar, **C. Colón-Ruiz**, M. Á. Tejedor-Alonso, and M. Moro-Moro (2018). Predicting of anaphylaxis in big data emr by exploring machine learning approaches. Journal of Biomedical Informatics. [Journal Impact Factor: 2.753, Q1]. https://doi.org/10.1016/j.jbi.2018.09.012

6.5.2. Preprints

These preprints contribute to the research context for SA and NER tasks in low annotated data scenarios. Their relation to the specific objectives is mentioned individually.

- **C. Colón-Ruiz**, T. W. Fitzgerald, I. Segura-Bedmar, E. Birney, and M. Herrero-Zazo. Automated Extraction and Classification of Drug Prescriptions in Electronic Health Records: Introducing the PRESNER Pipeline. medRxiv, 2023. [Preprint: https://doi.org/10.1101/2023.10.04.23296481]. Submitted for publication in Bioinformatics [Journal Impact Factor: 5.8, Q1].

  This preprint contributes to the research context of the NER task on medical prescriptions for low annotated data scenarios (SO4). Its methods are described in Chapter 5.

- **C. Colón-Ruiz** and I. Segura-Bedmar. Generative Strategies for Low Annotated Data Scenarios: Sentiment Analysis in Drug Reviews. 2023. Submitted for publication in Journal of Biomedical Informatics [Journal Impact Factor: 4.5, Q2].

  This preprint contributes to the research context of the SA task in drug reviews for low annotated data scenarios (SO2 and SO3). Their methods are described in Chapter 4.

- **C. Colón-Ruiz** and I. Segura-Bedmar. Semi-Supervised Generative Adversarial Network for Sentiment Analysis of drug reviews. TechRxiv, 2021. https://doi.org/10.36227/techrxiv.17075054

  This preprint contributes to the research context of the SA task in drug reviews for low annotated data scenarios (SO2). Their methods are described in Chapter 4.

6.5.3. Conferences

These publications contribute to the research context of the NER task in clinical texts (SO4) and provide the basis for the NER approaches described in Chapter 5.

- **C. Colón-Ruiz** and I. Segura-Bedmar (2019). Protected Health Information Recognition by BiLSTM-CRF. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019). Bilbao, Spain. 2019, Septiembre. CEUR, ISSN 1613-0073. 2421, 679- 686. https://ceur-ws.org/Vol-2421/MEDDOCAN_paper_6.pdf

- **C. Colón-Ruiz** and I. Segura-Bedmar (2019). Hulat-TaskAB at eHealth-KD Challenge 2019: Knowledge Recognition from Health Documents by BiLSTM-CRF. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019). Bilbao, Spain. 2019, Septiembre. CEUR, ISSN 1613-0073. 2421, 35 - 42. http://ceur-ws.org/Vol-2421/eHealth-KD_paper_3.pdf

- **C. Colón-Ruiz**, I. Segura-Bedmar, and P. Martínez (2017). Combining the BANNER tool with the DINTO ontology for the CEMP task of BioCreative V.5., BioCreative, Barcelona, España, April, 2017, Proceedings of the BioCreative V.5 Challenge Evaluation Workshop, pp: 104-107. https://biocreative.bioinformatics.udel.edu/media/store/files/2017/BioCreative_V5_paper14.pdf

- I. Segura-Bedmar, **C. Colón-Ruiz**, and P. Martínez (2017). LABDA at SemEval-2017 Task 10: Extracting Keyphrases from Scientific Publications by combining the BANNER tool and the UMLS Semantic Network, Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval), Vancouver, Canada, August, 2017, Association for Computational Linguistics, pp: 938-941. https://aclanthology.org/S17-2164/

6.6. Future Work

Given the growing interest in reducing dependency on annotated corpora, several future works are proposed. First, large language models for classification tasks can be applied to semi-supervised variational autoencoders (VAE) [211], and their performance against adversarial networks can be evaluated. Moreover, it is planned to investigate the use of these approaches for NER and RE tasks, learning to generate latent representations at the token level and obtaining synthetic sequences associated with their respective label sequences. In addition to the named generative methods, other semi-supervised approaches such as self- and co-training [216], graph-based [209], and perturbation-based [205], among others, are expected to be explored to reduce the dependency on annotated datasets and improve their performance by combining semantic and contextual features provided by large language models.

On the other hand, the emergence of large pre-trained language models such as GPT-3 [223], [301], GPT-4 [325], Llama2 [326], and T5 [302] provide language understanding capable of addressing few-data learning approaches with promising results. Specifically, such pre-trained models allow synthetic examples using zero- or few-shot techniques,

generating examples of a new domain or task with only a few or no training examples [235], [236]. In addition, we plan to explore the use of generative models and extend annotated corpora for IE tasks such as NER and RE synthetically.

While large language models provide state-of-the-art results, one of their inherent problems lies in the computational resources required for training and use. Therefore, the research community has recently increased its attention to knowledge distillation [327]. As the name suggests, knowledge distillation comprises methods that distill knowledge from large models (teacher models) into smaller models (student models) without significant performance loss. Specifically, the model's output and the intermediate layers features can be used to obtain a similarity or divergence error that approximates the student model behavior to the teacher model [328]. One of the most relevant examples is the DistilBERT model [329], which is 40% smaller and retains around 97% of the language understanding capabilities compared to BERT. Considering that future work is mostly focused on large language models, we also plan to investigate the efficiency of distilled language models applied to our work to reduce the computational requirements for future implementations.

Another relevant aspect is that deep learning models are considered “black boxes". Such models are enigmatic due to a limited understanding of their inner workings and interpretability [330]. The inherent opacity of deep learning models represents a major impediment, especially in healthcare, banking, and law, where the ability to justify decisions is of critical importance [331]. Several techniques have been employed to bring interpretability to the behavior of “black box" models, such as using heat maps to visualize neural activations [332], reducing the dimensionality of hidden layer representations [333], and examining attention mechanisms between features [334], among others. However, in recent years, and due to the increasing complexity of deep learning models, there has been a growing interest in research on explainers that are able to give a comprehensible interpretation of the behavior of such models [335]–[337], e.g., approaches based on Local Interpretable Model-agnostic Explanations (LIME) [338] or SHapley Additive exPlanations (SHap) [339]. In addition, research is underway to make language models generate comprehensible explanations of how neurons work in other language models [340]. With this in mind, we plan to research this area to provide interpretability to the different models described in this thesis through explainer-based approaches.

In summary, our future work in NLP will explore new approaches that reduce reliance on annotated corpora and allow pre-trained language models to address tasks with limited training data. The combination of generative models, semi-supervised approaches, and pre-trained transformer models will be an important area of future research, as will their interpretability and computational complexity.

6.7. Funding

This Thesis has been supported by:

- **DeepEMR project** (Extracción de información clínica usando deep learning y técnicas de Big Data), reference TIN2017-87548-C2-1-R, supported by the Ministry of Economy and Competitiveness - Government of Spain
- **ACCESS2MEET project** (Accesibilidad cognitiva y sensorial a los sistemas de videoconferencia), reference PID2020-116527RB-I00, supported by the Ministry of Science and Innovation under the 2020 Call for Knowledge Generation and Research Challenges Projects - Government of Spain
- **NLP4Rare project** (Natural Language Processing for Rare Diseases), reference NPL4RARE-CM-UC3M, supported by the Community of Madrid under the Interdisciplinary Projects Program for Young Researchers
- **EPUC3M17**, supported by the Multiannual Agreement with UC3M in the line of Excellence of University Professors

Bibliography

[1] N. Berente, B. Gu, J. Recker, and R. Santhanam, "Managing artificial intelligence.," *MIS quarterly*, vol. 45, no. 3, 2021.

[2] P. Cunningham, M. Cord, and S. J. Delany, "Supervised learning," *Machine learning techniques for multimedia: case studies on organization and retrieval*, pp. 21–49, 2008.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[5] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021.

[6] H. Li, "Deep learning for natural language processing: Advantages and challenges," *National Science Review*, vol. 5, no. 1, pp. 24–26, 2018.

[7] C. Shen, J. Ding, Z. Wang, D. Cao, X. Ding, and T. Hou, "From machine learning to deep learning: Advances in scoring functions for protein–ligand docking," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 10, no. 1, e1429, 2020.

[8] L. Wissler, M. Almashraee, D. M. Dı´az, and A. Paschke, "The gold standard in corpus annotation.," *IEEE GSC*, vol. 21, 2014.

[9] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National science review*, vol. 5, no. 1, pp. 44–53, 2018.

[10] D. Hao, L. Zhang, J. Sumkin, A. Mohamed, and S. Wu, "Inaccurate labels in weakly-supervised deep learning: Automatic identification and correction and their impact on classification performance," *IEEE journal of biomedical and health informatics*, vol. 24, no. 9, pp. 2701–2710, 2020.

[11] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, "Multiple instance learning: A survey of problem characteristics and applications," *Pattern Recognition*, vol. 77, pp. 329–353, 2018.

[12] E. Dai, K. Shu, Y. Sun, and S. Wang, "Labeled data generation with inexact supervision," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 218–226.

[13] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[14] P. Liu, L. Wang, R. Ranjan, G. He, and L. Zhao, "A survey on active deep learning: From model driven to data driven," *ACM Computing Surveys (CSUR)*, vol. 54, no. 10s, pp. 1–34, 2022.

[15] H. Shi, K. Livescu, and K. Gimpel, "Substructure substitution: Structured data augmentation for nlp," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 3494–3508.

[16] J. Yu, J. Wu, B. Wei, and Y. Liu, "Cvae-attention: Cvae based semi-supervised sentiment classification using attention," in *Proceedings of the 2019 the International Conference on Pattern Recognition and Artificial Intelligence*, 2019, pp. 68–75.

[17] K. Chowdhary and K. Chowdhary, "Natural language processing," *Fundamentals of artificial intelligence*, pp. 603–649, 2020.

[18] M. Birjali, M. Kasri, and A. Beni-Hssane, "A comprehensive survey on sentiment analysis: Approaches, challenges and trends," *Knowledge-Based Systems*, p. 107 134, 2021.

[19] D. Valle-Cruz, V. Fernandez-Cortez, A. López-Chau, and R. Sandoval-Almazán, "Does twitter affect stock market decisions? financial sentiment analysis during pandemics: A comparative study of the h1n1 and the covid-19 periods," *Cognitive computation*, pp. 1–16, 2021.

[20] R. Sandoval-Almazan and D. Valle-Cruz, "Sentiment analysis of facebook users reacting to political campaign posts," *Digital Government: Research and Practice*, vol. 1, no. 2, pp. 1–13, 2020.

[21] U. Naseem, I. Razzak, M. Khushi, P. W. Eklund, and J. Kim, "Covidsenti: A large-scale benchmark twitter data set for covid-19 sentiment analysis," *IEEE Transactions on Computational Social Systems*, 2021.

[22] C. Zuheros, E. Martínez-Cámara, E. Herrera-Viedma, and F. Herrera, "Sentiment analysis based multi-person multi-criteria decision making methodology using natural language processing and deep learning for smarter decision aid. case study of restaurant choice using tripadvisor reviews," *Information Fusion*, vol. 68, pp. 22–36, 2021.

[23] C. Colón-Ruiz and I. Segura-Bedmar, "Comparing deep learning architectures for sentiment analysis on drug reviews," *Journal of Biomedical Informatics*, vol. 110, p. 103 539, 2020.

[24] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50–70, 2022.

[25] A. Brandsen, S. Verberne, K. Lambers, and M. Wansleeben, "Can bert dig it?–named entity recognition for information retrieval in the archaeology domain," *Journal on Computing and Cultural Heritage (JOCCH)*, 2022.

[26] A. Kumar and B. Starly, ""fabner": Information extraction from manufacturing process science domain literature using named entity recognition," *Journal of Intelligent Manufacturing*, vol. 33, no. 8, pp. 2393–2407, 2022.

[27] Z. Li, D. Qu, C. Xie, W. Zhang, and Y. Li, "Language model pre-training method in machine translation based on named entity recognition," *International Journal on Artificial Intelligence Tools*, vol. 29, no. 07n08, p. 2 040 021, 2020.

[28] P. Kouris, G. Alexandridis, and A. Stafylopatis, "Abstractive text summarization: Enhancing sequence-to-sequence models using word sense disambiguation and semantic content generalization," *Computational Linguistics*, vol. 47, no. 4, pp. 813–859, 2021.

[29] R. Socher *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[30] S. Rosenthal, N. Farra, and P. Nakov, "Semeval-2017 task 4: Sentiment analysis in twitter," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 502–518.

[31] E. T. K. Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 2003, pp. 142–147.

[32] J. Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning*. Mit Press, 2008.

[33] A. Wong, J. M. Plasek, S. P. Montecalvo, and L. Zhou, "Natural language processing and its implications for the future of medication safety: A narrative review of recent advances and challenges," *Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy*, vol. 38, no. 8, pp. 822–841, 2018.

[34] N. Moore, D. Berdaï, P. Blin, and C. Droz, "Pharmacovigilance–the next chapter," *Therapies*, vol. 74, no. 6, pp. 557–567, 2019.

[35] P. Pilipiec, M. Liwicki, and A. Bota, "Using machine learning for pharmacovigilance: A systematic review," *Pharmaceutics*, vol. 14, no. 2, p. 266, 2022.

[36] S. Subramanian *et al.*, "A natural language processing system for extracting evidence of drug repurposing from scientific publications," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 13 369–13 381.

[37] E. Alsentzer *et al.*, "Publicly available clinical bert embeddings," *NAACL HLT 2019*, p. 72, 2019.

[38] D. S. Carrell *et al.*, "Challenges in adapting existing clinical natural language processing systems to multiple, diverse health care settings," *Journal of the American Medical Informatics Association*, vol. 24, no. 5, pp. 986–991, 2017.

[39] C. Colón-Ruiz and I. Segura-Bedmar, "Protected health information recognition bybilstm-crf," in *Proceedings of the Iberian Languages Evaluation Forum, IberLEF*, 2019.

[40] S. H. Oh, M. Kang, and Y. Lee, "Protected health information recognition by fine-tuning a pre-training transformer model," *Healthcare Informatics Research*, vol. 28, no. 1, pp. 16–24, 2022.

[41] *Informatics for integrating biology to the bedside, partners healthcare systems.* https://www.i2b2.org.

[42] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall, "2010 i2b2/va challenge on concepts, assertions, and relations in clinical text," *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 552–556, 2011.

[43] *National nlp clinical challenges.* https://n2c2.dbmi.hms.harvard.edu/.

[44] S. Henry, K. Buchan, M. Filannino, A. Stubbs, and O. Uzuner, "2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records," *Journal of the American Medical Informatics Association*, vol. 27, no. 1, pp. 3–12, 2020.

[45] A. Stubbs, M. Filannino, E. Soysal, S. Henry, and Ö. Uzuner, "Cohort selection for clinical trials: N2c2 2018 shared task track 1," *Journal of the American Medical Informatics Association*, vol. 26, no. 11, pp. 1163–1171, 2019.

[46] L. Kelly *et al.*, "Overview of the share/clef ehealth evaluation lab 2014," in *Information Access Evaluation. Multilinguality, Multimodality, and Interaction: 5th International Conference of the CLEF Initiative, CLEF 2014, Sheffield, UK, September 15-18, 2014. Proceedings 5*, Springer, 2014, pp. 172–191.

[47] F. Gräßer, S. Kallumadi, H. Malberg, and S. Zaunseder, "Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning," in *Proceedings of the 2018 International Conference on Digital Health*, 2018, pp. 121–125.

[48] M. Herrero-Zazo, I. Segura-Bedmar, P. Martı´nez, and T. Declerck, "The ddi corpus: An annotated corpus with pharmacological substances and drug–drug interactions," *Journal of biomedical informatics*, vol. 46, no. 5, pp. 914–920, 2013.

[49] C. Martı´nez-deMiguel, I. Segura-Bedmar, E. Chacón-Solano, and S. Guerrero-Aspizua, "The raredis corpus: A corpus annotated with rare diseases, their signs and symptoms," *Journal of Biomedical Informatics*, vol. 125, p. 103 961, 2022.

[50] J. R. A. Solares *et al.*, "Deep learning for electronic health records: A comparative review of multiple deep neural architectures," *Journal of biomedical informatics*, vol. 101, p. 103 337, 2020.

[51] A. Kormilitzin, N. Vaci, Q. Liu, and A. Nevado-Holgado, "Med7: A transferable clinical natural language processing model for electronic health records," *Artificial Intelligence in Medicine*, vol. 118, p. 102 086, 2021.

[52] J. Morrill, A. Kormilitzin, A. Nevado-Holgado, S. Swaminathan, S. Howison, and T. Lyons, "The signature-based model for early detection of sepsis from electronic health records in the intensive care unit," in *2019 Computing in Cardiology (CinC)*, IEEE, 2019, Page–1.

[53] M. Iman, H. R. Arabnia, and K. Rasheed, "A review of deep transfer learning and recent advancements," *Technologies*, vol. 11, no. 2, p. 40, 2023.

[54] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:52967399.

[55] Y. Liu *et al.*, "Roberta: A robustly optimized bert pretraining approach,"

[56] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[57] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[58] B. Zoph *et al.*, "Rethinking pre-training and self-training," *Advances in neural information processing systems*, vol. 33, pp. 3833–3845, 2020.

[59] S. M. Jiménez-Zafra, M. T. Martín-Valdivia, M. D. Molina-González, and L. A. Ureña-López, "How do we talk about doctors and drugs? sentiment analysis in forums expressing opinions for medical domain," *Artificial intelligence in medicine*, vol. 93, pp. 50–57, 2019.

[60] B. Biseda and K. Mo, "Enhancing pharmacovigilance with drug reviews and social media," *ArXiv*, vol. abs/2004.08731, 2020.

[61] B. Lim and S. Zohren, "Time-series forecasting with deep learning: A survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20 200 209, 2021.

[62] A. Gasparetto, M. Marcuzzo, A. Zangari, and A. Albarelli, "A survey on text classification algorithms: From text to predictions," *Information*, vol. 13, no. 2, p. 83, 2022.

[63] M. J. Er, R. Venkatesan, and N. Wang, "An online universal classifier for binary, multi-class and multi-label classification," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2016, pp. 003 701–003 706.

[64] A. Shiju and Z. He, "Classifying drug ratings using user reviews with transformer-based language models," in *2022 IEEE 10th International Conference on Healthcare Informatics (ICHI)*, IEEE, 2022, pp. 163–169.

[65] D. Croce, G. Castellucci, and R. Basili, "Gan-bert: Generative adversarial learning for robust text classification with a bunch of labeled examples," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2114–2119.

[66] H. Xu, B. Liu, L. Shu, and P. S. Yu, "Bert post-training for review reading comprehension and aspect-based sentiment analysis," in *Proceedings of NAACL-HLT 2019*, Association for Computational Linguistics, 2019, pp. 2324–2335.

[67] S. Žitnik, N. Blagus, and M. Bajec, "Target-level sentiment analysis for news articles," *Knowledge-Based Systems*, vol. 249, p. 108 939, 2022.

[68] Y. Fu, N. Lin, X. Yu, and S. Jiang, "Self-training with double selectors for low-resource named entity recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 1265–1275, 2023.

[69] J. Chen, Z. Wang, R. Tian, Z. Yang, and D. Yang, "Local additivity based data augmentation for semi-supervised ner," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 1241–1251.

[70] P. J. Worth, "Word embeddings and semantic spaces in natural language processing," *International Journal of Intelligence Science*, vol. 13, no. 1, pp. 1–21, 2023.

[71] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[72] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[73] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[74] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the association for computational linguistics*, vol. 5, pp. 135–146, 2017.

[75] M. E. Peters *et al.*, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018. DOI: 10.18653/v1/N18-1202. [Online]. Available: https://aclanthology.org/N18-1202.

[76] A. Vaswani *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[77] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[78] C. Sun, L. Huang, and X. Qiu, "Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence," in *Proceedings of NAACL-HLT 2019*, Association for Computational Linguistics, 2019, pp. 380–385. DOI: 10.18653/v1/N19-1035. [Online]. Available: https://www.aclweb.org/anthology/N19-1035.

[79] Y. Goldberg, *Neural network methods for natural language processing*. Springer Nature, 2022.

[80] T. Ali, D. Schramm, M. Sokolova, and D. Inkpen, "Can i hear you? sentiment analysis on medical forums," in *Proceedings of the sixth international joint conference on natural language processing*, 2013, pp. 667–673.

[81] A. Mishra, A. Malviya, and S. Aggarwal, "Towards automatic pharmacovigilance: Analysing patient reviews and sentiment on oncological drugs," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, IEEE, 2015, pp. 1402–1409.

[82] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of big data*, vol. 2, no. 1, pp. 1–21, 2015.

[83] S. Pouyanfar *et al.*, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–36, 2018.

[84] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[85] C. Goller and A. Kuchler, "Learning task-dependent distributed representations by backpropagation through structure," in *Proceedings of International Conference on Neural Networks (ICNN'96)*, IEEE, vol. 1, 1996, pp. 347–352.

[86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[87] X. V. Lin *et al.*, "Few-shot learning with multilingual generative language models," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 9019–9052.

[88] T. Kocmi *et al.*, "Findings of the 2022 conference on machine translation (wmt22)," in *Proceedings of the Seventh Conference on Machine Translation (WMT)*, 2022, pp. 1–45.

[89] E. Erdem *et al.*, "Neural natural language generation: A survey on multilinguality, multimodality, controllability and learning," *Journal of Artificial Intelligence Research*, vol. 73, pp. 1131–1207, 2022.

[90] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[91] L. Nanni, S. Ghidoni, and S. Brahnam, “Handcrafted vs. non-handcrafted features for computer vision classification,” *Pattern recognition*, vol. 71, pp. 158–172, 2017.

[92] Q. Li and X. Wang, “Image classification based on sift and svm,” in *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, IEEE, 2018, pp. 762–765.

[93] N. O’Mahony *et al.*, “Deep learning vs. traditional computer vision,” in *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1*, Springer, 2020, pp. 128–144.

[94] C. Colón-Ruiz, I. Segura-Bedmar, and P. Martı´nez, “Análisis de sentimiento en el dominio salud: Analizando comentarios sobre fármacos,” *Procesamiento del Lenguaje Natural*, vol. 63, pp. 15–22, 2019.

[95] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[96] D. Klein and C. D. Manning, “Accurate unlexicalized parsing,” in *Proceedings of the 41st annual meeting of the association for computational linguistics*, 2003, pp. 423–430.

[97] A. Graves and A. Graves, “Long short-term memory,” *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[98] K. Cho *et al.*, “Learning phrase representations using rnn encoder–decoder for statistical machine translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[99] A. Shewalkar, D. Nyavanandi, and S. A. Ludwig, “Performance evaluation of deep neural networks applied to speech recognition: Rnn, lstm and gru,” *Journal of Artificial Intelligence and Soft Computing Research*, vol. 9, no. 4, pp. 235–245, 2019.

[100] H. Abbasimehr and R. Paki, “Improving time series forecasting using lstm and attention models,” *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–19, 2022.

[101] P. Liu, X. Qiu, and X. Huang, “Recurrent neural network for text classification with multi-task learning,” in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2016, pp. 2873–2879.

[102] L. Liu *et al.*, “Empower sequence labeling with task-aware neural language model,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[103] A. Roy and S. Pan, "Incorporating medical knowledge in bert for clinical relation extraction," in *Proceedings of the 2021 conference on empirical methods in natural language processing*, 2021, pp. 5357–5366.

[104] X. Wu, S. Lv, L. Zang, J. Han, and S. Hu, "Conditional bert contextual augmentation," in *International Conference on Conceptual Structures*, 2018.

[105] Y. Su *et al.*, "Non-autoregressive text generation with pre-trained language models," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, pp. 234–243.

[106] N. Punith and K. Raketla, "Sentiment analysis of drug reviews using transfer learning," in *2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*, IEEE, 2021, pp. 1794–1799.

[107] X. Li, J. Feng, Y. Meng, Q. Han, F. Wu, and J. Li, "A unified mrc framework for named entity recognition," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 5849–5859.

[108] L. Lange, H. Adel, and J. Strotgen, "Nlnde: The neither-language-nor-domain-experts' way of spanish medical document de-identification," in *IberLEF@SEPLN*, 2020.

[109] V. Suárez-Paniagua, R. M. R. Zavala, I. Segura-Bedmar, and P. Martı´nez, "A two-stage deep learning approach for extracting entities and relationships from medical texts," *Journal of biomedical informatics*, vol. 99, p. 103 285, 2019.

[110] P. Ren *et al.*, "A survey of deep active learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.

[111] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine learning*, vol. 109, no. 2, pp. 373–440, 2020.

[112] Y. Chen and J. Z. Wang, "Image categorization by learning and reasoning with regions," *The Journal of Machine Learning Research*, vol. 5, pp. 913–939, 2004.

[113] B. Lutz, N. Pröllochs, and D. Neumann, "Sentence-level sentiment analysis of financial news using distributed text representations and multi-instance learning," in *Hawaii International Conference on System Sciences*, 2018.

[114] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 5, pp. 845–869, 2013.

[115] F. Muhlenbach, S. Lallich, and D. A. Zighed, "Identifying and handling mislabelled instances," *Journal of Intelligent Information Systems*, vol. 22, no. 1, pp. 89–109, 2004.

[116] V. S. Sheng, F. Provost, and P. G. Ipeirotis, "Get another label? improving data quality and data mining using multiple, noisy labelers," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 614–622.

[117] A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I. J. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Neural Information Processing Systems*, 2018.

[118] F. Yang, H. Zhang, and S. Tao, "Simplified multilayer graph convolutional networks with dropout," *Applied Intelligence*, pp. 1–16, 2022.

[119] S. Ding, Z. Zhu, and X. Zhang, "An overview on semi-supervised support vector machine," *Neural Computing and Applications*, vol. 28, pp. 969–978, 2017.

[120] P. Shayegh, Y. Li, J. Zhang, and Q. Zhang, "Semi-supervised text classification with deep convolutional neural network using feature fusion approach," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, pp. 363–366.

[121] S. Y. Feng *et al.*, "A survey of data augmentation approaches for nlp," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 968–988.

[122] J. Wei and K. Zou, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 6382–6388.

[123] R. Liu, G. Xu, C. Jia, W. Ma, L. Wang, and S. Vosoughi, "Data boost: Text data augmentation through reinforcement learning guided conditional generation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 9031–9041.

[124] C. Wang and D. Lillis, "Classification for crisis-related tweets leveraging word embeddings and data augmentation.," in *TREC*, 2019.

[125] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*.

[126] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," in *International Conference on Learning Representations*.

[127] V. Marivate and T. Sefara, "Improving short text classification through global augmentation methods," in *Machine Learning and Knowledge Extraction: 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, August 25–28, 2020, Proceedings 4*, Springer, 2020, pp. 385–399.

[128] N. Ng, K. Yee, A. Baevski, M. Ott, M. Auli, and S. Edunov, "Facebook fair's wmt19 news translation task submission," *WMT 2019*, p. 314, 2019.

[129] M. Bayer, M.-A. Kaufhold, and C. Reuter, "A survey on data augmentation for text classification," *ACM Computing Surveys*, vol. 55, no. 7, pp. 1–39, 2022.

[130] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 115–124. DOI: 10.3115/1219840.1219855. [Online]. Available: https://aclanthology.org/P05-1015.

[131] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang, "Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars)," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 193–202.

[132] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1422–1432.

[133] Y. D. Challenge, "Yelp dataset challenge," 2013.

[134] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[135] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and emotions in language," *Language resources and evaluation*, vol. 39, pp. 165–210, 2005.

[136] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis," *Computational linguistics*, vol. 35, no. 3, pp. 399–433, 2009.

[137] J. Gonzalo, M. Montes-y-Gómez, and P. Rosso, "Iberlef 2021 overview: Natural language processing for iberian languages," in *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021)*, CEUR Workshop, 2021, pp. 1–15.

[138] J. Gonzalo, M. Montes-y-Gómez, and F. Rangel, "Iberlef 2022 overview: Natural language processing for iberian languages," in *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2022), co-located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2022), A Coruña, Spain, September 20, 2022*, 2022.

[139] S. Jiménez-Zafra, F. Rangel, and M. M.-y. Gómez, "Overview of iberlef 2023: Natural language processing challenges for spanish and other iberian languages," in *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2023), co-located with the 39th Conference of the Spanish Society for Natural Language Processing (SEPLN 2023), CEUR-WS. org*, 2023.

[140] A. Mabrouk, R. P. D. Redondo, and M. Kayed, "Deep learning-based sentiment classification: A comparative survey," *IEEE Access*, vol. 8, pp. 85 616–85 638, 2020.

[141] Q. Li *et al.*, "A survey on text classification: From traditional to deep learning," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 2, pp. 1–41, 2022.

[142] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50–70, 2020.

[143] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, 2003, pp. 142–147.

[144] L. A. Ramshaw and M. P. Marcus, "Text chunking using transformation-based learning," in *Natural language processing using very large corpora*, Springer, 1999, pp. 157–176.

[145] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, "Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 546–555.

[146] P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii, "Brat: A web-based tool for nlp-assisted text annotation," in *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 2012, pp. 102–107.

[147] M. R. Cowie *et al.*, "Electronic health records to facilitate clinical research," *Clinical Research in Cardiology*, vol. 106, pp. 1–9, 2017.

[148] M. Tayefi *et al.*, "Challenges and opportunities beyond structured data in analysis of electronic health records," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 13, no. 6, e1549, 2021.

[149] Y. Wu, M. Jiang, J. Xu, D. Zhi, and H. Xu, "Clinical named entity recognition using deep learning models," in *AMIA annual symposium proceedings*, American Medical Informatics Association, vol. 2017, 2017, p. 1812.

[150] J. Xu, X. Xi, J. Chen, V. S. Sheng, J. Ma, and Z. Cui, "A survey of deep learning for electronic health records," *Applied Sciences*, vol. 12, no. 22, p. 11 709, 2022.

[151] A. E. Johnson *et al.*, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[152] A. Piad-Morffis *et al.*, "Overview of the ehealth knowledge discovery challenge at iberlef 2019," in *IberLEF@SEPLN*, 2021.

[153] M. Marimon *et al.*, "Automatic de-identification of medical texts in spanish: The meddocan track, corpus, guidelines, methods and evaluation of results.," in *IberLEF@ SEPLN*, 2019, pp. 618–638.

[154] H. Dalianis and H. Dalianis, "Evaluation metrics and evaluation," *Clinical Text Mining: secondary use of electronic patient records*, pp. 45–53, 2018.

[155] M. Gösgens, A. Zhiyanov, A. Tikhonov, and L. Prokhorenkova, "Good classification measures and how to find them," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 136–17 147, 2021.

[156] V. Van Asch, "Macro-and micro-averaged evaluation measures," *Belgium: clips*, vol. 49, 2013.

[157] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[158] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *Proceedings of the 2011 conference on empirical methods in natural language processing*, 2011, pp. 151–161.

[159] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 1556–1566.

[160] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 2012, pp. 1201–1211.

[161] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 655–665. DOI: 10.3115/v1/P14-1062. [Online]. Available: https://aclanthology.org/P14-1062.

[162] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1746–1751. DOI: 10.3115/v1/D14-1181. [Online]. Available: https://aclanthology.org/D14-1181.

[163] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 3485–3495. [Online]. Available: https://aclanthology.org/C16-1329.

[164] J.-C. Na, W. Y. M. Kyaing, C. S. Khoo, S. Foo, Y.-K. Chang, and Y.-L. Theng, "Sentiment classification of drug reviews using a rule-based linguistic approach," in *International conference on asian digital libraries*, Springer, 2012, pp. 189–198.

[165] A. Esuli and F. Sebastiani, "Sentiwordnet: A publicly available lexical resource for opinion mining.," in *LREC*, Citeseer, vol. 6, 2006, pp. 417–422.

[166] V. Bobicev, M. Sokolova, Y. Jafer, and D. Schramm, "Learning sentiments from tweets with personal health information," in *Canadian conference on artificial intelligence*, Springer, 2012, pp. 37–48.

[167] J. Carrillo-de-Albornoz, J. R. Vidal, and L. Plaza, "Feature engineering for sentiment analysis in e-health forums," *PloS one*, vol. 13, no. 11, e0207996, 2018.

[168] M. E. Basiri, M. Abdar, M. A. Cifci, S. Nemati, and U. R. Acharya, "A novel method for sentiment classification of drug reviews using fusion of deep and machine learning techniques," *Knowledge-Based Systems*, vol. 198, p. 105 949, 2020.

[169] I. Segura-Bedmar, C. Colón-Ruíz, M. Á. Tejedor-Alonso, and M. Moro-Moro, "Predicting of anaphylaxis in big data emr by exploring machine learning approaches," *Journal of biomedical informatics*, vol. 87, pp. 50–59, 2018.

[170] Y. Zhu *et al.*, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 19–27.

[171] S. Nagel, "Cc-news," *URL: http://web. archive. org/save/http://commoncrawl. org/2016/10/newsdatas* 2016.

[172] A. Gokaslan and V. Cohen, *Openwebtext corpus*, 2019.

[173] T. H. Trinh and Q. V. Le, "A simple method for commonsense reasoning," *arXiv preprint arXiv:1806.02847*, 2018.

[174] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda, "English gigaword fifth edition, june," *Linguistic Data Consortium, LDC2011T07*, vol. 12, 2011.

[175] J. Callan, M. Hoy, C. Yoo, and L. Zhao, "The clueweb09 dataset, 2009," *URL http://boston. lti. cs. cmu. edu/Data/clueweb09*, 2009.

[176] C. Crawl, "Common crawl," *URl: http://http://commoncrawl. org*, 2019.

[177] I. Turc, M. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: The impact of student initialization on knowledge distillation," *CoRR*, vol. abs/1908.08962, 2019.

[178] C. Colón-Ruiz, I. Segura-Bedmar, and P. Martınez, "Combining the banner tool with the dinto ontology for the cemp task of biocreative v. 5," *Proceedings of the BioCreative*, vol. 5, 2017.

[179] L.-H. Lee, K.-C. Lee, and Y.-H. Tseng, "The ntnu system at semeval-2017 task 10: Extracting keyphrases and relations from scientific publications using multiple conditional random fields," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 951–955.

[180] I. Segura-Bedmar, C. Colón-Ruiz, and P. Martı´nez, "Labda at semeval-2017 task 10: Extracting keyphrases from scientific publications by combining the banner tool and the umls semantic network," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 947–950.

[181] R. Leaman and G. Gonzalez, "Banner: An executable survey of advances in biomedical named entity recognition," in *Biocomputing 2008*, World Scientific, 2008, pp. 652–663.

[182] A. McCray, "The umls semantic network," in *Proceedings. Symposium on Computer Applications in Medical Care*, 1989, pp. 503–507.

[183] W. Ammar, M. E. Peters, C. Bhagavatula, and R. Power, "The ai2 system at semeval-2017 task 10 (scienceie): Semi-supervised end-to-end entity and relation extraction," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 592–596.

[184] C. Colón-Ruiz and I. Segura-Bedmar, "Hulat-taskab at ehealth-kd challenge 2019: Knowledge recognition from health documents by bilstm-crf.," in *IberLEF@ SEPLN*, 2019, pp. 35–42.

[185] F. Dernoncourt, J. Y. Lee, and P. Szolovits, "Neuroner: An easy-to-use program for named-entity recognition based on neural networks," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2017, pp. 97–102.

[186] W. Ling *et al.*, "Finding function in form: Compositional character models for open vocabulary word representation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1520–1530.

[187] S. Medina Herrera and J. Turmo Borras, "Talp-upc at ehealth-kd challenge 2019: A joint model with contextual embeddings for clinical information extraction," in *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019): co-located with 35th Conference of the Spanish Society for Natural Language Processing (SEPLN 2019): Bilbao, Spain, September 24th, 2019*, CEUR-WS. org, 2019, pp. 78–84.

[188] E. Martı´nez Cámara *et al.*, "Overview of tass 2018: Opinions, health and emotions," 2018.

[189] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *Proceedings of the 27th international conference on computational linguistics*, 2018, pp. 1638–1649.

[190] É. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning word vectors for 157 languages," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[191] F. Soares, M. Villegas, A. Gonzalez-Agirre, M. Krallinger, and J. Armengol-Estapé, "Medical word embeddings for spanish: Development and evaluation," in *Proceedings of the 2nd clinical natural language processing workshop*, 2019, pp. 124–133.

[192] R. Gupta, S. Sahu, C. Espy-Wilson, and S. Narayanan, "Semi-supervised and transfer learning approaches for low resource sentiment classification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, pp. 5109–5113.

[193] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, PMLR, 2014, pp. 1188–1196.

[194] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N project report, Stanford*, vol. 1, no. 12, p. 2009, 2009.

[195] Y. Li and J. Ye, "Learning adversarial networks for semi-supervised text classification via policy gradient," in *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, 2018, pp. 1715–1723.

[196] I. Triguero, S. Garcı´a, and F. Herrera, "Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study," *Knowledge and Information systems*, vol. 42, pp. 245–284, 2015.

[197] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[198] W. Xu, H. Sun, C. Deng, and Y. Tan, "Variational autoencoder for semi-supervised text classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[199] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling, "Semi-supervised learning with deep generative models," *Advances in neural information processing systems*, vol. 27, 2014.

[200] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *International Conference on Learning Representations*.

[201] Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick, "Improved variational autoencoders for text modeling using dilated convolutions," in *International conference on machine learning*, PMLR, 2017, pp. 3881–3890.

[202] I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[203] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, pp. 2234–2242, 2016.

[204] D. Croce, G. Castellucci, and R. Basili, "Kernel-based generative adversarial networks for weakly supervised learning," in *AI* IA 2019–Advances in Artificial Intelligence: XVIIIth International Conference of the Italian Association for Artificial Intelligence, Rende, Italy, November 19–22, 2019, Proceedings 18*, Springer, 2019, pp. 336–347.

[205] A. H. Li and A. Sethy, "Semi-supervised learning for text classification by layer partitioning," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2020, pp. 6164–6168.

[206] M. Sajjadi, M. Javanmardi, and T. Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[207] L. Samuli and A. Timo, "Temporal ensembling for semi-supervised learning," in *International Conference on Learning Representations (ICLR)*, vol. 4, 2017, p. 6.

[208] M. Stephen, X. Caiming, B. James, and R. Socher, "Pointer sentinel mixture models," *Proceedings of ICLR*, 2017.

[209] H. Zhao, J. Xie, and H. Wang, "Graph convolutional network based on multi-head pooling for short text classification," *IEEE Access*, vol. 10, pp. 11 947–11 956, 2022.

[210] M. Chen, Q. Tang, K. Livescu, and K. Gimpel, "Variational sequential labelers for semi-supervised learning," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 215–226.

[211] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013.

[212] L. Chen, W. Ruan, X. Liu, and J. Lu, "Seqvat: Virtual adversarial training for semi-supervised sequence labeling," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8801–8811.

[213] L. Huang and D. Chiang, "Better k-best parsing," in *Proceedings of the Ninth International Workshop on Parsing Technology*, 2005, pp. 53–64.

[214] C. Chelba *et al.*, "One billion word benchmark for measuring progress in statistical language modeling," in *Interspeech*, 2013.

[215] K. Clark, M.-T. Luong, C. D. Manning, and Q. V. Le, "Semi-supervised sequence modeling with cross-view training," in *Conference on Empirical Methods in Natural Language Processing*, 2018.

[216] C. Liang *et al.*, "Bond: Bert-assisted open-domain named entity recognition with distant supervision," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 1054–1064.

[217] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Pseudo-labeling and confirmation bias in deep semi-supervised learning," in *2020 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2020, pp. 1–8.

[218] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 31–36.

[219] A. Conneau *et al.*, "Unsupervised cross-lingual representation learning at scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8440–8451.

[220] V. Kumar, A. Choudhary, and E. Cho, "Data augmentation using pre-trained transformer models," in *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, 2020, pp. 18–26.

[221] M. Lewis *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Annual Meeting of the Association for Computational Linguistics*, 2019.

[222] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," *ArXiv*, vol. abs/1511.06709, 2015.

[223] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[224] B. Wang, *Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX*, https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

[225] B. Wang and A. Komatsuzaki, *GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model*, https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

[226] W. Satriaji and R. Kusumaningrum, "Effect of synthetic minority oversampling technique (smote), feature representation, and classification algorithm on imbalanced sentiment analysis," in *2018 2nd International Conference on Informatics and Computational Sciences (ICICoS)*, IEEE, 2018, pp. 1–5.

[227] J. Chen, Z. Yang, and D. Yang, "Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2147–2157.

[228] X. Dai and H. Adel, "An analysis of simple data augmentation for named entity recognition," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 3861–3867.

[229] I. Beltagy, K. Lo, and A. Cohan, "Scibert: A pretrained language model for scientific text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3615–3620.

[230] B. Ding *et al.*, "Daga: Data augmentation with a generation approach for low-resource tagging tasks," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6045–6057.

[231] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf, "Flair: An easy-to-use framework for state-of-the-art nlp," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics (demonstrations)*, 2019, pp. 54–59.

[232] J. Liu, Y. Chen, and J. Xu, "Low-resource ner by data augmentation with prompting," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 2022, pp. 4252–4258.

[233] R. Zhou *et al.*, "Melm: Data augmentation with masked entity language modeling for low-resource ner," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 2251–2262.

[234] F. Zhuang *et al.*, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[235] K. M. Yoo, D. Park, J. Kang, S.-W. Lee, and W. Park, "Gpt3mix: Leveraging large-scale language models for text augmentation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, pp. 2225–2239.

[236] S. Wang *et al.*, "Gpt-ner: Named entity recognition via large language models," *ArXiv*, vol. abs/2304.10428, 2023.

[237] A. Hussain and E. Cambria, "Semi-supervised learning for big social data analysis," *Neurocomputing*, vol. 275, pp. 1662–1673, 2018.

[238] I. Segura-Bedmar and P. Martínez Fernández, "Pharmacovigilance through the development of text mining and natural language processing techniques," 2015.

[239] S. A. Adams, "Using patient-reported experiences for pharmacovigilance?" In *Context Sensitive Health Informatics: Human and Sociotechnical Approaches*, IOS Press, 2013, pp. 63–68.

[240] M. Elbadawi, S. Gaisford, and A. W. Basit, "Advanced machine-learning techniques in drug discovery," *Drug Discovery Today*, vol. 26, no. 3, pp. 769–777, 2021.

[241] H. Liu, A. Gegov, and F. Stahl, "Categorization and construction of rule based systems," in *International conference on engineering applications of neural networks*, Springer, 2014, pp. 183–194.

[242] P. Lin and X. Luo, "A survey of sentiment analysis based on machine learning," in *CCF International Conference on Natural Language Processing and Chinese Computing*, Springer, 2020, pp. 372–387.

[243] O. Appel, F. Chiclana, J. Carter, and H. Fujita, "Successes and challenges in developing a hybrid approach to sentiment analysis," *Applied Intelligence*, vol. 48, no. 5, pp. 1176–1188, 2018.

[244] J. Chai and A. Li, "Deep learning in natural language processing: A state-of-the-art survey," in *2019 International Conference on Machine Learning and Cybernetics (ICMLC)*, IEEE, 2019, pp. 1–6.

[245] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using lstm for region embeddings," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16, 2016, pp. 526–534.

[246] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, e1253, 2018.

[247] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, 2018, pp. 328–339.

[248] B. N. Patro, V. K. Kurmi, S. Kumar, and V. Namboodiri, "Learning semantic sentence embeddings using sequential pair-wise discriminator," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 2715–2729.

[249] A. Yadav and D. K. Vishwakarma, "Sentiment analysis using deep learning architectures: A review," *Artificial Intelligence Review*, vol. 53, no. 6, pp. 4335–4385, 2020.

[250] C. Sindhu, B. Sasmal, R. Gupta, and J. Prathipa, "Subjectivity detection for sentiment analysis on twitter data," in *Artificial Intelligence Techniques for Advanced Computing Applications*, Springer, 2021, pp. 467–476.

[251] K. Kenyon-Dean *et al.*, "Sentiment analysis: It's complicated!" In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1886–1895.

[252] M. Bouazizi and T. Ohtsuki, "Sentiment analysis: From binary to multi-class classification: A pattern-based approach for multi-class sentiment analysis in twitter," in *2016 IEEE International Conference on Communications (ICC)*, IEEE, 2016, pp. 1–6.

[253] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?" In *China National Conference on Chinese Computational Linguistics*, Springer, 2019, pp. 194–206.

[254] S. Pyysalo, F. Ginter, H. Moen, T. Salakoski, and S. Ananiadou, "Distributional semantics resources for biomedical text processing," *Proceedings of Languages in Biology and Medicine*, Jan. 2013.

[255] 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," *Journal of the American Medical Informatics Association*, vol. 22, no. 3, pp. 671–681, 2015.

[256] Q. Li, S. Shah, X. Liu, and A. Nourbakhsh, "Data sets: Word embeddings learned from tweets and general data," in *Eleventh International AAAI Conference on Web and Social Media*, 2017.

[257] J. Wang, Z. Wang, D. Zhang, and J. Yan, "Combining knowledge with deep convolutional neural networks for short text classification," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 2915–2921. DOI: 10.24963/ijcai.2017/406. [Online]. Available: https://doi.org/10.24963/ijcai.2017/406.

[258] H. Ide and T. Kurita, "Improvement of learning for cnn with relu activation by sparse regularization," in *2017 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2017, pp. 2684–2691.

[259] V. Suárez-Paniagua and I. Segura-Bedmar, "Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction," *BMC bioinformatics*, vol. 19, no. 8, p. 209, 2018.

[260] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[261] J. Bao, L. Zhang, and B. Han, "Collaborative attention network with word and n-gram sequences modeling for sentiment classification," in *International Conference on Artificial Neural Networks*, Springer, 2019, pp. 79–92.

[262] Z. Min, "Drugs reviews sentiment analysis using weakly supervised model," in *2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, IEEE, 2019, pp. 332–336.

[263] I. Segura-Bedmar and P. Raez, "Cohort selection for clinical trials using deep learning models," *Journal of the American Medical Informatics Association*, vol. 26, no. 11, pp. 1181–1188, 2019.

[264] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[265] N. Houlsby *et al.*, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*, PMLR, 2019, pp. 2790–2799.

[266] V. Vapnik and V. Vapnik, *Statistical learning theory*, 1998.

[267] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*, Springer, 1998, pp. 137–142.

[268] H. Li, "Deep learning for natural language processing: Advantages and challenges," *National Science Review*, 2017.

[269] I. Spasic and G. Nenadic, "Clinical text data in machine learning: Systematic review," *JMIR medical informatics*, vol. 8, no. 3, e17984, 2020.

[270] S. Lee and W. Kim, "Sentiment labeling for extending initial labeled data to improve semi-supervised sentiment classification," *Electronic Commerce Research and Applications*, vol. 26, pp. 35–49, 2017.

[271] N. F. F. da Silva, L. F. Coletta, E. R. Hruschka, and E. R. Hruschka Jr, "Using unsupervised information to improve semi-supervised tweet sentiment classification," *Information Sciences*, vol. 355, pp. 348–365, 2016.

[272] F. H. Khan, U. Qamar, and S. Bashir, "A semi-supervised approach to sentiment analysis using revised sentiment strength based on sentiwordnet," *Knowledge and information Systems*, vol. 51, no. 3, pp. 851–872, 2017.

[273] C. Du, H. Sun, J. Wang, Q. Qi, and J. Liao, "Adversarial and domain-aware bert for cross-domain sentiment analysis," in *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, 2020, pp. 4019–4028.

[274] J. Zhou, J. Tian, R. Wang, Y. Wu, W. Xiao, and L. He, "Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 568–579.

[275] J. Liu, S. Zheng, G. Xu, and M. Lin, "Cross-domain sentiment aware word embeddings for review sentiment analysis," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 2, pp. 343–354, 2021.

[276] K. Fan, "Minimax theorems," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 39, no. 1, p. 42, 1953.

[277] A. Conneau and G. Lample, "Cross-lingual language model pretraining," *Advances in Neural Information Processing Systems*, vol. 32, pp. 7059–7069, 2019.

[278] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[279] M. Banko and E. Brill, "Scaling to very very large corpora for natural language disambiguation," in *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, 2001, pp. 26–33.

[280] Y. Wang, G. Huang, S. Song, X. Pan, Y. Xia, and C. Wu, "Regularizing deep networks with semantic data augmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[281] S. Kobayashi, "Contextual augmentation: Data augmentation by words with paradigmatic relations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, pp. 452–457.

[282] N. Malandrakis *et al.*, "Controlled text generation for data augmentation in intelligent artificial agents," *EMNLP-IJCNLP 2019*, p. 90, 2019.

[283] S. Qiu *et al.*, "Easyaug: An automatic textual data augmentation platform for classification tasks," in *Companion Proceedings of the Web Conference 2020*, 2020, pp. 249–252.

[284] A. Ollagnier and H. T. Williams, "Text augmentation techniques for clinical case classification.," in *CLEF (Working Notes)*, 2020.

[285] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.

[286] Y. Li, Q. Pan, S. Wang, T. Yang, and E. Cambria, "A generative model for category text generation," *Information Sciences*, vol. 450, pp. 301–315, 2018.

[287] H. Queiroz Abonizio and S. Barbon Junior, "Pre-trained data augmentation for text classification," in *Brazilian Conference on Intelligent Systems*, Springer, 2020, pp. 551–565.

[288] A. Anaby-Tavor *et al.*, "Do not have enough data? deep learning to the rescue!" In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 7383–7390.

[289] M. Bayer, M.-A. Kaufhold, B. Buchhold, M. Keller, J. Dallmeyer, and C. Reuter, "Data augmentation in natural language processing: A novel text generation approach for long and short text classifiers," *International Journal of Machine Learning and Cybernetics*, pp. 1–16, 2022.

[290] L. Gao *et al.*, "The pile: An 800gb dataset of diverse text for language modeling," 2021.

[291] P. Xu *et al.*, "Megatron-cntrl: Controllable story generation with external knowledge using large-scale language models," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 2831–2845.

[292] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2018.

[293] S. Maruf, F. Saleh, and G. Haffari, "A survey on document-level neural machine translation: Methods and evaluation," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–36, 2021.

[294] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Expert Systems with Applications*, vol. 165, p. 113 679, 2021.

[295] M. Post, "A call for clarity in reporting bleu scores," in *Proceedings of the Third Conference on Machine Translation: Research Papers*, 2018, pp. 186–191.

[296] R. Gontijo-Lopes, S. J. Smullin, and E. D. C. E. Dyer, "Affinity and diversity: Quantifying mechanisms of data augmentation,"

[297] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2978–2988.

[298] M. Bayer, M.-A. Kaufhold, and C. Reuter, "A survey on data augmentation for text classification," *ACM Computing Surveys*,

[299] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[300] S. Longpre, Y. Wang, and C. DuBois, "How effective is task-agnostic data augmentation for pretrained transformers?" In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 4401–4411.

[301] T. Brown *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[302] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer.," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.

[303] M. Lewis *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880.

[304] S. Sekine and C. Nobata, "Definition, dictionaries and tagger for extended named entity hierarchy.," in *LREC*, Lisbon, Portugal, 2004, pp. 1977–1980.

[305] S. Zhang and N. Elhadad, "Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts," *Journal of biomedical informatics*, vol. 46, no. 6, pp. 1088–1098, 2013.

[306] O. Etzioni *et al.*, "Unsupervised named-entity extraction from the web: An experimental study," *Artificial intelligence*, vol. 165, no. 1, pp. 91–134, 2005.

[307] G. Zhou and J. Su, "Named entity recognition using an hmm-based chunk tagger," in *Proceedings of the 40th annual meeting of the association for computational linguistics*, 2002, pp. 473–480.

[308] X. Liu, S. Zhang, F. Wei, and M. Zhou, "Recognizing named entities in tweets," in *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, 2011, pp. 359–367.

[309] M. Cho, J. Ha, C. Park, and S. Park, "Combinatorial feature embedding based on cnn and lstm for biomedical named entity recognition," *Journal of biomedical informatics*, vol. 103, p. 103 381, 2020.

[310] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional lstm-cnns," *Transactions of the association for computational linguistics*, vol. 4, pp. 357–370, 2016.

[311] C. Sun and Z. Yang, "Transfer learning in biomedical named entity recognition: An evaluation of bert in the pharmaconer task," in *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, 2019, pp. 100–104.

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

[313] F. Hassan, M. Jabreel, N. Maaroof, D. Sánchez, J. Domingo-Ferrer, and A. Moreno, "Recrf: Spanish medical document anonymization using automatically-crafted rules and crf.," in *IberLEF@ SEPLN*, 2019, pp. 727–734.

[314] L. Lange, H. Adel, and J. Strötgen, "Nlnde: The neither-language-nor-domain-experts' way of spanish medical document de-identification," 2019.

[315] V. Suárez-Paniagua, "Vsp at meddocan 2019," 2019.

[316] G. McInnes and R. B. Altman, "Drug response pharmacogenetics for 200,000 uk biobank participants," in *BIOCOMPUTING 2021: Proceedings of the Pacific Symposium*, World Scientific, 2020, pp. 184–195.

[317] W. Ning *et al.*, "Feature extraction for phenotyping from semantic and knowledge resources," *Journal of biomedical informatics*, vol. 91, p. 103 122, 2019.

[318] S. Zhou, N. Wang, L. Wang, H. Liu, and R. Zhang, "Cancerbert: A cancer domain-specific language model for extracting breast cancer phenotypes from electronic health records," *Journal of the American Medical Informatics Association*, 2022.

[319] M. Zhao *et al.*, "Development of a phenotype ontology for autism spectrum disorder by natural language processing on electronic health records," *Journal of Neurodevelopmental Disorders*, vol. 14, no. 1, pp. 1–12, 2022.

[320] E. Lynge, J. L. Sandegaard, and M. Rebolj, "The danish national patient register," *Scandinavian journal of public health*, vol. 39, no. 7_suppl, pp. 30–33, 2011.

[321] G. McInnes, A. Lavertu, K. Sangkuhl, T. E. Klein, M. Whirl-Carrillo, and R. B. Altman, "Pharmacogenetics at scale: An analysis of the uk biobank," *Clinical Pharmacology & Therapeutics*, vol. 109, no. 6, pp. 1528–1537, 2021.

[322] J. Serrà and A. Karatzoglou, "Getting deep recommenders fit: Bloom embeddings for sparse binary input/output networks," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 279–287.

[323] C. Sutton, A. McCallum, *et al.*, "An introduction to conditional random fields," *Foundations and Trends® in Machine Learning*, vol. 4, no. 4, pp. 267–373, 2012.

[324] C. Colón-Ruiz, T. W. Fitzgerald, I. Segura-Bedmar, E. Birney, and M. Herrero-Zazo, "Automated extraction and classification of drug prescriptions in electronic health records: Introducing the presner pipeline," *medRxiv*, 2023. DOI: 10.1101/2023.10.04.23296481.

[325] OpenAI, "Gpt-4 technical report," *ArXiv*, vol. abs/2303.08774, 2023.

[326] H. Touvron *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[327] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[328] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[329] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter," in *NeurIPS EMC2̂ Workshop*, 2019.

[330] A. B. Arrieta *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information fusion*, vol. 58, pp. 82–115, 2020.

[331] V. Hassija *et al.*, "Interpreting black-box models: A review on explainable artificial intelligence," *Cognitive Computation*, pp. 1–30, 2023.

[332] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in nlp," in *Proceedings of NAACL-HLT*, 2016, pp. 681–691.

[333] J. Shin, A. Madotto, and P. Fung, "Interpreting word embeddings with eigenvector analysis," in *32nd Conference on Neural Information Processing Systems (NIPS 2018), IRASL workshop*, 2018, pp. 73–81.

[334] S. Serrano and N. A. Smith, "Is attention interpretable?" In *Annual Meeting of the Association for Computational Linguistics*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:182953113.

[335] X. Zhao, W. Huang, X. Huang, V. Robu, and D. Flynn, "Baylime: Bayesian local interpretable model-agnostic explanations," in *Uncertainty in artificial intelligence*, PMLR, 2021, pp. 887–896.

[336] I. Ekanayake, D. Meddage, and U. Rathnayake, "A novel approach to explain the black-box nature of machine learning in compressive strength predictions of concrete using shapley additive explanations (shap)," *Case Studies in Construction Materials*, vol. 16, e01059, 2022.

[337] M. R. Zafar and N. Khan, "Deterministic local interpretable model-agnostic explanations for stable explainability," *Machine Learning and Knowledge Extraction*, vol. 3, no. 3, pp. 525–541, 2021.

[338] M. T. Ribeiro, S. Singh, and C. Guestrin, “" why should i trust you?" explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[339] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” *Advances in neural information processing systems*, vol. 30, 2017.

[340] S. Bills *et al.*, “Language models can explain neurons in language models,” *URL https://openaipublic. blob. core. windows. net/neuron-explainer/paper/index. html.(Date accessed: 14.05. 2023)*, 2023.