USING TRANSFORMER NETWORKS FOR DETECTION AND NORMALIZATION OF NAMED ENTITIES IN BIOMEDICAL TEXTS

by
İlkay Ramazan Pala
B.S., Computer Engineering, Boğaziçi University, 2018

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2021

ACKNOWLEDGEMENTS

Firstly, I would like to express my deep gratitude and admiration to my thesis advisor Prof. Dr. Arzucan Özgür, for her deep expertise, generosity, support and patience.

I want to thank Prof. Dr. Elif Özkırımlı for her valuable insights. I am also deeply thankful to members of the TABILAB, in particular Rıza Özçelik, Abdüllatif Köksal, Hilal Dönmez and Emrah Budur for their joyful collaboration and generous help in all aspects of thesis work.

I am grateful to members of the thesis jury, Prof. Dr. Suzan Üsküdarlı and Prof. Dr. Yusuf Yaslan who accepted to participate on short notice.

I would like to thank my family for their continuous encouragement and support throughout my academic journey and in my entire life. Finally I would like to thank my faithful friends for their friendship and support also during difficult times.

ABSTRACT

USING TRANSFORMER NETWORKS FOR DETECTION AND NORMALIZATION OF NAMED ENTITIES IN BIOMEDICAL TEXTS

The increasing difficulty of retrieving relevant information from rapidly growing literature has raised the interest for natural language processing (NLP) systems in the biomedical domain. In many of these systems, detection of named entities such as diseases, genes, and molecules (named entity recognition) and matching them to the corresponding entries in ontologies (normalization) are important intermediate steps. As these two tasks are related and datasets in this domain are relatively small, multi-task learning has been frequently used in the literature for this problem. Meanwhile, in recent years, the success of transformer-based pre-trained language models such as BERT in various NLP tasks has led them to be also applied in the biomedical domain. The different characteristics of biomedical text such as abbreviations and specific terminology motivated the development of new language models, which were trained specifically for this domain using a biomedical corpus. In this study, we propose a multi-task learning approach for named entity recognition and normalization by utilizing transformer-based pre-trained language models. To enable the optimal sharing of information, both tasks are formulated with text span embeddings obtained with a common encoder network. Promising results are obtained and compared with the results of state-of-the-art systems from the literature for commonly used named entity recognition datasets.

ÖZET

DÖNÜŞTÜRÜCÜ AĞLARI KULLANILARAK BİYOMEDİKAL METİNLERDE VARLIK İSİMLERİNİN TANINMASI VE NORMALİZASYONU

Çok hızlı ilerleyen literatür içinden ilgili yayınlara ulaşmanın gittikçe zorlaşması biyomedikal alanı hedefleyen doğal dil işleme sistemlerine olan ilgiyi arttırmaktadır. Bu sistemlerin önemli bir kısmında yayınlarda geçen hastalık, gen ve molekül adları gibi özel isimlerin tespit edilmesi (varlık ismi tanıma) ve ilgili ontolojilerdeki kayıtlarla eşlenmesi (normalizasyon) ara adımlarına ihtiyaç duyulmaktadır. Bu iki görevin birbiriyle yakın bir ilişki içinde olması ve bu alandaki veri kümelerinin küçük olması nedeniyle bu çoklu görev öğrenmesi literatürde sıkça başvurulan bir yaklaşım olmuştur. Bunun yanı sıra doğal dil işleme alanında son yıllarda BERT gibi dönüştürücü mimarisine dayanan önceden eğitilmiş dil modelleri kullanımıyla farklı görevlerde büyük başarılar elde edilmesiyle biyomedikal alanında da bu tarz modellerden faydalanılmaya başlanmıştır. Biyomedikal metinlerin kendilerine has bir terminolojiye sahip olmaları ve bu metinlerde kısaltmalara sıkça rastlanması gibi nedenlerle bu alana özgü dil modellerinin eğitilmesine ihtiyaç duyulmuştur. Bu çalışmada farklı biyomedikal veri kümeleri üzerinde transformer tabanlı dil modelleri ile çoklu görev öğrenmesi yaklaşımının etkili bir biçimde birlikte kullanılmasına çalışılmıştır. İki görev arasında bilgi paylaşımının en iyi düzeyde gerçekleşebilmesi için iki görevde de ortak bir ağ ile elde edilen metin aralıklarının vektör gösterimleri kullanılmıştır. Umut vadeden sonuçlar elde edilmiş ve sistemin performansı sıkça kullanılan veri kümeleri üzerinde literatürdeki en başarılı sistemlerle kıyaslanmıştır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . viii
LIST OF TABLES . . . ix
LIST OF SYMBOLS . . . x
LIST OF ACRONYMS/ABBREVIATIONS . . . xi
1. INTRODUCTION . . . 1
2. BACKGROUND . . . 7
  2.1. Named Entity Recognition . . . 7
  2.2. Named Entity Normalization . . . 7
  2.3. Sequence Tagging . . . 8
  2.4. Contrastive Learning . . . 9
  2.5. Pre-trained Language Models . . . 10
    2.5.1. BERT . . . 12
    2.5.2. BioBERT . . . 15
    2.5.3. SciBERT . . . 16
3. RELATED WORK . . . 18
  3.1. General Domain Named Entity Recognition and Normalization . . . 18
    3.1.1. Named Entity Recognition . . . 18
    3.1.2. Named Entity Normalization . . . 20
  3.2. Biomedical Named Entity Recognition and Normalization . . . 22
    3.2.1. General Overview . . . 22
    3.2.2. Joint Model of Zhao et al. . . . 23
    3.2.3. BioSyn . . . 24
4. METHODOLOGY . . . 29
  4.1. Sequence Tagging Approach . . . 29
  4.2. Span-based Approach . . . 33

4.2.1. Span-based NEN 33
4.2.2. Span-based NER 37
4.2.3. Pooling Functions 39
5. EXPERIMENTS AND RESULTS 41
5.1. Datasets 41
5.1.1. NCBI Disease Corpus 41
5.1.2. BioCreative V CDR Corpus 43
5.2. Results 45
5.2.1. Sequence Tagging Approach 45
5.2.2. Span-based NEN 48
5.2.3. Span-based NER 49
6. CONCLUSION 51
REFERENCES 52

LIST OF FIGURES

Figure 4.1. Outline of the BERT-based joint NER and NEN tagging model. . 33

Figure 4.2. Training pipeline of constrastive NEN model. 36

Figure 4.3. Example for creating span embeddings “ovarian cancer”. 40

Figure 5.1. An example article annotation from the NCBI disease dataset. . . 43

Figure 5.2. An example article annotation from the BC5CDR dataset. 44

LIST OF TABLES

Table 5.1. Statistics from datasets used in this work. 41

Table 5.2. An truncated disease example from the MEDIC dictionary. 43

Table 5.3. Results of the finetuned transformer models on the NCBI disease dataset. 46

Table 5.4. Results of the finetuned transformer models on the BC5CDR dataset. 46

Table 5.5. Results of the frozen transformer models on the NCBI disease dataset. 47

Table 5.6. Results of the frozen transformer models on the BC5CDR dataset. 47

Table 5.7. Comparison btw. Zhao et al. [1] and our BioBERT model. 48

Table 5.8. Accuracy scores of the top prediction of the NEN models. 49

Table 5.9. Accuracy scores of the top prediction of the span based NER models with different modeling choices. 50

LIST OF SYMBOLS

$Emb(s)$	Embedding or vector representation of mention s
$M(S, m, n)$	Match score for mention m from the sentence S and entity n
$\mathcal{L}$	Objective function
W	Maximum possible length of an entity mention
x_i	i'th token of a sentence
y_j^{NER}	NER label sequence of the j'th sample
y_j^{NEN}	NEN label sequence of the j'th sample
λ	Weighting scalar in the loss function
θ	Model parameters

LIST OF ACRONYMS/ABBREVIATIONS

CRF	Conditional Random Field
CUI	Concept Unique Identifier
LSTM	Long Short Term Memory Networks
NEN	Named Entity Normalization
NER	Named Entity Recognition
NLP	Natural Language Processing

1. INTRODUCTION

Information overload is considered to be one of the greatest problems for knowledge workers in today's age. For researchers, in particular, it is getting harder every year to keep up with relevant research and reach relevant information due to the ever-increasing number of researchers, journals, and published articles.

The same problem also applies in the biomedical field. In recent years, about two times every minute a new research article is included in the PubMed citation database [2], which currently contains about 32 million citations and was subject to 3.3 billion searches in the last year [3].

The situation is even worsened during times like the ongoing COVID-19 pandemic when it is critical to stay up-to-date with very fast-developing research to be able to make decisions under optimal circumstances. LitCovid [4], an effort to provide a daily updated hub for COVID-19 related literature in PubMed, is hosting 158576 articles as of today [5].

Named Entity Recognition (NER) and Named Entity Normalization (NEN) are among the most well-known and studied tasks in Natural Language Processing (NLP) because often achieving high accuracy in these tasks is crucial for many text mining systems where they are typically used in successive steps.

In NER, we aim to extract text spans that correspond to a real-word entity that is denoted by a proper name. For example, given the sentence "Ministry of Health announced the number of infected patients.", we expect to get "Ministry of Health" categorized as "Organization". In NEN or normalization, on the other hand, we would like to determine which particular entity or concept a given entity mention refers to. For example, in the sentence "Paris is the capital of France", the mention of "Paris" should be identified as Paris the city, not for example as the celebrity Paris Hilton.

Usually, the set of concepts that we want to map mentions to is given in the form of a dictionary, ontology, or knowledge graph. For this reason, NEN is also referred to as named entity linking (NEL) or named entity disambiguation (NED) in the literature.

Naturally, there has been great interest in both these tasks in the biomedical NLP domain, as many information retrieval and text mining systems require successful execution of these tasks in their intermediate steps. A non-comprehensive list is given below:

- **covidAsk** [6] A real-time question answering system for COVID-19 related questions. Broadly speaking, segments from COVID-19 related articles are embedded into vectors by a retrieval model trained for general-purpose question answering. From each research article appearing in the search results, they extract and normalize biomedical entities for enabling a quick overview of occurring entities, their descriptions, and synonyms.
- **VAPUR** [7] A search engine for finding related protein-chemical pairs from COVID-19 literature. Since biomedical compounds and proteins may occur in different articles in very different surface forms, traditional general-purpose systems may fail return relevant results. In VAPUR, an automatic bioNLP pipeline is employed to build an inverted index for protein-compound pairs. Entities are extracted and normalized from the COVID-related corpus and then a relation extraction model is used to find related protein-chemical pairs that occur in the same sentence. This way, when a compound such as "Favipiravir" is queried, via the inverted index the system returns all related proteins with corresponding articles in the literature.The performance of the system is measured through an evaluation by domain experts.
- **PKG** [8] (PubMed Knowledge Graph) To ease access to relevant information from PubMed, authors construct a knowledge graph from PubMed abstracts. To this end; authors with their affiliations and biomedical entities are extracted and entities or authors that correspond to the same real-world object are unified. They have demonstrated the usefulness of the knowledge graph with 3 examples. By

analyzing connected nodes of neurologist Stephen Silberstein they have identified collaboration patterns and key institutions. Observations of the CGRP molecule have shown interesting patterns in the change of interest in academic cycles with respect to other compounds. Finally, via an analysis of the bipartite author-entity network, they suggested collaboration opportunities between researchers who share a common interest.

While both NER and NEN are well-studied tasks in NLP, applying advances to the biomedical domain is not straightforward, as biomedical texts have characteristics such as domain-specific terminology and frequent use of abbreviations. In addition, biomedical NER datasets are usually much smaller than those studied in NLP. Because of this, often domain-specific approaches are required to obtain the best performance.

Named entity recognition and normalization tasks in the biomedical domain have been the subject of many studies in the literature. Earlier works were based on traditional machine learning algorithms with handcrafted features. For example, [9] used a joint CRF model with features like character n-grams, word stems, part-of-speech tags. As deep learning methods have become dominant in the NLP domain, state-of-the-art systems have adapted neural approaches. One example is [10], which used LSTM and word embeddings to improve upon existing models. More recently, following the trend in NLP, pre-trained language models have been applied to obtain state-of-the-art results. BioBERT [11]) was trained on 18B tokens from biomedical papers, whereas SciBERT [12]) was trained on a random collection of papers from Semantic Scholar making up 3.17B tokens.

As of now, the best results for the normalization task are achieved using BERT-based models in a contrastive learning fashion. In BioSyn [13]), entity mentions and their synonyms are embedded using a pre-trained BioBert model such that mentions mapping to the same entity have closer embeddings than non-synonyms. This allows the classification of unseen entities in the dataset as the embeddings are created from the surface representation of entity synonyms. SAPBERT [14]) improves upon BioSyn by pre-training the embedding model in a more general biomedical dictionary before the actual normalization dataset.

Several papers have explored the benefits of joint learning between these tasks. TaggerOne [9] is a transition-based model where states were the product of NER and NEN states. More recently, in [1], both tasks were learned together with a common encoder but two separate prediction heads. Furthermore, they have also shown that feeding the output of the NEN model as input to the NER model yielded even better performance. Collabonet [15], on the other hand, showed that combining models trained on datasets for different entity types (diseases, chemicals, species, etc.) can improve model performance on individual NER datasets for a specific entity type.

In this thesis, we tried to combine successful ideas from earlier works. As described earlier, in many papers multitask learning had improved the results. On the other hand, more recent studies had success with pre-trained transformer language models and contrastive learning for the NER task. Motivated by these works, we test three different hypotheses:

(i) Testing multitask learning between NER and NEN tasks together with transfer learning with a pre-trained language model. More specifically, we will replace word embeddings + LSTM networks in the classical sequence tagging approach of [1] with BERT-based transformer encoders.

(ii) Using sentence context with BERT models to learn mention embeddings in contrastive learning approach for normalization. We will the same learning objective as in BioSyn [13], but mention embeddings will be created with sentence context.

(iii) Finally, we will use a span classification approach for the NER task and train the joint encoder together with the normalization task described above. Since both tasks internally use span embeddings we hope that multitask learning will be useful in this setting too.

We hope to show that multitask learning can be still useful when used in conjunction with transfer learning. As biomedical NLP datasets are usually quite small, in general, multitask learning can be helpful as it provides more information. In this case, the normalization signal can help the model create a more informative entity representation and thus be more generalizable in NER. Also, the NER signal can help the model to better make use of sentence context, which could be harder for the model with the normalization task alone.

In this work, we propose to train a BERT-based model jointly on named entity recognition and normalization tasks. Both recognition and normalization tasks will be formulated via span representations instead of more usual sequence labeling approaches. A joint transformer encoder will be used to create embeddings for each span.

For normalization, we will use a similar approach to BioSyn. Synonym candidates will be drawn from the dictionary and based on embedding closeness the model will try to discriminate actual synonyms from others. However, unlike the BioSyn, the embedding for the entity mention will be created using the joint encoder. This allows the embeddings to capture contextual information. As mentioned in the BioSyn work, there are cases where identical mentions have been annotated with different entities. By extracting the mention representations from the sentence representation, we hope to handle such cases and even perhaps remove preprocessing steps like abbreviation resolution.

For the NER task, each span will be treated as a candidate entity and the model will try to predict whether a span forms an entity and of which class. Although this means $O(n^2)$ predictions, it can be done efficiently as we will explain in the methodology section.

The rest of this thesis is organized as follows: In Chapter 2, background information for NER and NEN tasks and several commonly used techniques will be given. In

Chapter 3, we will give an overview of related works in this area and describe several papers which directly inspired work in more detail. In Chapter 4, the methodology used in different settings will be discussed. Experiments and their results will be presented in Chapter 5 and the last chapter Chapter 6 will be reserved for discussion of the results and conclusion drawn from the study.

2. BACKGROUND

2.1. Named Entity Recognition

Named Entity Recognition (NER) is one of the most practically useful and therefore arguably one of the main tasks in all of the NLP domain. Given a text segment, typically a sentence, NER amounts to locating spans of tokens corresponding to entities of interest and classifying the extracted entity to a fixed set of broad types, such as PERSON, ORGANIZATION, LOCATION, etc. For example in the sentence "Merih Demiral was transferred to Juventus in 19/20 football season." the entities to be extracted could be the span "Merih Demiral" with type "PERSON", "Juventus" with type "ORGANIZATION" and "19/20 football season" with type "DATE". In biomedical NER datasets, entity types are usually diseases, chemicals, or species.

Many practical applications require NER as an intermediate step. For example, in information retrieval systems it is desirable to index occurrences of named entities. Automatic pipelines can be built via NER systems to extract relevant data from text, for example, to speed up customer support.

In NER datasets model performance is usually calculated with a span-based F1 score. In a given text, gold entities and predicted entities are contrasted. Predicted entities found in the gold entities set are treated as true positives, not found ones are counted as false positives. Similarly, gold entities not found in predicted entities are treated as false negatives. Note that two entity detections match only if both their spans and type match.

2.2. Named Entity Normalization

Named Entity Normalization (NEN), also referred to as Named Entity Disambiguation (NED) and Named Entity Linking (NEL) in the literature, is matching entity mentions found in a text to a unique identifier, which we will refer to as CUI (concept unique identifier) following [13]. Typically the set of unique identifiers to which the entities are matched to come from a knowledge base such as Wikidata or DBpedia. Therefore mentions are "linked" to the entries in the knowledge base which could be used for semantic annotation.

The difference between NER and NEN tasks is that in NER we are only concerned with detecting the mention from the text and classifying it into a broad entity type such as "DATE" or "ORGANIZATION". However, in NEN it is often assumed that the entity already has been found from the text and we are trying to "disambiguate" the found mention by assigning it to a very specific entity in the knowledge base. For example in the sentence "Jaguar has been making luxurious sedans and athletic sports cars for decades.", "Jaguar" refers to the automobile brand, not to the animal species. NEN can be seen as a fine-grained version of the NER, as we usually require not only a broad entity type such as PERSON, instead, we want to map the entity to a specific ontology entry that specifically refers to the person in question. Therefore we can say that NER and NEN tasks are closely related and complementary.

In the setting where entity mentions are not extracted, and the F1 score is calculated similarly to NER. However, in this case, we don't care about span matches and only use predicted CUI labels to find true positives, false positives, and false negatives. In the setting where entity mentions are already extracted, we measure accuracy in the top k entity predictions of the model.

2.3. Sequence Tagging

Sequence tagging defines a general approach that is applicable to a number of different NLP or other sequence learning tasks. Basically, given sequence of tokens, $[x_1, x_2, ..., x_n]$, each token will be classified (or "tagged") to a label class so that a sequence of labels of the same length will be obtained: $[y_1, y_2, ..., y_n]$.

While this approach is very simple, it can be used to formulate model different tasks. For example, in tasks like NER where we would like to extract spans with entity types from a sequence, sequence tagging can be used as follows:

- Tokens that are not inside an entity span will be assigned to outside label ("O" label).
- Tokens that are the first token of an entity span will be assigned to the begin label of entity type, e.g. "B-ORGANIZATION", "B-DATE" etc.
- Tokens that are part of an entity span but are not the initial token will be assigned to inside label of the entity type, e.g. "I-ORGANIZATION", "I-DATE" etc.

This way, any list of non-overlapping entity spans with types can be converted into unique a sequence of labels and vice versa. Since there are different tokens labels for start and other tokens of an entity type, there will not be any confusion about whether a sequence of labels with the same entity type is actually multiple separate entity spans of the same type. This particular encoding scheme is known as "IOB" encoding and there are also other encoding options like "IOBES" which also includes end and single token labels.

Sequence tagging is still the most commonly used formulation for NER to this day. Other tasks which are typically formulated as sequence tagging include part-of-speech tagging, semantic role labeling, and many others.

2.4. Contrastive Learning

In many machine learning tasks such as face recognition, there are thousands or millions of different labels encountered in the dataset. This leads to a data sparsity problem, as some of the label classes will contain very few samples in the training set. Also, in some settings, we may even want to handle label classes that are encountered at the training time, for example adding a new person to the face recognition system.

In such situations, contrastive learning can offer a remedy. In contrastive learning, the idea is to learn a mapping from samples to the embedding vector space, such that samples that have the same labels (similar or positive samples) have the close vectors and samples that have different labels (dissimilar or negative samples) have embeddings that are distant from each other. The network is trained to “contrast” similar and dissimilar samples.

Since the model never explicitly sees the labels during training, it can generalize to label classes not seen during training. For example, in face recognition, the model will hopefully embed two images of the same person into close vectors even if no image of that particular person was in the dataset.

Contrastive learning has many applications in different areas of machine learning. It is the de-facto approach for few-shot learning problems where only a few samples for each class is present in training dataset like facial recognition. It is also widely used for image retrieval and document retrieval (e.g. [16]), where the queries can not be known beforehand. It can be also adopted for pre-training purposes. For example in SimCLR [17], to learn image representations without labels, the model is trained to map augmentations of the image to close vectors in embeddings space and distant vectors otherwise. SimCLR had significantly improved the then state-of-the-art results in unsupervised object recognition.

2.5. Pre-trained Language Models

In recent years, there has been a huge surge of interest in pre-trained language models, which are now a vital part of almost all state-of-the-art systems in many tasks in the NLP domain, such as natural language inference, sentiment analysis, or question answering. In fact, the year 2018 has been described as “ImageNet moment” of NLP [18], during which many influential works using this methodology (e.g. ELMo [19], ULMFIT [20], GPT [21], BERT [22]) have been published.

Unlike traditional word vectors, which can be seen as shallow neural networks, these studies proposed transfer learning with deep networks with a high number of parameters.

Many of these models can be seen as “contextualized word embeddings”. In the classical word vector approach (for example word2vec [23]), a fixed representation for each word is learned during pre-training and these representations are independent of the context in which the word occurs. This causes a problem with words with multiple senses. Deep pre-trained language models don’t suffer from this problem, as they differentiate between senses or even refine the “meaning” of a word by processing the complex contextual information in their stacked layers.

Unlike in computer vision, where pre-training networks on a big labeled dataset such as ImageNet is a common practice, these networks are usually trained in an unsupervised way on large textual corpora, such as Wikipedia articles or book scans. This is very advantageous, as it is relatively straightforward to find large corpora for pre-training compared to more costly data annotation, for example by incorporating web crawls. This also makes them easy to apply to NLP tasks in other domains with smaller supervised datasets. For example to improve a named entity recognition system for the German language, one can pretrain a language model on German texts collected from the internet.

Another one of the promises of such networks is their wide applicability and re-usability. Even though it might be more costly to train these deeper unsupervised models on large corpora, it can be done once by perhaps a large company or institution with computational resources and shared with other practitioners to make use of these models in their own inquiries.

2.5.1. BERT

BERT [22], short for "Bidirectional Encoder Representations from Transformers", is arguably the most influential development in NLP in recent years. At the time of its release, it had simultaneously improved upon state-of-the-art systems in many tasks while adopting a unified approach in almost all of them.

The BERT model is based on the transformer architecture [24], which was originally proposed for neural machine translation. In contrast to CNN or LSTM networks, which are either based on convolution or recurrence operations, the transformer is based solely on attention mechanism. At each transformer layer, the representation of a token is obtained with a weighted sum of representations of other tokens in the sequence, where the weights are determined with a small network and depend on the queried token. This way, every token can directly influence every other token in the sequence in the next layer without requiring long recurrence or multiple layers. As transformer layers are not position-aware like LSTMs, position information is explicitly given by adding a learned position embedding to the initial token representation. Because of this, there is a maximum length of context which it can process, which was chosen to be 512 in this work.

Also, the model adopts a variant of subword tokenization, WordPiece, as in many of the previous works. From a large textual corpus, a subword vocabulary is learned and all word occurrences are expressed using subwords from this fixed subword vocabulary in a parsimonious way. For example the word "walking" may not be present in the subword dictionary, and will be instead segmented as "walk" + "###ing". This way, one can obtain still useful representations of words even though they don't occur in that specific form in the training set if they are composed of frequently seen subwords. This segmentation algorithm was found to be very beneficial for working with languages like Turkish and German where a specific occurrence of a word can be very sparse because of the composition rules of the language. Since these models are meant to be trained on large corpora, where many misspelling and obscure words can be found, this approach is particularly useful as otherwise many words will not be seen enough times in the training set to be able to learn meaningful word embeddings. Also, a reduction in vocabulary size results in a decrease in the number of parameters, thus a smaller memory footprint and faster execution. A subword vocabulary size of around 30.000 was chosen for the BERT model for a good efficiency/word coverage trade-off.

The main innovation in this work is the so-called “bidirectionality” of its token representations. Previously, language models were pre-trained in a left-to-right fashion, using the tokens that lie on the left side to predict the next token in the sequence. The token representations obtained from a language model trained this way will only contain information from the left context of the token, which is undesirable. Even though one can train a “bidirectional” language model via combining left-to-right and right-to-left representations to predict a token, as done in ELMo [19], it would be still suboptimal as information coming from both sides would not be fused until the last layer.

Another problem arises when we want to fine-tune a left-to-right language model directly on a downstream task instead of using its internal representations as features. For example, in GPT from OpenAI a left-to-right transformer was finetuned in several tasks via concatenating a linear layer on top and updating all parameters of the language model. However, since the language model was trained in a left-to-right way, it can only flow information from left-to-right in the finetuning phase too, which is very undesirable for tasks like natural language inference or question answering.

To overcome this problem the authors of BERT propose the MLM (masked language modeling) objective, also known as Cloze task in the literature. Some randomly chosen tokens in the original sentence or a longer sequence are hidden and the model tries to predict the missing tokens given the visible tokens in the context. The network is forced to make use of the full context from both sides to refine the representation of tokens so that the final hidden vectors for the masked tokens are representative enough to predict them. Since a transformer encoder allows direct interaction between every token pair via attention mechanism, tokens from the left and right context of the token to be predicted can interact in every layer without allowing the model to "cheat" as predicted token is only given as a mask token in the input. This allows the bidirectional flow of information throughout the network during pre-training. Since the interaction between the left and right context happens at every layer, the authors call BERT "deeply directional" as opposed to the "shallow bidirectionality" of ELMo.

Additionally, to encourage the model to output meaningful representations, not just for the masked tokens, the tokens randomly chosen for prediction are %80 of the time are masked, %15 of the time randomly replaced, and in the remaining %15 left as it is. This prevents the mismatch during training and finetuning, as otherwise during training the model would see only masked inputs and during finetuning only original sentences. This was shown to have a small positive influence on the results.

Since many important NLP tasks involve two sequences (e.g. question answering, natural language inference, etc.) the authors decided to incorporate such a task also to the pre-training scheme. To this end, they use NSP (Next Sentence Prediction) task, which is a binary classification task predicting whether two sentences directly follow one another in the corpus. Although in the original BERT paper this was claimed to be useful for tasks like question answering and natural language inference this was put into question in the later works [25]).

BERT was pre-trained on a combined dataset comprised of 800 million words BooksCorpus and English Wikipedia dump of size 2,500 million words. The authors also point out that it is important to use a dataset where we have access to full documents as only then we can ensure to have long sequences to fill up a context of 512 tokens.

At the time of writing BERT paper was cited 20571 times. Originally it had improved upon the previous best model %7.7 on the comprehensive GLUE benchmark. Because of its huge success in nearly all areas, it has been widely adopted and it has given rise to other transformer-based pre-trained models such as Roberta, Albert, BART, CTLR. Nowadays, all top submissions on GLUE and SuperGLUE benchmarks at least include a BERT-like subcomponent in their system, and in many tasks, they have even exceeded human performance.

2.5.2. BioBERT

As BERT has found success in so many different tasks, there have been naturally many efforts to apply it to other domains. Although we can assume that the BERT model has a good grasp of standard English text considering its strong performance on common English NLP tasks, it won't be as useful in the biomedical domain as in the biomedical domain there is a very specific terminology it uses. It is difficult to expect BERT to have a meaningful representation for words like "favipiravir" which occur rarely at best in the general domain corpus that BERT was trained on.

This has motivated researchers to pre-train BERT on large biomedical corpora to make them more effective for downstream tasks in this domain. Note that this means BioBERT uses the same WordPiece tokenizer as the original BERT model. In BioBERT [11], authors chose to continue pre-training of BERT on biomedical corpus which they have constructed using 2 different sources:

(i) PubMed abstracts (4.5 billion words)
(ii) PMC full-text articles (13.5 billion words).

They experimented with different pre-training setups: training for 270K steps (BioBERT v1.0) on (a) PubMed only, (b) PMC only, (c) PubMed and PMC, and (d) training for 1M steps on PubMed only (BioBERT v1.1).

They finetuned resulting pre-trained models and compared them to previous works on many different biomedical datasets for named entity recognition, relation extraction, and question answering. The results justified the in-domain training of BERT for the biomedical domain. On NER datasets for instance, while BERT was behind on state-of-the-art systems by %2.01 points on average, BioBERT models were the best performers in six out of nine cases.

2.5.3. SciBERT

SciBERT model was developed for similar motivations to BioBERT. However, instead of targeting the biomedical domain only, the authors wanted to train a common model which can be useful for NLP tasks in a variety of different scientific domains such as computer science and chemistry.

The pre-training corpus of SciBERT was constructed from full texts of 1.14 million research articles randomly chosen from Semantic Scholar. 18% of the articles were from the computer science domain and the other 82% were from the biomedical domain in a broad sense, making a 3.17 billion word dataset.

They try two different approaches for tokenizer vocabulary.

(i) BASEVOCAB - original vocabulary of the pre-trained BERT models is retained. This allows initializing the models from BERT checkpoints.
(ii) SCIVOCAB - the vocabulary is learned using the scientific corpus. In this setting, the models should be trained from scratch as their vocabulary would be mismatched to the original BERT.

In addition to different vocabulary setups, the authors also experiment with different transfer learning methods. Besides the more common finetuning approach, in which all parameters of the pre-trained language model are updated during training, they also try feature or embedding-based approach, in which the parameters of the language model are frozen. In the latter case, a two-layered BiLSTM is added in front of the BERT model.

They have evaluated SciBERT variants on a variety of tasks in different scientific domains and progressed the state-of-the-art in several tasks. In general, all variants have improved upon vanilla BERT models. SCIVOCAB approach has yielded slightly better performance over BASEVOCAB on average. Experiments also showed that finetuning approach is much more effective than the frozen embedding approach as for both BERT-base and SciBERT there was a gap of about +3F1 points on average. In fact, finetuned BERT-base model outperformed the feature-based SciBERT model.

3. RELATED WORK

3.1. General Domain Named Entity Recognition and Normalization

In this section, we will give a brief overview of the development of research in the general domain NER and NEN tasks. Research on biomedical NER and NEN tasks follows a very similar trajectory as the much more widely studied general domain English NER and NEN. Oftentimes, researchers have tried to apply advances in general NER tasks to biomedical NER. Therefore, it is useful to outline the development of these tasks in the general domain in order to understand the history of biomedical NLP better.

3.1.1. Named Entity Recognition

Many earlier works employed more formal methods for the NER task. Rule-based systems were developed that were relying on linguistic grammars and hand-crafted lexicons or gazetteers. One advantage of these systems is that even though the clever design of hand-crafted rules and lexicons by experts is necessary, they don't require training data. While such systems can be effective for some use cases, their reliance on gazetteers naturally meant that they were having problems generalizing unseen named entities and to domains other than the specific domain the system was developed for.

Other early works have used classical machine learning techniques with linguistic features. In order to represent each training sample, researchers were very carefully designing features like capitalization, part-of-speech tags, gazetteer match indicator, and word occurrence in the corpus. Learning algorithms included general methods like Decision Trees, Support Vector Machines (SVM), and Maximum Entropy models, but also sequence learning models like conditional random fields (CRF) and Hidden Markov models (HMM) were used (see [26]).

As deep learning methods were becoming popular in NLP, they started gaining prominence for NER tasks also. In one of the more influential papers, NLP from Scratch [27] from back in 2011, it was demonstrated that without using any major task-specific apriori knowledge or hand-crafted features, it was possible to train neural networks that are very competitive with the best systems that use traditional pipelines. By using very simple models by today's standards, a simple feed-forward network that took concatenation of vectors of neighboring words, they have reached near state-of-the-art scores in four core NLP tasks: chunking, part-of-speech tagging, NER, and semantic role labeling. However, they still needed to add traditional NLP features like suffix and part-of-speech tags, in order to surpass previous systems. This was an important milestone for the history of deep learning for NLP.

During the development of the field, other common techniques became part of the standard pipeline. These include: pre-trained word vectors (like word2vec [23]), character-level input representations [28]) and RNN based encoders (typically LSTM or GRU). The setup from Lample [29] has been very influential and taken as a baseline for many future works. They use pre-trained word embeddings in addition to learned embeddings generated via bidirectional LSTMs. Input representation of each word is created by concatenating different embedding types. An LSTM layer of hidden size 100 is used to contextualize the input representation of each word in order to obtain hidden vectors. Finally, a CRF layer operates on these hidden vectors to obtain loss function. Also, we should note that IOBES encoding scheme is used to create word-level features and illegal transitions such as I-LOC to B-PER are already disallowed in the CRF layer. One of the key conclusions of this work was that character-level features were used for the first time for the NER task.

As the importance of unsupervised pre-training became more generally understood in the NLP domain, researchers started to apply such techniques to the NER task also. One very successful work in this category is Flair [30]. In this study, authors train a character-level LSTM based language model on a large general domain corpus. After pre-training, the language model, the concatenation of the hidden vectors of start and end characters of a word is used as the input representation while finetuning the model for the downstream sequence labeling task. Flair had improved significantly upon the then state-of-the-art previous systems in many NER datasets in both English and German.

For general NER we also want to point to the original BERT implementation. In order to adapt the BERT model to the NER task, the authors tried both finetuning the BERT model directly on NER and using representation from its interior layers as features and using an LSTM layer to obtain final hidden vectors. The results showed that, even though BERT wasn't able to become the state-of-the-art model in NER unlike the other NLP tasks they have studied, its performance using in both finetuning and feature cases was very competitive.

Meanwhile, some recent works tried using span representations for named entity recognition. For example, [31] where NER is treated as dependency parsing which is very similar to our span classification formulation. Another related work from this is aspect is SpERT [32]). In this work, the authors address both NER and relation extraction tasks. In relation extraction, the goal is to classify the relationship between two given named entities in the same text segment. Here, NER is formulated as span classification and the same span embeddings for entities are also jointly used for relation extraction.

3.1.2. Named Entity Normalization

As discussed earlier NEN task is also studied under the names of Entity Linking or Entity Disambiguation in the literature. As the plurality of the names suggests, there are differences in its treatments in the literature. It is sometimes studied as an end-to-end task, where entity mentions are extracted from the text first and then the extracted mentions are mapped to concepts in the dictionary. In contrast, some studies only consider it from the disambiguation perspective and it assumed that correct entity mentions are already extracted from the text.

As with the NER task, earlier models were based on dictionary matching and classical ML algorithms with hand-crafted features. One of the most commonly used features is the string similarity score between the mention and the concept (e.g. [33], [34]). Since in many cases entries in knowledge bases have links to Wikipedia articles, some of the papers used the cosine similarity between tf-idf vectors of the context of the mention and Wikipedia article of the entity as a feature in order to benefit from contextual match (e.g. [33]). Several studies have used topic modeling techniques in order to represent entities and mentions as a weighted combination of topics (e.g. [33]). Papers focusing on document-level entity disambiguation often used entity mention graphs in order to benefit from consistency constraints between different entities found in a single document [34]).

Over time different types of neural models have emerged. While some models used neural networks to rank entity candidates with hand-crafted feature representations, others used deep learning to create dense representations for mentions, entities, and their context.

BERT model has also been applied to the NEN task. One of the successful works that use BERT and is close in its approach to ours is BLINK [35]. In this study, authors use two separate BERT encoders to embed mention context and entity description to the same embedding space, so that the corresponding vectors are close only if the mention indeed refers to the entity in the knowledge base. At inference, the mentioned context is embedded in to feature space and a fixed number of entity candidates are sampled from the knowledge base according to the distance of their description embedding to that of the mentioned context. Furthermore, a separate BERT model is used in order to select the correct entity among the list of candidate entities that were found by the embedding method. BLINK has improved upon the state-of-the-art in many different datasets.

3.2. Biomedical Named Entity Recognition and Normalization

In this section, we first give a general overview of related works in both tasks and then closely examine two related papers more closely in order to motivate our approaches.

3.2.1. General Overview

As hinted in the introduction of this chapter development of biomedical NER and NEN closely follows the development of their general domain counterparts. Because of the specificity of the biomedical terms and the relatively small size of the dataset, these two tasks are often handled together in biomedical literature. Therefore we summarize related work of both tasks in a common section.

Many early works in this area were based on dictionary matching, such as in [36]. However such methods tend to be fragile and don't generalize well to new entities. Likewise, CRF-based entity taggers operating on manually extracted features were proposed as they were popular in the NER research in the general domain.

Influential works under this category that were also made available as web APIs are tmVar [37] for gene mutations, tmChem [38] for chemicals, DNorm [39] for diseases and GNormPlus for genes. While all these studies had normalization steps that were based on matching the mention to the concepts in the dictionary via simple heuristics like abbreviation resolution and punctuation removal, DNorm had a machine learning-based normalization step. The normalization module was trained with a learning-to-rank objective, where positive and negative samples were drawn for each mention from the concept dictionary. Both mentions and the entities were represented using sparse features like tf-idf vectors. The matching score between an entity and a mention was calculated by the inner product of their feature vectors weighted in each dimension by a learnable importance score.

In a follow-up work, TaggerOne [40], authors combined the DNorm approach with CRF based NER model. Using features such as tf-idf and other lexical features, NER and NEN vectors are created for each token in the input sequence. Possible segmentations of the input sequence from a semi-Markov are scored with a weighted sum of their NER and NEN scores. The NER scores were based on matching the tag label with predictions whereas the NEN score was based on the similarity between the features vectors of the mention and the concept. Their results have shown that joint learning had significantly improved the performance in both tasks.

Many later works proposed character embedding + LSTM + CRF-based models in particular for the biomedical NER task. Among them, the approach of [1] is particularly interesting because it employs both joint learning and explicit feedback to improve the performance in both tasks. During the same period, for the NEN task, many papers suggested contrastive learning using deep neural networks based on CNN and LSTM layers. For example in BNE [41], a biomedical name encoder based on bidirectional LSTM networks was suggested which more or less replaced the role of the sparse vectors in DNorm's approach.

Another line of research following the recent development of pre-trained language models is the application of these models to NER and NEN tasks. In both SciBERT [12] and BioBERT [11], the models were benchmarked in their performance in biomedical NER datasets both becoming state-of-the-art in several of them. Also, several separate works studied the application of BERT-based models to normalization problems like Bert Ranking [42] and BioSyn [13].

3.2.2. Joint Model of Zhao et al.

An important work that has greatly influenced our first approach is [1]. In this work, the authors show that joint learning between biomedical NER and NEN tasks can improve the performance in both tasks significantly. Both tasks are modeled as sequence tagging problems as standard in the NER literature. Following [29], a pipeline of (1) embedding layer, (2) LSTM layer, and (3) CRF layer is used for model construction. The embedding layer contains both a pre-trained word embedding layer that was intended for biomedical NLP (PMC word2vec embeddings [43]) and a character-level CNN layer. The outputs of these two layers are concatenated to obtain the final word representation before the LSTM layer.

Since these two tasks are closely related, authors employ different variants of joint learning between the two tasks. In the most basic joint learning setting weights of the two models for both tasks are shared except the output layers. Experiments show that this joint learning scheme already brings great improvements in both tasks. Next, they introduce so-called explicit feedback mechanisms to inform one task with the outputs of other tasks more directly. More concretely, the output predictions of one task are turned into a vector and multiplied by a learnable matrix so that it has the same dimension as the hidden vector of the other task. The resulting vector is combined with the output of the bidirectional LSTM layer.

This procedure brings further improvements to both tasks performances. In particular feeding, the output of the NER task to the NEN model is responsible for much of the improvement. In our approach, we only tried to incorporate the simple joint learning scheme as it is easier to implement and it had already given great results.

3.2.3. BioSyn

A more recent and successful work was BioSyn [13] ("Biomedical Entity Representations with Synonym Marginalization"), which inspired one of our methods. As the title of the paper suggests their methodology is based on contrasting synonyms of a mention from other possible known entity mentions corresponding to other concepts.

The basic idea is to map entity mentions to embedding vectors with a common encoder. Similar entity mentions or more interestingly entity mentions corresponding to the same concept in the dictionary are expected to have close embedding vectors and mentions corresponding to distinct concepts should have embeddings that are far apart from one another. When such an embedding is found, the corresponding concept of an entity mention can be determined by finding the entity mention with the closest embedding vector to that of the mention in question and assigning the concept of that mention.

In order to obtain an embedding function with this property, authors employ contrastive learning which has seen great success in recent years particularly in computer vision. For a given entity annotation in the dataset, "similar" and "dissimilar" candidates need to be drawn and an objective function should be chosen that encourages embeddings of similar pairs to be closer and embeddings of dissimilar pairs to be more distant.

In the case of biomedical NEN, similar candidates are entity mentions that are map to the same concept as the entity in question and dissimilar candidates are those that map to different concepts. Also note that in this approach we only the entity mention itself in the annotation, disregarding the surrounding context of the mention entirely.

One advantage of using this embedding-based approach is that it allows fast inference compared to the previously mentions methods such as BERT Ranking [42]. In order to rank candidate entity mentions from the dictionary, it is not necessarily run the model N times to obtain similarity scores for entry in the dictionary. Rather, one can pre-compute embeddings of all candidates once and store them in a database. In order to make inference for a new entity mention, one needs to calculate the embedding of this new mention only and compare this embedding vector to the ones of the candidate mentions. Further, efficient algorithms exist that don't require $\mathcal{O}(N)$ distance calculations for N elements in the dictionary. For commonly used distance metrics such as Euclidean distance and cosine similarity, the complexity of the search can be reduced to $\mathcal{O}(log(N))$ with clever use of data structures. This ensures that inference can be done in a reasonable time even if the size of the dictionary is in the millions range.

Another benefit of adopting this approach is that it can potentially correctly predict concepts that are not found in the training set but existent in the dictionary. If the embedding function is generalizable enough, it will map semantically similar entity mentions to close vectors in the embedding space even if it has not encountered those exact mentions in the training set before. This would be of great use for practical applications as if the model is good enough only the output dictionary needs to be updated to detect new concepts, such as new diseases or chemicals.

For representing entity mentions, authors use both sparse and dense embeddings. The sparse representation corresponds to the commonly used tf-idf vectors of the entities. Since biomedical entity mentions are usually complex, often compound words comprised of Latin roots and suffice, tf-idf representation is based on character ngrams rather than words. For dense representations, a BioBERT based encoder is chosen. Since the BioBERT model was pre-trained on a large corpus of biomedical text, it is expected to already have meaningful representations of biomedical entity mentions even before training and to generalize well to mentions not seen in the training set of the NEN task but perhaps seen in the pre-training corpus. The dense embedding vector is obtained by feeding the mention to the BioBERT model enclosed in special tokens (like '[CLS] ovarian cancer [SEP]') and choosing the hidden vector corresponding to the '[CLS]' token as the embedding vector. We will denote this as

$$Emb(s) = BERT_{[CLS]}(s). \tag{3.1}$$

It is important to mention that both the entity mention appearing in the training set and synonym candidates are encoded with a shared BioBERT model so that they are embedded in the same embedding space:.

The similarity score of two mentions (say, m and n) is calculated by the weighted sum of the inner product of their sparse representations ($S_{sparse}(m, n)$) with the sum of the inner product of their dense representations ($S_{dense}(m, n)$):

$$S(m, n) = S_{dense}(m, n) + \lambda S_{sparse}(m, n). \tag{3.2}$$

The weighting factor λ is not fixed but learned during training.

Another novelty of this work is in the construction of the candidate set. In contrastive learning literature, the selection of negative samples was found to be of critical importance in many different studies. One option is to choose random entries from the dictionary, but it might be too easy for the model to distinguish the synonyms from the randomly sampled entity mentions so that no real learning occurs after initial epochs. Therefore, the authors choose entity mentions with the closest tf-idf match for the candidate set (sparse candidates). To make the task of the model even harder, they also add entity mentions with close dense vectors to the candidates set (dense candidates). Note that selected dense candidates for an entity mention will be different in each epoch as the weights of the model are updated. Following the intuition that dense candidates will be more difficult than the original negative samples after initial epochs and that the learning process should go from easier to harder examples; they increase the ratio of dense candidates in each epoch from 0% to 100%. This candidate selection process was named iterative candidate retrieval by the authors.

Authors have experimented with different loss functions including contrastive loss and hard entropy maximization, but they have found that the marginal maximum likelihood (MML) objective worked best. In the MML objective, similarity scores are treated as logits in a multiclass classification task. This way, a softmax operation gives probabilities of each candidate for getting chosen as the positive candidate. The probability of a candidate set is given by the sum of positive candidates in the candidate set, the negative log-likelihood of this probability is minimized:

$$\begin{aligned} P(n \mid m; \theta) &= \frac{exp(M(m,n)}{\sum_{n' \in N_{1:k}} exp(M(m,n')} \\ P(N_{1:k} \mid m) &= \sum_{\substack{n \in N_{1:k} \\ EQUAL(m,n)=1}} P(n \mid m; \theta) \\ \mathcal{L} &= \frac{1}{M} \sum_{i=1}^{M} log(P(N_{i,1:k} \mid m_i)). \end{aligned} \tag{3.3}$$

BioSyn has improved over the state-of-the-art in all datasets it was trained on. In fact in their error analysis on NCBI disease dataset authors claim that most of the errors were not due to the model but from other reasons (e.g. insufficient context, hyponym, or hypernym annotations) and the model has reached the upper bound performance.

4. METHODOLOGY

In the context of biomedical named entity recognition, we will usually refer to entities found in dictionaries or knowledge bases as concepts and to the text span corresponding to the concept as mention.

We have used different setups for each hypothesis we wanted to test. In all settings, the encoder transformer networks are initialized from one of the pre-trained language models and the input texts are tokenized with the corresponding pre-trained tokenizer of that model.

4.1. Sequence Tagging Approach

To test the first hypothesis, namely whether joint learning would be as beneficial with BERT-like language models, we adapt the commonly used sequence tagging formulation for both tasks as in [1]. This means that the model will map a sequence of n tokens to a sequence of n tags for each task. Note that NER and NEN tasks will have different output sequences and NEN will have a much larger output vocabulary.

More formally:

$$\begin{aligned}
D &= \{(S_j, y_j^{NER}, y_j^{NEN})\}_{j=1,k} \\
S_j &= [x_1, x_2, ..., x_n] \\
y_j^{NER} &= [y_{1j}^{NER}, y_{2j}^{NER}, ..., y_{nj}^{NER}] \\
y_j^{NEN} &= [y_{1j}^{NEN}, y_{2j}^{NEN}, ..., y_{nj}^{NEN}].
\end{aligned} \tag{4.1}$$

Here, given a dataset D of size k S_j is the j'th input sequence with tokens

$x_i, 1 < i < n$ and corresponding tag sequences for NER and NEN tasks, y_j^{NER} and y_j^{NEN}, both of length n.

Both named entity recognition and normalization are thought of as multi-class token classification problems. In the case of named entity recognition, target classes correspond to different entity types in the dataset, whereas in entity normalization case, there are as many classes as the number of entries in the target dictionary.

We use the ConLL format to represent the data format, following many works in the literature. In each line represents a token and contains the token itself, NER tag, and NEN tag of the token, separated by a tab character.

In this approach, we work at the sentence level and require that the input texts be split into sentences. Split sentences are then tokenized into words, either by whitespace or with more complicated methods. In this approach, most of the time we use already sentence split and tokenized versions of the biomedical entity datasets that are publicly available and were also used by previous works. They were first introduced by [44] and the preprocessing was done by standoff2conll tool [45].

Note that, for NER we use a tagging scheme like IOB or IOBES to be able to differentiate contingent mentions of the same type from each other. Since for the normalization task we don't care about the exact mention or how many times the entity type occurs in the sentence, we don't use a tagging scheme and try to predict the target class for each token directly.

One weakness of using this approach for entity normalization is that, because target classes are fixed during training, we cannot use the model to normalize into unseen entity types.

The input representation for finetuning BERT-based models in these tasks should be as close to the pre-training scheme as possible. Therefore each token is prepended by the special start token '[CLS]' and appended with '[SEP]'. Since the training is done in batches and sequences may be of different lengths, we also add '[PAD]' tokens to the end of the sequences so that they all have the same number of tokens as the longest sequence in the batch.

To obtain representations from a pre-trained BERT-based model, the input text must be processed by the model's specific pre-trained tokenizer that splits words into subwords. Unlike traditional word-level deep learning models for sequence tagging tasks, there is a token mismatch problem for BERT-like models that operate on subword tokens. This represents a problem, as the number of subword tokens as returned by the BERT tokenizer may differ from the original number of tokens in the tokenized sentence so that we don't have one tag label corresponding to each input subword token. Thus, there is a necessity to map or align hidden subword output representations from BERT to word-level tag sequences. This can be done in many different ways.

(i) To represent each word, simply use its first subword token
(ii) Pool all subword token representations of a word to get word representation (for example take the average)
(iii) Re-map IOB or IOBES encoding so that it works at subword level
(iv) Make predictions for each subword using the target of the word and take the majority in the inference time

We have chosen the first approach as it was easier to implement and chosen by the original BERT implementation [22] and BioBERT [11]. The second approach was also tried by Electra [46] and SciBERT [12] with a very similar performance. We implement this strategy by adding dummy tag labels (e.g. 'X') for not leading subword tokens word and masking them out in the loss function. This way we don't have to store the subword-to-word map during the forward pass of the model and inference, we simply ignore these masked subword tag predictions during decoding. Also, following standard practice, we apply dropout with a rate of 0.1 to the hidden BERT representations for regularization.

Hidden representations outputted by the shared BERT-based encoder are fed into two different classifier heads. These classifier heads are simply feedforward layers that map a vector of BERT's hidden size to a vector of size of the output vocabulary of each task. After that logits are obtained for both tasks and we are ready to calculate multi-class cross-entropy loss values for each token. Note that special start and end tokens (i.e. '[CLS]' and ['SEP']), pad tokens (['PAD']) and masked out non-leading subword tokens ('[MASK']) are not counted in the loss function thanks to masking.

We use a very simple joint learning scheme. For each sample, losses for both tasks are calculated and simply added together with a weight factor that was determined experimentally.

We have also experimented with the feature-based approach, where the weights of the BERT model are not updated during training. In this setup, the outputs after the BERT layer are encoded with a bidirectional LSTM layer. This approach, although yielding slightly inferior results in the literature (like in SciBERT [12]), is more cost-efficient as the BiLSTM layer has much fever parameters than the BERT model and thus the backward pass is less costly as gradients don't need to be calculated for BERT layers.

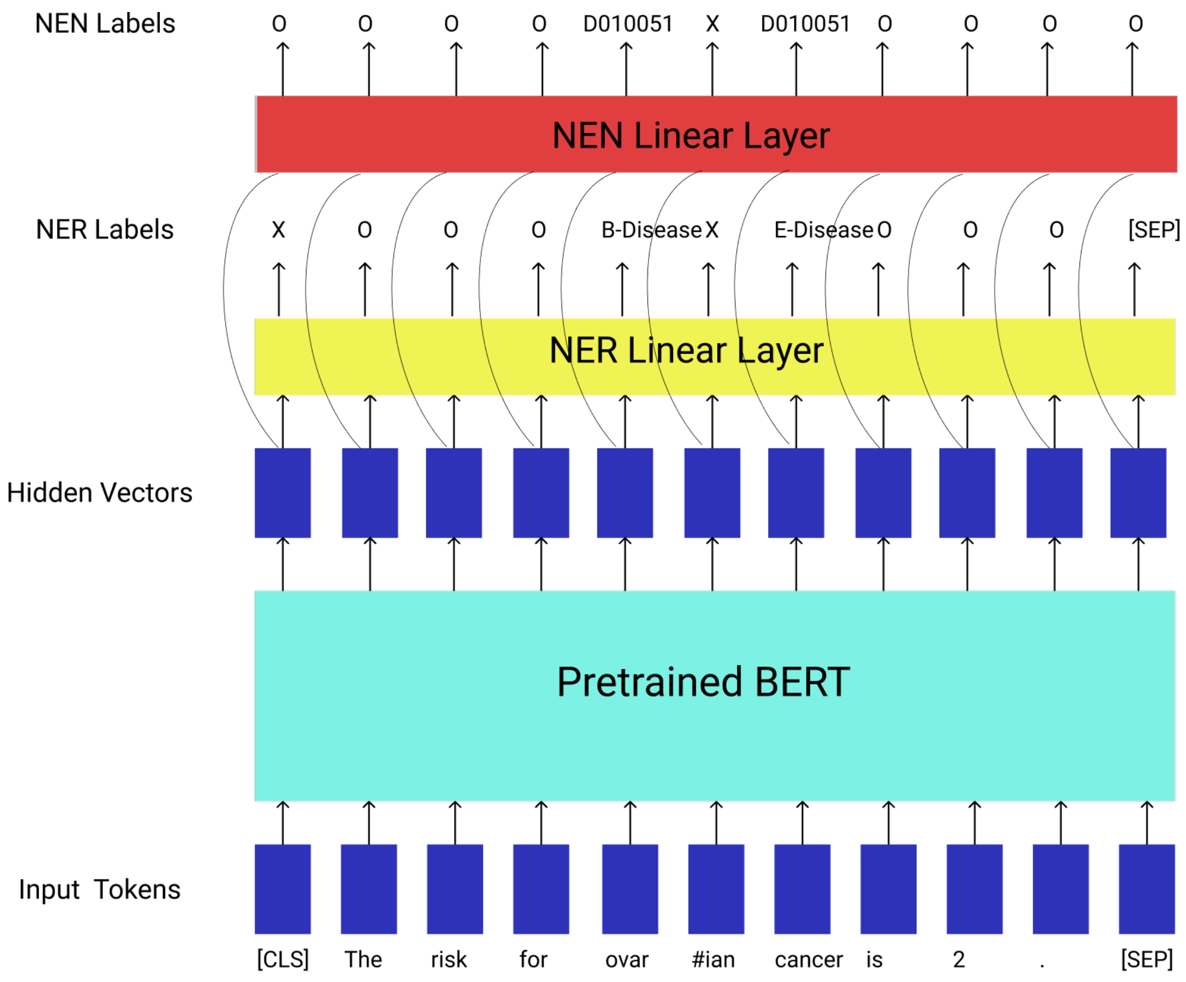
NEN Labels
O O O O D010051 X D010051 O O O O
NEN Linear Layer
NER Labels
X O O O B-Disease X E-Disease O O O [SEP]
NER Linear Layer
Hidden Vectors
Pretrained BERT
Input Tokens
[CLS] The risk for ovar #ian cancer is 2 . [SEP]

Figure 4.1. Outline of the BERT-based joint NER and NEN tagging model.

4.2. Span-based Approach

4.2.1. Span-based NEN

The BioSyn [13] has demonstrated the benefits of contrastive learning for the normalization task. The main idea is that mentions from the sentences and their synonyms are enforced to have similar embeddings. This is achieved by:

- getting synonym candidates by sampling synonyms and non-synonyms for each mention
- embedding the mention by the BERT encoder

- embedding all candidates for the entity mention by the BERT encoder
- predicting synonyms among the candidate set by using the cosine similarity between the candidate and the entity mention as logits
- finding concepts with closest embedding vector to mention embedding among the concept dictionary during the inference

While this approach has yielded excellent performance and improved upon the state-of-the-art models in all datasets, it treats the NEN task only as entity disambiguation and doesn't use sentence context for calculating mention embeddings. In fact, error analysis of the BioSyn model in the NCBI disease dataset has shown that some of the misclassification errors were due to context dependence [11].

This inspired our model which calculates entity mention embeddings using the sentence context. BERT representations of all tokens in the sentence are calculated by feeding the sentence to the BERT encoder. Next, representations of the tokens that correspond to the span that makes up the entity mention are pooled to get an embedding for the full mention rather than its individual tokens. These mention embeddings are used in the contrastive loss function of the BioSyn. Note that for embedding synonym candidates from the dictionary, we still cannot use context and therefore embed them the same way as in the BioSyn with a BERT encoder.

More formally, let S be a sentence containing the mention m. The mention m maps to the CUI c. In the dictionary for the CUI c several synonyms (also called synset) have been assigned. The set all of synonyms for all CUIs is denoted with $N = [n_1, n_2, \ldots, n_k]$ where $n_i \in N$ is a mention string. For the mention m, we will choose a mention string from the dictionary using our model and assign its CUI to m:

$$c^* = \mathrm{CUI}(\mathrm{argmax}_{n \in N} M(S, m, n)) \tag{4.2}$$

where is the $M(S, m, n)$ is the "synonymousness" score given by the model to the mention string n from the dictionary for the mention m appearing in the context S.

In BioSyn, the score was given by the inner product of mention embeddings of m and n using a BERT encoder (hidden vector of the '[CLS]' token). Formally:

$$\begin{aligned} Emb(s) &= BERT_{[CLS]}(s) \\ M(S, m, n) &= Emb(m) \cdot Emb(n). \end{aligned} \tag{4.3}$$

However, in our approach, we will calculate the mention representation dependent on the sentence S. The representation of m, will be formed by pooling representations of tokens in S that make up m:

$$Emb'(S, m) = pool(BERT(S)_{[x_i \in m]}) \tag{4.4}$$

where x_i are tokens in S that correspond to the location of m in S. Note that this means that mentions and candidates will be embedded in different ways. Therefore we might use different encoders for dataset mentions and dictionary candidates, for example, two BERT encoders that start with the same parameters but their parameters are not shared during training.

To prevent us from the burden of training two different BERT models during training, we might also use fixed embeddings for candidate embeddings using the BioSyn model. In this setup, the contextual mention encoder will be aligning the span embeddings to the space of the BioSyn model's embeddings.

At this point, we have calculated similarity scores for all concept candidates for the entity mention using BERT embeddings for each side. Different loss functions such as contrastive loss and triplet loss have been proposed in the contrastive learning literature. Following BioSyn [13], we use marginal maximum likelihood (MML) objective, which operates on all candidates at once instead of working on pairs. In this approach, similarity scores are interpreted as logits and we try to maximize the probability of choosing actual synonyms among the set of candidates. Formally, for the mention m

and candidate n from the candidate set $N_{1:k}$ the probability is given by:

$$P(n \mid m; \theta) = \frac{exp(M(S, m, n)}{\sum_{n' \in N_{1:k}} exp(M(S, m, n')}. \tag{4.5}$$

Then probabilities of all actual synonyms are added:

$$P(N_{1:k} \mid m) = \sum_{\substack{n \in N_{1:k} \\ EQUAL(m,n)=1}} P(n \mid m; \theta). \tag{4.6}$$

Finally the negative log-likelihood objective is maximized as follows:

$$\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} log(P(N_{i,1:k} \mid m_i)). \tag{4.7}$$

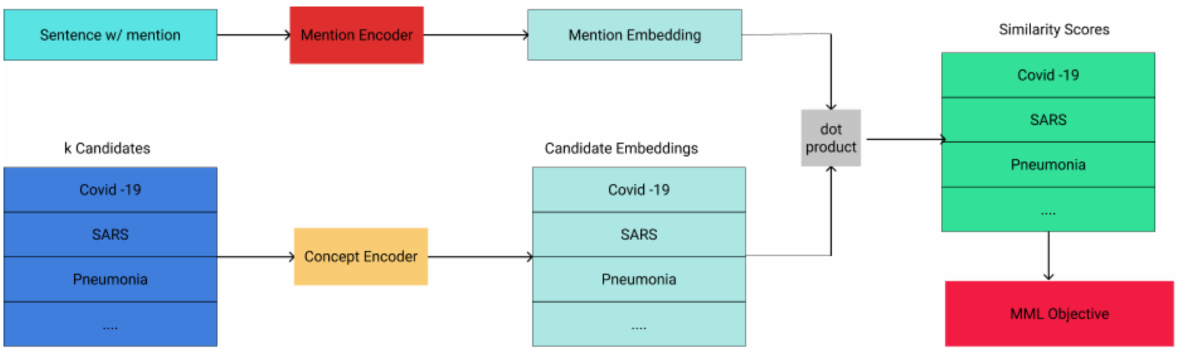

Figure 4.2. Training pipeline of constrastive NEN model.

For inference, embeddings of all mentions in the dictionary are calculated and stored. For a given sentence, a NER system is run to get the named entities in the sentence. Then mention embeddings for the named entities are obtained using the BERT-based mention encoder. From candidate embeddings the one with highest dot product with the calculated mention embedding is sought and its concept is assigned to the mention as given by the following formula:

$$\begin{aligned} c^* &= \mathrm{CUI}(\mathrm{argmax}_{n \in N} M(S, m, n)) \text{ where} \\ M(S, m, n) &= Emb(m) \cdot Emb(n). \end{aligned} \tag{4.8}$$

With the help of maximum-inner-product-search (MIPS) algorithms, we don't have to calculate products with all candidates in the dictionary to find the one with the highest score. With the use of special data structures the search complexity can be reduced from $\mathcal{O}(n)$ to $\mathcal{O}(log(n))$. For this, we use Faiss [47], a popular MIPS library by Facebook AI.

4.2.2. Span-based NER

Inspired by the success of previous studies, it is desirable to use joint learning with NER with the contrastive learning formulation of the NEN task too. However, when the NEN task is formulated using span embeddings as described above it is awkward to use the sequence tagging approach for NER as the inner representations have different requirements in both cases. For example, in NER tasks with sequence tagging formulation, a token representation should contain the information whether the token is at the start or the end of a named entity, whereas in span-based NEN token representations of a mention should give a distinctive representation when they are pooled. Therefore it makes to formulate the NER task using span embeddings so that both tasks will benefit from sharing the same encoder representations.

Different span-based NER formulations can be found in the literature. We use a simple scheme similar to SpERT [32]. Given a sequence of n tokens, there are $n*(n-1)$ possible entity spans that may or not correspond to a named entity. Each span will be classified into one of the entity categories or to the "non-entity" class. As the number of possible entity spans grows with $\mathcal{O}(n^2)$, the search space is limited to spans with a maximum length W. (for example, W=10 means only spans up to 10 tokens will be considered.)

In this NER task formulation, we will use the same span representation as in the span-based contrastive NEN formulation. The same BERT-based encoder will be used to obtain a sequence of hidden vectors for each token: $H = [h_1, h_2, ..., h_n]$. Then all valid possible span embeddings will be formed using the same pooling mechanism as in the NEN task. Note that, in the NEN task we were only considering spans that correspond to a named entity whereas in the NER task we consider all valid spans (i.e. all spans filtered by a maximum length). For the span $x_i, x_{i+1}, ..., x_j$ the representation will be constructed by pooling the BERT representation of the tokens that lie inside this span. Then this representation is fed to a linear layer and a softmax function to obtain the output distribution as it is standard in multiclass classification. We use the negative log-likelihood as the loss function. Note that output vocabulary includes entity categories and a none-class for spans that do not correspond to a named entity. Formally this can be expressed as follows:

$$\begin{aligned} S &= [x_1, x_2, ..., x_n] \\ s_{i,j} &= [x_i, x_{i+1}, ..., x_j] \text{ (for each valid span } s_{i,j}, 0 \leq i \leq j \leq n \text{)} \\ Emb(s_{i,j}) &= h = pool(BERT(S)_{[x_k \in s_{i,j}]}) \\ y' &= softmax(W \cdot h + b). \end{aligned} \tag{4.9}$$

During inference, all valid spans are considered and spans whose predictions are not the none-class are selected as NER predictions for the sentence. Note that because we are considering all valid spans it can occur that spans predicted to contain entities overlap or fully contain one another. This property makes the span-based NER approach more attractive for nested NER datasets as it is not possible to model overlapping entities with a sequence tagging approach. Yet since the datasets we will be using don't contain such nested entities, we will handle these overlapping entity predictions by choosing spans with higher scores in case of a collision.

Since now both tasks are using span representations for making predictions, we can share internal representations and use the same encoder network. As in the sequence tagging approach, we simply add loss functions weighted by a fixed parameter that was determined experimentally.

4.2.3. Pooling Functions

As explained earlier, in both span-based NER and NEN formulations hidden vectors corresponding to tokens of a span need to be converted to a single vector to be able to represent the span at once. We tested different choices of pooling functions, which were used in the literature. Note that different pooling operations can be combined by concatenating the resulting vectors.

We experimented with the following operations:

- average pooling
- max pooling
- attention mechanism
- width embeddings

To make it easier for the model to use sentence context, we also tried concatenating the output of the "[CLS]" token to the pooled vectors. After initial experiments with span-based NER, we found out max-pooling + [CLS] + width embeddings worked best and due to time limitations only performed an ablation study on the NER task.

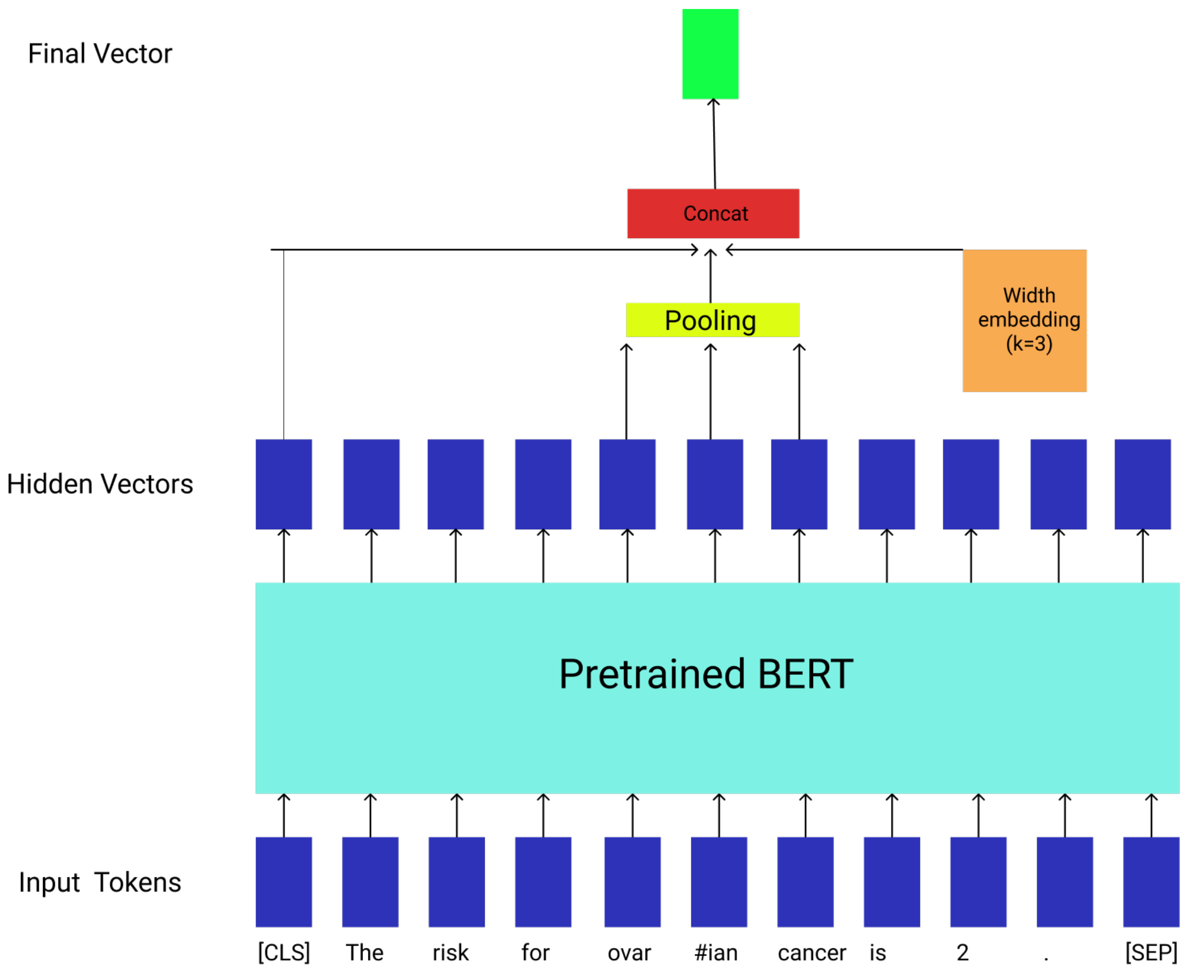

Figure 4.3. Example for creating span embeddings “ovarian cancer”.

5. EXPERIMENTS AND RESULTS

5.1. Datasets

In this section, we will give the details of the biomedical NER and NEN datasets on which the proposed methods were tested.

Table 5.1. Statistics from datasets used in this work.

Dataset	# of Documents			# of Mentions		
	Train	Dev	Test	Train	Dev	Test
NCBI Disease	592	500	500	5134	787	960
BC5CDR Disease	500	500	500	4182	4244	4424
BC5CDR Chemical	500	500	500	5203	5347	5385

5.1.1. NCBI Disease Corpus

NCBI Disease Corpus [48] is one the most commonly used datasets in the biomedical NER literature. It was constructed from the abstracts of PubMed articles. As the name suggests, the dataset is mainly concerned with disease mentions. Each abstract was annotated at both concept and mention levels. In a given abstract, all disease mentions were identified and they were mapped to their corresponding concept. As concepts sources, Medical Subject Headings (MeSH) and Online Mendelian Inheritance in Man (OMIM) ontologies have been used. Since the sources were merged in an earlier study to a single vocabulary called MEDIC [49], it was directly used whenever possible. The MEDIC disease dictionary is part of a larger effort named Comparative Toxicogenomics Database (CTD) which provides a public resource for health research and the dictionary is updated monthly since its first release in 2012.

14 different annotators were employed during the annotation process. Each abstract was given to at least two different annotators. The annotators did not start from scratch, rather they have received abstracts after they were annotated with the PubTutor tool as a pre-step. After an abstract was reviewed and annotated by both annotators their results were compared and conflicts were resolved by the annotator pair.

In the MEDIC dictionary, an entry has a CUI (concept unique identifier) like "MeSH:D010009", a preferred name like "Osteochondrodysplasias", a list of alternative CUIs, and a list of synonyms. When annotating a mention in a PubMed abstract, annotators were instructed to first try to map it to a concept with a matching preferred name. If there is not a match, next the mention is searched in synonym lists. Note the same synonym can be listed in many different entries. In this case, a CUI is chosen by an annotator from the alternative CUIs list that best describes the mention logically. In case when there is no specific entry that matches the mention the closest hypernym is chosen for annotation.

A single mention may contain more the one disease concept like in "colorectal, endometrial, and ovarian cancers". Such mentions are called composite mentions and were annotated by concatenating individual concept identifiers of the constituent diseases with '|': "MeSH:D010051|D016889|D015179".

Another non-standard situation is the case of multiple concept mappings, where a mention is annotated by combining two different concepts. For example, "inherited neuromuscular disease" was annotated by combining "Neuromuscular Disease" (MeSH:D009468) and "Genetic Diseases, Inborn" (MeSH:D030342), and the combined CUI is denoted by "MeSH:D009468+MeSH:D030342".

In the officially released version, the dataset already contains train, development, and test splits.

Table 5.2. An truncated disease example from the MEDIC dictionary.

DiseaseName	Ablepharon macrostomia syndrome
DiseaseID	MESH:C535557
AltDiseaseIDs	DO:DOID:0060550|OMIM:200110
Synonyms	Ablepharon-Macrostomia Syndrome|AMS

```
9949209|t|Genetic mapping of the copper toxicosis locus in ...
9949209|a|Abnormal hepatic copper accumulation is recognized as ...
9949209 23  39  copper toxicosis Modifier   OMIM:215600
9949209 158 185 hepatic copper accumulation SpecificDisease D008107
9949209 206 224 inherited disorder  DiseaseClass    D030342
9949209 272 299 hepatic copper accumulation SpecificDisease D008107
9949209 346 360 Wilson disease  SpecificDisease D006527
9949209 362 364 WD  SpecificDisease D006527
9949209 499 514 copper overload SpecificDisease D008107
9949209 523 553 non-Indian childhood cirrhosis  SpecificDisease OMIM:215600
9949209 637 653 copper toxicosis    SpecificDisease OMIM:215600
9949209 655 657 CT  SpecificDisease OMIM:215600
9949209 738 751 liver disease   DiseaseClass    D008107
9949209 777 779 WD  Modifier    D006527
9949209 814 816 CT  SpecificDisease OMIM:215600
9949209 999 1001    CT  SpecificDisease OMIM:215600
9949209 1147    1149    WD  SpecificDisease D006527
9949209 1174    1176    CT  SpecificDisease OMIM:215600
9949209 1261    1263    CT  SpecificDisease OMIM:215600
```

Figure 5.1. An example article annotation from the NCBI disease dataset.

5.1.2. BioCreative V CDR Corpus

The BioCreative V CDR dataset [50] was originally created for a challenge task for promoting advances in biomedical NLP research. The dataset was intended for disease named entity recognition and chemical-induced disease relation extraction. Therefore, unlike NCBI it contains both chemical and disease annotations. Since chemical-induced disease relation extraction is not our focus, we will describe the dataset only from NER and NEN perspectives. Similar to the NCBI disease dataset, this dataset was also created by annotating PubMed abstracts. Concept sources were CTD chemical dictionary for chemicals and MeSH dictionary for diseases.

Entity annotation was done by 4 people who were also indexers in the MeSH project. Like in NCBI dataset construction, each abstract was annotated independently by two different annotators. Disagreements between two annotations on the same abstract were resolved by a different and senior annotator.

This dataset is also split into train, development, and test sets in its official release.

```
8701013|t|Famotidine-associated delirium. A series of six cases.
8701013|a|Famotidine is a histamine H2-receptor antagonist used in ...
8701013 0   10  Famotidine  Chemical    D015738
8701013 22  30  delirium    Disease D003693
8701013 55  65  Famotidine  Chemical    D015738
8701013 156 162 ulcers  Disease D014456
8701013 324 332 delirium    Disease D003693
8701013 395 405 famotidine  Chemical    D015738
8701013 442 452 famotidine  Chemical    D015738
8701013 464 472 delirium    Disease D003693
8701013 537 547 famotidine  Chemical    D015738
8701013 573 583 famotidine  Chemical    D015738
8701013 689 699 famotidine  Chemical    D015738
8701013 CID D015738 D003693
```

Figure 5.2. An example article annotation from the BC5CDR dataset.

5.2. Results

5.2.1. Sequence Tagging Approach

In this approach, we are trying to understand the effects of joint learning in conjunction with transformer models. Therefore, we ran the experiments in the single-task mode for both NER and NEN tasks and a joint mode. Also, to see the influence of pre-trained transformer models we performed the same experiments with (1) the original BERT model, (2) BioBERT, and (3) SciBERT.

Another parameter we experimented with was finetuning the BERT model versus using it as features from frozen BERT layers. In the feature approach, a bidirectional LSTM module was added on top of the BERT layer to increase the number of learnable parameters and thus model complexity.

We have experimented with different parameter settings, but we have found the hyperparameters used in SciBERT [12] worked optimally in almost all cases.

Results are presented below in 5 different tables. In 5.3 and 5.4 the results of the finetuned models are shown, while 5.5 and 5.6 contain the results of the feature approach in NCBI and BC5CDR datasets respectively. Finally, in 5.7 we see the comparison between the models of Zhao et al. [1] and our best performing models.

The main conclusions we can draw from the results are:

- Domain-specific BERT variants outperform the vanilla BERT model in all tasks.
- BioBERT performs better on NCBI, while SciBERT has better results in BC5CDR.
- Joint learning between NER and NEN tasks has a small and not always positive influence on model performance.
- The performance of our NEN models is much behind that of Zhao et al. [1]. This might be indicating unexplained points in their work.

Table 5.3. Results of the finetuned transformer models on the NCBI disease dataset.

Model	Tasks	NER-P	NER-R	NER-F1	NEN-P	NEN-R	NEN-F1
BERT	NER	0.835	0.860	0.847	-	-	-
BERT	NEN	-	-	-	0.651	0.809	0.721
BERT	Joint	0.838	0.871	0.854	0.662	0.816	0.731
BioBERT	NER	0.867	0.890	0.878	-	-	-
BioBERT	NEN	-	-	-	0.682	0.832	0.749
BioBERT	Joint	0.847	0.897	0.871	0.673	0.823	0.741
SciBERT	NER	0.878	0.861	0.869	-	-	-
SciBERT	NEN	-	-	-	0.703	0.821	0.757
SciBERT	Joint	0.882	0.876	0.879	0.710	0.815	0.759

Table 5.4. Results of the finetuned transformer models on the BC5CDR dataset.

Model	Tasks	NER-P	NER-R	NER-F1	NEN-P	NEN-R	NEN-F1
BERT	NER	0.820	0.834	0.827	-	-	-
BERT	NEN	-	-	-	0.671	0.825	0.740
BERT	Joint	0.818	0.833	0.826	0.677	0.828	0.745
BioBERT	NER	0.854	0.866	0.860	-	-	-
BioBERT	NEN	-	-	-	0.683	0.846	0.758
BioBERT	Joint	0.852	0.868	0.860	0.681	0.853	0.757
SciBERT	NER	0.858	0.871	0.864	-	-	-
SciBERT	NEN	-	-	-	0.717	0.825	0.767
SciBERT	Joint	0.855	0.873	0.864	0.715	0.819	0.764

Table 5.5. Results of the frozen transformer models on the NCBI disease dataset.

Model	Tasks	NER-P	NER-R	NER-F1	NEN-P	NEN-R	NEN-F1
BERT	NER	0.813	0.831	0.822	-	-	-
BERT	NEN	-	-	-	0.628	0.784	0.697
BERT	Joint	0.814	0.845	0.829	0.644	0.793	0.711
BioBERT	NER	0.849	0.874	0.861	-	-	-
BioBERT	NEN	-	-	-	0.657	0.819	0.729
BioBERT	Joint	0.825	0.879	0.851	0.656	0.811	0.725
SciBERT	NER	0.850	0.835	0.842	-	-	-
SciBERT	NEN	-	-	-	0.676	0.800	0.733
SciBERT	Joint	0.856	0.850	0.853	0.697	0.787	0.739

Table 5.6. Results of the frozen transformer models on the BC5CDR dataset.

Model	Tasks	NER-P	NER-R	NER-F1	NEN-P	NEN-R	NEN-F1
BERT	NER	0.790	0.818	0.804	-	-	-
BERT	NEN	-	-	-	0.655	0.798	0.719
BERT	Joint	0.797	0.803	0.800	0.657	0.803	0.723
BioBERT	NER	0.829	0.845	0.837	-	-	-
BioBERT	NEN	-	-	-	0.660	0.845	0.730
BioBERT	Joint	0.836	0.856	0.846	0.659	0.835	0.736
SciBERT	NER	0.834	0.844	0.839	-	-	-
SciBERT	NEN	-	-	-	0.694	0.802	0.744
SciBERT	Joint	0.836	0.848	0.841	0.688	0.795	0.737

Table 5.7. Comparison btw. Zhao et al. [1] and our BioBERT model.

Model		NCBI-disease		BC5CDR	
		NER	NEN	NER	NEN
Zhao et al.	split	0.82	0.81	0.80	0.81
	Joint	0.86	0.87	0.86	0.87
BioBERT (ours)	split	0.88	0.75	0.86	0.76
	Joint	0.87	0.74	0.86	0.76

5.2.2. Span-based NEN

The arguably ideal setting with two finetuned BERT encoders, one for encoding context and the other for encoding mentions unfortunately did not fit into GPU memory. Therefore we had to conduct the experiments in the following two settings:

(i) frozen BERT features + LSTM for context encoder finetuned BERT encoder for mentions
(ii) finetuned BERT encoder for context encoder, pre-trained and fixed embeddings from BioSyn for mentions

After initial experiments, we decided to choose the BioBERT model as the encoder network and did not experiment with SciBERT due to time limitations.

In 5.8, we compare our models to the results published in the literature. Note that the full BioSyn model also makes use of sparse features, therefore we also include its dense-only performance. Our model with frozen BioBERT + LSTM encoder lies behind the dense BioSyn model, while the other setting with finetuned BERT context encoder and fixed candidate entity embeddings yields a slight improvement over the original dense BioSyn model. This shows that including sentence context for mention representation can indeed bring benefits even in this constrained setup with fixed candidate embeddings.

Table 5.8. Accuracy scores of the top prediction of the NEN models.

Model	NCBI-disease	BC5CDR-disease	BC5CDR-chemical
BNE	87.7	90.6	95.8
BERT-Ranking	89.1	-	-
BioSyn	91.1	93.2	96.6
BioSyn (dense only)	90.7	92.9	96.6
Ours (w/ BioSyn embeddings)	91.0	93.1	96.7
Ours (Bert features)	86.2	90.8	96.1

5.2.3. Span-based NER

In this section, we give the results of the span formulation for the NER task. As with the other experiments, after finding a hyperparameter setting that works reasonably well, we kept it and varied pre-trained transformer encoders.

Below we see the model performance in different settings. Once again vanilla BERT lies behind its domain-specific variants. Ablation study on pooling functions shows concatenating max pool, width, and “[CLS]” representation works best and self-attention on span tokens interestingly performs worse among choices even though it has more learnable parameters. Also, note that joint learning with the NEN task brings another small improvement to the results. When compared to the sequence tagging approach, it brings similar performance.

Table 5.9. Accuracy scores of the top prediction of the span based NER models with different modeling choices.

Model	NCBI-disease	BC5CDR
BERT	83.7	79.6
BioBERT	88.5	82.5
SciBERT (max pool, width, CLS)	88.2	83.0
SciBERT (max pool)	87.9	82.7
SciBERT (avg pool)	87.7	82.5
SciBERT (self attention)	86.5	80.6
SciBERT (joint w/ NEN)	88.5	83.2

6. CONCLUSION

In this thesis we studied named entity recognition and normalization tasks in the biomedical domain. In general, we tried to combine the benefits of using domain-specific pre-trained transformer language models with joint learning between two tasks. Inspired by the great improvements reported in Zhao et al. [1], we first modeled both tasks as sequence tagging problems and replaced word embedding + LSTM pipeline with pre-trained transformer language models such as BioBERT and SciBERT. Unfortunately, we did not see significant improvements in performance in this case.

In our second hypothesis, we proposed a novel model to bring contextual information to a very successful BERT-based normalization model BioSyn [13]. Because of computational constraints, we performed less optimal training setups, yet we saw slight but consistent improvement over the mention-only model. Finally, we also formulated the NER tasks using span embeddings and experimented with joint learning between span-based NER and NEN tasks. Span-based NER models performed very similarly to the sequence tagging NER models, but they have benefited from the joint training slightly.

As future work, we would like to study incorporating document-level context into both NER and NEN tasks as we saw a small improvement after adding sentence context. Another fruitful-looking approach is creating graph embedding of concepts from the ontology and using them in the normalization pipeline, to exploit the hierarchical structure between concepts. Finally, a BERT-based variant of TaggerOne model [9] that uses a semi Markov model to obtain and score segmentations would be an interesting direction. In this case, both tasks will be modeled in an end-to-end fashion and cascading errors won't pose a problem as correct but low scoring segmentations from the NER module will be boosted by the NEN module as they will have greater match scores.

REFERENCES

1. Zhao, S., T. Liu, S. Zhao and F. Wang, “A Neural Multi-Task Learning Framework to Jointly Model Medical Named Entity Recognition and Normalization”, *ArXiv*, Vol. abs/1812.06081, 2019.

2. Landhuis, E., “Scientific Literature: Information Overload”, *Nature*, Vol. 535, No. 7612, pp. 457–458, 2016.

3. NIH, *MEDLINE PubMed Production Statistics*, 2020, `https://www.nlm.nih.gov/bsd/medline_pubmed_production_stats.html`, accessed in August 2021.

4. Chen, Q., A. Allot and Z. Lu, “LitCovid: An Open Database of COVID-19 Literature”, *Nucleic Acids Research*, Vol. 49, pp. D1534 – D1540, 2021.

5. NIH, *LitCovid*, 2020, `https://www.ncbi.nlm.nih.gov/research/coronavirus`, accessed in August 2021.

6. Lee, J., S. S. Yi, M. Jeong, M. Sung, W. Yoon, Y. Choi, M. Ko and J. Kang, “Answering Questions on COVID-19 in Real-Time”, *ArXiv*, Vol. abs/2006.15830, 2020.

7. Köksal, A., H. Dönmez, R. Özçelik, E. Ozkirimli and A. Özgür, “Vapur: A Search Engine to Find Related Protein - Compound Pairs in COVID-19 Literature”, *bioRxiv*, 2020.

8. Xu, J., S. Kim, M. Song, M. Jeong, D. hyeon Kim, J. Kang, J. F. Rousseau, X. Li, W. Xu, V. Torvik, Y. Bu, C. Chen, I. A. Ebeid, D. Li and Y. Ding, “Building A PubMed Knowledge Graph”, *Scientific Data*, Vol. 7, 2020.

9. Leaman, R. and Z. Lu, “TaggerOne: Joint Named Entity Recognition and Normal-

ization with Semi-Markov Models", *Bioinformatics*, Vol. 32, No. 18, pp. 2839–2846, 2016.

10. Habibi, M., L. Weber, M. Neves, D. L. Wiegandt and U. Leser, "Deep Learning with Word Embeddings Improves Biomedical Named Entity Recognition", *Bioinformatics*, Vol. 33, No. 14, pp. i37–i48, 2017.

11. Lee, J., W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So and J. Kang, "BioBERT: A Pre-trained Biomedical Language Representation Model For Biomedical Text Mining", *Bioinformatics*, 2019.

12. Beltagy, I., K. Lo and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text", *Empirical Methods in Natural Language Processing*, 2019.

13. Sung, M., H. Jeon, J. Lee and J. Kang, "Biomedical Entity Representations with Synonym Marginalization", *Association for Computational Linguistics*, 2020.

14. Liu, F., E. Shareghi, Z. Meng, M. Basaldella and N. Collier, "Self-alignment Pre-training for Biomedical Entity Representations", *ArXiv*, Vol. abs/2010.11784, 2020.

15. Yoon, W., C. H. So, J. Lee and J. Kang, "CollaboNet: Collaboration of Deep Neural Networks for Biomedical Named Entity Recognition", *Computing Research Repository*, Vol. abs/1809.07950, 2018.

16. Cohan, A., S. Feldman, I. Beltagy, D. Downey and D. S. Weld, "SPECTER: Document-level Representation Learning Using Citation-informed Transformers", *ArXiv*, Vol. abs/2004.07180, 2020.

17. Chen, T., S. Kornblith, M. Norouzi and G. E. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations", *ArXiv*, Vol. abs/2002.05709, 2020.

18. Ruder, S., *NLP's ImageNet Moment Has Arrived*, 2018, `https://ruder.io/nlp-imagenet`, accessed in September 2021.

19. Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep Contextualized Word Representations", *North American Chapter of the Association for Computational Linguistics*, 2018.

20. Howard, J. and S. Ruder, "Universal Language Model Fine-tuning for Text Classification", *Association for Computational Linguistics*, 2018.

21. Radford, A. and K. Narasimhan, *Improving Language Understanding by Generative Pre-Training*, 2018, `https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf`, accessed in September 2021.

22. Devlin, J., M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

23. Mikolov, T., K. Chen, G. S. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space", *International Conference on Learning Representations*, 2013.

24. Vaswani, A., N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, "Attention Is All You Need", *ArXiv*, Vol. abs/1706.03762, 2017.

25. Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach", *ArXiv*, Vol. abs/1907.11692, 2019.

26. Li, J., A. Sun, J. Han and C. Li, "A Survey on Deep Learning for Named Entity

Recognition", *ArXiv*, Vol. abs/1812.09449, 2018.

27. Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. P. Kuksa, "Natural Language Processing (Almost) from Scratch", *Journal of Machine Learning Research*, Vol. 12, pp. 2493–2537, 2011.

28. Zhang, X., J. J. Zhao and Y. LeCun, "Character-level Convolutional Networks for Text Classification", *ArXiv*, Vol. abs/1509.01626, 2015.

29. Lample, G., M. Ballesteros, S. Subramanian, K. Kawakami and C. Dyer, "Neural Architectures for Named Entity Recognition", *North American Chapter of the Association for Computational Linguistics*, 2016.

30. Akbik, A., D. A. J. Blythe and R. Vollgraf, "Contextual String Embeddings for Sequence Labeling", *International Conference on Computational Linguistics*, 2018.

31. Yu, J., B. Bohnet and M. Poesio, "Named Entity Recognition as Dependency Parsing", *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6470–6476, Association for Computational Linguistics, Online, 2020.

32. Eberts, M. and A. Ulges, "Span-based Joint Entity and Relation Extraction with Transformer Pre-training", *ArXiv*, Vol. abs/1909.07755, 2020.

33. Cucerzan, S., "Large-Scale Named Entity Disambiguation Based on Wikipedia Data", *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 708–716, Association for Computational Linguistics, Prague, Czech Republic, 2007.

34. Hoffart, J., M. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater and G. Weikum, "Robust Disambiguation of Named Entities in Text", *Empirical Methods in Natural Language Processing*, 2011.

35. Wu, L. Y., F. Petroni, M. Josifoski, S. Riedel and L. Zettlemoyer, "Scalable Zero-shot Entity Linking with Dense Entity Retrieval", *Empirical Methods in Natural Language Processing*, 2020.

36. Hettne, K. M., R. H. Stierum, M. J. Schuemie, P. J. Hendriksen, B. J. Schijvenaars, E. M. v. Mulligen, J. Kleinjans and J. A. Kors, "A Dictionary to Identify Small Molecules and Drugs in Free Text", *Bioinformatics*, Vol. 25, No. 22, pp. 2983–2991, 2009.

37. Wei, C.-H., B. R. Harris, H.-Y. Kao and Z. Lu, "TmVar: A Text Mining Approach For Extracting Sequence Variants in Biomedical Literature", *Bioinformatics*, Vol. 29 11, pp. 1433–9, 2013.

38. Leaman, R., C.-H. Wei and Z. Lu, "TmChem: A High Performance Approach for Chemical Named Entity Recognition and Normalization", *Journal of Cheminformatics*, Vol. 7, pp. S3 – S3, 2015.

39. Leaman, R., R. Dogan and Z. Lu, "DNorm: Disease Name Normalization with Pairwise Learning to Rank", *Bioinformatics*, Vol. 29, pp. 2909 – 2917, 2013.

40. Leaman, R. and Z. Lu, "TaggerOne: Joint Named Entity Recognition and Normalization with Semi-Markov Models", *Bioinformatics*, Vol. 32, No. 18, pp. 2839–2846, 2016.

41. Phan, M. C., A. Sun and Y. Tay, "Robust Representation Learning of Biomedical Names", *Association for Computational Linguistics*, 2019.

42. Ji, Z., Q. Wei and H. Xu, "BERT-based Ranking for Biomedical Entity Normalization", *AMIA Joint Summits on Translational Science proceedings. AMIA Joint Summits on Translational Science*, pp. 269–277, 2020.

43. Pyysalo, S., F. Ginter, H. Moen, T. Salakoski and S. Ananiadou, "Distributional Semantics Resources for Biomedical Text Processing", *Proceedings of Languages*

in Biology and Medicine, 2013.

44. Wang, X., Y. L. Zhang, X. Ren, Y. Zhang, M. Zitnik, J. Shang, C. Langlotz and J. Han, "Cross-type Biomedical Named Entity Recognition with Deep Multi-Task Learning", *Bioinformatics*, Vol. 35 10, pp. 1745–1752, 2019.

45. Pyysalo, S., *standoff2conll*, `https://github.com/spyysalo/standoff2conll`, accessed in September 2021.

46. Clark, K., M.-T. Luong, Q. V. Le and C. D. Manning, "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators", *ArXiv*, Vol. abs/2003.10555, 2020.

47. Johnson, J., M. Douze and H. Jégou, "Billion-Scale Similarity Search with GPUs", *IEEE Transactions on Big Data*, Vol. 7, pp. 535–547, 2021.

48. Dogan, R., R. Leaman and Z. Lu, "NCBI Disease Corpus: A Resource For Disease Name Recognition and Concept Normalization", *Journal of biomedical informatics*, Vol. 47, pp. 1–10, 2014.

49. Davis, A. P., T. C. Wiegers, M. Rosenstein and C. Mattingly, "MEDIC: A Practical Disease Vocabulary Used at The Comparative Toxicogenomics Database", *Database: The Journal of Biological Databases and Curation*, 2012.

50. Li, J., Y. Sun, R. J. Johnson, D. Sciaky, C.-H. Wei, R. Leaman, A. P. Davis, C. Mattingly, T. C. Wiegers and Z. Lu, "BioCreative V CDR Task Corpus: A Resource for Chemical Disease Relation Extraction", *Database: The Journal of Biological Databases and Curation*, 2016.