CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Ensemble model of Bidirectional Encoder Representation from Transformers for Named Entity Recognition

Master's thesis in Computer science and engineering

Carl Jendle

Linus Schönbeck

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Ensemble model of Bidirectional Encoder Representation from Transformers for Named Entity Recognition

Carl Jendle
Linus Schönbeck

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Ensemble model of Bidirectional Encoder Representation from Transformers for Named Entity Recognition

Carl Jendle, Linus Schönbeck

Supervisor: Jonah Brown-Cohen, Department of Computer Science and Engineering
Advisor: Amanda Nilsson, Monocl AB
Examiner: Marina Axelson-Fisk, Mathematical Sciences

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2021

Ensemble model of Bidirectional Encoder Representation from Transformers for Named Entity Recognition

Carl Jendle, Linus Schönbeck
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Named entity recognition (NER) has been widely modeled using Bidirectional Encoder Representations from Transformers (BERT) in state of the art implementations since its appearance in 2018. Various configurations based on BERT models currently hold 4 out of 5 top positions on the GLUE leaderboard, an acknowledged benchmark for natural language processing and understanding. Relying on BERT architecture, a range of NER model designs were investigated to predict entities in a comparatively small set of medical press releases.
The performance of all investigated model designs proved to be boosted with transfer learning using the publicly available datasets Conll2003 and BC5CDR early on in the project. Transfer learning was therefore implemented in the best named entity recognition system found, the separate submodel system under Section 6.3.6. This final design consisted of two submodels, each classifying different entity subsets independently. The Conll and BC5CDR datasets were used for transfer learning in the respective submodels prior to the introduction of medical press release data. The separate submodel system reached an F1-score of 0.79 (Conll model) and 0.78 (BC5CDR model).
The effect of pre-training a selection of publicly available BERT models on the medical press releases was also investigated, but was given less emphasis due to insufficient amounts of data.

Keywords: Transfer learning, natural language processing, named entity recognition, BERT, conditional random field.

Acknowledgements

First and foremost, we would like to thank our advisors at Monocl: Amanda Nilsson and Henrik Alburg. Amanda - thank you for your stellar support and the effort you have put into helping us. As for you Henrik, your help and support has been highly appreciated whenever you haven't been busy saving the world.

We would also like to thank our Chalmers supervisor, Jonah Brown Cohen, for your invaluable input and insights along the way as well as your soothing presence and infectious optimism.

Lastly, we would like to thank Monocl employees Yurii Khoroshchak, Khrystyna Tkachuk and Uliana Pylypiv. Without your help in providing the medical press releases, there would be no model.

Carl Jendle and Linus Schönbeck, Gothenburg, May 2021

Contents

List of Figures

List of Tables

1
Introduction

Finding experts within the field of medicine and health science is generally a tedious task. Issues arise when a candidate needs to fulfil an array of criteria in order to be considered for a task provided by a stakeholder. Typical criteria include knowledge within a certain field of medicine, experience in leadership or research, geo-location, previous collaborations and projects et cetera. Sifting through candidates has traditionally been taxing on resources and has generally required the aid of consultants or professional connections. Monocl is actively working on the facilitation of finding these candidates based on a stakeholder's needs by navigating their internal database.

Monocl's internal database is based on information from several public biomedical literature databases. One of the databases, PubMed, contains a vast knowledge base regarding experts and research. Announcements and publications tied to medicine and health science are uploaded to this platform at a staggering rate of more than a million documents per year [2]. Key information from the articles is stored in meta tags and is easily extracted and integrated to Monocl's database today.

Monocl aims to expand the information flow into their platform by integrating more information sources. Biomedical news articles offer time-critical news updates from pharmaceutical companies, often communicated through their websites. Rapidly extracting key information from these sources gives Monocl the opportunity to update their customers with the broadcasts. Hence, biomedical news articles are one information source of particular interest for Monocl. The lack of meta-tags and unstructured text of the website articles are presently an impediment to the integration of information. No biomedical news articles are integrated to the platform today due to the fact that the key information must be extracted from the unstructured text of the article.

1.1 Problem

The problem at hand lies in efficiently processing and extracting the information deemed valuable for Monocl for integration with their database. Information of interest can for instance be the names of the involved medical doctors, information about a new drug that is being tested or the expected outcome of a clinical trial.

The occurrence of previously unseen diseases, drugs or medical doctors is likely in text gathered from biomedical news articles. Thus, simply referring to an existing database for string matching may omit information in an article that is unmistakably relevant. A model that takes context into consideration is anticipated to perform

better in such a case.
Given the continuous influx of biomedical news from a range of websites, manual processing of documents is costly and requires qualified employees to handle. Finding a way to automate this process could potentially have a profound impact on the extraction of information, assuming that the automated approach generates accurate results.

1.2 Goals and Challenges

The goal is, in essence, to create a transfer-learning based named entity recognition model. The model will extract both general entities (names of medical doctors, organisations and locations) as well as clinical entities (drugs and diseases). The input is to be gathered from biomedical news articles, which contain information about investigators, drugs, collaborations and expected outcomes.
The model needs to be as accurate as possible according to metrics defined later in the document, while still being computationally feasible to implement. Seeing as the resulting model is domain-specific and fine-tuned on similar text, the ambition is to reach near state-of-the-art results for named entity recognition.
While semantically equal, a pair of written texts can be, and most often are, inherently different. Extracting entities of notion, such as investigators, drugs etc. and recognising them as such poses a problem if a high accuracy is to be achieved by a suitable metric. The exponential growth complexity of generated text, the scarcity of labelled data, the possibility of previously unknown entities makes any brute-force approach impossible. Thus, more sophisticated methods are required to tackle the matter at hand.
A core issue of this project consists of the scarcity of labelled or annotated data. This poses a challenge as no conventional, supervised learning is achievable. Furthermore, adding more entities such as drugs that are not pre-trained on a large data set in the same manner as BERT can be problematic. Subsequent distinguishing between entities that have traits in common and appear in the same context is at risk of achieving a low accuracy if the pre-training is insufficient.

1.3 Ethical considerations

The biomedical news data used for training and evaluating as well as potential future usage is gathered from publicly available sources and is accessible to any actor. The only apparent ethical consideration to take into account is that of storing text data that may contain personal information regarding out of scope individuals. Individuals that are not experts in the field of medicine are of no interest to the project or Monocl. It means that information regarding these individuals is never stressed upon, used or abused in any way. Furthermore, in the event of personal and sensitive information being present in the gathered text data it is anticipated to make out only a negligible amount of the raw data used. The information is of no use for the models performance and it's presence is simply a consequence of being infeasible to filter out.

Storage of the aforementioned type of text data can be reliant on the GDPR regulations regarding legitimate interest, making it compliant with EU legislation [3]. Exceptions to this principle can be made if the information is considered to be intrusive.
Monocl however is operating under a Swedish Publication License which allows for the storage of the raw data as well as any information extracted from the target public documents. This also applies to the database to which the information is to be funnelled. Swedish constitutional laws regarding freedom of speech safeguard the storage and handling of the information provided that it has been gathered from public sources [4].

2

Theory

This project is connected to Natural Language Processing (NLP) and is connected to a specific branch of NLP called named entity recognition. This method aims to locate and classify named entities in unstructured text. Given the nature of the data to be processed, a good choice of model will be the Bidirectional Encoder Representation from Transformers.

2.1 Bidirectional Encoder Representation from Transformers

The Bidirectional Encoder Representation from Transformers (BERT) is pre-trained on a large corpus of unlabelled text from Wikipedia as well as the Brown Corpus, amounting to a total of approximately 3.3 billion words [5]. BERT is pre-trained on two tasks: Masked Language Modeling and Next Sentence Prediction. Both tasks are unsupervised, which makes it possible to pre-train BERT with unlabelled text. In Masked Language Modeling BERT masks 15% of the tokens in the pre-training data and predicts which token that hides behind each mask. See Figure 2.1 for visualisation of the masking process. During Next Sentence Prediction BERT takes two sentences from the pre-training data as input. These sentences can either be a correct sequence or an incorrect sequence. In case of an incorrect sequence the second sentence is randomly chosen from the training data. The training objective is for BERT to predict whether the first sentence is followed by the second sentence or not. The Next Sentence Prediction is visualised in Figure 2.2.

These data showed that over 310,000 deaths	These data showed that over [MASK] [MASK]
were associated with COVID-19, placing	were associated with COVID-19, placing
it third among the leading causes of death	it third among the leading causes of [MASK]
in 2020. According to the mathematicians,	in 2020. [MASK] to the mathematicians,
there was enough distance between the number	there was enough distance between the number

Figure 2.1: Masked language modeling visualised in a sample text from a medical press release. 15% of the text has been masked out with the [MASK] token in the text to the right.

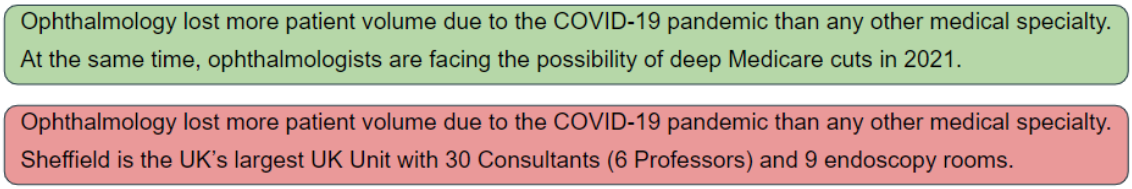

Figure 2.2: Next sentence prediction visualised with an example from a medical press release. The two sentences in green are the actual, correct sequence of sentences. The sentences in red are a false sequence of sentences. The second sentence has been randomly chosen from the training data.

The model was developed by Google engineers and implementation of this encoder architecture helped in surpassing state-of-the-art performance in many NLP tasks. Implementations using various types of BERT currently holds the top three positions in the General Language Understanding Evaluation (GLUE) leaderboard, a test that consists of nine diverse natural language understanding tasks [6]. Specifically, the article [7] showcases how state-of-the-art results for named entity recognition, as well as other tasks, were achieved with a fine-tuning approach on BERT. This was evaluated on a biomedical language counterpart to GLUE, aptly named BLUE. There are two types of BERT: $\text{BERT}_{\text{BASE}}$ that contains 109 million parameters and $\text{BERT}_{\text{LARGE}}$ containing 340 million parameters. They are structurally equal but $\text{BERT}_{\text{LARGE}}$ has larger feature vectors, more encoding blocks and more self-attention layers. In this report we are describing the parameters for $\text{BERT}_{\text{BASE}}$.

BERT is based on the concept of transformers, particularly the encoder part of the transformer. As the name suggests, BERT operates in a bidirectional manner in order to use the contextual information prior to and after any word token. This means that instead of reading input in sequence from left-to-right (or right-to-left) BERT takes a chunk of unstructured text as input directly. First off, the input text is split into tokens in BERTs word vocabulary, consisting of roughly 30.000 elements. This vocabulary contains a variety of different tokens such as whole words, sub-words and single characters. The single characters include alphabetic characters of different languages, integers and special characters.
Each token is then encoded to a feature vector in a high-dimensional vector space of dimension $d = 768$. In terms of input length, BERT has a maximum capacity of 512 encoded tokens per calculation. Compared to simple context-free word embedding methods like word2vec, BERT takes the position in the sentence and token type for each token into account by integrating it into the input feature vector. The token type is simply a vector that signifies whether the investigated word is in sentence A or B, which is a crucial part for the Next Sentence Prediction pre-training. The feature vectors are then used as input to the encoder part of the transformer. The encoder consist of $N = 12$ encoder blocks stacked on top of each other. One encoder block consists of a multi-headed self-attention mechanism, add & normalise layer, a 2-layer feed forward network and another add & normalise layer [8], see Figure 2.3.

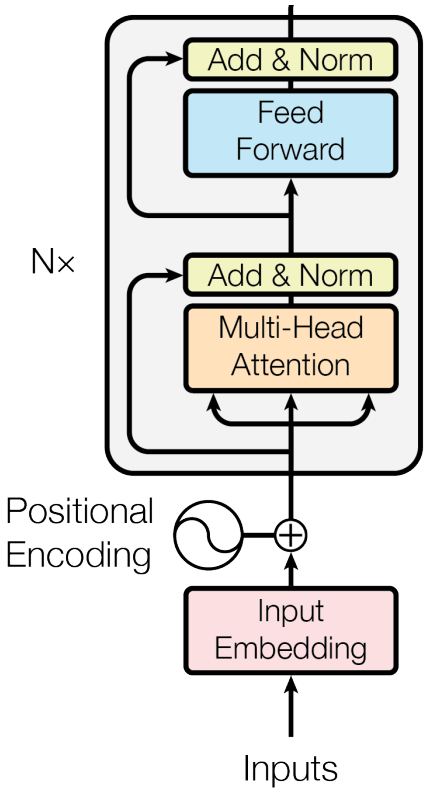

Figure 2.3: Encoder part of a transformer. Source: [1]

The multi-headed self-attention layer computes refined embeddings (new updated feature vectors) by means of weighted averages of all other feature vectors [1]. This is described more in detail in Section 2.2. After the multi-headed self-attention layer an add & normalise layer adds the input and output vectors from the multi-headed self-attention layer and normalises the sum. A residual connection over the multi-headed self-attention layer is necessary because it leads to stronger gradients and better learning [8]. The normalisation reduces co-variate shift and makes higher learning rates possible, leading to faster training. The feed-forward neural network layer is a fully connected one-hidden layer applied individually to each feature vector. This neural network has 768 input neurons, 3072 hidden neurons and 768 output neurons [9] and thus it has the same input and output dimensions. The weights and biases for the neural network are shared between different feature vectors. The last part of the encoder block is another add & normalise layer. Overall BERT is mapping input vectors of dimension $d = 768$ through $N = 12$ encoder blocks to output vectors of the same dimension as the input vectors. It is possible to add layers on these final output vectors and train them for fine-tuning BERT on different Natural Language Processing tasks.

2.2 Attention in BERT

The multi-headed self-attention layer in BERT consists of $h = 12$ parallel self-attention heads. Each self-attention head's update process is independent and has

it's own parameters. The update steps in a self-attention head is described below: [10]

First off, the Query, Keys and Values for all n feature vectors need to be calculated.
For all $i \in \{1, 2, \ldots, n\}$:

$$\text{Query:} \quad q_i = W_Q x_i \tag{2.1}$$
$$\text{Keys:} \quad k_i = W_K x_i \tag{2.2}$$
$$\text{Values:} \quad v_i = W_V x_i \tag{2.3}$$

W_Q, W_K, W_V are matrices with trainable parameters. These matrices are of dimension $[64, 768]$ and project the feature vectors of dimension 768 down to query, key and value vectors of dimension 64. Now, to find weights to update feature vector x_i, use Query q_i and the Keys k_j for all feature vectors:
For all $i, j \in \{1, 2, \ldots, n\}$:

$$Z_{ji} = k_j^T q_i / \sqrt{d} \tag{2.4}$$

d above is the dimension of the query/key vectors. Next, calculate the weights for x_i and update the feature vector $x_i \to y_i$.
For all $i \in \{1, 2, \ldots, n\}$:

$$\text{Weights:} \quad [W_{1i}, W_{2i}, ..., W_{ni}] = \text{softmax}(Z_{1i}, Z_{2i}, ..., Z_{ni}) \tag{2.5}$$
$$\text{New embedding:} \quad y_i = \sum_j W_{ji} v_j \tag{2.6}$$

Observe in Equation 2.5 that the weights are calculated by softmax and therefore also are summed to unity. Observe also in Equation 2.6 that the new feature vector y_i is a weighted average of the Values v_j and NOT the feature vectors x_j. Thus, one self-attention layer maps feature vectors x_i of dimension 768 to refined feature vectors y_i of dimension 64. The multi-headed self-attention layer produces $h = 12$ new vector representations y_i of the feature vector x_i. These representations are different because the parameters for W_Q, W_K, W_V vary between different self-attention layers. All 12 feature vectors y_i from different layers are then concatenated by stacking them on top of each other. This generates larger feature vectors of dimension $d = 12 \cdot 64 = 768$. These vectors are then stacked in a matrix $\widetilde{Y}$, one column for every feature vector. $\widetilde{Y}$ is then multiplied with a matrix of dimension $[768, 768]$, W_0:

$$Y = W_0 \widetilde{Y} \tag{2.7}$$

The matrix W_0 contains trainable parameters. In this way information from all $h = 12$ layers are gathered in the matrix Y and the multi-headed self-attention layer has input and output vectors of the same dimension.

2.2.1 Interpretation of the weights in BERT's self-attention layers

The update weights for every token i, $W_{1i}, W_{2i}, ..., W_{ni}$, in every self-attention layer have a logical interpretation. The tokens with large weights for an update of feature vector x_i are likely tokens that have a linguistic connection to the investigated token i. For example, consider the sentence "Simon is ill and he is going to test himself for covid-19 today.". In Figure 2.4 the weights for the word "himself" are investigated in the example sentence for both BERT-base-uncased and Clinical BERT.

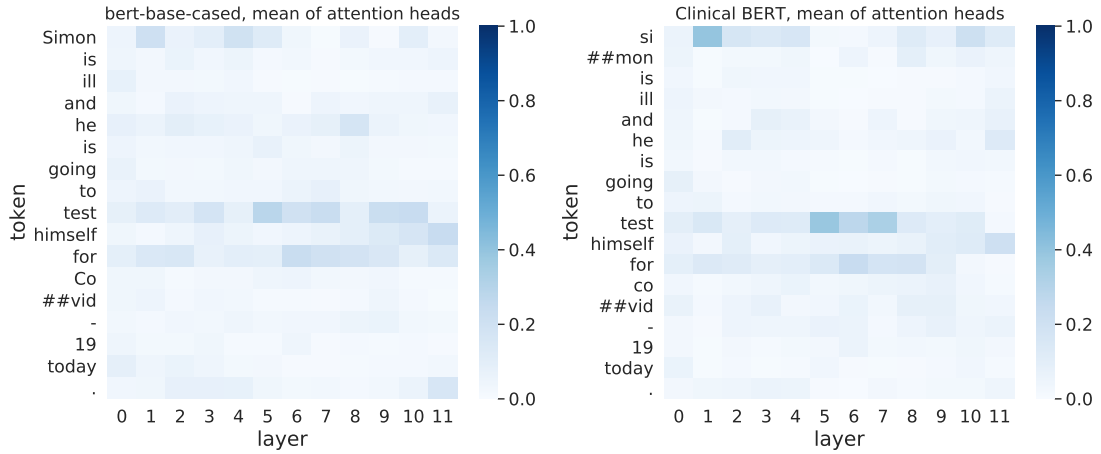

Figure 2.4: Averaged weights over all 12 self-attention heads for every self-attention layer. Comparison of weights between BERT-base-cased and Clinical BERT for the word "himself" in the example sentence stated above. Notice also that the two models have different vocabularies and hence also different tokenizations of the sentence.

It can be seen in Figure 2.4 that both BERT base uncased and Clinical BERT have large weights for the tokens corresponding to the words "Simon", "test", "himself" and "for". Arguably also the word "he" in BERT base uncased. These words are all of linguistic importance to the word "himself" in the sentence. The words "test" and "for" are the neighbouring words to "himself" and describe the context. Of course the weight of the investigated word "himself" is large. "himself" is referring to "Simon" and both models capture this relation by assigning a large weight.

2.3 Transfer Learning for named entity recognition

There are a variety of machine- and deep-learning methods and the intention is to use transfer learning by proceeding from an already pre-trained BERT model. Since BERT was introduced in 2018, several further pre-trained open source BERT models have appeared on Github. It is of particular interest to proceed from a BERT model that has been pre-trained in the medical domain in order to get the best possible starting-point with a high language understanding from the beginning. Choosing

the model with the most similar domain to the biomedical news articles will not only increase the performance of the model, but also decrease the pre-training time. The specific BERT models are discussed in Section 2.7.

2.4 Conditional Random Field

A standard prediction process in a classification problem is to individually predict a class for every feature. In that case no consideration is taken into the sequence of entities. In order for the model to take the sequence of entities into account a Conditional Random Field (CRF) is used in the classifier - more specifically a Linear Chain CRF. By using Linear Chain CRF the loss function is not the regular Cross-entropy loss, as an additional transition part is added to the loss function, L:

$$\text{loss, } L = -\log(p(\mathbf{y}|\mathbf{X})) \tag{2.8}$$

$$p(\mathbf{y}|\mathbf{X}) = \exp\Bigg(\underbrace{\sum_{k=1}^{K} a^{(L+1)}\left(\mathbf{x}_k\right)_{y_k}}_{\text{Emission part}} + \underbrace{\sum_{k=1}^{K-1} V_{y_k,y_{k+1}}}_{\text{Transition part}}\Bigg) \underbrace{/Z(\mathbf{X})}_{\substack{\text{Partition}\\ \text{function}}} \tag{2.9}$$

$$Z(\mathbf{X}) = \sum_{y_1}\sum_{y_2}\cdots\sum_{y_K}\exp\left(\sum_{k=1}^{K} a^{(L+1)}\left(\mathbf{x}_k\right)_{y_k} + \sum_{k=1}^{K-1} V_{y_k,y_{k+1}}\right) \tag{2.10}$$

The emission part is derived from calculating the probability of the correct entity y_k given the input token $\mathbf{x_k}$ for every token $k = 1, 2, ..., K$ in the K tokens long sequence. $a^{(L+1)}\left(\mathbf{x}_k\right)$ is the activation function for the input $\mathbf{x_k}$. Thus, in our case, this is the emission scores received from the feed-forward layer after BERT. Hence $a^{(L+1)}\left(\mathbf{x}_k\right)_{y_k}$ is the emission score for the correct entity y_k. This is just like in regular Cross-entropy loss.
The transition part of the loss function is a measurement of how likely it is that the entity y_k is followed by entity y_{k+1}. $V_{y_k,y_{k+1}}$ is a $K \times K$ matrix containing values for every transition $y_k \rightarrow y_{k+1}$. The transition part is making the model consider predictions of sequences of entities and not just individual predictions of entities. The partition function is a normalization constant for the likelihood $p(\mathbf{y}|\mathbf{X})$. By looking at Equation 2.10 we see that the normalizing constant is just the sum over all possible sequences of entities for the numerator. It is also mathematically possible to find a sequence of entities $\mathbf{y}$ given the emission scores $\mathbf{X}$ that minimizes the loss function in Equation 2.8. This sequence is of interest because it is the most probable sequence of entities given a sequence of tokens. The sequence can finally be found by the Viterbi algorithm. [11]

2.5 Uniform Manifold Approximation and Projection

In order to visualize the high dimensional, contextualized tokens in the BERT model, Uniform Manifold Approximation and Projection (UMAP) is used. This can be ap-

plied to UMAP builds a high-dimensional graph representation of a dataset and subsequently creates a lower dimensional graph to be as structurally similar as possible.

Two UMAP parameters used for creating the high dimensional graph are are of k, the number of nearest neighbours and r_{max}, the max radius.

In constructing the high-dimensional graph, radii around each data vertex expands until they intersect with k other radii, adding weighted, directed edges between vertices at each intersection.

Setting a small r_{max} leads to more isolated simplices whereas a high r_{max} may lead to a too high connectivity between clusters. As the radius growth of each vertex depends on the number of intersections with other radii being less than k, the value of k also has an impact on connectivity.
Each vertex considers its own k nearest neighbours based on outgoing edges and calculates values for $\rho_i = min\{\|x_i - x_{ij}\|_2 | 1 \leq j \leq k\}$ where x_{ij} is the j:th nearest neighbour for vertex x_i and σ_i such that $log_2(k) = \sum_{j=1}^{k} exp(-\frac{\|x_i - x_{ij}\|_2 - \rho_i}{\sigma_i})$
The σ_i and ρ_i values help defining the weight function w for outgoing edges, $w(x_i, x_{ij}) = exp(-\frac{\|x_i - x_{ij}\|_2^2 - \rho_i}{\sigma_i})$. These weights are viewed as probabilities of membership connection between two vertices and the nearest neighbour x_{ij} is guaranteed to be locally connected to the origin x_i.

Viewing X as the set of high-dimensional data points, a directed graph $\vec{G} = (V, E, w)$ can be created where the vertices V refer to the points in X and the edges E with respective weights w are derived from the radius intersections.

Figure 2.5: 2D construction representation of high-dimensional graph for an increasing maximum distance and nearest neighbours $k = 2$.

Figure 2.6: 2D construction representation of high-dimensional graph for an increasing maximum distance and nearest neighbours $k = 2$. The central node stops expanding as it reaches 2 nearest neighbours and the final edges added are directed.

By letting A be the weighted adjacency matrix of the directed, acyclic graph $\vec{G}$, the symmetric matrix B is considered

$$B = A + A^T - A \cdot A^T \tag{2.11}$$

The resulting symmetrical matrix B is the weighted adjacency matrix for the undirected UMAP graph G.
While there are a number of methods for reducing the dimension of a graph, UMAP relies on a force directed graph layout algorithm in low-dimensional space. The forces $\vec{F}_r$ and $\vec{F}_a$ denote a repulsive and an attractive force between each vertex pair $(\mathbf{y_i}, \mathbf{y_j})$ where $\mathbf{y_i}, \mathbf{y_j} \in \mathbb{R}^d$ and d is the target dimension.

$$\vec{F}_a = \frac{-2ab\|\mathbf{y_i} - \mathbf{y_j}\|_2^{2(b-1)}}{1 + \|\mathbf{y_j}\|_2^2} w(x_i, x_j)(\mathbf{y_i} - \mathbf{y_j}) \tag{2.12}$$

$$\vec{F}_r = \frac{2b}{((\epsilon + \|\mathbf{y_i} - \mathbf{y_j}\|_\mathbf{2}^\mathbf{2})(\mathbf{1} + \mathbf{a}\|\mathbf{y_i} - \mathbf{y_j}\|_\mathbf{2}^\mathbf{2b}))} (1 - w(x_i, x_j))(\mathbf{y_i} - \mathbf{y_j}) \tag{2.13}$$

a and b are hyperparameters and ϵ is a small constant used for numerical stability. The system stabilizes as a force equilibrium is reached, at which point the weighted graph H constituted by the points $\{\mathbf{y_i}\}_{i=1...N}$ approximates the original, high-dimensional graph G as well as the method allows. [12] [13]

2.6 Named Entities

A named entity in named entity recognition corresponds to a physical or abstract real-world object and carries semantic significance. An instance of a named entity has, apart from it's entity adhesion, a name string. In terms of entity classification, a variety of entities can be divided into separate classes. In order to extract correct entity-string pairs, entities must first be properly classified. As previously mentioned, each entity has an adhering name. Determining string boundaries for these names can be more or less problematic and matching is divided into the exact and partial categories. In the case of discontinuous entity names, a string boundary has an exact matching if the name is fully encapsulated. For instance, extracting

only a first name of a medical doctor and leaving out the surname or extracting "Biopharma" from "Hansa Biopharma" would yield a partial boundary matching. Working with the IOB-format is a common practice in named entity recognition, where text is annotated into three categories - beginning of entity, inside entity and other. This format allows entities that consist of more than one word to be mapped to a single entity for future extraction.

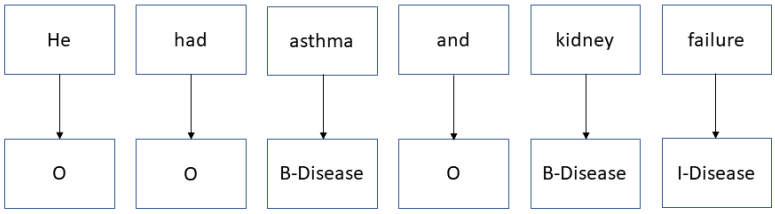

Figure 2.7: IOB-annotation of an arbitrary sentence. Two separate disease entities are tagged as well as the omitted entities.

In general text, a majority of the entities will always be classified as omitted, or "other". Viewing named entity recognition as a classification problem means that there is an imbalance of annotated data between labels. This can be addressed by weighting the cost function based on the observed inverse frequency of each label during the training.

2.7 Related Work

Several publications have been made regarding clinical named entity recognition, i.e. research that has been done to classify medical terms as drugs, diseases, symptoms etc. These articles mainly revolve around text from Electronic Health Records, for instance in the study "Incorporating dictionaries into deep neural networks for the Chinese clinical named entity recognition" [14]. Some research has focused on named entity recognition for publications from PubMed. Examples include articles where documents from PubMed have been examined for names of diseases or chemicals [15][16]. However, we can not find research specified on clinical named entity recognition for press releases or any other biomedical news sources. Neither have we found research that is trying to capture the overall picture of a clinical trial text by identifying several entities such as scientists/medical doctors, drugs, names of hospitals/research facilities, expected outcome of the trial etc. The difference lies in that most research seems to focus on finding specific entities such as the names of diseases and chemicals. BERT is also a somewhat new (2018) word embedding model and it is interesting to see how well it can perform in this area. BERT has been specialized in the biomedical domain by pre-training it unsupervised on PubMed abstracts (4.5B words) and PubMed articles (13.5B words) in the article [17]. BERT has also been specialized in the clinical domain by starting with BERT and the pre-trained BioBERT model in [17] and pre-train these two models further

on 2 million MIMIC (Medical Information Mart for Intensive Care) notes in the Clinical BERT study [18]. Another notable flavor of BERT in the biomedical domain is the BlueBERT model, which differs from the Clinical BERT model only in the sense that the Wikipedia and Brown corpora are excluded from the pre-training [7].

3
Methods

Named Entity Recognition is interchangeable with a multi-class classification problem and an array of designs for the task is conceivable. Based on the available annotated data, the aim is to derive an ensemble model of low dimensional classifiers with BERT models as a key component. The methods described below are reliant on the pre-existing BERT architecture and corresponding libraries for pre-training and pre-processing.

3.1 BERT Model

All of the BERT-base models, BioBERT and Clinical BERT are open source models available at Huggingface [19]. They all have the same structure, which includes 12 encoding blocks, all with 12 parallel self-attention layers and a feed-forward network described in section 2.1. Each self-attention layer has matrices for querys, keys and values. The model also includes three matrices for encoding tokens, these are for word embedding, position embedding and token embedding. As mentioned in 2.1, BERT does not simply encode each token based on the token itself, but also the token position in the sentence and which sentence the token is part of. All of the 109 million parameters, including the ones mentioned above, are reachable in the model and can be gauged. It is a huge advantage that BERT's parameters are non-proprietary components. For example the analysis of how the word embedding vectors are altered during pre-training and how the self-attention layers weights are divided for an input sentence would not be possible without it.

3.2 Intended Workflow

As the intention is to develop an ensemble model of separate BERT classifiers, each separate model will be specialized in extracting a subset of the target entities. The reasoning behind this is based on the quality of the data available and is discussed in detail in Section 4. First off, we are using BERT Base provided by Google Research as well as BioBERT and Clinical BERT, two separate model checkpoints that are further trained on biomedical data [17] [18]. These models are to be pre-trained unsupervised on a corpus consisting of biomedical news articles to make the BERT models domain specific. The reason for this further stage of training is due to the text structure in the news articles being different from the text structure in the training data the BERT models have previously been trained on. The pre-training consists of two tasks - Masked Language Modelling and Next Sentence Prediction and are

performed by using scripts provided by Google Research. The resulting models are subsequently ported from Tensorflow to PyTorch via the Transformer library provided by HuggingFace [20]. For this extra pre-training step the unstructured text needs to be split into smaller parts, since BERT has a maximum input length of 512 tokens. Tokenization of the input strings and encoding to input vectors makes the text ready for training. To verify that the pre-training is heading in the right direction it is important to evaluate the training continuously. Metrics, such as loss function and accuracy, are calculated at given checkpoints for the pre-training evaluation set and logged in Tensorboard. Fine-tuning BERT for named entity recognition requires annotated training data with relevant entities. The entities this project has settled to include:

- Person
- Organisation
- Geo-locations
- Chemicals
- Diseases

External annotated data sets discussed in Section 4.2 are pre-processed and used to fine-tune BERT. In this way BERT is able to classify every token. Following fine-tuning, the model will finally be evaluated on the manually annotated set of news articles.

3.3 Initializing CRF transitions

Some transitions are logically impossible and as the model is trained, the penalty for illegal transitions increases. If such transitions are known beforehand, a common practice is to initialize low values for these specific transitions. Empirical studies and those of our own suggest that this step speeds up convergence and improves overall accuracy - especially if a shortage of data is an issue.
When working with annotated data on an IOB-format, the transitions below are logically impossible

Transition from	Transition to
O	$I - Entity_i$
$I - Entity_i$	$I - Entity_j$
$B - Entity_i$	$I - Entity_j$

Table 3.1: Illegal transitions for the conditional random field for all $i \neq j$.

As the Viterbi algorithm is used to decode the final sequence, these transitions are initialized to low values.

3.4 Evaluation

Following training, the generated output is to consist of a collection of extracted entity-string pairs. The extracted entity-string pairs are stored separately in an

annotation file which denotes the entity ID, character off-sets in the file, type of entity and the string found in between the character off-sets. This standard is commonly referred to as brat standoff format [21]. These string-entity pairs need to be extracted and parsed for future integration into the Monocl database.
Due to the negative impact of false positives and false negatives, the evaluation metric has to penalise these occurrences. Furthermore, partially correct output must be considered better than fully incorrect output.
Results can be divided into the following categories when compared to a gold standard annotation:

- COR - Correct, entity-string matches golden standard annotation
- INC - Incorrect, no match in system output and golden standard annotation
- MIS - Missing, entity-string pair in golden standard annotation not recognized by the system
- PAR - Partial, the string in system output is not fully captured
- SPU - Spurious, inferred by the system but not present in the golden standard annotation

Here, the golden standard annotation signifies the manual annotation made by an expert and are assumed to be correct. Based on these categories, evaluation can be done via recall, precision and F1-score, taking the following measurements into account:

- Strict - correctness of both entity type and string boundary
- Exact - boundary matching for string regardless of type
- Partial - partial boundary matching for string regardless of type
- Type - type match for entity, some overlap of string boundary required

The number of gold standard annotations contributing to the final score are referred to as "possible" (POS),

$$POS = COR + INC + PAR + MIS = TP + FN \tag{3.1}$$

and the number of annotations produced by the named entity recognition system is referred to as "actual" (ACT),

$$ACT = COR + INC + PAR + SPU = TP + FP \tag{3.2}$$

When considering the strict evaluation, the precision (P) and recall (R) are calculated through

$$P_{strict} = \frac{COR}{ACT} = \frac{TP}{TP + FP} \tag{3.3}$$

$$R_{strict} = \frac{COR}{POS} = \frac{TP}{TP + FN} \tag{3.4}$$

and for the partial counterpart

$$P_{partial} = \frac{COR + 0.5 \cdot PAR}{ACT} \tag{3.5}$$

$$R_{partial} = \frac{COR + 0.5 \cdot PAR}{POS} \tag{3.6}$$

$$F1_{strict} = \frac{2P_{strict} \cdot R_{strict}}{P_{strict} + R_{strict}} \tag{3.7}$$

$$F1_{partial} = \frac{2 \cdot P_{partial} \cdot R_{partial}}{P_{partial} + R_{partial}} \tag{3.8}$$

Furthermore, assuming a small evaluation set, the robustness of scores must be analysed for a range of elements used in the evaluation set.

The importance of recall and precision is generally task dependent. A generalized version of the F_1 score is the F_β score:

$$F_\beta = (1 + \beta^2) \cdot \frac{P \cdot R}{(\beta^2 \cdot P + R)} \tag{3.9}$$

With this modification, recall is considered β times as important as precision in F_β score. Higher values of β promote a high recall at the expense of a lower precision.

3.5 Recall over Precision

Precision is more important than recall if the cost of false positives is high - for instance if you are to defuse an MBV-78-A1 anti-personell mine. Wrongfully classifying a wire as safe to cut will have devastating results and the alternative to simply avoid it altogether might be more appealing.
Recall on the other hand is more important if the cost of false negatives is high. In testing a patient for a treatable, deadly disease, the consequences of sending a diseased patient away are direr than investigating a false positive further.
In this task, the recall is considered to be of greater importance as the model output is to be filtered after its generation. Omitted entities can not be added with ease, but posterior filtering mitigates the negative effect of false positives. By modifying the weights in the cross-entropy loss for a linear feedforward network, the recall for an entity or class increases as its loss weight goes up. The trade off between precision and recall is asymptotical to the worst score, 0, as either approaches the perfect score 1.

3.6 Baseline comparison

As various BERT model implementations are widely occurring in the GLUE leader board, suitable baseline comparisons are best centered around similar designs. A simple baseline comparison can be done by comparing a single BERT-model to the proposed ensemble model. Another baseline comparison can be done by using the RoBERTA based named entity recognition training pipeline provided by spaCy.

3.7 Limitations

In order to pre-train the BERT-model properly, the amount of unstructured or non-annotated test data needs to be of sufficient size. In the project the sparcity of

biomedical press releases may restrict the pre-training step. If no performance boost is achieved from pre-training the model, the pre-training step can be abandoned for other tasks.

The number of entities considered will be restricted, not only in order to promote overall accuracy but also to leverage publicly available datasets. This helps in avoiding unnecessary complexity to the model while still extracting the most relevant entities.

The required computation time for downstream tasks is not negligible and the tuning of hyperparameters must be done strategically as any exhaustive search will be too time- and resource consuming. Consulting related works for approximate hyperparameters will help minimizing the time put into this step.

4
Data

To make full usage of the BERT architecture for named entity recognition, one can either rely solely on annotated data or a combination of a small amount of annotated data and a greater portion of unstructured text. These datasets, both stemming from the same area of expertise and having semantic similarity, are used for two separate tasks - the pre-training and the downstream task.

4.1 Unstructured Dataset

Pre-training a BERT model prior to a downstream task is heavily dependent on vast amounts of unstructured text data.
As the annotated data consists of text scraped from various medical news outlets, the unstructured text used for the pre-training is gathered from the same sources in order to maximize semantic similarity and relevance.
A significant change in performance is not anticipated in our case. Comparing our current volume of training data to that of other articles suggests that we have way to few press releases for pre-training. Therefore, a careful choice of model checkpoint is deemed more important as stated in Section 2.3.

4.2 Annotated Datasets

Generating a sufficient amount of text data for our model is costly as it requires both text retrieval and manual annotation by a qualified person. However, subsets of the entities are found in a collection of publicly available datasets. Leveraging these datasets could help minimizing the amount of work put into generating a dataset. Another benefit of using well studied datasets is the possibility to compare model performance to pre-existing benchmarks.
The publicly available datasets considered and their respective annotations are the BC5CDR (disease and chemical dataset, 3 MB [22]) and the Conll2003 (person, location and organisation dataset, 4.5 MB [23]). These datasets are already splitted into training, validation and test set. Hence, this enables a standard for a fair international state-of-the-art competition between different named entity recognition models.
A custom dataset, Monocl2021, has also been tagged and contains 1.5 MB worth of annotations for persons, locations, organisations, diseases and chemicals. The text is gathered from publicly available sources for biomedical news and has been annotated by Monocl and is used for evaluation of the finalized ensemble model.

The datasets vary in terms of volume as well as raw count of annotations.

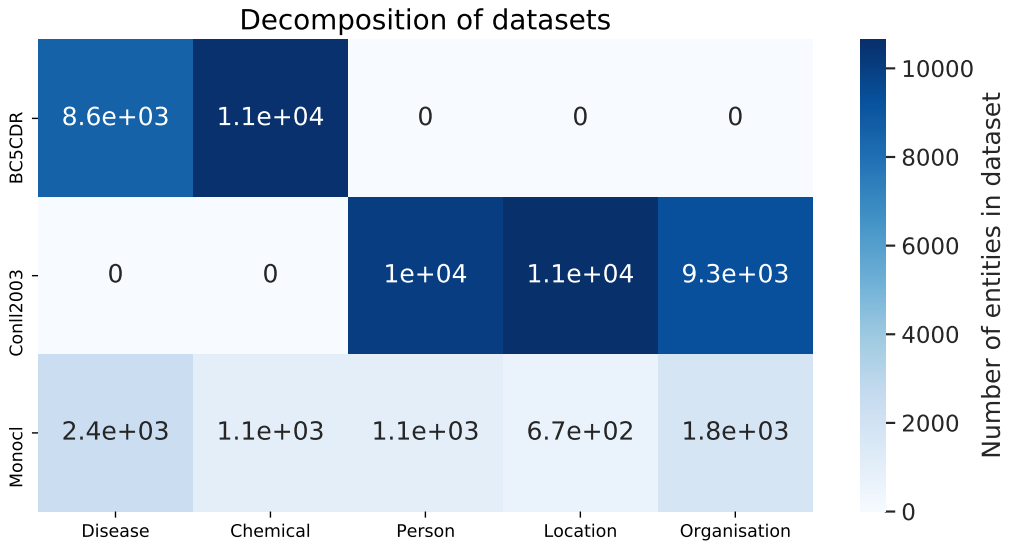

Figure 4.1: Raw entity count of the considered datasets.

4.3 Entity density in datasets

In terms of target classes or entities, the data is imbalanced with the 'Other' class being more frequent. Comparing the entity density in the considered public datasets to the Monocl data, it is evident that the medical press releases are more sparse in terms of target entities.

	Person	Location	Organisation	Disease	Chemical
Conll2003	0.062	0.058	0.058	0	0
BC5CDR	0	0	0	0.046	0.058
Monocl	0.014	0.004	0.008	0.016	0.008

Table 4.1: Table of entity density in the datasets based on total word count.

4.4 Conflicting data

A shortcoming of using separate datasets in the model is the potential occurrence of non-annotated entities in the public datasets. Seeing as locations and organisations are out of scope for the NCBI dataset, these entities are simply left unannotated. The severity of this fact can be measured by simply using all available data in a naive manner, making no adjustments to the model to handle the ambiguous data.

4.5 Pre-processing

The input shape needed for pre-training and fine-tuning differs as the methods are unsupervised and supervised respectively. Both methods need full sentences as input

and the fine-tuning input needs annotations for each word.

4.5.1 Pre-training

In order to transform raw text to data suitable for the pre-training of the BERT-model, the text needs to be segmented into sentences. These sentences are used for the unsupervised next sentence prediction (NSP) as well as the masked language modelling (MLM). This step is facilitated by the spaCy library which provides flexible pipelines for sentence segmentation [24]. Following this, the sentences are cast to tensorflow examples and used for the NSP and MLM tasks.

4.5.2 Fine-tuning

The pre-processing from training data to eligible input to BERT is handled via the tokenisation method described in 2.1. The tokenisation process varies with an upper- or lower-cased argument as the vocabulary is case sensitive. On one hand, countries, names etc. are generally upper cased and a lower cased representation of them may not be present in the vocabulary. On the other hand, words that are positioned in the beginning of a sentence and thus upper cased are at also at risk of not being present in the vocabulary, despite the presence of a lower cased counterpart. Depending on the entities to be extracted, this may have an impact as pre-trained word embeddings are likely to be better than the piecewise tokenisation. Choosing separate casings for two different parts of the ensemble model may provide an overall increase in performance score. However, the length of a tokenised string can differ based on the casing.
To showcase this behaviour, consider the following sentence tokenized by BERT with and without lower-casing:

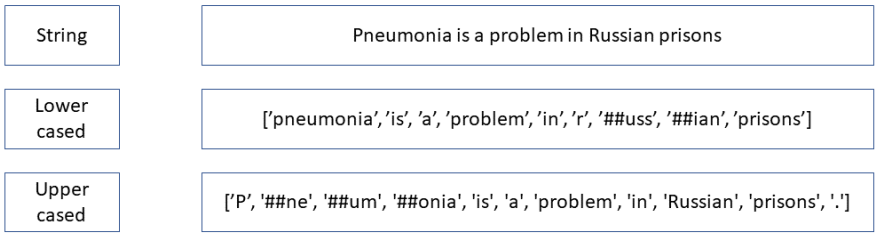

Figure 4.2: Example string tokenised with lower and upper case settings.

This disparity of token lengths must also be addressed if ensemble models are to be used as the output of each model must be mapped together.

5
Model

In order to circumvent the issue of contradictory data, an ensemble model of three separate BERT-based classifiers is proposed. Each model is responsible for classifying a subset of the entities considered based on the annotated datasets. The individual models are merged in a unifying classifier in order to achieve an ensemble model.

5.1 Individual Models

Each individual model consist of a BERT encoder and a simple feed-forward network which reduces the dimension of the encoder output to the number of true entity labels in the subset. These models are trained in a vacuum and use different pre-trained BERT models. The choice of BERT model is based on the performance on each subset of the entity recognition task. In Figure 5.1 the three separate BERT-based models are listed. The starting BERT models are selected as BERT-base cased, BioBERT and Clinical BERT. Notable also is that the different feed-forward networks maps down to different dimensions, depending on the number of entities in the subset. The output is mapped to the label dimension $2 \cdot n_{entities} + 1$.

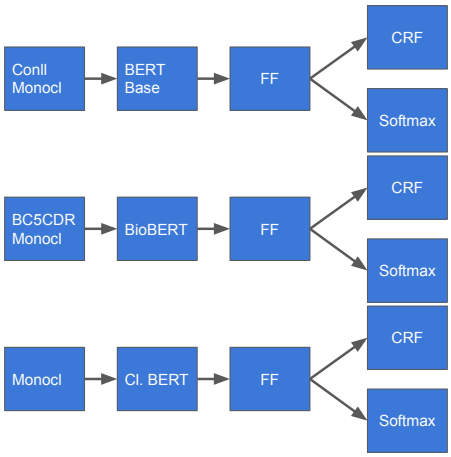

Figure 5.1: The three individual models. Each model is trained and operates individually. Text data is tokenized and each contextualized word embedding is reduced from 768 to the number of target labels, followed by softmax classification or further processing in the CRF.

5.2 Classification with unconnected individual models

In order to leverage the pre-trained BERT models available and circumvent out of scope entities present in the public datasets, multiple models can work in parallel.

If the target entities for the models are disjoint, a rule-based classifier can be used in order to lower model complexity. This method generates double output and each model has precedence over the other when classifying a singular token as one of its own entities.

In order for this method to prove effective, the contextualized tokens must be easily separable on entity level. In the fringe case of a singular token being classified as a relevant entity by both models, cruder methods such as maximum score or a frequency based classifier can be used if a lower complexity is to be retained.

Denoting the subset of entities for two models as S_1 and S_2, the following rule based reduction can be applied:

Final classification method		
Model 1 output	Model 2 output	Output result
O	O	O
$E_1 \in S_1$	O	E_1
O	$E_2 \in S_2$	E_2
$E_1 \in S_1$	$E_2 \in S_2$	$\omega(E_1, E_2)$

Table 5.1: Rule based logic for reducing aggregated output from submodels. Each submodel has classification precedence if one of its entities is predicted. In the case of conflicting output, the binary, frequency based function $\omega(E_1, E_2)$ is used.

Intending the model to consider linguistic dependency, the predictions will be made as sequences of entities and not just individual independent entities. This is accomplished by implementing a Linear Chain Conditional Random Field module lastly in the classifier. An advantage with the CRF module is that it comprehends if an entity transition is reasonable or not, based on the frequency this transition has occurred in the training data. This means that sequences containing forbidden transitions, for example 'B-disease' to 'I-chemical', will never be predicted by the CRF module.

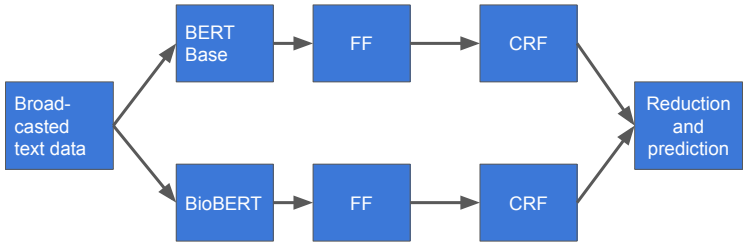

Figure 5.2: The complete model implementing submodels and rule based reduction. The submodels are trained independently on their respective subsets of entities and work in parallell. Their output is reduced for final predictions based on the logic in 5.1.

5.3 Connected Ensemble Model

In order to generate singular classifications for all words, the separate models can be unified by concatenating their outputs and add a final classification layer. This can for instance be done with a feed-forward network followed by a conditional random field module. The encoder-decoder pairs from different models are subsequently put together into an ensemble model. Concretely, every input sequence is individually and independently given to each individual model. The output vectors from each model are concatenated with each other, in order to get a concatenated vector with the information from every individual model. The concatenated vector are fed through the feed-forward network. This feed-forward layer maps the concatenated vectors down to entity vectors. An entity vector has one element for each entity of interest. In this way every token of the tokenized sentence has an emission score for every possible entity. The point in training the feed-forward layer is to find weights and biases that combines the information from the individual models in the best possible way in order to get a reliable ensemble model. The training of the feed-forward network is performed after the training of the individual models. The individual models parameters are kept fixed during this training. In order to include all the relevant entities the training is performed on the Monocl dataset. The output from the feedforward network is subsequently processed in a CRF layer and predictions are made.

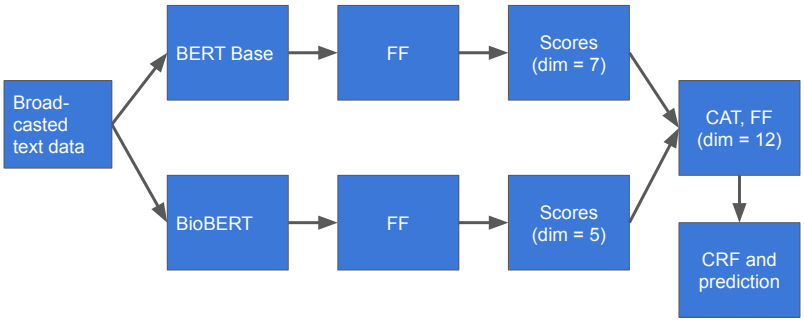

Figure 5.3: The complete Ensemble model. Outputs are concatenated and fed into the CRF in order to generate the best scoring sequence.

6

Results

6.1 Data visualization of high-dimensional token vectors

Sentence and word representations for the BERT CLS tokens and separate word tokens visualized using the UMAP method described in Section 2.5.

6.1.1 Sentence embeddings

The BERT CLS tokens prepended to all sentence examples were evaluated using the UMAP method to visualize sentence similarities or dissimilarities between the datasets. The comparison is depicted in Figure 6.1 below.

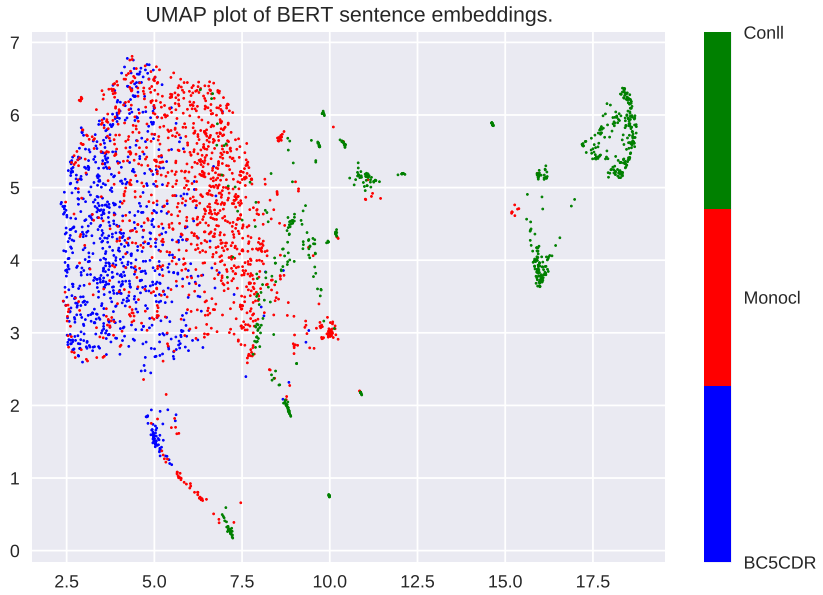

Figure 6.1: UMAP plot of BERT CLS-tokens for sentences. Sentences from the Conll and BC5CDR datasets are fairly disjoint while sentences from the Monocl dataset is distributed across the BC5CDR and Conll datasets.

6.1.2 Word embeddings

To visualize entity separability, entity token embeddings were plotted using the UMAP reduction and is presented in Figure 6.2. All 'Other' tokens were excluded in these measurements.

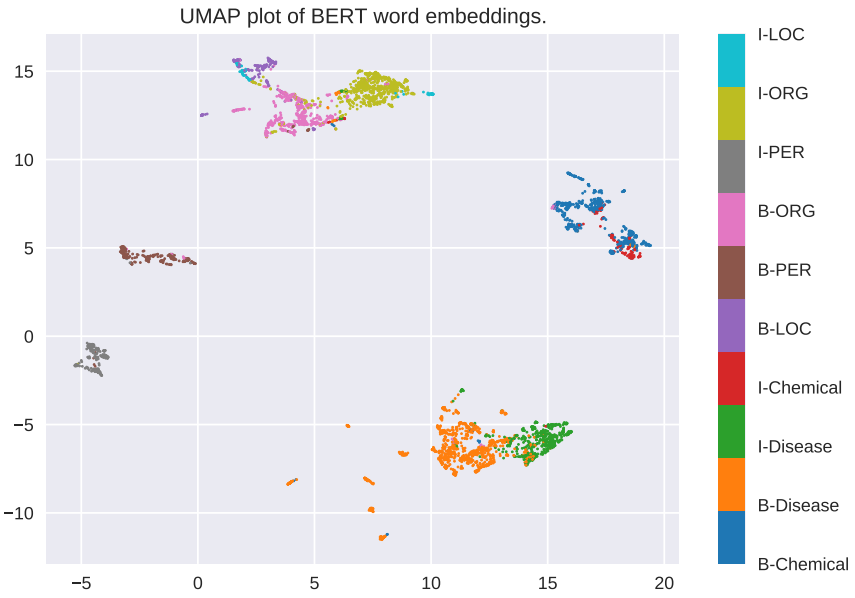

Figure 6.2: UMAP plot of contextualized word embeddings with original labels. Most B/I entity pairs are clearly separable.

A heterogeneous region was identified within the organisation cluster with chemicals and diseases present. This may impact inter-model misclassifications if no extra complexity is introduced to the model.

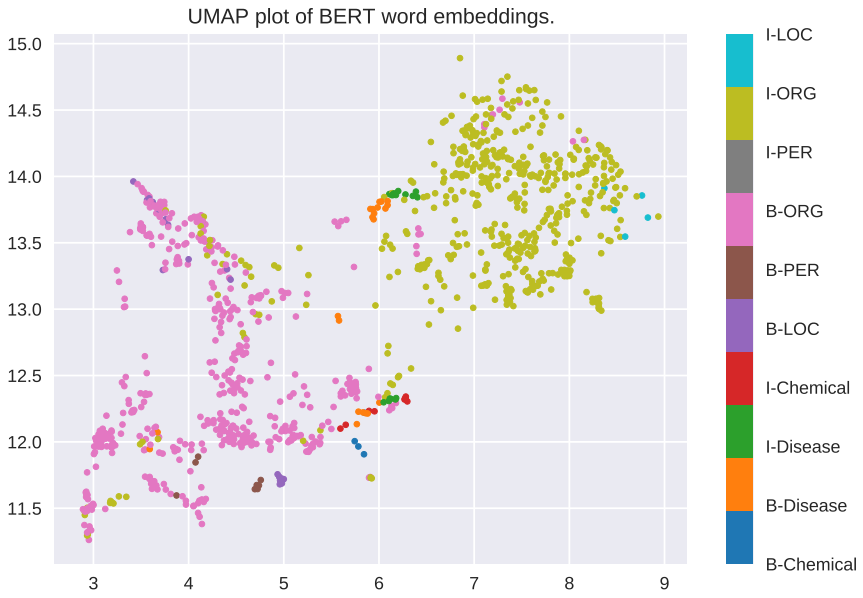

Figure 6.3: Zoom view of the B/I organisation cluster. Some chemical and disease word embeddings are also mapped to this region.

6.2 Pre-training of the BERT model

Following pre-training on the biomedical press releases, the training and validation loss functions of the next sentence prediction (NSP) and masked language modelling (MLM) were gauged. The total pre-training loss function is the next sentence prediction loss function added with the masked language modelling loss function. One interesting approach was to see if Clinical BERT or BioBERT performed best for the pre-traning tasks. The results are visualised below.

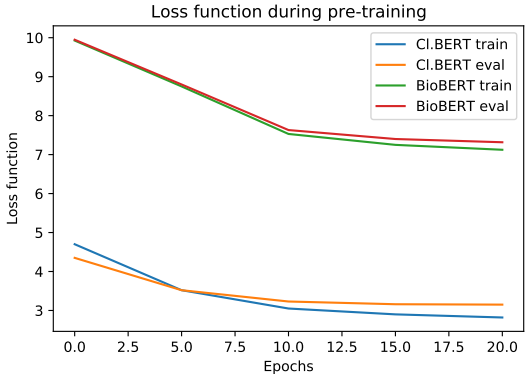

Figure 6.4: Training and validation loss for pre-training Clinical BERT and BioBERT on the biomedical news articles.

To get a more detailed picture of the pre-training, the different pre-training tasks are

visualised separately. Figure 6.5 visualises the loss functions as well as the accuracy of the next sentence prediction and masked language modelling for Clinical BERT.

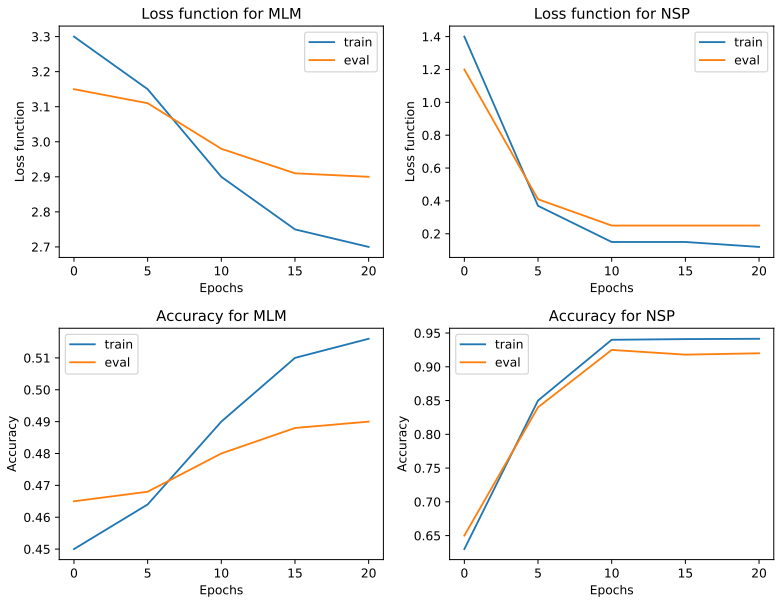

Figure 6.5: Loss function and accuracy for masked language modelling and next sentence prediction for Clinical BERT during pre-training.

Another interesting approach is to see how BERT's word embeddings are changing during pre-training. The PCA comparison of word embeddings between the pre-trained model and the original checkpoint, for Clinical BERT, is presented in Figure 6.6. As the embedded dimension from each word is reduced from 768 to 2 dimensions, some information is lost.

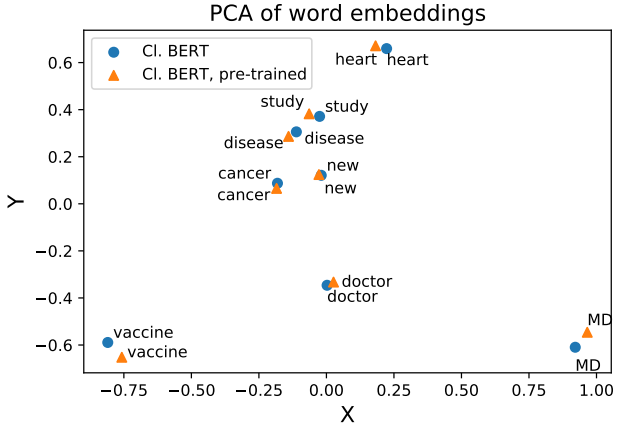

Figure 6.6: PCA plot of arbitrary word embeddings before and after pre-training. A slight shift is visible for each word.

The impact of the pre-training on attention was measured and visualized with heatmaps. See Section 2.2.1 for theory. Three BERT models were used - the BERT

Base Cased, Clinical BERT and the model pre-trained from the Clinical BERT checkpoint. The sentence "The patient had pneumonia, but the doctor prescribed him some medicine", with attention spanning to and from "prescribed" is depicted below.

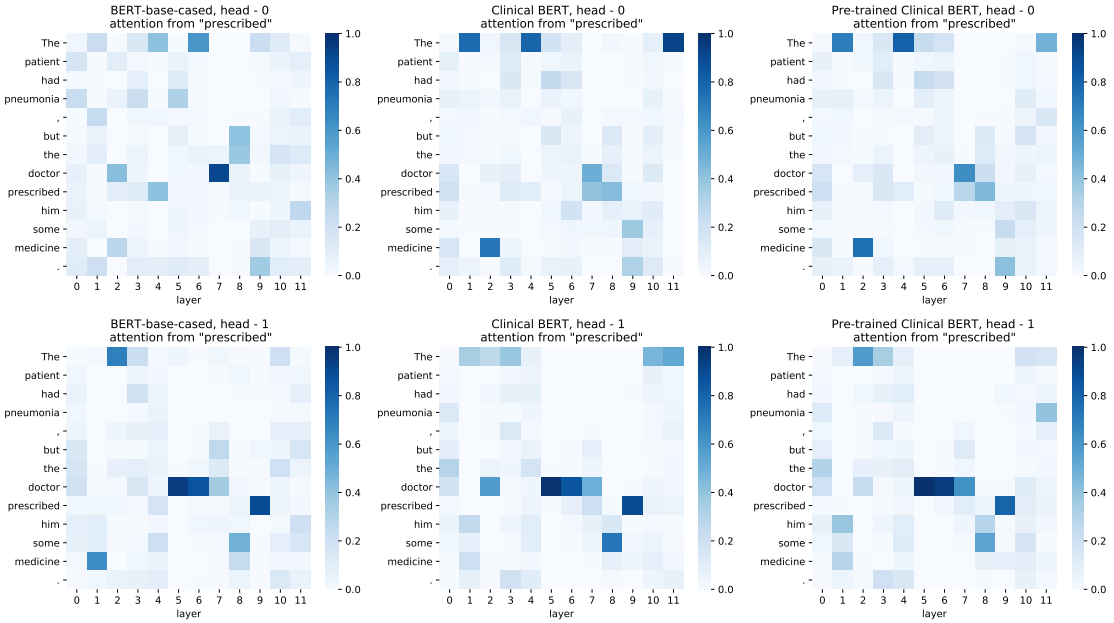

Figure 6.7: Heatmaps of the 2 first attention heads, all layers, for three separate models.

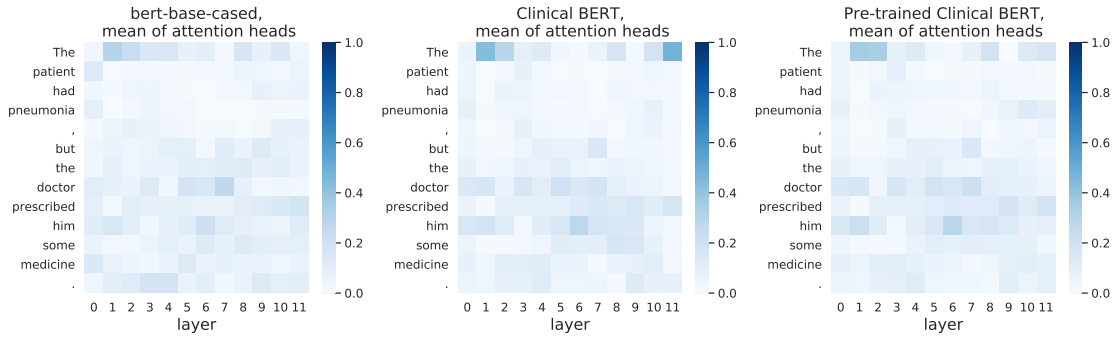

Figure 6.8: Mean of all 12 attention heads for each BERT model.

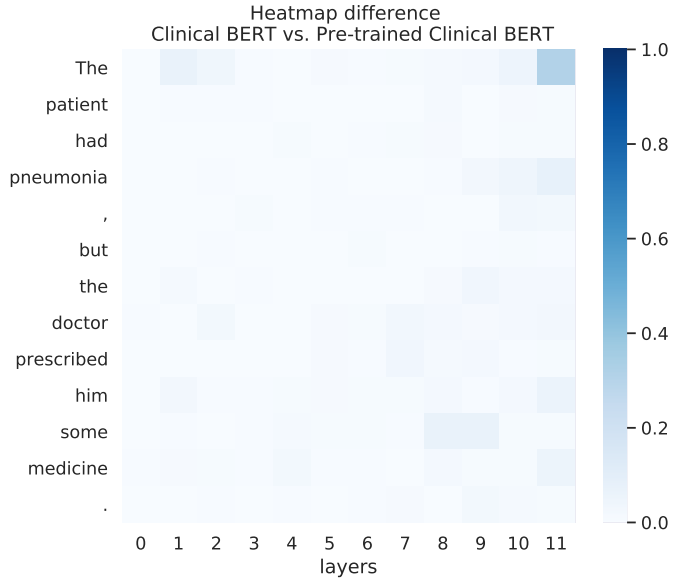

Figure 6.9: Heatmap visualization of mean attention difference between Clinical BERT and the pre-trained Clinical BERT model.

6.3 Named Entity Recognition Performance

In order to get the best performance from every BERT submodel each data set mentioned in 4.2 are investigated separately. Table 6.1 visualises the best measured results for every data set. Though, the Conll2003 data set contains entities 'Times' and 'Quantity' in addition to 'Person', 'Organisation' and 'Location'. Uninterested in entities outside the scope, all entities except 'Person', 'Organisation' and 'Location' are treated as the 'Other' entity in this data set. Hence, because of exclusion of entities, the comparison against the state of the art results for the Conll2003 data set can be a bit misleading.
During the pre-training, Figure 6.4 indicated Clinical BERT to be better than BioBERT for the pre-training tasks. Hence, the intention was to use Clinical BERT for the named entity recognition. However, during the named entity recognition training, BioBERT had a better performance than Clinical BERT for the BC5CDR dataset. As a result, BioBERT was the preferred model.

Dataset	SOTA F1-score	spaCy score	F1-score	Models used
Conll2003	0.94	0.93	0.92	BERT Base Cased + CRF
BC5CDR	0.89	0.90	0.91	BioBERT + CRF
Monocl	-	0.71	0.76	BERT Base Cased + CRF

Table 6.1: Table over the best measured F1-scores for the test set. Different BERT models were used to increase performance for the different data sets. The BC5CDR state of the art score refers to the best result of 2019. The simulations resulting in this Table are visualized in Figures 6.10-6.12.

Table 6.1 visualises the best results for each dataset, without any transfer learning.

Section 6.3.6 deals with the model system with the best overall results (F1-score) on the Monocl dataset. Section 6.3.7 is about the more recall favoured model system, which is of special interest for Monocl.

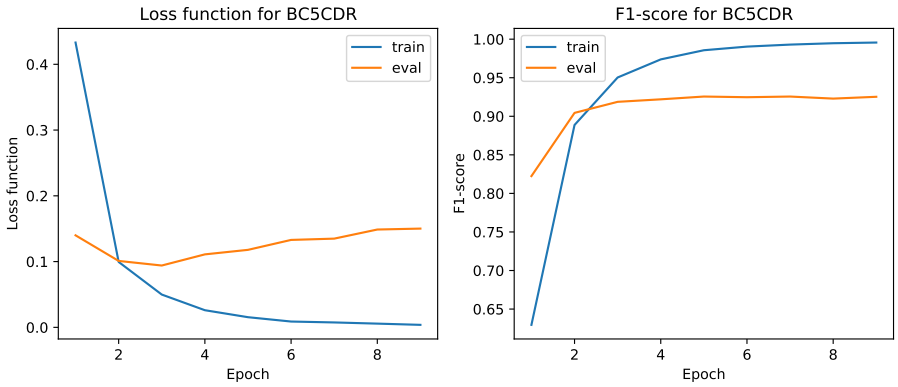

Figure 6.10: Loss function and F1-score for the BC5CDR data set.

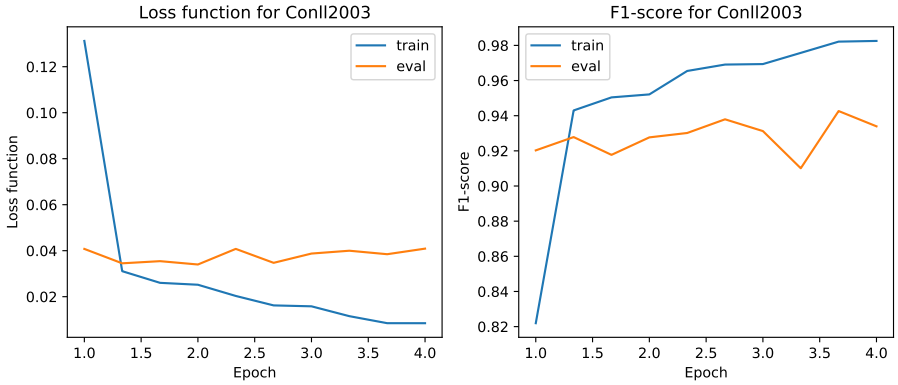

Figure 6.11: Loss function and F1-score for the Conll2003 data set.

6.3.1 Best performing model design

The highest F1 score was achieved using a design of unconnected, individual models per the description of Section 5.2 and the results are covered in Section .

6.3.2 Information about the simulations

Mentioned in Section 4.2, Conll2003 and BC5CDR are already splitted into training, validation and test set. The Monocl dataset at the other hand, are pre-processed into one file. The dataset are too small to fix a sufficiently amount of data for an evaluation set and test set that guarantees a generalized and functional evaluation and testing. All data that are included in the evaluation and test set, are data that could be a part of the training set. The intention is to use as much data as possible for training, but still has a reasonable validation and test set. Hence, the split 70% for training set, 15% for evaluation set and the remaining 15% for test set were chosen for all simulations when the Monocl dataset were used for training, evaluation

and test set. The actual splitting process of the data included a stochastic process, where 15% of the articles where randomly chosen as evaluation set and 15% for test set. Bias between the different sets were avoided by not letting examples from the same medical press release end up in different sets. 15% of the Monocl dataset for an evaluation/test set are in reality data from 36 medical press releases. This is somewhat small. As a fact the model's results are highly dependent on which articles that were chosen for the evaluation and test set. In order to counteract inaccurate results several (ten as default) iterations were made for every simulation. The results presented are the averaged results over the iterations. Throughout the simulations, the test set are evaluated on the model that performed the lowest result on the loss function for the evaluation set.

6.3.3 Monocl BERT model

A natural first investigation, would be to measure the performance of an individual BERT model trained, evaluated and tested on the Monocl data set. Information about the simulation can be found in Section 6.3.2. The results are visualised in Figure 6.12 and Table 6.2.

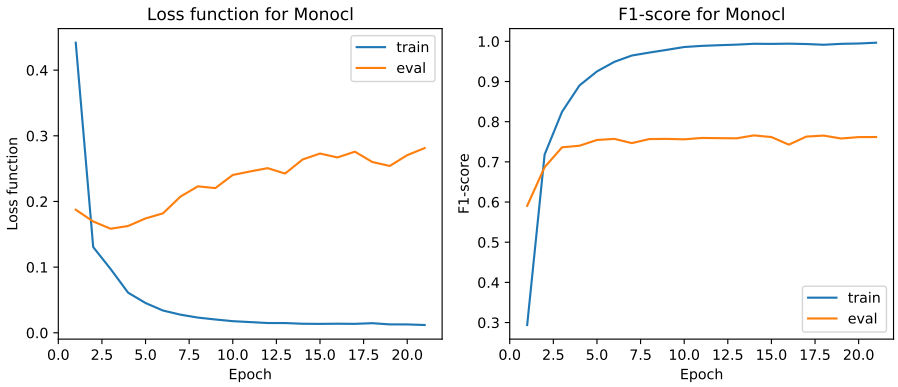

Figure 6.12: Loss function and F1-score for the Monocl data set. This plot is for one of the iterations of the Monocl BERT model.

	Precision	Recall	F1-Score
Person	0.9042	0.9195	0.9104
Organisation	0.6809	0.7175	0.6969
Location	0.7455	0.7675	0.7555
Disease	0.7553	0.7693	0.7601
Chemical	0.7397	0.7268	0.7309
Overall	0.7566	0.7659	0.7605

Table 6.2: Average performance of the 10 iterations on the test set.

6.3.4 Single model naive approach

The results of using all available data from the Conll- and BC5CDR datasets in conjunction with the Monocl dataset is depicted below in table 6.3. The measurement was done by using a single Clinical BERT model without addressing the lack of out of scope annotations for the public datasets.

	Precision	Recall	F1-Score
Person	0.6625	0.8977	0.7499
Organisation	0.6058	0.4848	0.5386
Location	0.6556	0.5735	0.6118
Disease	0.6724	0.7669	0.7165
Chemical	0.6961	0.7709	0.7316
Overall	0.6625	0.7097	0.6853

Table 6.3: Average performance of the naive approach. The results are averaged over 10 iterations on the test set.

6.3.5 Performance of Conll- and BC5CDR model on the Monocl data set

Table 6.1 and Figures 6.10-6.11 shows great results for the BC5CDR and Conll datasets. One interesting investigation would be to evaluate the performance of these models on the Monocl dataset. The evaluation were done for the FULL Monocl dataset in Tables 6.4-6.5.

	Precision	Recall	F1-Score
Person	0.8380	0.9081	0.8717
Organisation	0.5287	0.7524	0.6210
Location	0.6831	0.8767	0.7679
Overall	0.6289	0.8177	0.7109

Table 6.4: Table over the evaluation of the Conll2003 model on the Monocl data set for the entities person, location and organisation.

	Precision	Recall	F1-Score
Disease	0.5429	0.4873	0.5136
Chemical	0.7113	0.7847	0.7462
Overall	0.6194	0.6075	0.6134

Table 6.5: Table over the evaluation of the BC5CDR model on the Monocl data set for the entities disease and chemical.

Comparison between the measured F1-score of 0.919 in Table 6.1 with the F1-score of 0.7109 from Table 6.6 concludes that there is a large difference in performance between evaluating on the Conll2003 or Monocl data set. A similar conclusion can

be drawn by comparing the measured F1-score of 0.908 in Table 6.1 for the BC5CDR data set with the F1-score of 0.6134 from Table 6.7 for the Monocl data set.

6.3.6 Training the Conll- and BC5CDR model on the Monocl data set

In order to increase the performance, the primal idea was to use the Conll2003 and BC5CDR datasets for transfer learning. This is due to the small size of the Monocl dataset. Concretely, instead of training a plain BERT model on the Monocl data set from scratch, proceeding from the Conll- and BC5CDR model might improve the results. The Conll model was trained on the Monocl dataset for the entities person, organisation and location. BC5CDR likewise, but for the entities chemical and disease. Information about the simulation can be found in Section 6.3.2. The results are visualised in Tables 6.6-6.7.

	Precision	Recall	F1-Score
Person	0.9047	0.9449	0.9238
Organisation	0.7275	0.7107	0.7184
Location	0.7855	0.7937	0.7889
Overall	0.7861	0.7870	0.7863

Table 6.6: Table over the results on the test set for the Conll model trained on the Monocl data set. The results are averaged over 10 iterations.

	Precision	Recall	F1-Score
Disease	0.7723	0.7800	0.7747
Chemical	0.7755	0.7811	0.7734
Overall	0.7739	0.7805	0.7772

Table 6.7: Table over the results on the test set of the BC5CDR model trained on the Monocl data set. The results are averaged over 10 iterations.

6.3.7 Recall Favoured Monocl model

According to the preference of recall over precision, discussed in Section 3.5, a model focused on F2-score was trained. The simulation are made just as in Section 6.3.6, proceeding from the Conll- or BC5CDR model, but now with focus on F2-score for all of the training. The results are visualised in Tables 6.8-6.9.

	Precision	Recall	F1-Score	F2-score
Person	0.8694	0.9625	0.9131	0.9423
Organisation	0.5904	0.8037	0.6794	0.7495
Location	0.6634	0.8612	0.7483	0.8127
Overall	0.6735	0.8593	0.7545	0.8144

Table 6.8: Table over the results on the test set for the Conll recall model trained on the Monocl data set. The results are averaged over 10 iterations.

	Precision	Recall	F1-Score	F2-Score
Disease	0.6280	0.8723	0.7262	0.8093
Chemical	0.7474	0.8990	0.8147	0.8640
Overall	0.6730	0.8834	0.7618	0.8314

Table 6.9: Table over the results on the test set of the BC5CDR recall model trained on the Monocl data set. The results are averaged over 10 iterations.

6.3.8 Connected Ensemble Model

The best performing BC5CDR and Conll models were used as submodels and concatenating their output scores to a feed-forward layer and CRF classifier. The results of these measurements are listed below in Table 6.10.

	Precision	Recall	F1-Score
Disease	0.7233	0.7441	0.7336
Chemical	0.7685	0.7565	0.7624
Person	0.8645	0.8845	0.8744
Location	0.6139	0.6278	0.6208
Organisation	0.6770	0.6233	0.6490
Overall	0.7305	0.7252	0.7278

Table 6.10: Table of the ensemble model results on the withheld test set. The results are averaged over 10 iterations.

6.3.9 Impact of transfer learning

The results from Sections 6.3.3 and 6.3.6 imply that transfer learning actually increases the performance of the model. In order to confirm the impact of transfer learning, a more detailed investigation between models starting from scratch and models using transfer learning is necessary. Concretely, a Monocl model (without transfer learning) are compared against the separate model system (started from the Conll/BC5CDR models). This investigation is carried out just as the ones in Sections 6.3.3 and 6.3.6, but with one difference. The difference is that only a subset of the training set is used for training here, the remaining part is unused. The percentage of the training set used varies during the investigation. The results are visualised in Figure 6.13. Notice that the Figure does not contain a measurement for the Monocl model and no data. Without training on any data at all, the results are just dependent on the randomized initial parameters for the BERT model. Hence, this is not interesting to include in Figure 6.13.

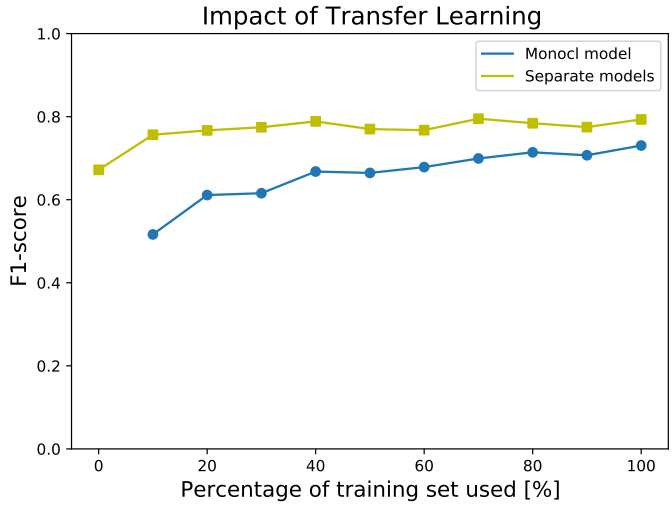

Figure 6.13: Performance of the Monocl model and the separate model system. Both models plotted towards the percentage of data in the training set that was actually used. The results of the separate models are combined into one graph.

6.3.10 Classifications

Analysing both submodels classification errors are crucial to get a deeper understanding of their limitations. Moreover, the submodels are meant to complement each other, and not have a conflict about the entity. Hence a comparison between the two submodels and conflicting classifications are of great interest.

6.3.10.1 Classification errors for the individual submodels

The classifications for the submodels from Section 6.3.7 are visualized in Tables 6.11-6.12. The classifications are categorized by the metrics described in Section 3.4. Both submodels were evaluated on the full Monocl data set. Remember that each submodel were trained for 70% of the Monocl data set, so there is a bias between the model and a part of the data. The amount of misclassifications would probably be higher for an unseen data set. Hence, this section is about investigating the actual misclassifications of the model, rather than the performance of the model. More types of misclassifications were discovered by evaluating on all data available. The results are discussed in detail in Section 7.1.3.

	Correct	Partial	Missing	Spurious	Incorrect	**Possible**	**Actual**
Person	1036	26	3	71	5	1070	1138
Organisation	1407	295	24	750	17	1743	2469
Location	535	97	3	88	16	651	736
Overall	2978	418	30	909	38	3464	4343

Table 6.11: Table over the Conll recall models classifications. The evaluation is over the full Monocl data set.

	Correct	Partial	Missing	Spurious	Incorrect	**Possible**	**Actual**
Disease	1731	224	435	510	19	2409	2484
Chemical	774	83	252	143	6	1115	1006
Overall	2505	307	687	653	25	3524	3490

Table 6.12: Table over the BC5CDR recall models classifications. The evaluation is over the full Monocl data set.

6.3.10.2 System-wide classification errors

The goal with the separate submodel approach is a classification that works independently with high performance for each model, but also without conflict between the submodels. Hence, an investigation regarding conflicting classifications are necessary. The investigation included both internal classification errors and external classification errors (between the submodels). Figure 6.14 visualises all classification errors for the Monocl dataset. A partial matching is enough to be registered as a classification in the investigation, since most of the misclassifications are partial matches. The results are discussed under Section 7.1.3.

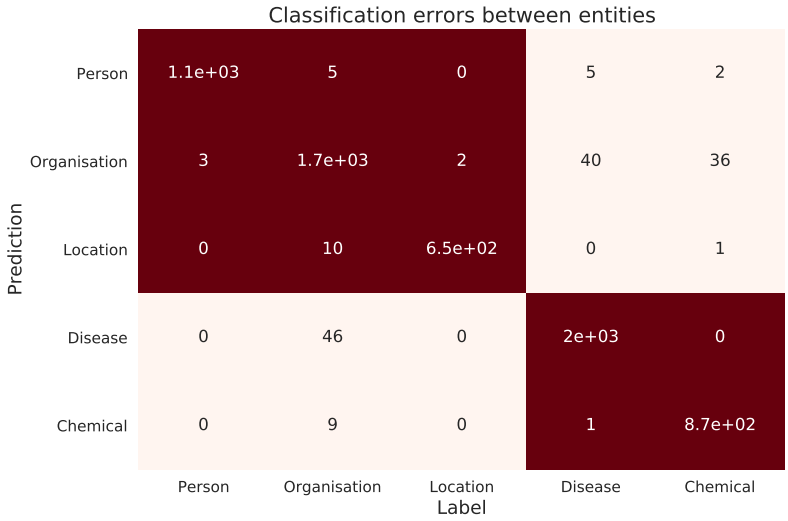

Figure 6.14: Classification errors for the individual submodels and between the submodels. The predicted entity is on the y-axis and the correct entity on the x-axis. NOTE: The coloring in the figure is NOT proporitonal to the number in every box. Instead, internal submodel classification errors are colored in red and external (between the submodels) are colored in white.

7
Conclusion

7.1 Discussion

7.1.1 Transfer Learning with Conll and BC5CDR

The best results were achieved by leveraging the separate, specialized BERT models and training the models with the Conll and BC5CDR datasets prior to the final fine-tuning.
In terms of Monocl data volume, the difference in performance between using the public datasets for transfer learning and omitting them is higher for lower amounts of Monocl data used for training. While these results are less surprising, what is more interesting is that the performance is also boosted when the entire Monocl dataset is used, as seen in Figure 6.13. Despite sentence level differences between datasets in the text data, visualized in Figure 6.1, using the public datasets increases the performance for all measured amounts of high quality data.
Whether or not this approach is beneficial given an even greater amount of high quality data is however hard to infer from the results.

7.1.2 Pre-training

The pre-training step of the BERT model does not yield a clear performance boost for the named entity recognition and any model. Still, the results in Section 6.2 are proving that the model parameters are changing during pre-training. The lack of performance boost is most likely due to the size of our available amount of unstructured text data reaching 10 MB where the data size of the text used to train the original BERT model amounted to 16 GB - a factor $1.6 \cdot 10^3$ higher. Because of this lack of data, the project resources was redirected early on to the named entity recognition.

7.1.3 Misclassifications

Several interesting discoveries were made from the investigation leading up to the results in Section 6.3.10.1. First off, the partial matches varies in quality. The best partial match differs with one or a few tokens. Classifying "The U.S. Department of Veterans Affairs" instead of "U.S. Department of Veterans Affairs" or "lung cancer" instead of "non-small cell lung cancer" are of such type. This classifications are informative and not far from correct. Many of these classification can be matched to an organisation, disease etc in Monocl's platform. However, not all partial matches

live up to this quality. Occasionally crucial information were missed and the partial matches became too weak. Barely identifying disease tokens "##hil" and "##ia" in "severe hemophilia" are not informative. From this partial match it is impossible to identify a disease, and hence this classification is useless. Secondly, incorrect annotated tokens appeared investigating the spurious misclassifications. The tokens were classified correct by the submodels, but their corresponding label was incorrect. The problem were tracked down to the annotation of the medical press releases. Annotating more than two hundreds of medical press releases manually takes days. Making mistakes along the way are natural. Some specific entities were missed in the text and therefore marked with the 'Other' label in the dataset. Still, this is something that has to be analysed, but accepted in the project. Besides, the upside is that the model actually makes a correct prediction in these cases, even if they are declared as spurious classifications in the statistics.

Regarding the system-wide classification investigation a couple of interesting repeated classification errors were observed. One common disagreement in classification between the submodels are when an entity string hides inside a longer entity string. If the two entities are of types handled by different submodels, this leads to a disagreement between the submodels. "National Comprehensive Cancer Network" is an organisation, but "cancer" is a disease itself. Several medical organisations includes a name of a disease or a chemical in their company name, which can explain the confusion between organisation and disease/chemical in Figure 6.14. Furthermore, it appeared that the submodels also had disagreements regarding the aforementioned entities when the string boundary was correctly identified by both submodels. Organisation names can easily be mistaken for chemical/disease names and vice-versa. Specially abbreviations are hard to classify. "COVAX" and "Breyanzi" are potential names of both organisations and chemicals, and "HCC" can be an abbreviation of anything. The context of the word has to determine the correct label. Nevertheless, the correct model in the conflict varies. For clarification: "COVAX" is an organisation, "Breyanzi" is a chemical and "HCC" stands for Hepatocellular carcinoma and is a type of liver cancer.

For the downstream task, the two conflicts between submodels just mentioned are not critical. The "entity string inside a longer entity string" case does not have to be a disadvantage. Quite the opposite! If the system identifies "National Comprehensive Cancer Network" as well as "Cancer", it picks up two entities. Organisations with names including a disease/chemical are presumably working with this disease/chemical. If the "National Comprehensive Cancer Network" is mentioned in a medical press release, it is hard to not relate to the disease cancer itself. Hence, identifying both entities are not considered a problem in this case. In the case regarding models disagreeing about the same text string, the easiest solution for the downstream task is to identify the text string as both entities to Monocl. This is done despite the knowledge that one of the classifications are wrong. There is a logical explanation for this. Monocl will only integrate a text string from the model to their platform if they find an organisation, disease etc with that name. If no match is found in the platform the information is discarded. Hence, the intention is that the incorrect classification is not going to be matched into Monocl's platform and therefore discarded.

7.1.4 Connected Ensemble Model

The primary idea with the Ensemble model was that the extra layer should handle classification errors between the different submodels, without losing anything in performance. During the project the question arised if the classification errors between the submodels was of such importance, that an Ensemble model approach would outrival a separate submodel approach. Hence, the investigation of the conflicting classifications in Section 6.3.10.2 and the related discussion in Section 7.1.3 was of great importance. Also, the clear performance drop between the separate submodels in Sections 6.3.6 and the Ensemble model in Section 6.3.8 was taken into account. In total, it was concluded that the extent and severity of the classification errors between the separate models, would not justify the performance drop that comes with the Ensemble model. Hence, the separate model approach was chosen over the Ensemble model.

7.2 Conclusion

The best results were generated by using a BioBERT and BERT base model separately and relying on transfer learning with public datasets prior to introducing the target data. The separate models reached F1-scores of 0.79 and 0.78 as well as F2-scores of 0.81 and 0.83 for the recall centered approach. The final reduction from the sporadic, conflicting output was simply handled by selecting the entity with the highest frequency in order to comply with the CRF model. This design outperforms not only the proposed ensemble model and the single BERT encoder design, but also the independent spaCy baseline which was on par with the achieved results for the public datasets. The impact of the transfer learning diminishes as the amount of high quality data increases, but the final result proves to benefit from the transfer learning aswell.

Further unsupervised pre-training of the BERT models yielded little to no improvement, likely due to insufficient volumes of unstructured text data. The difference in attention heatmaps posterior to pre-training is almost indiscernible and no boost in performance was noticeable in the measurements. As mentioned in the discussion, the sheer scale of data used pales in comparison to the amount of text data used to derive the BioBERT, Clinical and Base BERT models.

There is a noticable difference in performance between entities, with the person entity being the easiest and the organisation entity being the hardest to classify. Inter-model misclassifications, when occuring, typically consisted of the organisation entity being wrongfully predicted as a disease/chemical or vice versa.

Bibliography

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[2] P. Fontelo and F. Liu, "A review of recent publication trends from top publishing countries," *Systematic reviews*, vol. 7, no. 1, pp. 1–9, 2018.

[3] "When can we rely on legitimate interests?," Information Commissioner's Office. Available at: `https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/legitimate-interests/when-can-we-rely-on-legitimate-interests/`, copied 2021-03-17.

[4] "Yttrandefrihetsgrundlag (1991:1469)," Sveriges Riksdag. Available at: `https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/yttrandefrihetsgrundlag-19911469_sfs-1991-1469`, copied 2021-03-17.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[6] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *arXiv preprint arXiv:1804.07461*, 2018.

[7] Y. Peng, S. Yan, and Z. Lu, "Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets," *arXiv preprint arXiv:1906.05474*, 2019.

[8] L. Svensson, *Transformers - Part 3 - Encoder*. YouTube, Oct 2020. Available at: `https://www.youtube.com/watch?v=SsLzwRXH0UI&t=12s`, copied 2021-03-17.

[9] C. McCormick, *BERT Research - Ep. 7 - Inner Workings IV - FFN and Positional Encoding*. YouTube, Feb 2020. Available at: `https://www.youtube.com/watch?v=YIEe7d7YqaU&t=656s&ab_channel=ChrisMcCormickAI`, copied 2021-03-17.

[10] L. Svensson, *Transformers - Part 2 - Self attention complete equations*. YouTube, Oct 2020. Available at: `https://www.youtube.com/watch?v=ER_KqqtoikA&t=24s`, copied 2021-03-17.

[11] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[12] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[13] E. Becht, L. McInnes, J. Healy, C.-A. Dutertre, I. W. Kwok, L. G. Ng, F. Ginhoux, and E. W. Newell, "Dimensionality reduction for visualizing single-cell data using umap," *Nature biotechnology*, vol. 37, no. 1, pp. 38–44, 2019.

[14] Q. Wang, Y. Zhou, T. Ruan, D. Gao, Y. Xia, and P. He, "Incorporating dictionaries into deep neural networks for the chinese clinical named entity recognition," *Journal of biomedical informatics*, vol. 92, p. 103133, 2019.

[15] R. I. Dogan and Z. Lu, "An improved corpus of disease mentions in pubmed citations," in *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pp. 91–99, 2012.

[16] R. Leaman, C.-H. Wei, and Z. Lu, "tmchem: a high performance approach for chemical named entity recognition and normalization," *Journal of cheminformatics*, vol. 7, no. 1, pp. 1–10, 2015.

[17] J. Lee, 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*, vol. 36, no. 4, pp. 1234–1240, 2020.

[18] E. Alsentzer, J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. McDermott, "Publicly available clinical bert embeddings," *arXiv preprint arXiv:1904.03323*, 2019.

[19] E. Alsentzer. Huggingface repocitory. Available at: `https://huggingface.co/emilyalsentzer/Bio_ClinicalBERT`, copied 2021-03-17.

[20] Huggingface repocitory for the Transformers library. Available at: `https://huggingface.co/transformers/`, copied 2021-03-17.

[21] Brat standard format. A format for marking entities in a text. Available at: `https://brat.nlplab.org/standoff.html`, copied 2021-03-17.

[22] MTL-BioInformatics. Available at: `https://github.com/cambridgeltl/MTL-Bioinformatics-2016/tree/master/data`, copied 2021-03-19.

[23] Conll2003 dataset. Available at: `https://github.com/davidsbatista/NER-datasets/tree/master/CONLL2003`, copied 2021-03-19.

[24] spaCy github. Available at: `https://github.com/explosion/spaCy`, copied 2021-03-19.

A

Appendix 1