ZURICH UNIVERSITY OF APPLIED SCIENCES

MASTER THESIS

Ontology-Aware Biomedical Text Classification

Author:
Jasmin SAXER

Supervisors:
Dr. Ahmad AGHAEBRAHIMIAN
Dr. Manuel GIL

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Biomedical String Analysis
Institute of Computational Life Sciences

February 2, 2023

Abstract

Text Classification has become an exciting field in Machine Learning and Natural Language Processing. In the biomedical field, Text Classification is used to tackle the ever-increasing number of scientific publications.

The biomedical domain is rich in structured knowledge, such as ontologies. Ontologies are collections of semantic knowledge on a specific domain and are proved to be useful in many Natural Language Processing tasks, including Text Classification.

The contribution of this work is to incorporate biomedical ontologies into Text Classification. The TextGCN model from Yao, Mao, and Luo, 2018 and a simple Convolutional Neural Network model (Kim, 2014) were used as the baselines. Ontological information was incorporated into the TextGCN model, by adding word-to-ontology and document-to-ontology connections into the text document graph. For the Convolutional Neural Network model pretrained UMLS embeddings were used as input to incorporate ontology information. Further, a new architecture combining the TextGCN with the Convolutional Neural Network model was proposed. The datasets Ohsumed, CRAFT and MedOBO were utilized to evaluate the new models.

The incorporation of ontology information into the TextGCN model improved the classification of the MedOBO dataset. There was not any improvement for the Ohsumed and CRAFT dataset. Incorporating ontology resources into a CNN model did not improve the classification of any of the datasets. The new proposed architecture also did not improve the text classification for any dataset. The Conclusion is that incorporating biomedical ontologies with Text Classification has the potential to improve the system performance and that more research on this topic is needed.

Zusammenfassung

Die Textklassifizierung hat sich zu einem spannenden Gebiet des maschinellen Lernens und der Verarbeitung natürlicher Sprache entwickelt. Im biomedizinischen Bereich wird die Textklassifizierung eingesetzt, um die ständig wachsende Zahl wissenschaftlicher Veröffentlichungen zu bewältigen. Im medizinischen Bereich werden Ontologien verwendet, um semantisches Wissen über bestimmte Bereiche zu sammeln.

Der Beitrag dieser Arbeit besteht darin, verfügbare biomedizinische Ontologien in Textklassifizierungsmethoden einzubeziehen. Das TextGCN-Modell von Yao, Mao, and Luo, 2018 und ein einfaches CNN-Modell (Kim, 2014) wurden als Grundlage verwendet. Ontologie-Informationen wurden in das TextGCN-Modell integriert, indem Wort-zu-Ontologie- und Dokument-zu-Ontologie-Verbindungen in den Textdokument-Graphen eingefügt wurden. Für das CNN-Modell wurden vortrainierte UMLS-Einbettungen als Input verwendet, um Ontologie-Informationen einzubinden. Außerdem wurde eine neue Modellarchitektur vorgeschlagen, die das TextGCN mit dem CNN-Modell kombiniert. Die Datensätze Ohsumed, CRAFT und MedOBO wurden zur Evaluierung der neuen Methoden herangezogen.

Die Einbindung von Ontologie-Informationen in das TextGCN-Modell verbesserte die Klassifizierung des MedOBO-Datensatzes im Vergleich zum TextGCN-Basismodell. Für die Datensätze Ohsumed und CRAFT ergab sich keine Verbesserung. Die Einbeziehung von Ontologieressourcen in ein CNN-Modell führte bei keinem der Datensätze zu einer Verbesserung der Klassifizierung. Die neu vorgeschlagene TextGCN CNN-Architektur verbesserte die Textklassifikation für keinen Datensatz. Die Schlussfolgerung lautet, dass die Einbeziehung bestehender biomedizinischer Ontologien das Potenzial hat, die Klassifizierung biomedizinischer Texte zu verbessern, und dass weitere Forschung zu diesem Thema sehr relevant ist.

Acknowledgement

First of all, I would like to thank Dr Manuel Gil who introduced me to the project team and supervised me from the beginning of the master's program.

Most importantly, I would like to thank Dr Ahmad Aghaebrahimian for his efforts. The excellent support, the valuable tips, and the diverse discussions have contributed decisively to the success of my project.

I would also like to thank the rest of the Computational Genomics team for their warm welcome and inclusion in the team, and contributing ideas and helpful comments.

Contents

List of Abbreviations

H_0	Null hypothesis
Adam	adaptive moment estimation
ANN	Artificial neural networks
BERT	Bidirectional Encoder Representations from Transformers
CBOW	Continuous Bag-of-Words
CHEBI	Chemical Entities of Biological Interest
CL	Cell Ontology
CNN	Convolutional Neural Network
CRAFT	Colorado richly annotated full text
CUI	Concept Unique Identifiers
CUI	Concept Unique Identifiers
DDA	drug-disease associations
DDI	drug–drug interactions
DL	Deep Learning
EEG	Electroencephalography
ElMO	Embeddings from Language Model
FFN	Feed-forward neural networks
GANs	Generative Adversial Networks
GCN	Graph Convolutional Neural Network
GloVe	Global Vectors
GNN	Graph Neural Network
GO	Gene Ontology
KG	Knowledge graph
LSTM	Long Short-Term Memory
MEDLINE	Medical Literature Analysis and Retrieval System Online
MeSH	Medical Subject Headings
MKE	medical knowledge embedding
MONDO	MONDO Disease Ontology
MOP	Molecular Process Ontology
NCBI	National Center for Biotechnology Information
NCBITaxon	NCBI Taxonomy
NCBO	National Centre for Biomedical Ontology
NCR	Neural Concept Recognizer
NER	Named Entity Recognition
NLP	Natural language processing
NLTK	Natural Language Toolkit
OBO	Open Biological and Biomedical Ontologies
OMIM	Online Mendelian Inheritance in Man
OWL	Web Ontology Language

PMI	Point-wise mutual information
PPI	Protein-protein interactions
PR	Protein Ontology
RDF	Resource Description Framework
RE	Relation extraction
ReLu	rectified linear unit
RNN	Recurrent Neural Network
SO	Sequence Ontology
StonKGs	Sophisticated Transformer trained on biomedical text and Knowledge Graphs
tanh	Tangens hyperbolicus
TC	Text Classification
TextGCN	Text Graph Convolutional Network
TF	Term Frequency
TF-IDF	Term Frequency-Inverse Document Frequency
UBERON	uber-anatomy ontology
UMLS	Unified Medical Language System
VGCN	Vocabulary Graph Convolutional Network
XLNet	Generalized Autoregressive Pretraining for Language Understanding

Chapter 1

Introduction

1.1 Overview

Text Classification (TC) is an integral task in Machine Learning and Natural language processing (NLP). Assigning labels to text by finding patterns for each label is the main goal of TC.

An important step before classification is the representation of text such that it can be used computationally. The better the representation of a text is, the better can the text classifier find common patterns of classes.

In the biomedical field, TC is used to tackle the ever-increasing number of scientific publications. TC can facilitate knowledge extraction, literature-based discovery, and many other NLP tasks.

The biomedical domain uses ontologies to collect semantic knowledge of specific domains. Recent studies have been made on using ontologies for different NLP tasks, including text classification.

The contribution of this work is the incorporation of biomedical ontologies into TC methods. In Chapter 1, the literature of TC, biomedical TC, and ontology-based NLP tasks are reviewed. The research gap is elucidated and the research questions of the thesis are defined. A theoretical background on the topics NLP, TC, Deep Learning (DL), ontology and evaluation is provided in Chapter 2. Materials and methods applied to answer the research questions are provided in Chapter 3. In Chapter 4, the results are presented and discussed. Chapter 5 in the end, answers the research questions, draws the conclusion, and identifies further research opportunities.

1.2 Literature Review

1.2.1 Text classification

Recent work in text classification (TC) develops on improving traditional text classification methods. Vaswani et al., 2017 proposed a new network architecture called Transformers. Transformers use an encoder and decoder and are based only on attention mechanisms. Neli Arabadzhieva - Kalcheva and Ivelin Kovachev, 2022 compared the accuracy of traditional TC methods (Bernoulli Naive Bayes classifier (Murphy, 2012), Gaussian Naive Bayes classifier (Raschka, 2017), Multinomial Naive Bayes classifier (McCallum and Nigam, 1998) and Support Vector Machines (Kalcheva, Karova, and Penev, 2020)) with Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) and Generalized Autoregressive Pretraining for Language Understanding (XLNet) (Yang et al., 2020). They showed that for short TC of movie reviews the XLNet model gave the highest accuracy, followed by different BERT models. They concluded that the context awareness of these models improved the classification.

To benefit from Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) simultaneously, Jang et al., 2020; G. Liu and Guo, 2019 proposed Attention-based bi-LSTM and CNN hybrid models. They were trained on movie reviews and both showed higher accuracy than non-hybrid models.

Yao, Mao, and Luo, 2019 proposed a Text Graph Convolutional Network (TextGCN) where they built a single text graph for a corpus and learned the TextGCN by feeding the graph into a two-layer Graph Convolutional Neural Network (GCN). They compared their model with 13 different state-of-the-art TC and embedding methods. They showed that the TextGCN outperformed multiple state-of-the-art methods on four out of five different benchmark datasets (e.g., newsgroup posts, medical literature). The model did not perform better than CNN and LSTM-based models on the movie review dataset. They stated that this is because it ignores word order which is important for sentiment classification, and there were fewer connections in the graph for this specific dataset.

Building on TextGCN, Lu, P. Du, and Nie, 2020 developed the VGCN-BERT model, which combines BERT with a Vocabulary Graph Convolutional Network (VGCN). To train the model they fed the vocabulary graph embedding and the sequence of word embeddings to BERT transformer. They concluded that VGCN-BERT can take advantage of both mechanisms as it outperformed both BERT and VGCN alone.

Biomedical text classification In the biomedical field, text and document classification is evolving by using new combinations of methods (i.e., hybrid systems) or adding additional information to models.

Balabin et al., 2022 proposed a new Transformer model for better representations of biological knowledge named StonKGs (sophisticated transformer trained on biomedical text and knowledge graphs). As the name indicates the model was trained on biomedical text encoded using BioBERT and knowledge graphs encoded with node2vec (Grover and Leskovec, 2016).

Combining label prediction and label counts in one neural network (Jingcheng Du et al., 2019), concatenating CNN and Recurrent Neural Network (RNN) layers (Ibrahim et al., 2021), combining bag-of-words and word embeddings using weighting methods (Ahmed, Dilmaç, and Alpkocak, 2020), and using images along titles and abstracts (Li et al., 2021) are some new hybrid methods in document classification.

A new model called LitMC-BERT for multi-label classification uses a specific module for each label to learn label-based features (Q. Chen et al., 2022). Furthermore, some studies investigated the impact of using full-text articles with different weights applied to different sections (Oliveira Gonçalves et al., 2021)).

1.2.2 Ontology-based natural language processing

Some work has already been done recently on using ontologies for TC or characterisation. One way to use ontologies is for vocabulary expansion. Slater et al., 2020 improved the characterisation of clinical text through ontology-based vocabulary. They expanded ontology terms with other ontologies and synonyms by matching and mapping the terms. Koutsomitropoulos, Andriopoulos, and Likothanassis, 2020 used an ontology to expand keywords of articles which were then embedded using doc2vec (Le and Mikolov, 2014) for multi-label TC. Denecke, 2022 applied the ontology to semantically enrich the vector of clinical text embedding in German.

The Unified Medical Language System (UMLS, Bodenreider, 2004) is often used for improving NLP tasks. Shanavas et al., 2020 used UMLS to find concepts in medical documents, weighted those concepts, and created a concept graph for each medical text document. The concept graphs were compared using a graph kernel, and the documents were classified using the comparison results.

Other thesauri were used to improve automatic subject indexing. Willis and Losee, 2013 showed that using a weighted random walk for selecting terms in thesaurus based on the text improves subject indexing. The algorithm starts the random walk with a set of terms produced by matching the thesaurus with the text. Sanchez-Pi, Martí, and Bicharra Garcia, 2016 used ontologies for generating similarities based on the jumps needed in the ontology to get to the search term starting from the text. The similarities are then used to classify the text, where the most similar terms are mapped to the text. Koutsomitropoulos and Andriopoulos, 2022 used a pretrained Embeddings from Language Model (ElMo, Peters et al., 2018) plus embedded Medical Subject Headings (MeSH, Medicine (U.S.), 1960) terms for MeSH indexing documents. Similarly, Xu et al., 2018 proposed a new model called biomedical resource LSTM (BR-LSTM), which combines biomedical resources with lexical information and entity position information combined for the task of extracting drug-drug interactions. Also using a combination for enriched embeddings Lou et al., 2020 proposed a new representation model for biological entities. The new model first learns the vector representation from structured axioms of ontologies and unstructured texts independently and then fuses them together by optimizing a specific objective function.

Further work includes methods using ontologies for interaction prediction or concept recognition, which to the best of my knowledge, were not yet applied to the task of TC. Maldonado, Goodwin, et al., 2017 proposed a new method that first generates knowledge graphs (KG) from the medical concepts and relations expressed in Electroencephalography (EEG) reports, and then learns medical knowledge embeddings (MKE) from the associated KG. This new MKE show high accuracy of the medical concepts automatically identified from clinical text and also show promising results of correctness and completeness of relations produced. Arbabi et al., 2019 proposed a new method called Neural Concept Recognizer (NCR) for concept recognition by using the hierarchical structure of a biomedical ontology as implicit prior embedding to better learn the embedding of terms. The final embedding of a medical concept is the combination of the raw concept embedding and the embedding of its parents. The NCR uses a CNN to encode input phrases and ranks the medical concepts based on the similarity in that space.

Further Maldonado, Yetisgen, and Harabagiu, 2019 proposed a new method using Generative Adversarial Networks (GANs) to learn knowledge embeddings from UMLS and applying scoring functions from translational embeddings methods including TransE, TransD and DistMult.

One way to integrate an ontology into TC is to project it into an embedding space. The different methods for embedding graphs can be grouped into three main types: graph-based approaches, syntactic approaches and model-theoretic approaches (Kulmanov, Smaili, et al., 2020). Graph-based embeddings try to preserve the graph structure in the embedding which is usually lost in the two other techniques.

Perozzi, Al-Rfou, and Skiena, 2014 developed one of the first graph embedding methods by random walks called DeepWalk. DeepWalk generates a corpus of sentences by making random walks starting from each node in the graph. It embeds these sentences with Word2Vec (Mikolov, Sutskever, et al., 2013; Mikolov, K. Chen, et al., 2013) to generate the embeddings. Building on this method Node2Vec (Grover and Leskovec, 2016) was developed which generates embeddings using 'biased' random walks where the distance of the walk from the original node can be decided. Translational embedding methods are graph-based embeddings that model relations in knowledge graphs as translation operations. The first translational embedding model is TransE (Bordes et al., 2013). TransE defines the translation operation as the addition of the relation vector to the node vector yielding the corresponding node vector. TransH (Wang et al., 2014) improves TransE by using a translation operation in the relation-specific hyperplane and can thus model one-to-many and many-to-many relations better than TransE.

Syntactic representation of ontologies embeds ontologies by accounting only for the set of axioms, without generating a graph-based intermediate representation (Kulmanov, Smaili, et al., 2020). Smaili, Gao, and Hoehndorf, 2018 developed the Onto2vec algorithm which generates embeddings of ontology classes and instances by taking into account the logical axioms of ontologies. Building on Onto2vec Smaili, Gao, and Hoehndorf, 2019 developed a new algorithm called OPA2vec which uses transfer learning. They applied a pretrained word2vec model trained with biomedical literature texts on ontology. They tuned the model to additionally incorporate the axioms from an ontology into the

embedding. They could show that the new algorithm significantly outperforms existing methods for gene-disease association prediction.

Semantic representations uses the semantics of the underlying logic and gives constraints on how symbols should be interpreted. EL embeddings (Kulmanov, Liu-Wei, et al., 2019) generate embeddings by optimizing a set of loss functions that correspond to different normal forms of axioms in ontologies. Anc2vec (Edera, Milone, and Stegmayer, 2022) uses three features of Gene Ontology (GO) terms to embed them: ontological uniqueness, ancestor hierarchy and membership to sub-ontologies. The neural network can thus embed the terms that they are as unique as their corresponding terms in the GO and that the distance reflects the semantic similarity between the corresponding GO terms.

These embedding techniques have been applied to different NLP tasks. Alshahrani, Thafar, and Essack, 2021 reviewed different KG embedding techniques for protein-protein interactions (PPI) prediction. They showed that Walking RDF/OWL (Resource Description Framework / Web Ontology Language) followed by TransE gave the highest performance.

Graph embedding of Gene Ontology Annotation graphs were compared to information content methods and word embeddings for predicting PPI (Zhong and Rajapakse, 2020; Zhong and Rajapakse, 2020). They concluded that graph embeddings improve learning vector representation for PPI and calculating the similarity of proteins. Jianzong Du et al., 2021 compared different graph embedding methods of a PPI network for network feature extraction. They showed that Node2vec gave better results than DeepWalk. Yue et al., 2020 applied 11 different representative graph embedding methods on biomedical networks for drug-disease associations (DDA) prediction, drug–drug interactions (DDI) prediction, PPI prediction, protein function prediction and medical term semantic type classification. In their work, they trained the graph embeddings with the task-specific networks first and then used them as feature inputs to build a classifier to predict the links or labels.

1.3 Research gap

Because of the exponential growth of biomedical documents being published, it is very difficult for researchers to be aware of relevant work being done. Additionally, labelling biomedical documents manually is a very time-consuming and expensive task requiring experts. The automatic classification of biomedical documents is thus an important task to label documents efficiently and cost-effectively.

To improve biomedical TC many different methods have been used. With the advent of neural networks, they were used to improve TC. Similarly, with the rise of biomedical ontologies, they have been used for various NLP tasks, showing that they are very effective (see section 1.2). Only a few works have been done using existing biomedical ontologies for TC. To my knowledge, there hasn't been any work on using biomedical ontology embeddings to improve biomedical TC or to include biomedical ontologies in graph-based TC methods. Ibrahim et al., 2021 concluded that the most interesting future work would be to investigate whether different combinations of graph-based

embedding approaches also have the potential to significantly increase classification performance. Trying to address this gap, this thesis aims to answer the following research questions:

1.4 Research question

1. Does biomedical ontologies improve biomedical text classification?
    (a) Does integrating ontologies with TextGCN improve biomedical text classification?
    (b) Does integrating pretrained UMLS ontology embeddings with CNN improve biomedical text classification?
2. Does a model architecture that combines CNN and TextGCN improve text classification?
3. Does the new model improve biomedical text classification for assigning ontologies to text?

To answer these questions, the null hypothesis "H_0 : The new model does not improve the baseline." for every one of the research questions is assessed.

Chapter 2

Theoretical Background

This chapter provides background information on the key concepts presented in this thesis.

2.1 Natural Language Processing

Natural Language Processing (NLP) is a subfield of Linguistics and Computer Sciences which develops on making computers able to 'understand' human language. NLP in general and text mining in particular are often used in the biomedical field for processing textual data and extracting insights. This field has become interesting in the biomedical field as scientific publications and clinical documents have increased enormously in recent years (Dash et al., 2019). These different documents come with different challenges. From scientific publications mostly abstracts were used which have a different linguistic composition than the article body. But further research has shown that only with the article body the full potential of biomedical text mining can be reached (Oliveira Gonçalves et al., 2021). Each section in the article body (e.g., introduction, method and material, results, discussion) has its differences. These differences can lead to different performances of NLP tools. Clinical documents have an even larger variability in structure, which changes with the type of document, different hospital or even department it came from (Locke et al., 2021; Kevin Bretonnel Cohen and Demner-Fushman, 2014).

Tokenization is often the first step in NLP. Tokenization is a process which separates texts into a list of individual elements such as paragraphs, sentences, or tokens, depending on the required granularity. Named Entity Recognition (NER) is the task of recognizing named entities, which is everything that can be referred to with a proper name (a drug, disease, protein, genes etc.). This is a very useful task in NLP. Relation extraction (RE) is the extraction of very focused types of relationships. In the biomedical domain, RE is mostly done between two entities, like protein-protein interaction (PPI) (Zhang et al., 2018), gene and disease (Bhasuran and Natarajan, 2018), protein and its subcellular location (Cejuela et al., 2018), protein and its function (Chiang and H.-C. Yu, 2005) etc. As another task in NLP, Text Classification (TC), also called text categorization, is the task of assigning labels or tags to textual units like sentences, queries, paragraphs and documents (Minaee et al., 2022). This task is further described in section 2.2.

The main input to NLP tasks are word embeddings. Word embeddings are numerical vectors which are learned based on the context the words appear in the text (Chiu and Baker, 2020; Khattak et al., 2019).

There are many different users who can benefit from biomedical NLP systems. Some possible users and the tools they can apply are as follows. Database curators who build databases on genes in specific model organisms are called model organism database curators. They apply different NLP tools including document triaging and NER tools to work with scientific publications. Bench scientists can use text mining for information extraction or summarizing publications. Clinicians apply NLP systems for question-answering.

There are many different resources and tools for accessing biomedical knowledge. The National Center for Biotechnology Information (NCBI) is a database which can be used for research in molecular biology information. The MEDLINE (Medical Literature Analysis and Retrieval System Online) database has high-quality medical information, and provides citations with basic information about each article. And, the GENIA project provides many tools for NLP (Kevin Bretonnel Cohen and Demner-Fushman, 2014).

2.2 Text classification

Text classification (TC), the task of labelling text with predefined categories can be divided into the following four sub-tasks: Feature extraction, dimensionality reductions (optional), classifier selection and evaluation. The text data can be web data, emails, chats, social media posts, user reviews, literature and more (Kowsari et al., 2019). Depending on the number of classes, TC tasks falls into three categories; binary for two classes, multi-class for more than two classes, and multi-label for more than two classes when one or more than one lable may be assigned to each text.

Before feature extraction can be applied, the textual data should first be cleaned to remove noises such as web or email addresses or irrelevant characters like emojis. The next step is tokenization. Tokenization is the process which separates the text into individual elements, named tokens. Journal articles and clinical text bring challenges for tokenization, for example, the use of hyphens is different than in normal English language (Kevin Bretonnel Cohen and Demner-Fushman, 2014).

2.2.1 Feature extraction

Feature extraction is the task of transforming raw data into features which can be used mathematically (Kowsari et al., 2019). Feature selection, on the other hand, defines the process of selecting a small subset of features from the original feature set (Y. Liu et al., 2020). Common techniques of feature extraction are Term Frequency-Inverse Document Frequency (TF-IDF), Term Frequency (TF) (Luhn, 1957), Point-wise Mutual Information (PMI) or word embeddings.

Term Frequency - Inverse Document Frequency (TF-IDF) is the product of Term Frequency (TF or bag-of-words) and Inverse Document Frequency (IDF). TF is the frequency of a word in the document, and can be used as the raw count or the log_{10} of the raw count plus one. The IDF (Jones, 1972) is used to give more relevance to words that occur only in a few documents which can be helpful to discriminate those papers from the remaining corpus. IDF is calculated as the fraction N/df_t, in which N is the total number of documents in the corpus and df_t is the document frequency of term t (count of documents with t in the document). IDF is usually also squashed using the log_{10}. Therefore, the TF-IDF weighted value $w_{t,d}$ is calculated as followed:

$$w_{t,d} = tf_{t,d} \times idf_t = log_{10}(count(t,d)+1) \times log_{10}(\frac{N}{df_t}) \tag{2.1}$$

Point-wise Mutual Information (PMI) is a common metric used to measure word association (Yao, Mao, and Luo, 2018). PMI calculates the probability of two words occurring together while also accounting for the frequency of the individual terms. Thus PMI of two words a, b is formally defined as

$$PMI(a,b) = log\frac{p(a,b)}{p(a)p(b)} \tag{2.2}$$

$$p(a,b) = \frac{\#W(a,b)}{\#W} \tag{2.3}$$

$$p(a) = \frac{\#W(a)}{\#W} \tag{2.4}$$

The range of words where the words are defined as occurring together is called a sliding window (W), as it is a fixed number of words which slides over a text. The probability of two words co-occurring is defined by the number of sliding windows in a corpus ($\#W$) that contain both words a and b ($\#W(a,b)$). The probability of one term (a), is the number of sliding windows in a corpus containing the term ($\#W(a)$). These two formulas are described in equation 2.3 and 2.4.

Word embeddings are theoretically any numerical representations of words. The most simple method for embedding words is *one-hot encoding* (Khattak et al., 2019). One-hot encoding is often used as a starting point for more complex embedding methods. In one-hot encoding, the words are represented as vectors with zeros and a single one. The dimension of the vector is the size of the vocabulary. Each dimension of the vector is assigned to a word in the vocabulary. As an example, the word *cat* in the corpus *The cat is in the box.* would be assigned the vector [0 1 0 0 0 0 0]. There are several techniques (e.g., Word2Vec (Mikolov, Sutskever, et al., 2013; Mikolov, K. Chen, et al., 2013)) that project each of these one-hot vectors into low-dimensional dense representations known as word embeddings by learning context-based prediction. These embeddings often have a much lower dimensionality than the vocabulary size and contain information about the latent semantics of the linguistic data (Khattak et al., 2019).

Word2vec is one of the most basic and often used methods of learning word embeddings. Word2vec makes static embeddings, which means that each word in the vocabulary has one fixed embedding irrespective of the context it may appear. The learning of the embeddings starts with random vectors and then shifts them so that the word and its context are more similar to each other. This is done by gradient descent and minimizing a loss function (Jurafsky and Martin, 2020). Word2vec uses a prediction-based method to represent words and was implemented with two different model architectures: the Continuous Bag-of-Words (CBOW) and the Skip-gram Models. The CBOW architecture predicts the current word based on the context that is given and the Skip-gram architecture predicts context words around the current word. Thereby, each word is represented by the context it is in.

Since the invention of Word2vec, more complex methods have been developed. Two other kinds of static embeddings are FastText (Bojanowski et al., 2017) and GloVe (Pennington, Socher, and Manning, 2014). *Global Vectors (GloVe)* learns word embeddings from a term co-occurrence matrix. The *FastText* model extends GloVE and Word2Vec, by handling out-of-vocabulary terms using internal sub-word information. *Embedding from Language Models (ELMo)* contextualizes the word and character embedding using a bi-directional recurrent neural network (Peters et al., 2018). Thus, ELMo represents homonyms with different embeddings. *BERT* is also a contextualized word representation model and uses a multi-layer bi-directional transformer-encoder architecture (Devlin et al., 2019).

2.3 Deep Learning

Deep learning (DL) is a type of Machine Learning using deep Artificial Neural Networks (ANN) (Skansi, 2018). DL is a subfield of Machine Learning, and thus a subfield of statistics and Artificial Intelligence. Deep Learning has been used widely in TC (Minaee et al., 2022). It combines the feature extraction, dimension reduction and classification process into one learning process.

Neuron The building blocks of ANNs are single computational units called neurons (Jurafsky and Martin, 2020). An example of a neural unit is shown in figure 2.1. A neural unit first calculates a weighted sum $\sum$ of the inputs $x_1 ... x_n$ plus a bias b resulting in the intermediate output z. Further, a non-linear function is applied to z which is called the activation function. Three popular non-linear functions used are sigmoid σ, Tangens hyperbolicus (tanh) and rectified linear unit (ReLU). The identity function is used to produce a linear output. The diagrams of these functions are shown in figure 2.2.

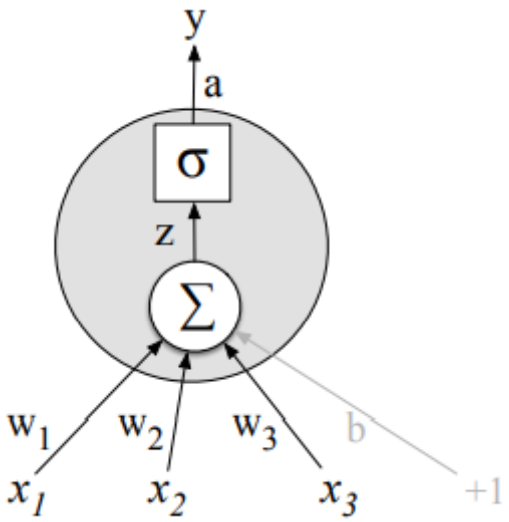

FIGURE 2.1: A example of a neural unit, taking in 3 inputs x_1, x_2, x_3 and a bias b and calculating the weighted sum $\sum$ giving the intermediate output z. Further, the activation function is displayed as the Sigmoid function σ resulting in the activation value a which is in this example the same as the final output y (Jurafsky and Martin, 2020).

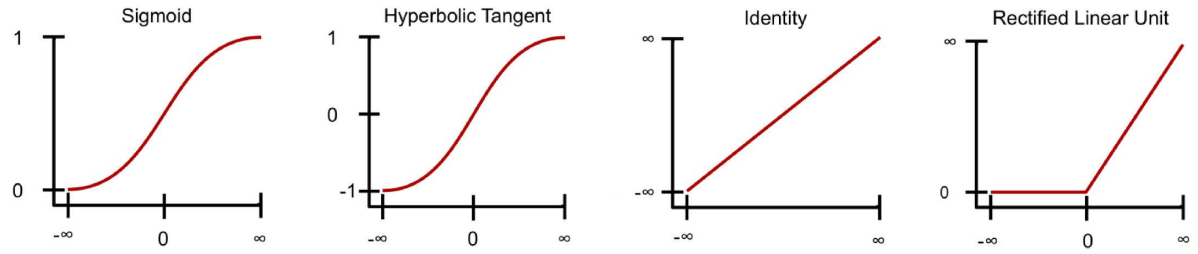

FIGURE 2.2: Diagram of four different activation functions: Sigmoid, hyperbolic tangent (tanh), identity, and rectified linear unit (ReLU) used in neural networks (Choi et al., 2020).

Activation functions for output layer The last layer in a neural network is called the output layer, as it directly outputs the predictions. In classification tasks, the number of output nodes is the number of classes.

For multi-class classification tasks, the Softmax activation function is used (Jurafsky and Martin, 2020). The Softmax function adjusts the output vector so that the sum is 1, thereby allowing this new vector to be interpreted as the probability of class membership. The highest probability can then be used as the final class prediction. Formally, the Softmax function is

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{2.5}$$

For the multi-label classification tasks, the Sigmoid activation function is used (Jurafsky and Martin, 2020). The Sigmoid function maps a real-valued number into the range of [0,1], which can be interpreted as a probability. The classification problem can be looked at as a binary classification independently for each class. Therefore, the Sigmoid function can be applied to each output separately, resulting in a probability score for each class. The final predictions are those classes which beat a defined probability threshold. Formally, the Sigmoid function is

$$Sigmoid(x_i) = \frac{1}{1 + \exp(-x_i)} \tag{2.6}$$

2.3.1 Training neural networks

Training neural networks is the process of fine-tuning the weights and biases of the neurons in the network, to improve the output of the network (Jurafsky and Martin, 2020). First, a loss function is used to find the distance between the target and the prediction of the model. Second, an optimization algorithm is applied to find the values that minimize the loss function. Third, an algorithm called error backpropagation is used to apply the values, which minimizes the loss, over all the nodes in the neural network.

Loss functions are equations which compare the target output with the predicted output of a neural network and result in a value called the loss. While training a neural network the goal is to minimize this loss. A commonly used loss function for classification tasks is *cross-entropy*. Cross-entropy calculates the loss between two probability vectors. The larger the distance between the target and the prediction is, the larger is also the loss. Formally the cross-entropy is calculated as

$$H(p, q) = -\sum_{x \in X} p(x) \log q(x) \tag{2.7}$$

where:

H = cross-entropy loss
$p(x)$ = true probability distribution (target)
$q(x)$ = model's predicted probability distribution
X = classes

Optimization algorithms are used to find the weights of the neural network which minimizes the loss function and are applied iteratively after each training step. The learning rate η is a parameter which defines the weight of the move in the direction of the minimum of the loss function. Using a higher learning rate means that the move in each step is larger.

Stochastic gradient descent algorithm is a basic optimization method which minimizes the loss function by calculating the gradient after each training step and changes the weights in the opposite direction of the gradient using a fixed learning rate.

Adam (Kingma and Ba, 2017), which is derived from adaptive moment estimation, is an adaptive learning rate optimization algorithm. Adam uses squared gradients to scale learning rates, thus computing individual learning rates for different parameters. Further, it applies moving averages of the gradient instead of the gradient itself.

Regularization techniques are applied to avoid model over-fitting. Over-fitting refers to a model's poor generalization performance due to excessive complexity, resulting in training data being memorized rather than learning the underlying pattern.

Dropout is one of the most important regularization methods. During the training of the network, some random units and their connections are dropped. This makes the final model less dependent on individual neurons and thus can reduce over-fitting.

Early stopping is a method where the training of the model is stopped depending on the validation error. A sign of over-fitting is if the training loss is decreasing but the validation loss is increasing. One method for early stopping is to check if the validation loss is higher than the mean of the validation loss of the past few epochs known as patience hyper-parameter. If it is higher the training is stopped.

L2 regularization is a technique that adds a regularization term to the loss function (Hoerl and Kennard, 1970). The regularization term is the sum of all squared weights of the neural network, which is then weighted by a scalar alpha. The scalar alpha is an additional hyperparameter, which determines how much the regularization is applied. By adding this regularization term to the loss function, an additional subtraction is introduced. Thus, the weights are made a little smaller in each learning step and guide them towards zero. Smaller weights reduce the impact of those neurons, thus also reducing the complexity of the neural network. Having a less complex model can help with over-fitting, as it can store less information on the training data.

2.3.2 Feed Forward Neural Network

Feed-Forward Neural networks (FFN) are the simplest kind of ANN, where the output of each unit is passed to the next higher layer, feeding it forward in the network. The architecture of a standard deep FFN is shown in figure 2.3. An FFN has three kinds of nodes: input units, hidden units and output units. The number of features used defines the number of input nodes. For the input nodes, no activation function is used, thus the input is forwarded to the first hidden layer. The core of the network are the hidden layers built from the hidden units. Hidden units are built of neurons with often the activation function ReLu. The number of units in the output layer and the activation function applied depends on the task. The units are all fully connected, thus each unit in each layer takes as input the outputs from all the units of the previous layer (Jurafsky and Martin, 2020).

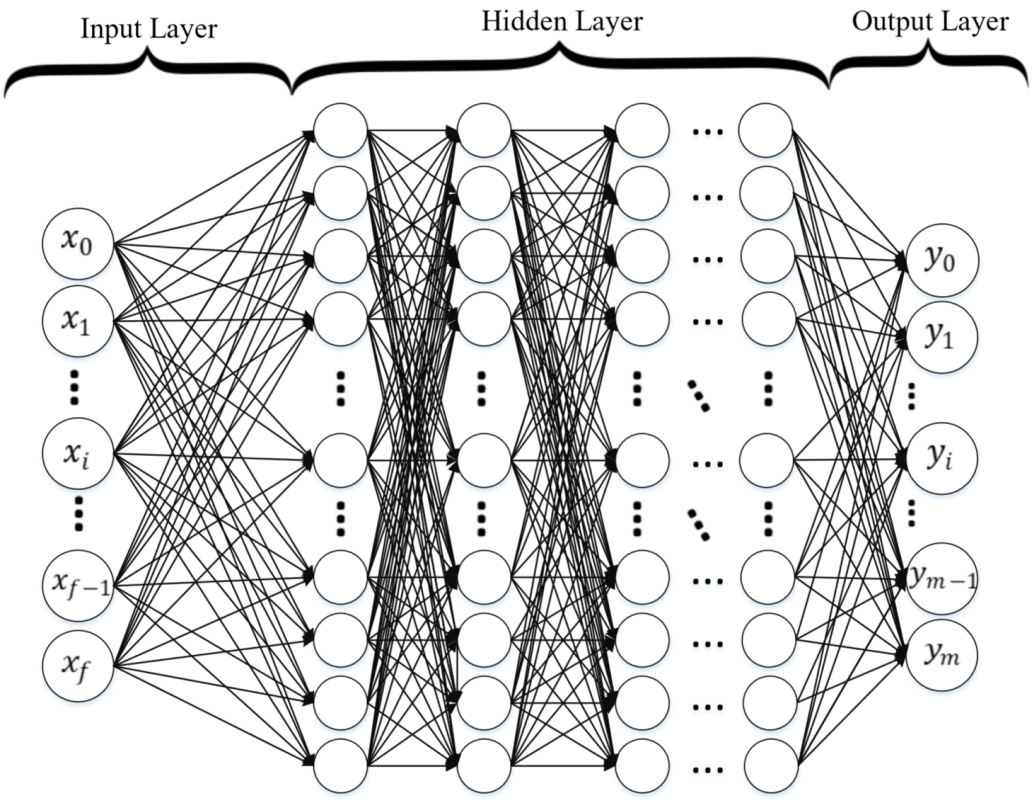

FIGURE 2.3: Standard deep feed-forward neural network (Kowsari et al., 2019).

FFN networks are one of the simplest methods for text representation and achieve high accuracy on different TC tasks (Minaee et al., 2022). When FFN networks are applied for a TC task, textual features are extracted in the first step (Section 2.2.1). If word embeddings are chosen to represent a text, the sum or average over all the word embeddings in the text is taken. The features are then used as input to the first layer of the FFN. The output of the final layer is then used as input to a classifier like logistic regression, Naïve Bayes or support vector machines (SVM). Two models which use this technique are the Deep Average Network (DAN) (Iyyer et al., 2015) and the doc2vec model (Le and Mikolov, 2014).

2.3.3 Convolutional Neural Network

Convolutional neural networks (CNN) are neural networks with convolutional layers, pooling layers and fully connected layers (Kowsari et al., 2019). The architecture of a CNN with text data as input is shown in figure 2.4. The core of CNNs are the convolutional layers. Convolutional layers, perform convolution on the input data. Conceptually, this means that a set of learnable parameters known as the kernel or filter slides over the input data and performs matrix multiplication. The resulting matrices are called feature maps. While training a CNN model, the kernel learns the representation of the input data.

Pooling layers are applied to the feature maps after a convolutional layer. They are used to reduce the computational complexity of the CNN, while still preserving important information. They also compose fixed-size vectors which is required for fully-connected neurons or classification layer in the next step. Max pooling is the most often used pooling algorithm, in which a filter slides over the feature map and the maximum value in each pooling window is selected.

Finally, one or multiple fully connected layers are added to the architecture. These layers are used to flatten the output of the pooling layer to the number of classes for the classification task.

CNN networks initially were developed for image classification, as they can learn parse interaction, parameter sharing, and equivariant representation. Those properties are also very helpful for textual data, as they can recognize patterns in text, such as key phrases. Therefore CNNs are effectively used for TC as well.

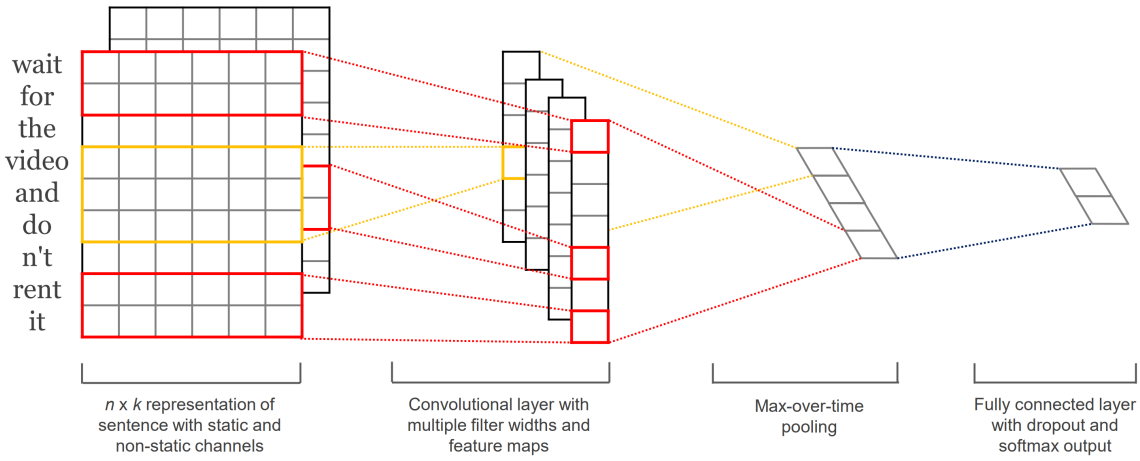

FIGURE 2.4: Convolutional neural network (CNN) architecture for text classification (Kim, 2014).

2.3.4 Graph Neural Network

Graph neural networks (GNN) are neural networks which train on graphs (Z. Wu et al., 2021). A general GNN architecture is shown in figure 2.5. A graph is a data structure consisting of entities (nodes) and relations (edges) between these entities. Formally, a graph is defined as $G = (V, E)$, with V referring to the set of nodes and E to the set of edges. Computationally, a graph can be implemented with an adjacency matrix. It can also have a node feature matrix and an edge feature matrix. The adjacency matrix A is an $n * n$ Matrix with 1 or a weight if an edge exists between two nodes otherwise 0. The node or edge feature matrix represents the feature vectors of the nodes or edges in the graph.

GNNs can be applied for three different levels of analytics: node level, edge level and graph level. Node level refers to using the GNN to classify nodes or do node regression. The output of edge-level

analytics can be used for edge classification and link prediction tasks. Graph classification tasks are called analytics on the graph level.

The main propagation operation for GNN are convolution operators, also applied are recurrent operators and skip connections. Zhou et al., 2020 describe different methods and applications of GNNs.

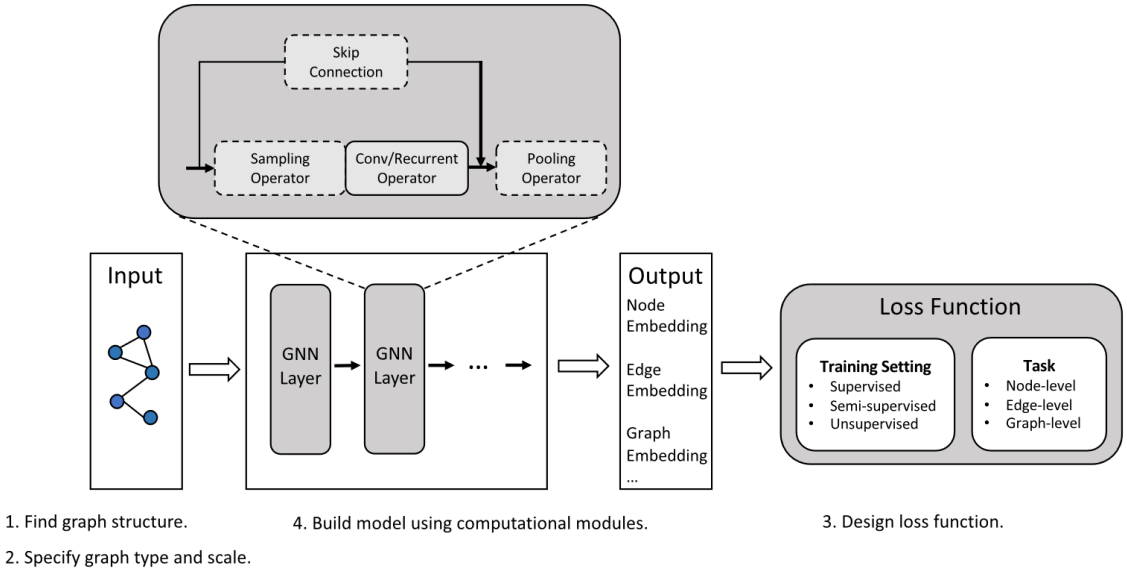

FIGURE 2.5: A general graph neural network (GNN) architecture (Zhou et al., 2020).

Graph Convolutional Neural Network (GCN) are convolutional neural networks which use convolution operators as propagation modules (Zhou et al., 2020). Graph convolution is very similar to 2-D convolution, as both use the neighbouring entities and the central entity to update the central entity. The difference is that in graph convolution the number of neighbouring entities (nodes) is unordered and variable in size.

GCNs can be grouped into two different methods: spectral approach and spatial approach. The spectral approaches use the spectral representation of the graph input and apply the mathematical foundations of graph signal processing. Spatial approaches, on the other hand, use the topology of the graph to learn information. Meaning that a node is updated by the convolution of the node's feature with its neighbouring nodes' features. Similar to CNN, GCN uses pooling to create smaller representations of the nodes.

2.4 Ontology

Ontologies are graphs of knowledge pertaining to a domain which contain the concepts, the terms and unique identifiers connected with relations (is-a, has-part, etc.) (Kevin Bretonnel Cohen and Demner-Fushman, 2014). An excerpt from the Gene Ontology (GO) is shown in figure 2.6.

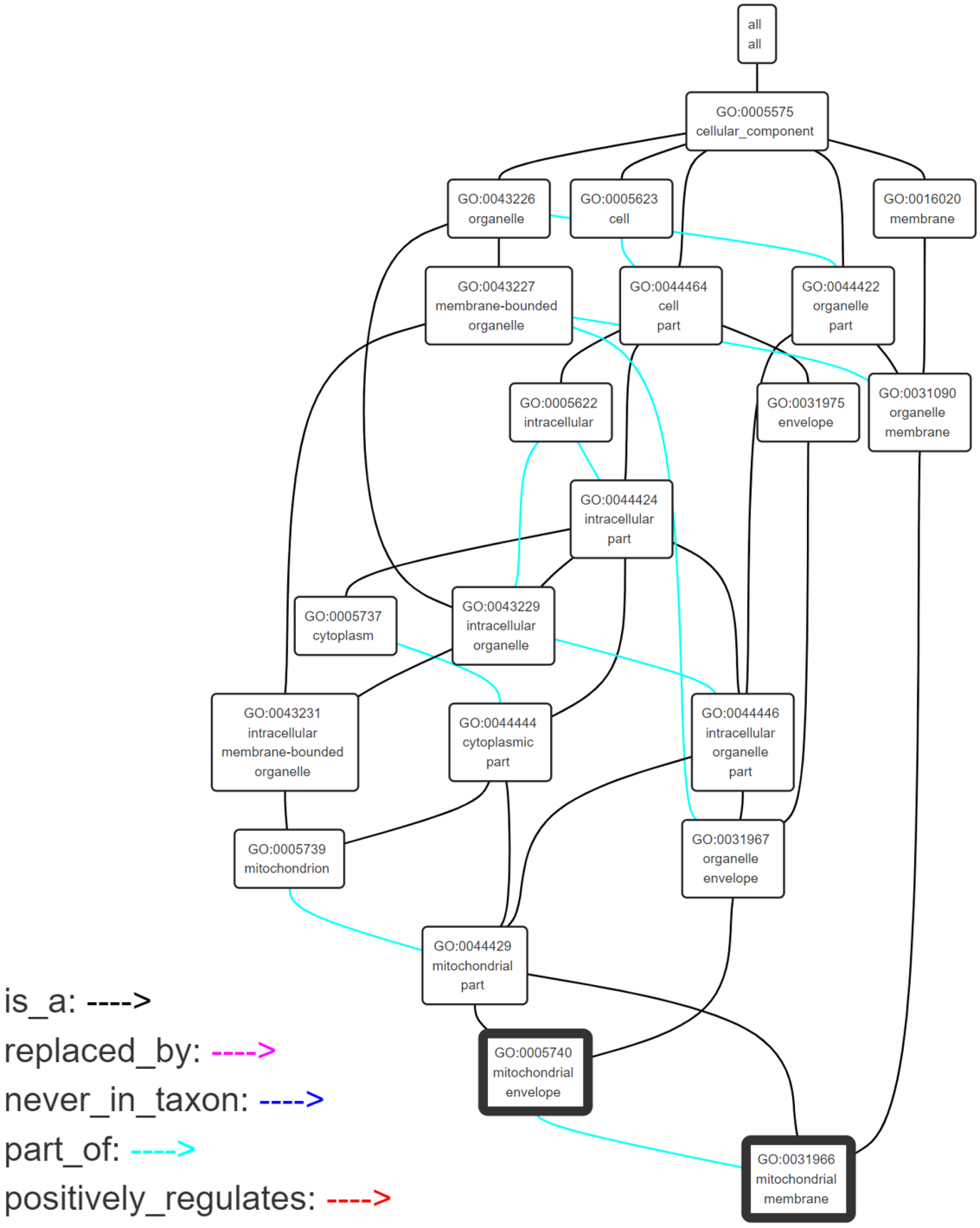

FIGURE 2.6: An excerpt from the Gene Ontology (Carbon and Mungall, 2018) which was created by using the Visualization tool from AmiGO 2 (Carbon and Mungall, 2018).

Willem Nico Borst and W. N. Borst, 1997 defines an ontology as a formal specification of a shared conceptualization. Conceptualization refers to a model which identifies the relevant concepts in the domain. Shared means that an ontology encompasses consensual knowledge. Explicit refers to the type of concepts in an ontology and the constraints on these concepts are explicitly defined. Formal defines that the ontology should be machine-readable. The Web Ontology Language (OWL) is an ontology language and was developed for the Semantic Web ("OWL 2 Web Ontology Language Document Overview (Second Edition)" 2012). Ontologies with the OWL syntax contain classes, properties, individuals, and data values and are stored as Resource Description Framework (RDF) files. OWL's formal semantics specifies how to derive logical consequences, i.e., facts that are not literally present in the ontology but are implied by the semantics (H. Wu and Yamaguchi, 2014). In the biomedical domain, ontologies are being used to share and analyze data in distributed sources (H. Wu and Yamaguchi, 2014).

The Open Biological and Biomedical Ontologies (OBO) Foundry[1] was created to facilitate the development, harmonization, application and sharing of science-based ontologies (Smith et al., 2007). It includes a set of organizational and structural principles which can be applied to compose ontologies in the OBO format using the OBO Editor. As different applications may require either OWL or OBO format, several systems have been developed to translate the syntax deterministically (Moreira and Musen, 2007; Tirmizi et al., 2011).

BioPortal[2] is a web portal which gives access to a library of biomedical ontologies and terminologies and was developed by the national centre for Biomedical Ontology (NCBO) (Whetzel et al., 2011). It contains over 1050 ontologies and over 12 million total classes by the time of this writing.

With so many ontologies available, the integration of information becomes a great problem, as not all resources use the same vocabulary. To overcome this barrier of the variety of names used to express the same concept and the absence of a standard format for distributing terminologies, the Unified Medical Language System (UMLS)[3] was developed Bodenreider, 2004. The UMLS is a repository of biomedical vocabularies, containing concepts and their relations to each other. The UMLS is a data resource with three knowledge sources: the Metathesaurus which contains biomedical concepts, names, and relationships, the semantic network which provides a textual description of semantic types and defines important relations and the SPECIALIST lexicon which provides general biomedical lexicon with syntactic, morphological and orthographic information). The Semantic Network provides high-level categories to categorize every Metathesaurus concept. The core component of the UMLS is the Metathesaurus. The following vocabularies were integrated into the UMLS Metathesaurus: NCBI taxonomy, Gene Ontology, the Medical Subject Headings (MeSH), Online Mendelian Inheritance in Man (OMIM) and the Digital Anatomist Symbolic Knowledge Base. The concepts in the UMLS are constructed as clustered synonyms and linked to other concepts by different relationships. The specific structure of the UMLS makes it possible to perform tasks

[1] https://obofoundry.org
[2] https://bioportal.bioontology.org
[3] http://umlsks.nlm.nih.gov

such as collecting various terms used to name a concept, extracting the relations of one concept to other concepts or getting a set of concepts for a given category, using the list of concepts that were assigned to a semantic type.

2.5 Evaluation

2.5.1 Metrics

To evaluate a TC system the first step is to build a confusion matrix (Jurafsky and Martin, 2020). Figure 2.7 shows the layout of a confusion matrix. For binary or multi-class classification tasks the accuracy can be calculated to get the percentage of how many observations the system predicted correctly. The accuracy score is calculated as displayed in figure 2.7 or as followed:

$$accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i) \tag{2.8}$$

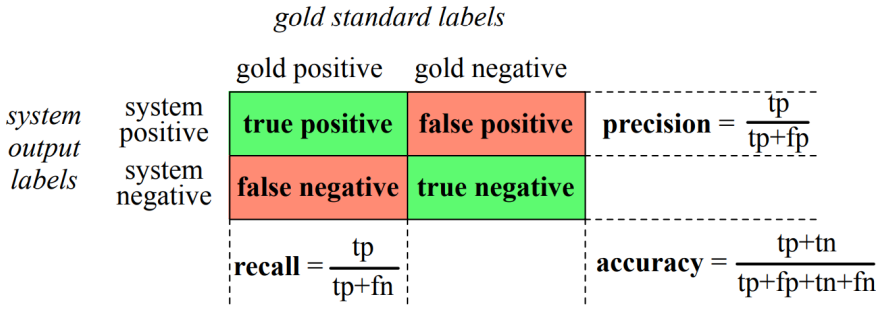

FIGURE 2.7: Confusion matrix which displays how well a binary classification method performs (Jurafsky and Martin, 2020).

When the goal is to find rare observations or observations which are not completely balanced, the metrics precision, recall and F1-score can be used. The equations of precision, recall and F1 score can be found from equation 2.9 to 2.11. Precision measures the ability of the system to accurately predict positive instances among all instances predicted as positive. Recall measures the capability to find all positive observations. F1-score combined these two measures, where 1 denotes that precision and recall are weighted equally.

To use these metrics for multi-label classification they can be averaged in different ways: micro, macro and weighted. The micro-averaged metrics calculate the metrics globally by counting the total over all classes. The macro-averaged metrics are calculated by measuring the metrics for each class and taking the unweighted mean. Thus not taking label imbalance into account. The weighted metrics calculate the metrics for each label and weigh it by their support. The support for a class is

the number of true instances for this class. The weighted metrics thus account for label imbalance. The equations of micro, macro and weighted metrics can be found in the appendix section A.3.

$$precision = \frac{TP}{TP + FP} \tag{2.9}$$

$$recall = \frac{TP}{(TP + FN} \tag{2.10}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{2.11}$$

2.5.2 Bootstrap Test

The bootstrap test is a test to check whether two systems which perform differently on the same dataset, performed differently by accident or not (Jurafsky and Martin, 2020). The test can be applied to any evaluation metric.

The bootstrap test assumes the null hypothesis H_0: Model A is not better than Model B. H_0 described in other words: Model A is accidentally better than Model B. The algorithm calculates a onesided empirical p-value which describes the percentage of samples where model A accidentally performed better than model B. To reject the H_0, a threshold is defined for example $a = 0.05$, if the p-value is smaller than the threshold the H_0 can be rejected. Concluding, that model A is better than model B, indeed, and the observed difference was not accidental.

The bootstrap algorithm uses bootstrapping, which refers to repeatedly drawing samples from a dataset with replacement. Thus creating many virtual test sets from one test set, assuming that the virtual test sets are representative of the population. The p-value is calculated by counting how many times Model A is accidentally better than Model B. This is done by counting how often the virtual test set results in a difference higher or equal to twice the original difference.

Chapter 3

Materials and methodology

The implementation of all methods and the materials can be found at the GitHub page of the thesis[1].

3.1 Corpora

This section describes the following three corpora which were used for the text classification task: Ohsumed (Joachims, 1998), Colorado richly annotated full text (CRAFT, K. Bretonnel Cohen et al., 2017) and MedOBO[2].

3.1.1 Ohsumed

The Ohsumed dataset contains medical abstracts from the MeSH categories of the year 1991 which are categorized into 23 cardiovascular diseases (Joachims, 1998). To recreate the results from Yao, Mao, and Luo, 2018 the same version of the Ohsumed dataset was used which contains the first 20'000 abstracts of the year 1991. As Yao, Mao, and Luo, 2018 focused on multi-class text classification the documents with more than one label were excluded. This resulted in 7400 documents, 3357 in the training set and 4043 in the test set. 336 documents of the training set (10%) were used as the development set. Figure 3.1 shows the label distribution of this dataset. The description of the categories is in the appendix A.1.

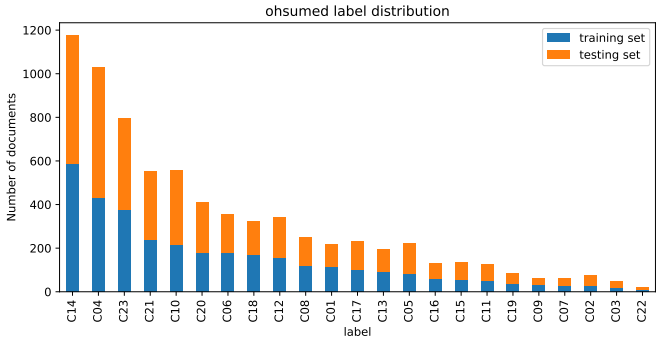

FIGURE 3.1: Distribution of the labels of the Ohsumed dataset.

[1] https://github.zhaw.ch/saxerja1/Ontology-Aware-Biomedical-Text-Classification
[2] https://github.com/acg-team/MEDOBO

3.1.2 Colorado richly annotated full text corpus

The Colorado richly annotated full text (CRAFT) corpus (K. Bretonnel Cohen et al., 2017) version 5.0.2 consists of 97 full-text articles annotated in the OBO format, organized into 9 ontologies and total of 11 different classes: Chemical Entities of Biological Interest (CHEBI), Cell Ontology (CL), Molecular Process Ontology (MOP), Gene Ontology (biological process (GO_BP), cellular component (GO_CC), and molecular function (GO_MF)), NCBI Taxonomy (NCBITaxon), Protein Ontology (PR), Sequence Ontology (SO), uber-anatomy ontology (UBERON), and MONDO Disease Ontology (MONDO). MONDO annotations weren't used, as they were made with a different annotation tool (Knowtator v2 annotation tool). The concept annotations were used to change the original dataset into a multi-label text classification dataset. For this, the corresponding ontologies of the concept annotations for each article were used as labels. The corpus was split into training, validation and testing sets according to the description in the GitHub folder of CRAFT[3]. The distribution of the ontologies over the articles is shown in figure 3.2a. The number of labels per document ranges from 7 to 10 and is shown in figure 3.2b.

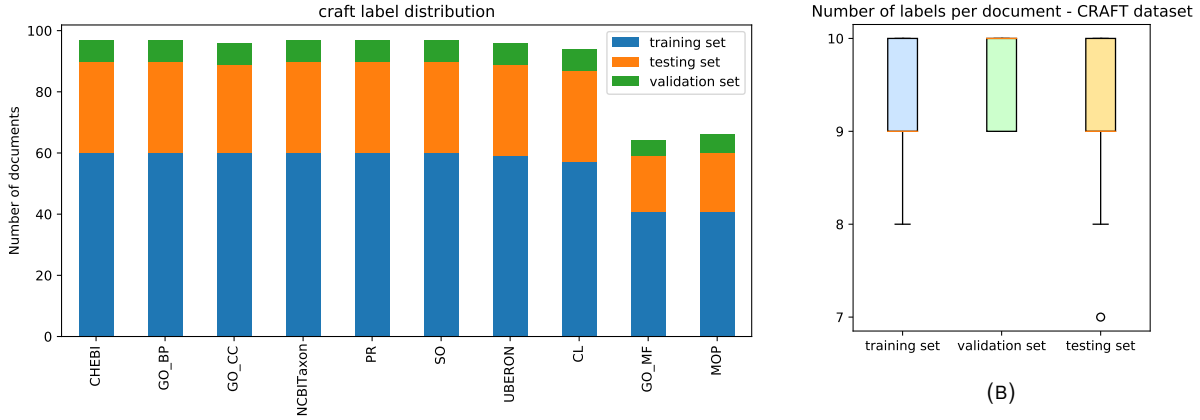

FIGURE 3.2: The distribution of the labels of the CRAFT dataset (A) and the number of labels per document (B)

3.1.3 MedOBO

The MedOBO corpus[4] is a dataset of automatically tagged MEDLINE abstracts with OBO ontologies. From the extracted abstracts of MEDLINE, MeSH terms were extracted. The terms were expanded using UMLS. Then using exact matching, the terms were mapped to ontologies from the OBO Foundry, which are used as labels. The dataset consits of more than 18 million abstracts among which 10 different training datasets from 100,000 to 1,000,000 instances are sampled. The testing and development sets are sampled with 100'000 instances each. The training set includes 83 unique

[3] https://github.com/UCDenver-ccp/CRAFT
[4] https://github.com/acg-team/MEDOBO

ontology labels. To make the training feasible for this project a sample from the original dataset was extracted: 10'000 articles for testing, 2'000 articles for development and 10'000 articles for training. The down-sampled dataset included 76 OBO ontologies[5]. The training set included 74 unique labels, the validation set 65 unique labels and the testing set 72 unique labels. The distribution of the labels which are mapped to more than 1000 articles is shown in fig 3.3a. The number of labels per document ranges from 0 to 39 and is shown in figure 3.3b.

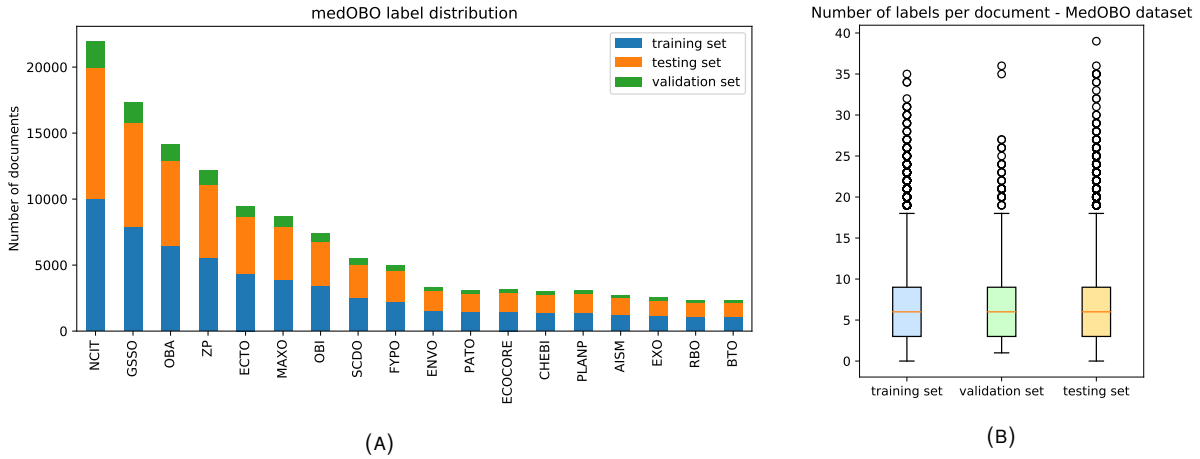

FIGURE 3.3: The distribution of the labels of the MedOBO 10k dataset (A) and the number of labels per document (B). Figure (A) shows only the ontologies with more than 1000 articles in the training set.

3.2 Preprocessing data

All the text data was preprocessed using the code from Yao, Mao, and Luo, 2018, specifically the python file called 'remove words'[6]. The sentences were tokenized using the split function from python. Infrequent words (< 5 occurrences) and stop words were removed. The English stop words from the Natural Language Toolkit (NLTK) library (Bird et al., 2009) were used for this purpose.

3.3 Model

All models were implemented and trained using Tensorflow (Developers, 2022) .

3.3.1 TextGCN

The TextGCN model architecture was proposed by Yao, Mao, and Luo, 2018 and can be found in the Github page of TextGCN[7]. It is a graph convolutional network for text classification. First, a global

[5]For the list of the names please refer to table A.3 in the appendix
[6]https://github.com/yao8839836/text_gcn/blob/master/remove_words.py
[7]https://github.com/yao8839836/text_gcn

text graph with word nodes and document nodes is created, which is implemented as an adjacency matrix. Two different weights were added: document-word edges as TF-IDF and word-word edges as PMI. Formally, the weights of the adjacency matrix are defined as followed:

$$A_{ij} = \begin{cases} \text{PMI}(i,j) & i,j \text{ are words, PMI}(i,j) > 0 \\ \text{TF-IDF}(i,j) & \text{i is document, j is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Further, an identity matrix was implemented and used as the feature matrix. Thus every word and document is represented as a one-hot vector. The text graph is then fed into a two-layer GCN, where the second-layer node embeddings have the size of the number of different labels. For the multi-class classification task, the Softmax classifier is then applied to these embeddings and the categorical cross-entropy error over all labelled documents is used as a loss function. For the multi-label classification task, the classifier and loss function was adapted. The Sigmoid classifier and the binary cross-entropy over all labelled documents were used. The model was trained using the Adam optimizer.

Ontologies were added to the adjacency matrix in two ways: word-ontology edges and document-ontology edges. The word-ontology edge was set to 1 if the word occurred as a term in the ontology, otherwise, it was set to 0. The document-ontology edge was set to 1 if the document had a word which matched with a term in the ontology, otherwise, it was also set to zero. The adjacency matrix with ontology connections is shown in figure 3.4. The feature matrix was also adjusted to include one-hot vectors for the ontologies. Figure 3.5 illustrates a flow chart and the graph of the model architecture.

Using the MedOBO dataset, the hyperparameters were tuned using the grid search technique. The best hyperparameters are shown in table 3.1. All the experiments on hyperparameter tuning can be found in the appendix in table B.3.

TABLE 3.1: This table shows the hyperparameters used for the TextGCN model.

Parameter	**Value**
dropout probabilty	0.5
sigmoid threshold	0.3
l2 reg lambda	0.0
starting learning rate	0.02
dimension of hidden layer	200

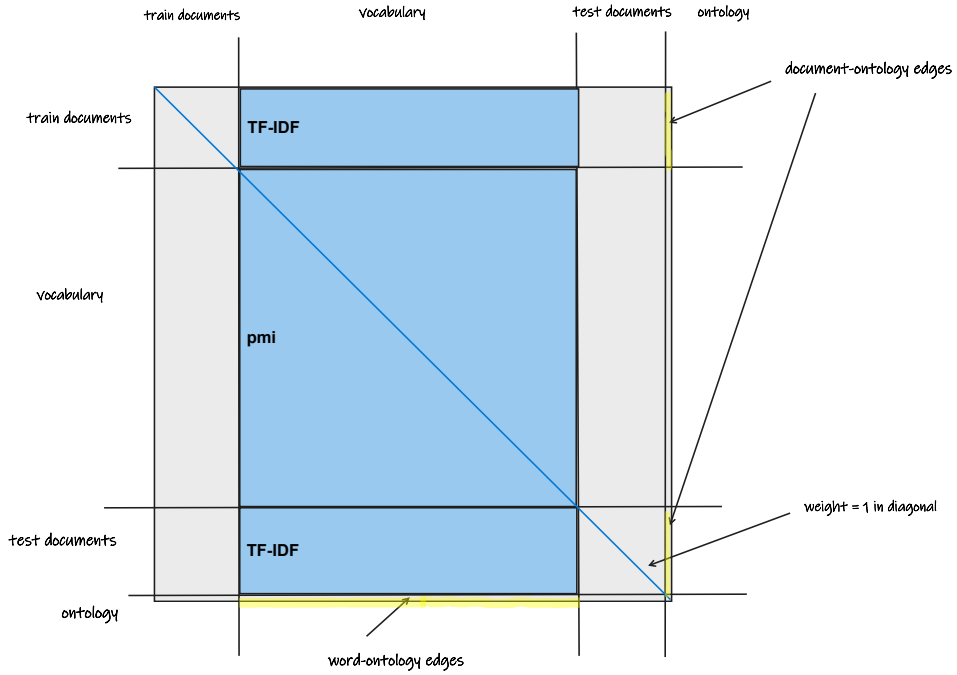

FIGURE 3.4: Adjacency matrix with ontology connections for implementing the text document ontology graph as input for the TextGCN model.

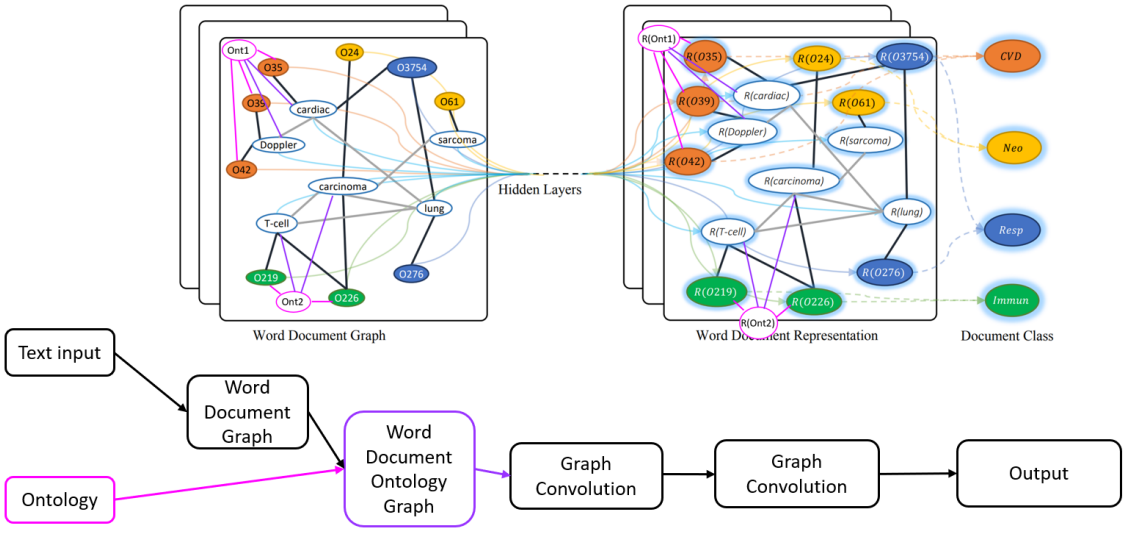

FIGURE 3.5: TextGCN model architecture with adding ontology connections as a flow chart and as a graph. The word document graph figure was adopted from (Yao, Mao, and Luo, 2018).

3.3.2 CNN

The CNN model for TC proposed by Kim, 2014 was used as a baseline. A blog post and specific description of this CNN model can be found on Denny's Blog[8]. The CNN architecture consists of an embedding layer followed by three convolutional layers, max-pooling, dropout and a final dense layer. For multi-class classification, the categorical cross-entropy error over all labelled documents is used as a loss function. For the multi-label classification, the binary cross-entropy over all labelled documents was used and the sigmoid function was applied to the output of the last layer. The model was trained using the Adam optimizer from TensorFlow.

To incorporate ontology information into the model, an embedding layer was added to include pretrained embeddings from ontologies. The ontology embeddings were then appended to either trained with the model or pretrained BioASQ embeddings. The new larger embedding was then fed into the convolution layers. The adapted architecture of the CNN model can be seen in figure 3.7 as graph and in figure 3.6 as flow chart.

Similar to the TextGCN model above, the hyperparameters of the CNN model were tuned on the MedOBO dataset. The best hyperparameters are shown in table 3.2. All the experiments on hyperparameter tuning of the CNN model can be found in the appendix in table B.2.

TABLE 3.2: The hyperparameters used for the CNN model.

Parameter	Value
filter sizes	3, 4, 5
number of filters	128
dropout probability	0.5
sigmoid threshold	0.3
l2 reg lambda	0.0
starting learning rate	0.001
dimension of hidden layer	200

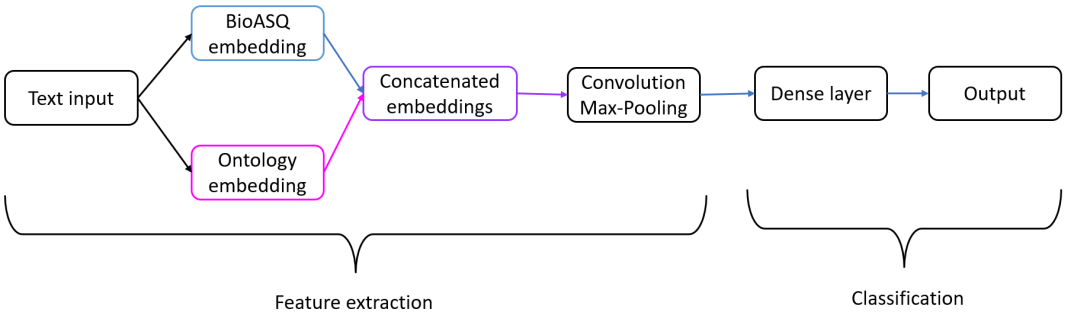

FIGURE 3.6: CNN model architecture with ontology embeddings as a flow chart.

[8] dennybritz.com/posts/wildml/implementing-a-cnn-for-text-classification-in-tensorflow

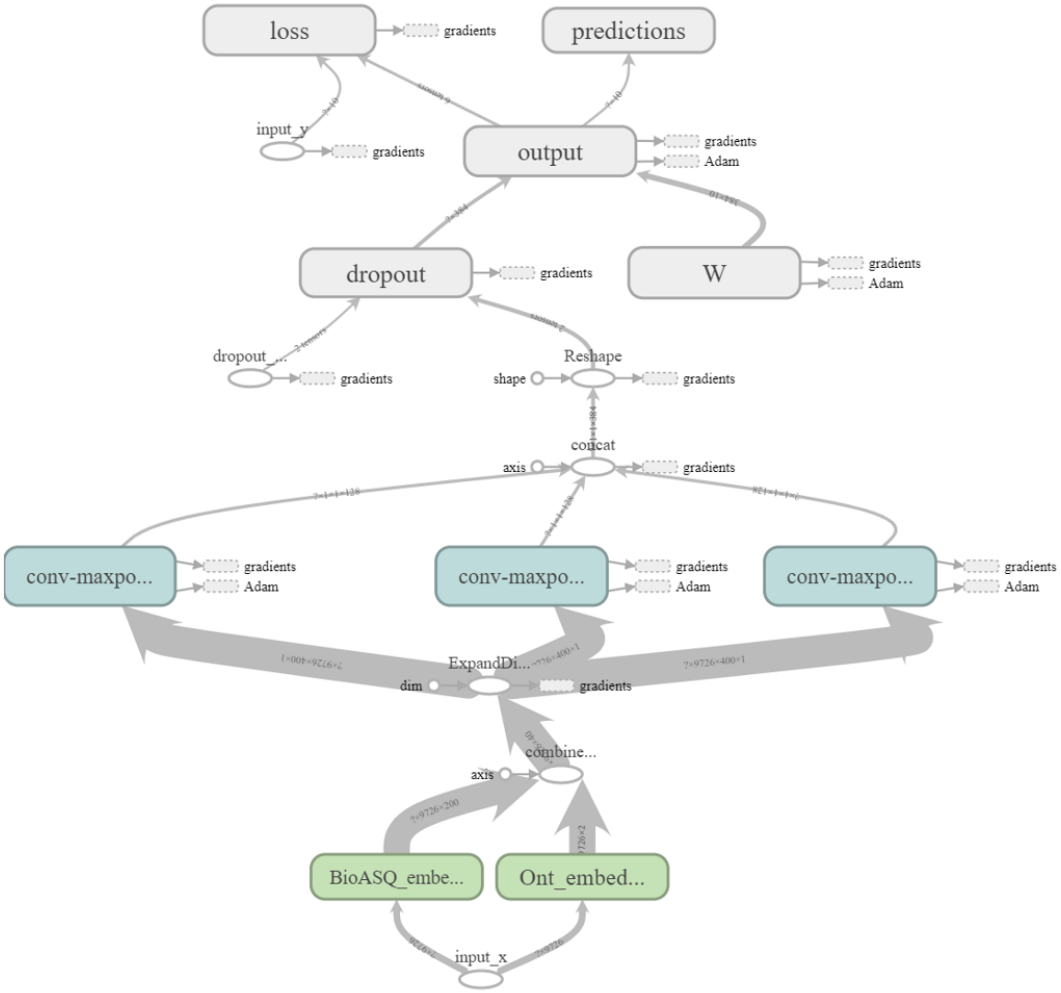

FIGURE 3.7: CNN model architecture with ontology embeddings as graph.

3.3.3 TextGCN-CNN

This thesis proposes a new architecture called TextGCN-CNN that combines the TextGCN and CNN architectures in a hybrid architecture enabling isolating ontological from sequence feature sets such that the comparison between the two turns feasible. The proposed architecture is shown in figure 3.8 as a simple flow chart in 3.8a and the whole architecture in 3.8b. As displayed in the flow chart the feature extraction of both TextGCN and CNN is independent. The feature extraction of the CNN features is the same as described in section 3.3.2. The feature extraction of the TextGCN was adjusted as follows: The output dimension of the second graph convolution layer was set to the same size as the output dimension of the hidden layer of the CNN architecture. Which was set to a dimension of 200. Then a mask was used to select only the relevant document features. The masked

features of TextGCN are then appended to the features of CNN. As the final layer, a dense layer is used with the number of labels as the output dimension.

Similar to the TextGCN or CNN models, for the multi-class classification softmax, was applied to the final scores and the categorical cross-entropy error over all labelled documents is used as a loss function. For the multi-label classification, the binary cross-entropy over all labelled documents was used and the sigmoid function was applied to the output of the last layer. The model was trained using the Adam optimizer from TensorFlow.

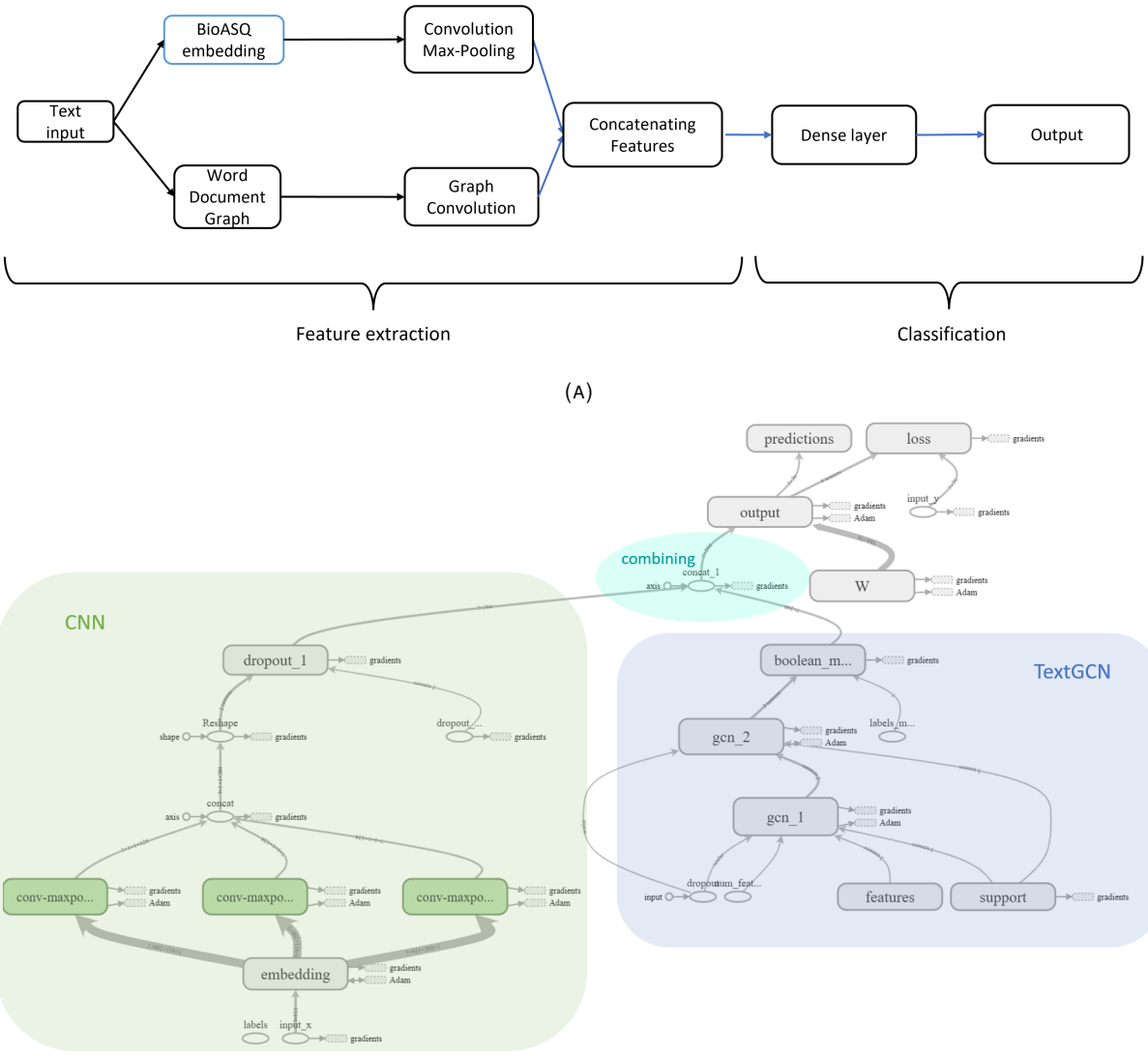

FIGURE 3.8: Text GCN-CNN model architecture: (A) flow chart, and (B) the architecture.

The incorporation of ontological information into the model was done as described in section 3.3.1

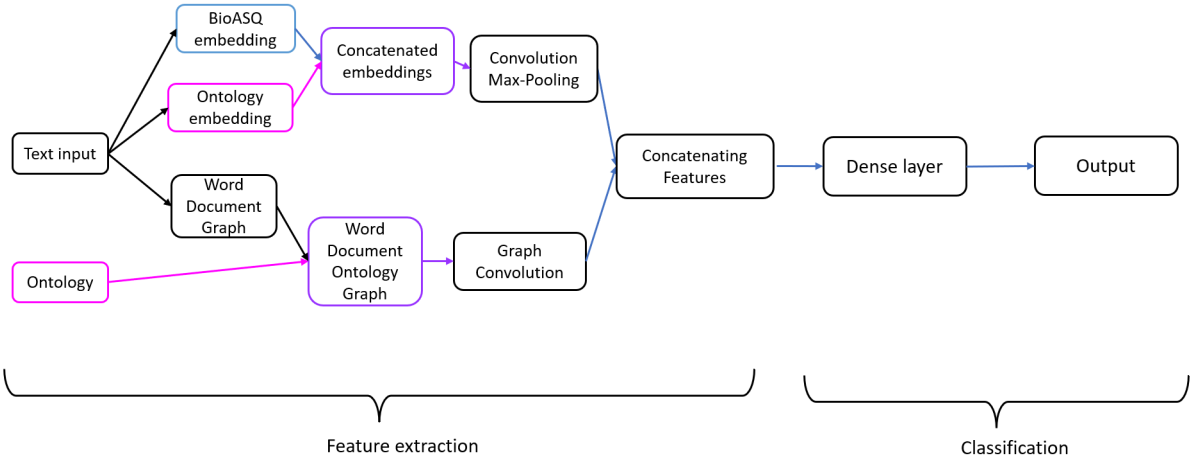

FIGURE 3.9: TextGCN-CNN model architecture incorporating ontology information as a flow chart.

and 3.3.2. Figure 3.9 shows a flow chart of the TextGCN-CNN model incorporating ontology information both in the TextGCN part of the model and the CNN part of the model.

Using the Ohsumed dataset the starting learning rate of the TextGCN-CNN model was tuned. The used hyperparameters are shown in table 3.3. All the experiments on hyperparameter tuning can be found in the appendix in table B.1.

TABLE 3.3: This table shows the hyperparameters used for the TextGCN CNN model.

Parameter	**Value**
filter sizes	3, 4, 5
number of filters	128
dropout probability	0.5
sigmoid threshold	0.3
l2 reg lambda	0.0
starting learning rate	0.01
dimension of hidden layer	200

3.4 Pretrained embeddings

For the CNN component in all models, three different pretrained embeddings were used. As general biomedical pretrained embeddings, the embeddings from the BioASQ challenge by Pavlopoulos, Kosmopoulos, and Androutsopoulos, 2014 were used. BioASQ embeddings[9] were trained using the Word2vec algorithm on 10'876'004 English abstracts of biomedical articles from PubMed. The

[9] http://bioasq.lip6.fr/tools/BioASQword2vec/.

pretrained embeddings include 1,701,632 unique words embedded in a vector space of the dimension of 200.

To include biomedical information in the form of pretrained embeddings two different methods of UMLS embeddings were used. Maldonado, Yetisgen, and Harabagiu, 2019 used GANs to learn knowledge embeddings from the Metathesaurus and the Semantic Network of the UMLS. The pretrained embeddings[10] contain 3'210'963 Concept Unique Identifiers (CUIs) mapped to vectors with a dimension of 50. To use the embeddings, first, the preferred names for each CUI were extracted from the MrCONSO file from UMLS. The preferred names were cleaned in the same way as all the text data. A new mapping of preferred names to embeddings was made, resulting in 3'208'679 unique terms as some of the CUIs couldn't be found in the UMLS of 2017, 2018 and 2021 combined. In another apparoach, Beam et al., 2019 used an extremely large collection of multi-modal medical data to learn clinical concept embeddings[11]. The embeddings were trained using the GloVE algorithm on a concept co-occurrence matrix. The concept co-occurrence matrix was constructed using all concepts from the collection. 109'054 unique CUIs are mapped to vectors with a dimension of 500. To use these embeddings the same method as above explained was applied. This generated a term to embedding mapping with 107'787 unique terms.

3.5 Evaluation

For all the experiments while training the model, the model was saved if the validation loss decreased and the validation micro F1-score increased. The last saved model was then used to evaluate the testing set.

For the multi-class classification task accuracy was used as the evaluation measure. It was implemented using the classification report function from Scikit-learn. For the multi-label classification task the micro, macro and weighted F1-Score, precision and recall were used to evaluate the test set. It was implemented using the same classification report function from Scikit-learn (Pedregosa et al., 2011).

Confusion matrices were used to visualize the classification accuracy for each class. The multi-label confusion matrix function from Scikit-learn was applied. This function was used to compute a confusion matrix for each class.

3.5.1 Bootstrap Algorithm

To check if one model is not accidentally better than another model the bootstrap algorithm was applied. The bootstrap algorithm was implemented as described in Daniel Jurafsky and James H. Martin, 2020. The pseudo-code of the algorithm is shown in Algorithm 1.

[10] https://GitHub.com/r-mal/umls-embeddings
[11] https://github.com/beamandrew/cui2vec

For the Ohsumed and CRAFT dataset the number of virtual test sets (n_sets) was set to 10'000. For the MedOBO dataset, n_sets was set to 1'000 due to the larger size of the test samples as full abstracts and time constraints. The threshold for the p-value for rejection of H_0 was set at 0.05.

Algorithm 1 Bootstrap Algorithm

H_0: A `is not better than` B. ▷ Null Hypothesis
$\alpha \leftarrow 0.05$ ▷ Threshold for p-value
$counter \leftarrow 0$ ▷ Number of times A is better than B
$\delta(test\_set) \leftarrow metric_A(test\_set) - metric_B(test\_set)$
for $n = 1$ `to` n_sets **do** ▷ Do for number of virtual test sets
  $new\_set \leftarrow$ `empty list`
  for $s = 1$ `to` set_size **do** ▷ Do for size of the virtual test set
    $x \leftarrow$ `random sample from` $test\_set$
    $new\_set \leftarrow new\_set + x$
  end for
  $\delta(new\_set) \leftarrow metric_A(new\_set) - metric_B(new\_set)$ ▷ How much A is accidentally better than B
  if $\delta(new\_set) \geq 2 * \delta(test\_set)$ **then**
    $counter \leftarrow counter + 1$
  end if
end for

p-value($test\_set$) $\leftarrow \frac{counter}{n\_sets}$ ▷ Onesided empirical p-value
if p-value($test\_set$) $< \alpha$ **then**
  H_0 `can be rejected.`
else
  H_0 `can not be rejected.`
end if

Chapter 4

Results and discussion

4.1 Ohsumed

Classifying the Ohsumed dataset is a multi-class classification problem. As the Ohsumed dataset has 23 classes, the random classification accuracy is 4.3 %. The following subsections report the performance of different models on the Ohsumed dataset.

4.1.1 TextGCN

The results of classifying with the TextGCN model by Yao, Mao, and Luo, 2018 and with additional connections to OBO Ontologies are shown in table 4.1. The TextGCN Baseline (BS) yielded an accuracy of 68.91 % which is the range 0.6836 ± 0.0056 as Yao, Mao, and Luo, 2018 described.

TABLE 4.1: Results of TextGCN and additional information from ontologies on ohsumed dataset. Ontology (top10) refers to the top 10 ontologies with the most active developers from the OBO Foundry. Ontology (all) refers to all of the ontologies from OBO Foundry with an OBO format. Minus (-) refers to without the feature, plus (+) refers to with the following feature. More information on the model can be found in section 3.3.1.

Description	Accuracy
TextGCN Baseline (BS)	68.91 %
BS - pmi	65.65 %
BS - TF-IDF	14.17 %
BS + ontology (top10) word-ont	69.01 %
BS + ontology (top10) doc-ont	68.93 %
BS + ontology (top10) doc-ont and word-ont	68.91 %
BS + ontology (all) word-ont	68.84 %
BS + ontology (all) doc-ont	68.46 %
BS + ontology (all) doc-ont and word-ont	68.49 %

The *BS - pmi* model was trained without the word-word connections in the text graph which are weighted by the PMI. The *BS - TFIDF* model was trained without the word-document connections in the text graph which are weighted by the term TF-IDF. These experiments showed that the word-document connections weighted by TF-IDF gave the most information to the model. The word-word connections weighted by the PMI only give little information to the model.

To integrate ontologies into the model, the top 10 ontologies[1] with the most active developers from the OBO Foundry were used to add ontology-word and ontology-document connections to the text graph in models *word-ont* and *doc-ont* accordingly. The results show that both the *word-ont* and *doc-ont* connections make only very little positive change in the accuracy of the model. To verify whether this change in accuracy is statistically relevant, the bootstrap algorithm, as described in Section 3.5.1, was applied. The Null hypothesis (H_0) is defined as The model BS + ontology (top10) word-ont is not better than BS. The resulting p-Value is 0.1587 which is higher than the threshold of 0.05. Thus the H_0 can not be rejected and the model with ontology did not give better results.

Further, all the ontologies from OBO Foundry were added to the text graph, resulting in worse accuracy than the BS. Figure 4.1 shows the mean of the ontology-document connections per label. What can be seen is that all ontologies have a very similar mean over all labels. Thus not giving any additional information to separate the labels.

Concluding that for the Ohsumed dataset, adding ontologies to the text graph and using TextGCN does not improve the model.

As the difference between adding word-ont, doc-ont or both edges was negligible, all further experiences were made using both word-ont and doc-ont edges. This decision was made, as it is consistent with the TextGCN Text-Document graph having both word-word and document-word edges.

4.1.2 CNN

The results of using a CNN model to classify the Ohsumed dataset is shown in table 4.2. The CNN baseline model yielded an accuracy of 55.26 %. Pretrained embeddings from the BioASQ challenge were used instead of training the embeddings with the model. The word coverage i.e., the ratio of the dataset words that are found in the embeddings vocabulary for the Ohsumed dataset with the BioASQ embeddings was 99.8 %. BioASQ embeddings improved the classification by about 12 %. This could be because the BioASQ embeddings were trained with more data and a more complex architecture and therefore contain more information. To integrate ontology into the CNN model, pretrained embeddings of the UMLS by Maldonado, Yetisgen, and Harabagiu, 2019 were concatinated with the BioASQ embeddings. The word coverage of the UMLS embeddings was 22.5 %. The low coverage is due to the types of terms in UMLS. UMLS terms often consist of multiple words and the mapping was done using exact matching, thus the UMLS terms with multiple words weren't used. The resulting model produced worse results than the model that included only BioASQ

[1] For the list of the names please refer to table A.2 in the appendix.

embeddings. The worse result might be because of the too low coverage of the words, as the other 78 % of embeddings were randomly generated, leading to disrupt the original BioASQ distribution hence less informative embeddings.

TABLE 4.2: Results of CNN on Ohsumed dataset. The following abbreviations were used: CNN: CNN Baseline, BioASQ: with pretrained BioASQ embeddings, UMLS: with pretrained UMLS embeddings from Maldonado, Yetisgen, and Harabagiu, 2019, train: embedding trained with the model.

Description	Accuracy
CNN	55.26 %
CNN_{BioASQ}	67.62 %
CNN_{BioASQ} + UMLS	60.87 %

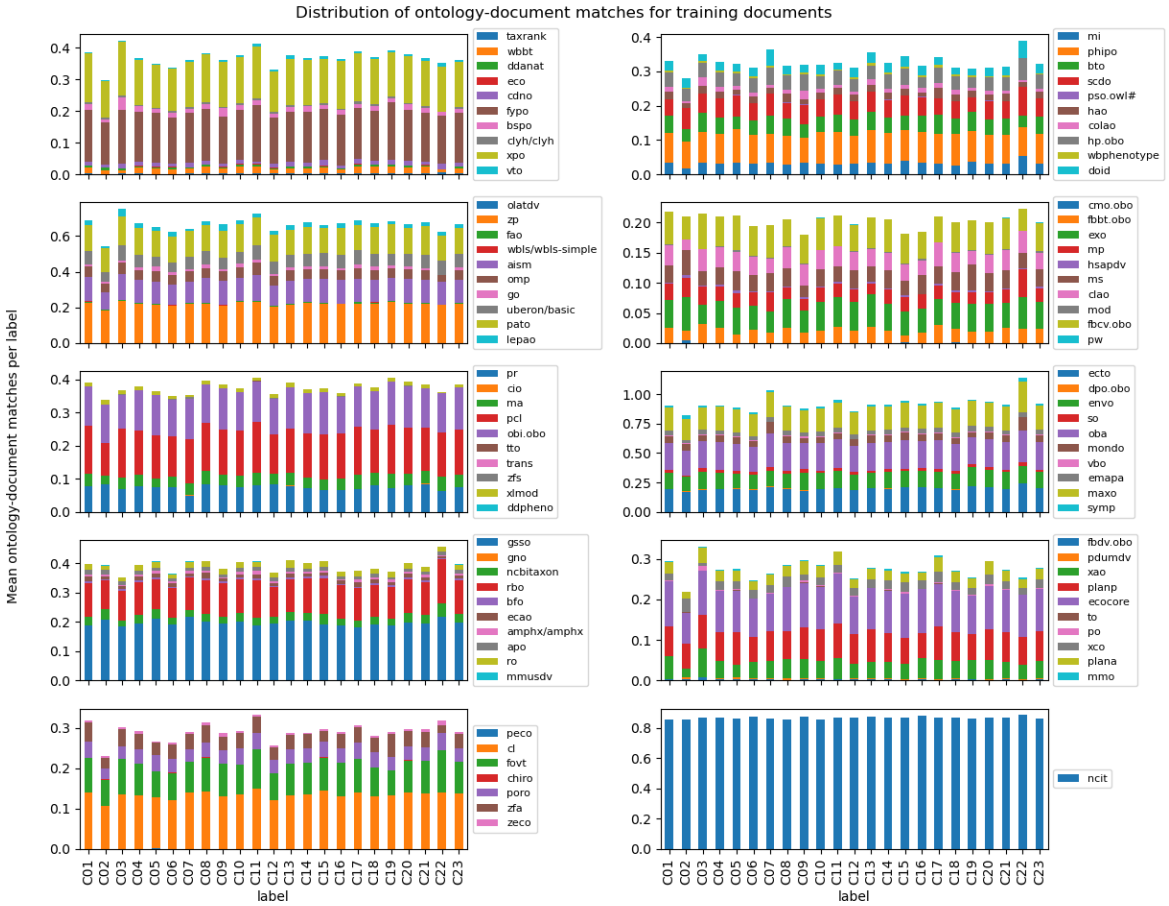

FIGURE 4.1: Ohsumed training documents mean of exact matches of all OBO Foundry ontologies.

4.1.3 TextGCN-CNN

TextGCN-CNN is a hybrid model of TextGCN and CNN which are trained at the same time. The features of TextGCN and CNN are combined before the classification layer of the architecture. More information on the new architecture can be found in Section 3.3.3.

The results of the TextGCN-CNN model with the Ohsumed dataset are shown in table 4.3. As a baseline model, the TextGCN-CNN was trained with the pretrained BioASQ embeddings for the CNN input (BS_{BioASQ}) and resulted in an accuracy of 67.0 %. In the same way, the ontology connections were added to the TextGCN model, they were also added to the TextGCN-CNN model. Using the top 10 ontologies didn't improve the accuracy. Adding all of the ontologies improved the accuracy by 1.19 % though. To check if the positive change in accuracy is statistically relevant a bootstrap algorithm was applied. The Null Hypothesis H_0: $BS_{BioASQ} + Ontology(all)$ is not better than BS_{BioASQ} was assumed. The p-value resulted in 0.0128, as the p-value is smaller than 0.05 the H_0 can be rejected thus $BS_{BioASQ} + Ontology(all)$ is better than BS_{BioASQ}.

TABLE 4.3: Results of TextGCN-CNN model on Ohsumed dataset. BioASQ refers to the pretrained BioASQ embeddings. Ontologies (top10) refers to the top 10 ontologies with the most active developers from the OBO Foundry. Ontology (all) refers to all of the ontologies from OBO Foundry with an OBO format.

Description	Accuracy
BS_{BioASQ}	67.00 %
BS_{BioASQ} + Ontology (all)	68.19 %
BS_{BioASQ} + Ontology (top10)	66.96 %

4.1.4 Comparing different architectures

The best results from all the three model architectures TextGCN, CNN and TextGCN-CNN using the Ohsumed dataset are shown in table 4.4. The TextGCN-CNN architecture did not improve either the TextGCN or CNN model.

TABLE 4.4: Results of Ohsumed dataset, comparing the best result from the different model architectures.

Description	Accuracy
CNN_{BioASQ}	67.62 %
TextGCN	68.91 %
$TextGCN - CNN_{bioASQ}$ + Ontology (all)	68.19 %

Yao, Mao, and Luo, 2018 focused with the TextGCN architecture on capturing global word co-occurrence information. CNNs, on the other hand, are designed to detect local and position-invariant

patterns (Minaee et al., 2022). So the hypotheses was, that the combination of both would improve the result, as it should both detect local and global features. Looking more closely at the details of both architectures, there are some ways for both architectures to capture local and global features. The TextGCN architecture uses TF-IDF to make edges between documents and words, which incorporates both local and global parameters. Nevertheless, it does not take word order into account, so less semantic meaning is captured. The CNN architecture applies max-over-time pooling layer which aggregates the information from all features thus capturing some global information about the text. Further, the use of the pre-trained BioASQ embeddings already provides some understanding of global patterns in the language, improving the capture of global patterns in the text data. Those features makes it possible for both TextGCN and CNN to get similar results.

Very recently Zeng et al., 2022 proposed an ensemble method combining TextGCN with CNN which is very similar to the TextGCN-CNN architecture. The differences is that they use a simplified boosting algorithm to make CNN learn the samples misclassified by GCN. They compare parallel and serial ensemble of GCN and CNN, and concluded that the serial ensemble is superior to the parallel. As the serial ensemble can fully fuse global and local knowledge, whereas the parallel ensemble looses information by concatenating the features. There results on the Ohsumed dataset are 0.6878 for the parallel ensemble, 0.6883 for the serial ensemble, 0.6833 with the TextGCN and 0.5844 with the CNN. By using the simplified boosting algorithm they got improved results: GCN-CNN-serial-SB got 0.7185, GCN-CNN-parallel-SB 0.7017.

This shows that the combination of the two architectures can bring improvements, but must be combined in a very specific way so that they can build on each other.

4.2 CRAFT

The Colorado richly annotated full text (CRAFT) corpus was used as a multi-label classification problem of 10 ontologies assigned to the documents. The following subsections report the performance of different models on the CRAFT dataset.

4.2.1 TextGCN

The TextGCN model architecture used for the CRAFT dataset resulted in a macro F1 score of 0.9509 and a micro F1 score of 0.9583. On the other hand, the recall was 1.0 and the precision was 0.92, which means that all labels are assigned for all documents. This is also evident from the confusion matrices in figure 4.2. This could be because the dataset is very small and the labels are very similar. Thus resulting in an over-fitted model. Adding the same ontologies as the labels of the CRAFT dataset into the text graph did not change the result of the classification. Probably the model architecture is too complex for this dataset, and thus adding more information does not change the results of the classification.

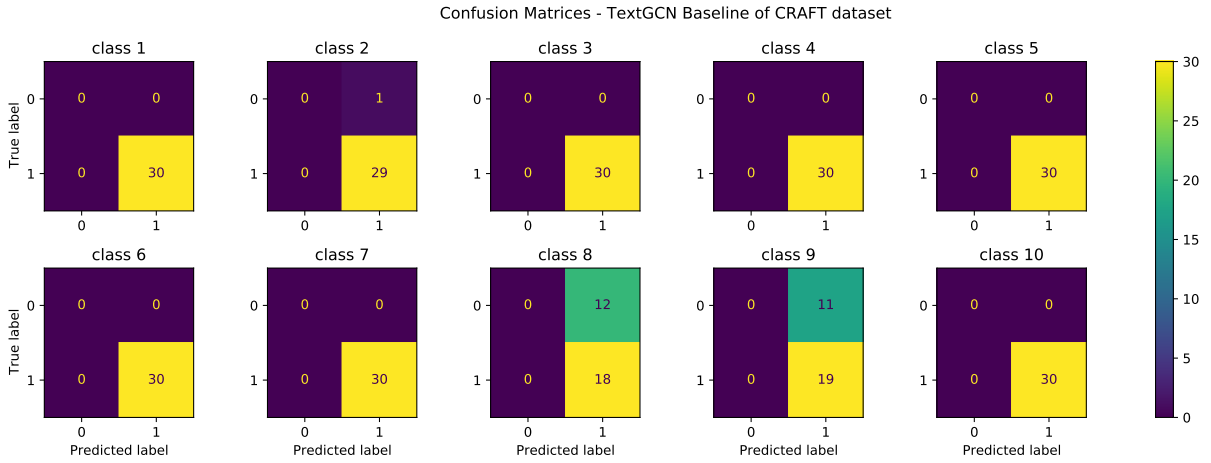

FIGURE 4.2: Confusion matrices for each class in the CRAFT dataset with the TextGCN Baseline model. Macro Scores: Precision 0.9200, Recall 1.000, F1 0.9509. Micro Scores: 0.9200, Recall 1.000, F1 0.9583.

4.2.2 CNN

TABLE 4.5: Results of CNN on Craft dataset. The following abbreviations were used: CNN: CNN Baseline, BioASQ: with pretrained BioASQ embeddings, UMLS: with pretrained UMLS embeddings, train: embedding trained with the model.

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
CNN	0.9200	1.0000	0.9509	0.9200	1.0000	0.9583
CNN_{BioASQ}	0.9221	1.0000	0.9525	0.9231	1.0000	0.9600
CNN_{BioASQ} + UMLS	0.9200	1.0000	0.9509	0.9200	1.0000	0.9583
CNN_{train} + UMLS	0.9232	0.9667	0.9407	0.9343	0.9783	0.9558

The results of using the CNN model with the CRAFT dataset are shown in table 4.5. The baseline CNN model gave a macro F1 score of 0.9509 and a micro F1 score of 0.9583, which is the same as the TextGCN model. Thus having the same problem of mapping all labels to all documents. Using the pretrained BioASQ embeddings did not change the results significantly. The word coverage for the CRAFT dataset with the BioASQ embeddings was 98.56 %. As with the Ohsumed dataset, pretrained UMLS embeddings were added. The word coverage of the UMLS embeddings was 20.70 %. The model with UMLS and BioASQ embedding resulted in the same F1 scores as the baseline. Further, an experiment was performed training the embeddings with the model and adding the UMLS embeddings called *BS + train, UMLS*. BS + train, UMLS model improved the precision but worsened the recall resulting in a worse F1-score. However, this means that the classification of true negatives improved slightly for this experiment.

4.2.3 TextGCN-CNN

The main results of the TextGCN-CNN model with the CRAFT dataset are shown in table 4.6. The baseline TextGCN-CNN model with the pretrained BioASQ embeddings as input for the CNN part gave a macro F1 score of 0.9466 and a micro F1 score of 0.9577. The recall is smaller than 1.000, but the precision is higher than 0.9200. Meaning that the number of predicted negatives increased which can be seen in figure 4.3. The true negative prediction of classes 8 and 9 increased, but the positive prediction also decreased. This means that in this experiment, not all labels were mapped to all documents. However, the prediction of negative classes is still difficult.

Adding the pretrained UMLS embeddings to the BioASQ embeddings improved the macro F1 score by 0.06 %. Resulting in mapping all labels to all documents. A bootstrap algorithm was applied to the two models to check if the difference is significant. It resulted in a p-value of 0.6968. Thus, the null hypothesis that the model with UMLS embeddings does not perform better than the baseline could not be rejected. Concluding that the observed differences explained beforehand was just coincidental.

TABLE 4.6: Results of TextGCN-CNN and additional information from ontologies on CRAFT dataset. BS + bioASQ: TextGCN-CNN with BioASQ embeddings, BS + BioASQ, UMLS: TextGCN-CNN, CNN with bioASQ and UMLS embeddings

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
BS_{BioASQ}	0.9236	0.9787	0.9466	0.9315	0.9855	0.9577
BS_{BioASQ} + UMLS	0.9200	1.0000	0.9509	0.9200	1.0000	0.9583

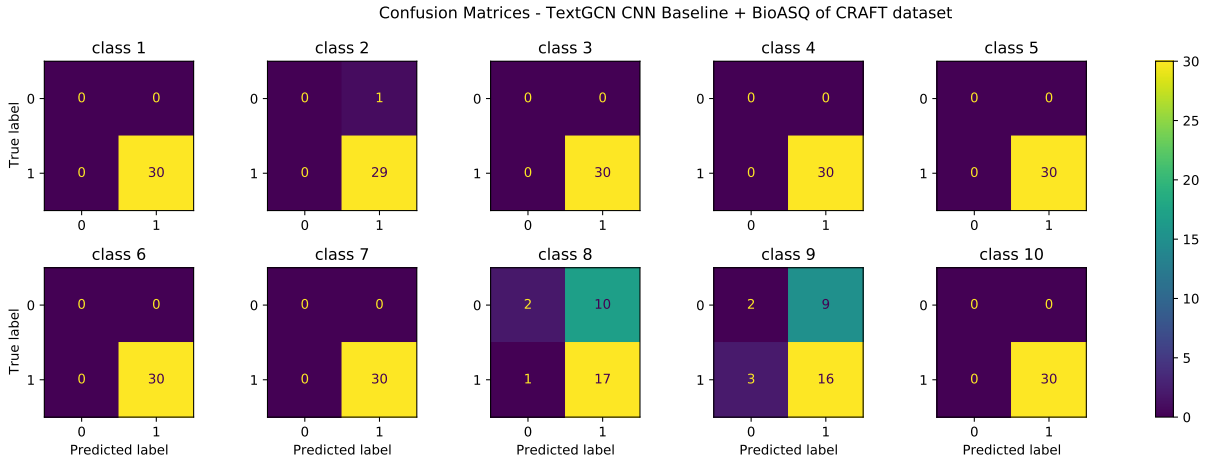

FIGURE 4.3: Confusion matrices for each class in the CRAFT dataset from the results of the TextGCN-CNN Baseline + BioASQ model. Macro Scores: Precision 0.9236, Recall 0.9787, F1 0.9466. Micro Scores: Precision 0.9315, Recall 0.9855, F1 0.9577.

4.3 MedOBO

The MedOBO corpus was used as a multi-label classification problem of 74 ontologies assigned to the documents. The following subsections report the performance of different models on the MedOBO dataset.

4.3.1 TextGCN

TABLE 4.7: Results of TextGCN and additional information from ontologies on MedOBO dataset. *Ontology* refers to the TextGCN with ontology connections. *All* refers to including all ontologies from the MedOBO development dataset. $\pm 1std$ refers to including only the ontologies in the range of $\pm 1std$.

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
TextGCN Baseline (BS)	0.2152	0.1614	0.1546	0.6426	0.7059	0.6727
BS + Ontology (all)	0.2092	0.1603	0.1543	0.6468	0.7031	0.6738
BS + Ontology ($\pm 1std$)	0.2215	0.1589	0.1539	0.6490	0.6998	0.6734

The results of the TextGCN model on the MedOBO dataset are shown in table 4.7. The baseline TextGCN model gave a micro F1 score of 0.6727 and a macro F1 score of 0.1546. These different results stem from the class imbalance in the dataset. Meaning that the reliability of classifying and ability to detect positive samples is quite good with a micro F1 score of 0.67. But the classes with little support do not perform well resulting in the macro F1 score of 0.15. This is also reflected in the F1 score weighted by support, which is 0.6339.

Adding all the ontologies from the development set as connections in the text graph gave an improvement of the micro F1 score by 0.11 %, but a decrease in the macro F1 score by 0.03%. To check if the improvement in the micro F1 score is reliable a bootstrap algorithm was performed. Resulting in a p-value of 0.0 for 1000 tests. The two compared models performed the same for all of the virtual test sets. This could be because the difference is very small, and thus only a few documents make the difference, it could happen that these documents were not sampled in the virtual tests. This could be changed by making more tests. Due to time constraints doing more tests was not possible. The results are still valid, as there was not any accidental advantage. Thus the difference observed for the original test set is not accidental. Further, an experiment was made using only the ontologies in the range of mean ±1 standard deviation (±1 std) using the label distribution to add word-ontology and document-ontology connections. Thus, ontologies labeled for only a few or the majority of documents are not used. The objective is to to provide less superfluous information and utilize just the essential ontologies. Adding only the ±1 std ontologies improved the BS model by 0.07 %. Thus only including the more prevalent ontologies performs worse than using all ontologies.

This could mean that using all ontologies gives information to the rare classes, thus improving the classification of the rare classes. An interesting addition for future work would be to include only the ontologies of the rare classes.

4.3.2 CNN

TABLE 4.8: Results of CNN on MedOBO dataset. CNN: CNN Baseline, BioASQ: with pretrained BioASQ embeddings, UMLS: with pretrained UMLS embeddings from 1: Maldonado, Yetisgen, and Harabagiu, 2019 or 2: Beam et al., 2019, train: embeddings trained with model.

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
CNN	0.1782	0.1368	0.1339	0.6387	0.6721	0.6550
CNN_{BioASQ}	0.1819	0.1208	0.1160	0.6835	0.6592	0.6711
CNN_{BioASQ} + UMLS 1	0.1819	0.1208	0.1160	0.6835	0.6592	0.6711
CNN_{BioASQ} + UMLS 2	0.1819	0.1208	0.1160	0.6835	0.6592	0.6711

The results of the CNN model on the MedOBO dataset are shown in table 4.7. The baseline model called *CNN* resulted in a macro F1 Score of 0.1339 and a micro F1 Score of 0.6550. Similar to the TextGCN model, there is a big difference between the micro and macro results caused by the class imbalance. Using the pretrained BioASQ embeddings instead of training the embeddings (*CNN + BioASQ*) gave an increase in the micro F1 score and a decrease in the macro F1 score. This means that the BioASQ embeddings improved the classification globally, but per class, it worsened the classification. The word coverage for the MedOBO dataset with the BioASQ embeddings was 93.9 %. Adding the pretrained UMLS embeddings from Maldonado, Yetisgen, and Harabagiu, 2019 (UMLS 1) and Beam et al., 2019 (UMLS 2) did not change the results. The word coverage for adding UMLS 1 was 14.6 % and for UMLS 2 was 6.7 %. Due to the low word coverage of the UMLS embeddings, the additional information from the UMLS probably did not improve the classification because of the high proportion of randomly initialized embeddings.

4.3.3 TextGCN-CNN

The results of the TextGCN-CNN model on the MedOBO dataset is shown in table 4.9. The baseline TextGCN-CNN model *BS* gave a macro F1 score of 0.1183 and a micro F1 score of 0.6637. Adding the pretrained BioASQ embeddings to the input of the CNN part (i.e., BS_{bioASQ}) improved both the macro and micro F1 score. Adding all the ontologies from the development set as connections in the text graph (i.e., *+ Ontology (all)*) gave an improvement of the micro F1 score for both BS and BS_{bioASQ}. But it decreased the macro F1 score of BS while improving it for BS_{bioASQ}. Further, an experiment with adding only the ontologies in the range of mean ± 1 standard deviation (± 1 *std*)

using the label distribution was conducted. The results show that using only ± 1 std ontologies is a little better than using all of the ontologies. All changes were trivial, hence the bootstrap test deemed unnecessary.

TABLE 4.9: Results of TextGCN-CNN and additional information from ontologies on MedOBO dataset. BS: TextGCN-CNN Baseline, BS_{bioASQ}: with pretrained BioASQ embeddings, UMLS: with pretrained UMLS embeddings from Maldonado, Yetisgen, and Harabagiu, 2019, Ontology: with ontology connections in the graph.

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
BS	0.1349	0.1354	0.1183	0.5909	0.6833	0.6337
BS + Ontology (all)	0.1279	0.1342	0.1173	0.5932	0.6815	0.6343
BS + Ontology ($\pm 1std$)	0.1334	0.1342	0.1174	0.5931	0.6819	0.6344
BS_{bioASQ}	0.2141	0.1245	0.1263	0.6877	0.6493	0.6679
BS_{bioASQ} + UMLS	0.1955	0.1336	0.1302	0.6575	0.6702	0.6638
BS_{bioASQ} + Ontology (all)	0.2060	0.1306	0.1326	0.6803	0.6564	0.6681
BS_{bioASQ} + Ontology ($\pm 1std$)	0.2078	0.1313	0.1334	0.6798	0.6575	0.6685

4.3.4 Comparing different architectures

The best results from all three model architectures TextGCN, CNN and TextGCN-CNN are shown in table 4.10. The new combined architecture did not improve either of the two separate architectures. All three architectures gave very similar F1 scores.

To establish a baseline for the MedOBO dataset, Aghaebrahimian, 2022 used a Naive Bayes (NB) model and a DL model. The NB model result is 0.73 micro F1-score. The DL model, where a bidirectional Gated Recurrent Unit (biGRU) C. a. O. Yu et al., 2019 was used, reached a micro F1-score of 0.75 and a macro F1-Score of 0.26. Both of these results were gained using all the MedOBO data. Thus using classification methods which are scalable to use more data, can learn with more data and thus perform better.

TABLE 4.10: Results of MedOBO dataset, comparing best result from different architectures. CNN refers to CNN + BioASQ, TextGCN refers to TextGCN + Ontology (all), TextGCN-CNN refers to BS_{bioASQ} Ontology ($\pm 1std$).

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
CNN	0.1819	0.1208	0.1160	0.6835	0.6592	0.6711
TextGCN	0.2092	0.1603	0.1543	0.6468	0.7031	0.6738
TextGCN-CNN	0.2078	0.1313	0.1334	0.6798	0.6575	0.6685

4.4 Answering the Research Questions

1. Does biomedical ontologies improve biomedical text classification?
   In some experiments, classification was improved by incorporating existing biomedical ontologies.

   (a) Does integrating ontologies with TextGCN improve biomedical text classification?
   The classification of the MedOBO dataset was improved by including biomedical ontology information. However, it did not improve the classification of the Ohsumed and CRAFT dataset.

   (b) Does integrating pretrained UMLS ontology embeddings with CNN improve biomedical text classification?
   Using the Ohsumed, MedOBO and CRAFT datasets, the use of the two different pretrained biomedical ontology embeddings from UMLS did not improve the classification.

2. Does a model architecture that combines CNN and TextGCN improve text classification?
   Using the Ohsumed, CRAFT, and MedOBO datasets with the new model architecture to classify documents did not improve the results.

3. Does the new model improve biomedical text classification for assigning ontologies to text?
   This question was answered using the CRAFT and MedOBO datasets for the classification task. Because these two datasets contain ontologies as classes. The Ohsumed dataset contains various cardiovascular diseases as classes assigned to the documents. Incorporating ontology information did improve the classification of the MedOBO dataset. However, it did not improve the classification of the CRAFT dataset.

Chapter 5

Conclusion and Outlook

5.1 Conclusion

In this thesis, the knowledge within ontologies in the biomedical field was utilized and incorporated into the biomedical text classification task. Further, a new DL architecture was proposed which combines the TextGCN and CNN architectures in a hybrid system. Three datasets Ohsumed, CRAFT and MedOBO were used to test the new approaches.

The incorporation of ontology information into the TextGCN model improved the classification of the MedOBO dataset compared to the baseline TextGCN model. There wasn't any improvement for the Ohsumed and CRAFT dataset. The MedOBO dataset is a larger dataset than Ohsumed or CRAFT, so the model may have been too complex for the smaller datasets. Further, all the documents in the Ohsumed dataset are related to cardiovascular issues, thus the assigned ontologies to all samples are very similar, hence ontological information has no distinctive power. The CRAFT dataset is a very small dataset with only 97 documents and 10 labels where most of the documents are assigned to all 10 classes. As a result, the model is rapidly overfitted, and more data does not improve the classification.

Incorporating ontology resources into a CNN model did not improve the classification of any of the datasets. The new proposed TextGCN-CNN architecture also did not improve the text classification for any dataset. Although, combining CNN with TextGCN in TextGCN-CNN enhances the speed and efficiency of the training. Zeng et al., 2022 very recently proposed an ensemble architecture with TextGCN and CNN, which did improve the classification. Thus, there is a possibility to improve the results by combining the two architectures in a very specific way, such that they can build on each other.

Concluding that the incorporation of biomedical ontologies has the potential to improve biomedical text classification and that more research on this topic is highly required.

5.2 Outlook

There are many different aspects that could be further explored, especially in terms of datasets, ontology integration, and the model architecture. The datasets which were used are unbalanced,

which is a common problem with multi-class and multi-label datasets. Up-sampling or down-sampling could be applied to multi-class datasets. Algorithms like LP-ROS (Charte et al., 2013) could be applied to the multi-label datasets, to improve the imbalance in the dataset. LP-ROS is a random oversampling algorithm designed for multi-label datasets.

One of the main problems with incorporating ontology information into the models was the mapping of the terms to the articles. An improvement could be to use MetaMap (Aronson, 2001), which is an application for finding UMLS concepts in biomedical articles. This could increase the connections from articles to ontologies in the TextGCN graph or increase the coverage of embeddings used in the CNN model.

Furthermore, methods such as Node2vec could be used to embed ontologies such as GO, CHEBI or NCBI to estimate embeddings for each ontology and use these embeddings instead of the pretrained UMLS embeddings. The classification could be improved by employing more appropriate losses for imbalanced data such as the weighted losses. Finally, the incorporation of ontology information could also be researched with different neural network architectures, such as BiGRU, Transformers or different graph-based approaches.

Bibliography

Yao, Liang, Chengsheng Mao, and Yuan Luo (Nov. 2018). *Graph Convolutional Networks for Text Classification*. arXiv: 1809.05679 [cs].

Kim, Yoon (Oct. 2014). "Convolutional Neural Networks for Sentence Classification". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1746–1751. DOI: 10.3115/v1/D14-1181.

Vaswani, Ashish et al. (Dec. 2017). *Attention Is All You Need*. arXiv: 1706.03762 [cs].

Neli Arabadzhieva - Kalcheva and Ivelin Kovachev (June 2022). "Comparison of BERT and XLNet Accuracy with Classical Methods and Algorithms in Text Classification". In: *2021 International Conference on Biomedical Innovations and Applications (BIA)*. Vol. 1, pp. 74–76. DOI: 10.1109/BIA52594.2022.9831281.

Murphy, Kevin P. (2012). *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning Series. Cambridge, MA: MIT Press. ISBN: 978-0-262-01802-9.

Raschka, Sebastian (Feb. 2017). *Naive Bayes and Text Classification I - Introduction and Theory*. DOI: 10.48550/arXiv.1410.5329. arXiv: 1410.5329 [cs].

McCallum, Andrew and Kamal Nigam (1998). "A Comparison of Event Models for Naive Bayes Text Classification". In.

Kalcheva, Neli, Milena Karova, and Ivaylo Penev (Oct. 2020). "Comparison of the Accuracy and the Execution Time of Classification Algorithms for Bulgarian Literary Works". In: *2020 International Conference Automatics and Informatics (ICAI)*, pp. 1–5. DOI: 10.1109/ICAI50593.2020.9311373.

Devlin, Jacob et al. (May 2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. DOI: 10.48550/arXiv.1810.04805. arXiv: 1810.04805 [cs].

Yang, Zhilin et al. (Jan. 2020). *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. DOI: 10.48550/arXiv.1906.08237. arXiv: 1906.08237 [cs].

Jang, Beakcheol et al. (Jan. 2020). "Bi-LSTM Model to Increase Accuracy in Text Classification: Combining Word2vec CNN and Attention Mechanism". In: *Applied Sciences* 10.17, p. 5841. ISSN: 2076-3417. DOI: 10.3390/app10175841.

Liu, Gang and Jiabao Guo (Apr. 2019). "Bidirectional LSTM with Attention Mechanism and Convolutional Layer for Text Classification". In: *Neurocomputing* 337, pp. 325–338. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2019.01.078.

Yao, Liang, Chengsheng Mao, and Yuan Luo (July 2019). "Graph Convolutional Networks for Text Classification". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01, pp. 7370–7377. ISSN: 2374-3468. DOI: 10.1609/aaai.v33i01.33017370.

Lu, Zhibin, Pan Du, and Jian-Yun Nie (2020). “VGCN-BERT: Augmenting BERT with Graph Embedding for Text Classification”. In: *Advances in Information Retrieval*. Ed. by Joemon M. Jose et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 369–382. ISBN: 978-3-030-45439-5. DOI: 10.1007/978-3-030-45439-5_25.

Balabin, Helena et al. (Mar. 2022). “STonKGs: A Sophisticated Transformer Trained on Biomedical Text and Knowledge Graphs”. In: *Bioinformatics* 38.6, pp. 1648–1656. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btac001.

Grover, Aditya and Jure Leskovec (Aug. 2016). “Node2vec: Scalable Feature Learning for Networks”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’16. New York, NY, USA: Association for Computing Machinery, pp. 855–864. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939754.

Du, Jingcheng et al. (Nov. 2019). “ML-Net: Multi-Label Classification of Biomedical Texts with Deep Neural Networks”. In: *Journal of the American Medical Informatics Association* 26.11, pp. 1279–1285. ISSN: 1527-974X. DOI: 10.1093/jamia/ocz085.

Ibrahim, Muhammad Ali et al. (Apr. 2021). “GHS-NET a Generic Hybridized Shallow Neural Network for Multi-Label Biomedical Text Classification”. In: *Journal of Biomedical Informatics* 116, p. 103699. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2021.103699.

Ahmed, Nizar, Fatih Dilmaç, and Adil Alpkocak (Oct. 2020). “Classification of Biomedical Texts for Cardiovascular Diseases with Deep Neural Network Using a Weighted Feature Representation Method”. In: *Healthcare (Basel, Switzerland)* 8.4, E392. ISSN: 2227-9032. DOI: 10.3390/healthcare8040392.

Li, Pengyuan et al. (July 2021). “Utilizing Image and Caption Information for Biomedical Document Classification”. In: *Bioinformatics* 37.Supplement_1, pp. i468–i476. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btab331.

Chen, Qingyu et al. (2022). “LitMC-BERT: Transformer-Based Multi-Label Classification of Biomedical Literature with an Application on COVID-19 Literature Curation”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–1. ISSN: 1545-5963, 1557-9964, 2374-0043. DOI: 10.1109/TCBB.2022.3173562.

Oliveira Gonçalves, Carlos Adriano et al. (Jan. 2021). “Classification of Full Text Biomedical Documents: Sections Importance Assessment”. In: *Applied Sciences* 11.6, p. 2674. ISSN: 2076-3417. DOI: 10.3390/app11062674.

Slater, Luke T. et al. (July 2020). *Improved Characterisation of Clinical Text through Ontology-Based Vocabulary Expansion*. DOI: 10.1101/2020.07.10.197541.

Koutsomitropoulos, Dimitrios A., Andreas D. Andriopoulos, and Spiridon D. Likothanassis (Dec. 2020). “Semantic Classification and Indexing of Open Educational Resources with Word Embeddings and Ontologies”. In: *Cybernetics and Information Technologies* 20.5, pp. 95–116. DOI: 10.2478/cait-2020-0043.

Le, Quoc V. and Tomas Mikolov (May 2014). *Distributed Representations of Sentences and Documents*. arXiv: 1405.4053 [cs].

Denecke, Kerstin (2022). “Does Enrichment of Clinical Texts by Ontology Concepts Increases Classification Accuracy?” In: *MEDINFO 2021: One World, One Health – Global Partnership for Digital Innovation*, pp. 602–606. DOI: 10.3233/SHTI220148.

Bodenreider, O. (Jan. 2004). “The Unified Medical Language System (UMLS): Integrating Biomedical Terminology”. In: *Nucleic Acids Research* 32.90001, pp. 267D–270. ISSN: 1362-4962. DOI: 10.1093/nar/gkh061.

Shanavas, Niloofer et al. (July 2020). “Ontology-Based Enriched Concept Graphs for Medical Document Classification”. In: *Information Sciences* 525, pp. 172–181. ISSN: 0020-0255. DOI: 10.1016/j.ins.2020.03.006.

Willis, Craig and Robert M. Losee (2013). “A Random Walk on an Ontology: Using Thesaurus Structure for Automatic Subject Indexing”. In: *Journal of the American Society for Information Science and Technology* 64.7, pp. 1330–1344. ISSN: 1532-2890. DOI: 10.1002/asi.22853.

Sanchez-Pi, Nayat, Luis Martí, and Ana Cristina Bicharra Garcia (Sept. 2016). “Improving Ontology-Based Text Classification: An Occupational Health and Security Application”. In: *Journal of Applied Logic*. SOCO13 17, pp. 48–58. ISSN: 1570-8683. DOI: 10.1016/j.jal.2015.09.008.

Koutsomitropoulos, Dimitrios A. and Andreas D. Andriopoulos (2022). “Thesaurus-Based Word Embeddings for Automated Biomedical Literature Classification”. In: *Neural Computing & Applications* 34.2, pp. 937–950. ISSN: 0941-0643. DOI: 10.1007/s00521-021-06053-z.

Peters, Matthew E. et al. (June 2018). “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202.

Medicine (U.S.), National Library of (1960). *Medical Subject Headings: Main Headings, Subheadings and Cross References Used in the Index Medicus and the National Library of Medicine Catalog*. U.S. Department of Health, Education, and Welfare, Public Health Service.

Xu, Bo et al. (2018). “Leveraging Biomedical Resources in Bi-LSTM for Drug-Drug Interaction Extraction”. In: *IEEE Access* 6, pp. 33432–33439. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2845840.

Lou, Peiliang et al. (2020). “A Representation Model for Biological Entities by Fusing Structured Axioms with Unstructured Texts.” In: *Bioinformatics* 37.8, pp. 1156–1163. DOI: 10.1093/bioinformatics/btaa913.

Maldonado, Ramon, Travis R. Goodwin, et al. (2017). “Deep Learning Meets Biomedical Ontologies: Knowledge Embeddings for Epilepsy”. In: *AMIA ... Annual Symposium proceedings. AMIA Symposium* 2017, pp. 1233–1242. ISSN: 1942-597X.

Arbabi, Aryan et al. (May 2019). “Identifying Clinical Terms in Medical Text Using Ontology-Guided Machine Learning”. In: *JMIR Medical Informatics* 7.2, e12596. DOI: 10.2196/12596.

Maldonado, Ramon, Meliha Yetisgen, and Sanda M. Harabagiu (May 2019). “Adversarial Learning of Knowledge Embeddings for the Unified Medical Language System”. In: *AMIA Summits on Translational Science Proceedings* 2019, pp. 543–552. ISSN: 2153-4063.

Kulmanov, Maxat, Fatima Zohra Smaili, et al. (May 2020). *Machine Learning with Biomedical Ontologies*. Preprint. Bioinformatics. DOI: 10.1101/2020.05.07.082164.

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena (Aug. 2014). "DeepWalk: Online Learning of Social Representations". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. New York, NY, USA: Association for Computing Machinery, pp. 701–710. ISBN: 978-1-4503-2956-9. DOI: 10.1145/2623330.2623732.

Mikolov, Tomas, Ilya Sutskever, et al. (2013). "Distributed Representations of Words and Phrases and Their Compositionality". In: *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc.

Mikolov, Tomas, Kai Chen, et al. (Sept. 2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv: 1301.3781 [cs].

Bordes, Antoine et al. (2013). "Translating Embeddings for Modeling Multi-relational Data". In: *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc.

Wang, Zhen et al. (June 2014). "Knowledge Graph Embedding by Translating on Hyperplanes". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 28.1. ISSN: 2374-3468. DOI: 10.1609/aaai.v28i1.8870.

Smaili, Fatima Zohra, Xin Gao, and Robert Hoehndorf (July 2018). "Onto2Vec: Joint Vector-Based Representation of Biological Entities and Their Ontology-Based Annotations". In: *Bioinformatics* 34.13, pp. i52–i60. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty259.

— (June 2019). "OPA2Vec: Combining Formal and Informal Content of Biomedical Ontologies to Improve Similarity-Based Prediction". In: *Bioinformatics* 35.12, pp. 2133–2140. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty933.

Kulmanov, Maxat, Wang Liu-Wei, et al. (Aug. 2019). "EL Embeddings: Geometric Construction of Models for the Description Logic EL++". In: pp. 6103–6109. DOI: 10.24963/ijcai.2019/845.

Edera, Alejandro A, Diego H Milone, and Georgina Stegmayer (Mar. 2022). "Anc2vec: Embedding Gene Ontology Terms by Preserving Ancestors Relationships". In: *Briefings in Bioinformatics* 23.2, bbac003. ISSN: 1467-5463, 1477-4054. DOI: 10.1093/bib/bbac003.

Alshahrani, Mona, Maha A. Thafar, and Magbubah Essack (Feb. 2021). "Application and Evaluation of Knowledge Graph Embeddings in Biomedical Data". In: *PeerJ Computer Science* 7, e341. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.341.

Zhong, Xiaoshi and Jagath C. Rajapakse (Dec. 2020). "Graph Embeddings on Gene Ontology Annotations for Protein–Protein Interaction Prediction". In: *BMC Bioinformatics* 21.Suppl 16, p. 560. ISSN: 1471-2105. DOI: 10.1186/s12859-020-03816-8.

Du, Jianzong et al. (Nov. 2021). "Graph Embedding Based Novel Gene Discovery Associated With Diabetes Mellitus". In: *Frontiers in Genetics* 12, p. 779186. ISSN: 1664-8021. DOI: 10.3389/fgene.2021.779186.

Yue, Xiang et al. (Feb. 2020). "Graph Embedding on Biomedical Networks: Methods, Applications and Evaluations". In: *Bioinformatics* 36.4, pp. 1241–1251. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz718.

Dash, Sabyasachi et al. (June 2019). “Big Data in Healthcare: Management, Analysis and Future Prospects”. In: *Journal of Big Data* 6.1, p. 54. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0217-0.

Locke, Saskia et al. (June 2021). “Natural Language Processing in Medicine: A Review”. In: *Trends in Anaesthesia and Critical Care* 38, pp. 4–9. ISSN: 2210-8440. DOI: 10.1016/j.tacc.2021.02.007.

Cohen, Kevin Bretonnel and Dina Demner-Fushman, eds. (2014). *Biomedical Natural Language Processing*. Natural Language Processing (NLP) volume 11. Amsterdam ; Philadelphia: John Benjamins Publishing Company.

Zhang, Yijia et al. (May 2018). “A Hybrid Model Based on Neural Networks for Biomedical Relation Extraction”. In: *Journal of Biomedical Informatics* 81, pp. 83–92. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2018.03.011.

Bhasuran, Balu and Jeyakumar Natarajan (July 2018). “Automatic Extraction of Gene-Disease Associations from Literature Using Joint Ensemble Learning”. In: *PLoS ONE* 13.7, e0200699. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0200699.

Cejuela, Juan Miguel et al. (Jan. 2018). “LocText: Relation Extraction of Protein Localizations to Assist Database Curation”. In: *BMC Bioinformatics* 19.1, p. 15. ISSN: 1471-2105. DOI: 10.1186/s12859-018-2021-9.

Chiang, Jung-Hsien and Hsu-Chun Yu (Aug. 2005). “Literature Extraction of Protein Functions Using Sentence Pattern Mining”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.8, pp. 1088–1098. ISSN: 1558-2191. DOI: 10.1109/TKDE.2005.132.

Minaee, Shervin et al. (Apr. 2022). “Deep Learning–based Text Classification: A Comprehensive Review”. In: *ACM Computing Surveys* 54.3, pp. 1–40. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3439726.

Chiu, Billy and Simon Baker (Dec. 2020). “Word Embeddings for Biomedical Natural Language Processing: A Survey”. In: *Language and Linguistics Compass* 14.12. ISSN: 1749-818X, 1749-818X. DOI: 10.1111/lnc3.12402.

Khattak, Faiza Khan et al. (Jan. 2019). “A Survey of Word Embeddings for Clinical Text”. In: *Journal of Biomedical Informatics*. Articles Initially Published in Journal of Biomedical Informatics: X 1-4, 2019 100, p. 100057. ISSN: 1532-0464. DOI: 10.1016/j.yjbinx.2019.100057.

Kowsari, Kamran et al. (Apr. 2019). “Text Classification Algorithms: A Survey”. In: *Information* 10.4, p. 150. ISSN: 2078-2489. DOI: 10.3390/info10040150.

Liu, Yong et al. (May 2020). “A New Feature Selection Method for Text Classification Based on Independent Feature Space Search”. In: *Mathematical Problems in Engineering* 2020, e6076272. ISSN: 1024-123X. DOI: 10.1155/2020/6076272.

Luhn, H. P. (Oct. 1957). “A Statistical Approach to Mechanized Encoding and Searching of Literary Information”. In: *IBM Journal of Research and Development* 1.4, pp. 309–317. ISSN: 0018-8646, 0018-8646. DOI: 10.1147/rd.14.0309.

Jones, Karen Spärck (1972). “A Statistical Interpretation of Term Specificity and Its Application in Retrieval”. In: *Journal of Documentation* 28, pp. 11–21.

Jurafsky, Daniel and James H. Martin, eds. (Dec. 2020). *Speech and Language Processing*. Third Edition Draft.

Bojanowski, Piotr et al. (June 2017). *Enriching Word Vectors with Subword Information*. DOI: 10.48550/arXiv.1607.04606. arXiv: 1607.04606 [cs].

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

Skansi, Sandro (2018). *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Undergraduate Topics in Computer Science. Cham: Springer International Publishing. DOI: 10.1007/978-3-319-73004-2.

Choi, Rene Y. et al. (Feb. 2020). “Introduction to Machine Learning, Neural Networks, and Deep Learning”. In: *Translational Vision Science & Technology* 9.2, p. 14. ISSN: 2164-2591. DOI: 10.1167/tvst.9.2.14.

Kingma, Diederik P. and Jimmy Ba (Jan. 2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs].

Hoerl, Arthur E. and Robert W. Kennard (Feb. 1970). “Ridge Regression: Biased Estimation for Nonorthogonal Problems”. In: *Technometrics* 12.1, pp. 55–67. ISSN: 0040-1706. DOI: 10.1080/00401706.1970.10488634.

Iyyer, Mohit et al. (2015). “Deep Unordered Composition Rivals Syntactic Methods for Text Classification”. 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)*. Beijing, China: Association for Computational Linguistics, pp. 1681–1691. DOI: 10.3115/v1/P15-1162.

Wu, Zonghan et al. (Jan. 2021). “A Comprehensive Survey on Graph Neural Networks”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1, pp. 4–24. ISSN: 2162-2388. DOI: 10.1109/TNNLS.2020.2978386.

Zhou, Jie et al. (Jan. 2020). “Graph Neural Networks: A Review of Methods and Applications”. In: *AI Open* 1, pp. 57–81. ISSN: 2666-6510. DOI: 10.1016/j.aiopen.2021.01.001.

Carbon, Seth and Chris Mungall (July 2018). *Gene Ontology Data Archive*. DOI: 10.5281/zenodo.7504797.

Borst, Willem Nico and W. N. Borst (Sept. 1997). “Construction of Engineering Ontologies for Knowledge Sharing and Reuse”. In.

“OWL 2 Web Ontology Language Document Overview (Second Edition)” (2012). In: p. 7.

Wu, Hongyan and Atsuko Yamaguchi (Aug. 2014). “Semantic Web Technologies for the Big Data in Life Sciences”. In: *Bioscience Trends* 8.4, pp. 192–201. ISSN: 1881-7823. DOI: 10.5582/bst.2014.01048.

Smith, Barry et al. (Nov. 2007). “The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration”. In: *Nature Biotechnology* 25.11, pp. 1251–1255. ISSN: 1546-1696. DOI: 10.1038/nbt1346.

Moreira, Dilvan A. and Mark A. Musen (July 2007). “OBO to OWL: A Protege OWL Tab to Read/Save OBO Ontologies”. In: *Bioinformatics (Oxford, England)* 23.14, pp. 1868–1870. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btm258.

Tirmizi, Syed Hamid et al. (Mar. 2011). “Mapping between the OBO and OWL Ontology Languages”. In: *Journal of Biomedical Semantics* 2 Suppl 1, S3. ISSN: 2041-1480. DOI: 10.1186/2041-1480-2-S1-S3.

Whetzel, Patricia L. et al. (July 2011). “BioPortal: Enhanced Functionality via New Web Services from the National Center for Biomedical Ontology to Access and Use Ontologies in Software Applications”. In: *Nucleic Acids Research* 39.Web Server issue, W541–W545. ISSN: 0305-1048. DOI: 10.1093/nar/gkr469.

Joachims, Thorsten (1998). “Text Categorization with Support Vector Machines: Learning with Many Relevant Features”. In: *Machine Learning: ECML-98*. Ed. by Jaime G. Carbonell et al. Vol. 1398. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 137–142. DOI: 10.1007/BFb0026683.

Cohen, K. Bretonnel et al. (2017). “The Colorado Richly Annotated Full Text (CRAFT) Corpus: Multi-Model Annotation in the Biomedical Domain”. In: *Handbook of Linguistic Annotation*. Ed. by Nancy Ide and James Pustejovsky. Dordrecht: Springer Netherlands, pp. 1379–1394. ISBN: 978-94-024-0881-2. DOI: 10.1007/978-94-024-0881-2_53.

Bird et al. (2009). *Natural Language Processing with Python*. O’Reilly Media Inc. ISBN: 9780596516499.

Developers, TensorFlow (May 2022). *TensorFlow*. Zenodo. DOI: 10.5281/zenodo.6574269.

Pavlopoulos, Ioannis, Aris Kosmopoulos, and Ion Androutsopoulos (Mar. 2014). “Continuous Space Word Vectors Obtained by Applying Word2Vec to Abstracts of Biomedical Articles”. In.

Beam, Andrew L. et al. (Aug. 2019). *Clinical Concept Embeddings Learned from Massive Sources of Multimodal Medical Data*. DOI: 10.48550/arXiv.1804.01486. arXiv: 1804.01486 [cs, stat].

Pedregosa, Fabian et al. (2011). “Scikit-Learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12.85, pp. 2825–2830. ISSN: 1533-7928.

Daniel Jurafsky and James H. Martin (Dec. 2020). *Speech and Language Processing*. Third Edition draft.

Zeng, Fang et al. (Dec. 2022). “Simplified-Boosting Ensemble Convolutional Network for Text Classification”. In: *Neural Processing Letters* 54.6, pp. 4971–4986. ISSN: 1573-773X. DOI: 10.1007/s11063-022-10843-4.

Aghaebrahimian, Ahmad (July 2022). *GitHub - Acg-Team/MEDOBO*. github.com/acg-team/MEDOBO.

Yu, C. a. O. et al. (June 2019). “BGRU: New Method of Chinese Text Sentiment Analysis”. In: *Journal of Frontiers of Computer Science & Technology* 13.6, p. 973. ISSN: 1673-9418. DOI: 10.3778/j.issn.1673-9418.1806018.

Charte, Francisco et al. (2013). “A First Approach to Deal with Imbalance in Multi-label Datasets”. In: *Hybrid Artificial Intelligent Systems*. Ed. by Jeng-Shyang Pan et al. Lecture Notes in Computer

Science. Berlin, Heidelberg: Springer, pp. 150–160. ISBN: 978-3-642-40846-5. DOI: 10.1007/978-3-642-40846-5_16.

Aronson, A. R. (2001). “Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program.” In: *Proceedings of the AMIA Symposium*, pp. 17–21. ISSN: 1531-605X.

List of Figures

List of Tables

Appendix A

Additional information on materials and methodology

A.1 Ohsumed

TABLE A.1: This table shows the categories associated to the id of the Ohsumed dataset.

Label	ID
Bacterial Infections and Mycoses	C01
Virus Diseases	C02
Parasitic Diseases	C03
Neoplasms	C04
Musculoskeletal Diseases	C05
Digestive System Diseases	C06
Stomatognathic Diseases	C07
Respiratory Tract Diseases	C08
Otorhinolaryngologic Diseases	C09
Nervous System Diseases	C10
Eye Diseases	C11
Urologic and Male Genital Diseases	C12
Female Genital Diseases and Pregnancy Complications	C13
Cardiovascular Diseases	C14
Hemic and Lymphatic Diseases	C15
Neonatal Diseases and Abnormalities	C16
Skin and Connective Tissue Diseases	C17
Nutritional and Metabolic Diseases	C18
Endocrine Diseases	C19
Immunologic Diseases	C20
Disorders of Environmental Origin	C21
Animal Diseases	C22
Pathological Conditions, Signs and Symptoms	C23

TABLE A.2: Ontologies from the OBO Foundry with highest number of active developers (Social), the underlined ID's represent the top 10 relevant ontologies.

ID	Title	Description	Social	OBO Format
doid	Human Disease Ontology	An ontology for describing the classification of human diseases organized by etiology.	279	Yes
hp	Human Phenotype Ontology	A structured and controlled vocabulary for the phenotypic features encountered in human hereditary and other disease.	210	yes
bfo	Basic Formal Ontology	The upper level ontology upon which OBO Foundry ontologies are built.	198	Yes
go	Gene Ontology	An ontology for describing the function of genes and gene products	165	yes
mondo	Mondo Disease Ontology	A global community effort to harmonize multiple disease resources to yield a coherent merged ontology.	146	yes
foodon	Food Ontology	A broadly scoped ontology representing entities which bear a "food role". It encompasses materials in natural ecosystems and agriculture tha...	129	No
envo	Environment Ontology	An ontology of environmental systems, components, and processes.	103	Yes
uberon	Uberon multi-species anatomy ontology	An integrated cross-species anatomy ontology covering animals and bridging multiple species-specific ontologies	100	yes
cl	Cell Ontology	The Cell Ontology is a structured controlled vocabulary for cell types in animals.	87	Yes
so	Sequence types and features ontology	A structured controlled vocabulary for sequence annotation, for the exchange of annotation data and for the description of sequence objects...	73	Yes
ro	Relation Ontology	Relationship types shared across multiple ontologies	65	Yes
upheno	Unified phenotype ontology (uPheno)	The uPheno ontology integrates multiple phenotype ontologies into a unified cross-species phenotype ontology	59	No
obi	Ontology for Biomedical Investigations	An integrated ontology for the description of life-science and clinical investigations	57	Yes
pato	Phenotype And Trait Ontology	An ontology of phenotypic qualities (properties, attributes or characteristics)	55	Yes

A.2 MedOBO

TABLE A.3: The 76 Ontologies from the sampled MedOBO 10k corpus.

Abbreviation	Description
TTO	Teleost taxonomy ontology
AISM	Ontology for the Anatomy of the Insect Skeleto Muscular system
XCO	Experimental condition ontology
LEPAO	Lepidoptera Anatomy Ontology
FBDV	Drosophila development
OBA	Ontology of Biological Attributes
PHIPO	Pathogen Host Interaction Phenotype Ontology
XLMOD	HUPO-PSI cross-linking and derivatization reagents controlled vocabulary
ONTOAVIDA	OntoAvida: ontology for Avida digital evolution platform
CDNO	Compositional Dietary Nutrition Ontology
FOVT	FuTRES Ontology of Vertebrate Traits
UNIT	Units of measurement ontology (UO)
GO	Gene Ontology
MA	Mouse adult gross anatomy
AMPHX	The Amphioxus Development and Anatomy Ontology
NCBITAXON	NCBI organism
ZECO	Zebrafish Experimental Conditions Ontologyal
SO	Sequence types and features ontology
EMAPA	Mouse Developmental Anatomy Ontology
PORO	Porifera Ontology
DDANAT	Dictyostelium discoideum anatomy
MONDO	Mondo Disease Ontology
SYMP	Symptom Ontology
ECTO	Environmental conditions, treatments and exposures ontology
BSPO	Biological Spatial Ontology
MMO	Measurement method ontology
DPO	Drosophila Phenotype Ontology
FBBT	Drosophila gross anatomy
ZFA	Zebrafish anatomy and development ontology
RO	Relation Ontology
VTO	Vertebrate Taxonomy Ontology
MAXO	Medical Action Ontology
GSSO	Gender, Sex, and Sexual Orientation (GSSO) ontology
PLANP	Planarian Phenotype Ontology
FBCV	FlyBase Controlled Vocabulary
BFO CLASSES ONLY	Basic Formal Ontology

TABLE A.3: The 76 Ontologies from the sampled MedOBO 10k corpus.

Abbreviation	Description
FYPO	Fission Yeast Phenotype Ontology
PSO	Plant Stress Ontology
NCIT	NCI Thesaurus OBO Edition
OBI	Ontology for Biomedical Investigations
FAO	Fungal gross anatomy
APO	Ascomycete phenotype ontology
PO	Plant Ontology
WBPHENOTYPE	C. elegans phenotype
HP	Human Phenotype Ontology
PR	PRotein Ontology (PRO)
XENOPUS ANATOMY	Xenopus Anatomy Ontology
COLAO	Coleoptera Anatomy Ontology (COLAO)
XPO	Xenopus Phenotype Ontology
PLANA	planaria-ontology
EXO	Exposure ontology
MCO	Microbial Conditions Ontology
MOP	Molecular Process Ontology
CLYH	Clytia hemisphaerica Development and Anatomy Ontology
ZFS	Zebrafish developmental stages ontology
DOID	Human Disease Ontology
MI	Molecular Interactions Controlled Vocabulary
MPHENO	Mammalian Phenotype Ontology (MP)
UBERON	Uberon multi-species anatomy ontology
OMP	Ontology of Microbial Phenotypes
ECAO	The Echinoderm Anatomy and Development Ontology
TAXRANK	Taxonomic rank vocabulary
WBBT	C. elegans Gross Anatomy Ontology
HAO	Hymenoptera Anatomy Ontology
CL	Cell Ontology
PSI-MS	Mass spectrometry ontology (MS)
CLAO	Collembola Anatomy Ontology
ENVO	Environment Ontology
ZP	Zebrafish Phenotype Ontology
RBO	Radiation Biology Ontology
SCDO	Sickle Cell Disease Ontology
PATO	Phenotype And Trait Ontology

TABLE A.3: The 76 Ontologies from the sampled MedOBO 10k corpus.

Abbreviation	Description
ECOCORE	An ontology of core ecological entities
CHEBI	Chemical Entities of Biological Interest
BTO	BRENDA tissue / enzyme source
CMO	Clinical measurement ontology

A.3 Evaluation

Specific equations used as evaluation measures for the mulit-label classification task.

$$precision_{micro} = \frac{\sum_{classes} TP(class)}{\sum_{classes}(TP(class) + FP(class))} \tag{A.1}$$

$$recall_{micro} = \frac{\sum_{classes} TP(class)}{(\sum_{classes} TP(class) + FN(class)} \tag{A.2}$$

$$F1_{micro} = 2 * \frac{precision_{micro} * recall_{micro}}{precision_{micro} + recall_{micro}} \tag{A.3}$$

$$precision_{macro} = \frac{\sum_{classes} precision(class)}{n(classes)} \tag{A.4}$$

$$recall_{macro} = \frac{\sum_{classes} recall(class)}{n(classes)} \tag{A.5}$$

$$F1_{macro} = \frac{\sum_{classes} F1(class)}{n(classes)} \tag{A.6}$$

$$precision_{weighted} = \sum_{classes} (\frac{support(class)}{\sum_{classes} support(class)} * precision(class)) \tag{A.7}$$

$$recall_{weighted} = \sum_{classes} (\frac{support(class)}{\sum_{classes} support(class)} * recall(class)) \tag{A.8}$$

$$F1_{weighted} = \sum_{classes} (\frac{support(class)}{\sum_{classes} support(class)} * F1(class)) \tag{A.9}$$

Appendix B

Additional results

B.1 Ohsumed

TABLE B.1: Results of TextGCN-CNN model on Ohsumed dataset. Different starting learning rates (lr) were tested. The code was corrected by adding a seed. The new results are called 'seed'.

Description	Accuracy
TextGCN-CNN Baseline, CNN with BioASQ embeddings (BS)	67.00 %
BS, lr = 0.001	67.7 %
BS, lr = 0.005	66.9 %
BS, lr = 0.010	68.4 %
BS, lr = 0.015	63.2 %
BS, lr = 0.020	65.7 %
BS seed, lr = 0.001	65.64%
BS seed, lr = 0.005	67.50%
BS seed, lr = 0.010	67.00%
BS seed, lr = 0.015	65.69%
BS seed, lr = 0.020	62.45%

Different starting learning rates for the Adam optimizer were tested as CNN uses 0.001 and TextGCN uses 0.02. Different values between 0.001 and 0.02 were tested, and 0.01 gave the best results. A code without seed was corrected, resulting in different results, where the learning rate 0.005 gave the best results and is 0.05 % better than the learning rate 0.01. As all the models were already trained as this was corrected, the further models were trained with a starting learning rate of 0.01.

B.2 MedOBO

TABLE B.2: Results of tuning hyperparameters of CNN on MedOBO dataset using the baseline model of CNN with pretrained BioASQ embeddings.

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
CNN_{BioASQ}	0.1915	0.0806	0.0867	0.7980	0.5392	0.6436
filter sizes "4,5,6"	0.1952	0.0791	0.0870	0.8106	0.5278	0.6393
filter sizes "2,3,4"	0.1554	0.0750	0.0804	0.8061	0.5210	0.6330
num filters '64'	0.1375	0.0764	0.0789	0.7928	0.5324	0.6370
dropout keep prob 0.25	0.1265	0.0706	0.072	0.7991	0.5141	0.6257
softmax threshold 0.4	0.1816	0.0993	0.1031	0.7478	0.5989	0.6651
softmax threshold 0.3	0.1819	0.1208	0.1160	0.6835	0.6592	0.6711
l2 reg lambda 0.01	0.1270	0.0726	0.0745	0.7947	0.5206	0.6291
l2 reg lambda 0.001	0.1717	0.0790	0.0844	0.7995	0.5359	0.6417
l2 reg lambda 0.0001	0.1659	0.0792	0.0845	0.7958	0.5369	0.6412

TABLE B.3: Results of tuning hyperparameters of TextGCN using the MedOBO dataset.

	Macro			Micro		
Description	Precision	Recall	F1 Score	Precision	Recall	F1 Score
TextGCN Baseline (BS)	0.2174	0.1049	0.1161	0.7596	0.5818	0.6589
BS, L2 reg = 0.1	0.0463	0.0528	0.0449	0.7422	0.4471	0.5580
BS, L2 reg = 0.01	0.0463	0.0528	0.0449	0.7422	0.4471	0.5580
BS, L2 reg = 0.001	0.0469	0.0529	0.0450	0.7424	0.4473	0.5582
BS, L2 reg = 0.0001	0.0391	0.0526	0.0443	0.7427	0.4461	0.5574
BS, dropout 0.25	0.0391	0.0526	0.0443	0.7427	0.4461	0.5574
BS, softmax threshold 0.4	0.0558	0.0803	0.0638	0.6433	0.5754	0.6075
BS, softmax threshold 0.3	0.2152	0.1614	0.1546	0.6426	0.7059	0.6727
BS, softmax threshold 0.2	0.1076	0.0942	0.0698	0.5849	0.6270	0.6052
BS, softmax threshold 0.1	0.1149	0.1286	0.0897	0.4870	0.7077	0.5770

Appendix C

Declaration of Originality

Zürcher Hochschule
für Angewandte Wissenschaften

Life Sciences und Facility Management

Education Unit

Statement of Authorship for Student Work at the School of Life Sciences and Facility Management

By submitting the enclosed

☐ Project
☐ Literature review
☐ Course work
☐ Minor paper
☐ Bachelor's thesis
☒ Master's thesis (tick as appropriate)

the student affirms independent completion of the(ir) work without outside help.

The undersigned student declares that all printed and electronic sources used are correctly identified in the text and in the bibliography, i.e. that the work does not contain any plagiarism (no parts that have been taken in part or in full from another's text or work without clear labelling and without citing the source).

In the event of misconduct of any kind, Paragraph 39 and Paragraph 40 of the General Academic Regulations for Bachelor's and Master's degree programmes at the Zurich University of Applied Sciences (dated 29 January 2008) and the provisions of the Disciplinary Measures of the University Regulations shall apply.

Location, date: Horgen, 02.02.2023 Student signature: Jasmin Faxer

Note on submitting the Statement of Authorship:

Direct submission of the work: This Statement of Authorship is to be inserted in the appendix of the ZHAW version of all work with original signatures and date (copies will not be accepted).

Submission of the work via Complesis: The Statement of Authorship should be made directly in Complesis by clicking as directed and should not be inserted in the appendix of the work.

Erlassverantwortliche/-r		LeiterIn Stabsbereich Bildung	Ablageort	2.05.00 Lehre Studium
Beschlussinstanz		LeiterIn Stab	Publikationsort	Public
Genehmigungsinstanz				
Version	Beschluss	Beschlussinstanz	Inkrafttreten	Beschreibung Änderung
1.0.0	15.03.2022	LeiterIn Stab	15.03.2022	Originalversion

Appendix D

Master's Thesis Topic, Form and Registration

N-FO-Master's Thesis Topic, Form and Registration

General Information		
Student name		Jasmin Saxer
Start date of studies		08.08.2021
Course type		☒ Full-time ☐ Part-time
Specialisation		☐ V1: Food and Beverage Innovation ☐ V2: Pharmaceutical Biotechnology ☐ V3: Chemistry for the Life Sciences ☒ V5: Applied Computational Life Sciences
Institute / workplace		ICLS
Thesis title		Ontology-Aware Biomedical Text Classification
Centre / group		ACGT
Confidential		Confidential storage/correction ☐ yes ☒ no Confidentiality agreement ☐ yes ☒ no Confidential poster * ☐ yes ☒ no * If the poster is not confidential, we will hang it up at your graduation ceremony.
V1, V2 or V3: **Timetable Master's Thesis (40 ECTS)**		**Milestone 1:** proposal / literature research (10 ECTS) ☐ AS ☐ SS **Milestone 2:** experimental strategy I (10 ECTS) ☐ AS ☐ SS **Milestone 3:** experimental strategy II (10 ECTS) ☐ AS ☐ SS **Milestone 4:** final conclusions (10 ECTS) ☐ AS ☐ SS
V5: **Timetable Master's Thesis (30 ECTS)**		**Milestone 1:** proposal / literature research (10 ECTS) ☒ AS 2022 ☐ SS **Milestone 2:** experimental strategy (10 ECTS) ☒ AS 2022 ☐ SS **Milestone 3:** final conclusions (10 ECTS) ☒ AS 2022 / 2023 ☐ SS
Thesis deadline (corresponds to Milestone 3 or 4)		**CW 2** ☒**, Year: 2023 Monday at 12:00 o'clock** (Admissions Office Grüental Campus) **CW 27** ☐**, Year:** **Friday at 12:00 o'clock** (Admissions Office Grüental Campus) Please note: the deadline can only be chan ged for justifiable reasons. A written request for extension must be sent to and approved by the programme director. The costs amount to a reduced semester fee (see brochure for Master's Thesis MSLS).
Thesis supervisors	1.	Zurich University of Applied Sciences Name: Aghaebrahimian Ahmad Address: ☐ Grüental ☐ Reidbach Postfach, 8820 Wädenswil Phone: 058 934 45 04 Email: agha@zhaw.ch
	2.	Name: Manuel Gil Address: Schloss 1, 8820 Wädenswil Phone: +41 (0) 58 934 57 44 Email: manuel.gil@zhaw.ch

Compensation for second supervisor (if external)	☐ yes ☐ no
Topic description	
Topic description • **Background** • **Objectives (for example planned experiments, investigations)** • **Equipment**	Text classification is an important topic in biomedical text analytics with numerous applications. Biomedical text classification can be improved by integrating ontologies. An Ontology is the model of a domain, which contains the terms, their descriptions, and their inter-relationships. The knowledge contained in an ontology can instruct the decisions made by machine learning algorithms via imposing explicit rules on the loss function, adding additional inputs, etc. One task in biomedical text classification is to assign relevant ontologies to a text. Knowing the target ontologies given a text helps other text analytics tasks such as Named Entity Recognition, Named Entity Linking, or Relation Extraction to make more informed decisions. The plan is to train an unsupervised neural network with the information of ontologies and the articles to classify PubMed articles. As a benchmark the Colorado Richly Annotated Full-Text (CRAFT) Corpus will be used. Which is a collection of 67 articles from the PubMed Central Open Access subset, where each has been annotated along a number of different axes spanning structural, coreference, and concept annotation. The equipment needed is access to the cluster and a PC.
Comments (for example necessary purchases, budget plan, additional conditions)	
General Terms and Conditions	
Formal requirements	In addition to a written composition, the following requirements must also be fulfilled, in accordance with the brochure for Master's Thesis MSLS (Merkblatt zur Masterarbeit): • Poster: as an alternative (agreed upon in writing with the supervisors) a web site or publication may be submitted. • Oral examination in form of a presentation of the work in a colloquium or to a panel composed of the partners involved: - the format will be determined by the supervisor; - the examination will take place until CW 04 or CW30 after the submission date; - the oral examination is not graded, it will be assessed as "pass" or "fail".
Important information and guidelines	The document Guidelines for writing semester project work, literature reviews, semester assignments, Bachelor's and Master's theses must be read. The requirements within the brochure for Master's Thesis MSLS must be fulfilled. Plagiarism violates copyright. Violations of copyright will be dealt with in accordance with subsection 39 of the General examination regulations for Bachelor and Master programmes at the Zurich University of Applied Sciences (Rahmenprüfungsordnung für Bachelor- und Masterstudiengänge der ZHAW), 29.01.08.
Grading deadline	3 weeks after the actual delivery of the Master's thesis, if it was submitted to the Master's Administration Office on time: CW30 (SS) / CW04 (AS).
Deadline topic form	The topic form must be delivered to the Administration Office the latest two weeks before the start of the semester in which the thesis is to be submitted.

Signature supervisor 1	**Signature student**
_____	_____
Place, date Wädenswil, 19.07.2022	Place, date Winterthur, 19.07.2022

Erlassverantwortliche/-r		Leiter/in Stabsbereich Bildung	Ablageort	2.05.00 Erlasse Lehre Studium
Beschlussinstanz		Leiter/in Stab	Publikationsort	Public
Genehmigungsinstanz				
Version	Beschluss	Beschlussinstanz	Inkrafttreten	Beschreibung Änderung
1.0.0	01.07.2021	Leiter/in Stab	01.07.2021	Überführung ins GPM
1.1.0	01.07.2021	LeiterIn Stab	01.07.2021	Inhaltliche Änderungen

Appendix E

Declaration of consent on the ZHAW Digitalcollection

Zürcher Hochschule
für Angewandte Wissenschaften

Life Sciences and Facility Management

Education Unit

Declaration of consent and release for the electronic publication of a Bachelor's/Master's thesis on the ZHAW Digital Collection at the Department of Life Sciences and Facility Management

Bachelor's / Master's thesis (tick as appropriate)

☐ Bachelor's thesis
☒ Master's thesis

Work title:	Ontology-Aware Biomedical Text Classification
Student name:	Jasmin Saxer
Name of 1st corrector:	Ahmad Aghaebrahimian

What keywords do you suggest for public online search?

Text Classification | Ontology | Graph convolutional networks

Convolutional neural networks

The student signing the form gives his/her consent to the electronic publication of the thesis in the ZHAW Digital Collection (based on § 16 para. 1 lit. B FaHG).

☒ The signing student agrees that

a) his/her work (full text) is published in digital form in the ZHAW Digital Collection and referenced in relevant directories (e.g. Google Scholar). The right to publish the work elsewhere is not affected by this declaration.
b) his/her work (full text) is published under the post-use licence granted by the Department.
c) the file is converted into other file formats or otherwise technically modified for the purpose of long-term availability.
d) the descriptive data as well as the work itself is permanently stored electronically and publicly accessible and can only be removed in case of infringement of third party rights.

The signing student assures that the publication of the thesis does not conflict with any rights of third parties, in particular with regard to illustrations contained in the full text or other content protected by copyright.

☐ The signing student does not consent to electronic publication.

Location, date: Horgen, 02.02.2023 Student signature: 

Zürcher Hochschule
für Angewandte Wissenschaften

Life Sciences and Facility Management

Education Unit

Note on submitting the declaration of consent and publication:

Direct submission of the work: This declaration of consent and publication must be inserted in the appendix of the ZHAW version of all student work with original signatures and date (copies will not be accepted).

Submission of the work via Complesis: The declaration of consent and publication should be made directly in Complesis by clicking as directed and should not be inserted in the appendix of the work.

Issuing authority		Head of Education	Storage location	2.05.00 Lehre Studium
Decision-making authority		Head of Education	Place of publication	Public
Approving authority				
Version	Decision	Decision-making authority	Valid from	Description Modification
1.0.0	13.12.2022	Head of Education	13.12.2022	Original