BIOMEDICAL NAMED-ENTITY RECOGNITION

by
Jian Shu (James) Wu
Supervisor: Professor Gary Bader
April 2019

BIOMEDICAL NAMED-ENTITY RECOGNITION

Jian Shu (James) Wu

Biomedical Named-Entity Recognition (BioNER) is an important area of research in the field of Biomedical Information Extraction. BioNER is a unique challenge in natural language processing (NLP) due to the limited amount of labelled BioNER corpora available for both training and evaluating BioNER models. This report presents a novel approach to BioNER which uses transfer learning techniques in NLP. A new architecture, BERT, which has shown state-of-the-art results in multiple NLP tasks is applied to BioNER. We call this model BERT-Bio-NER. BERT-Bio-NER shows promising results in BioNER and has the potential to avoid problems of over fitting associated with current state-of-the-art BioNER models. This report presents BERT-Bio-NER as not only a viable candidate for replacing current state-of-the-art BioNER models, but also as a promising future research direction for developing models for biomedical information extraction in general.

ACKNOWLEDGEMENTS

This work would not have been possible without the help and guidance of my supervisor Professor Gary Bader and my mentor John Giorgi.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

PubMed contains more than 28 million citations for biomedical literature [1]. Such an overwhelming quantity of published literature makes it difficult for scientists to stay up to date in their research field. Because of this, Information Extraction (IE) from biomedical literature has been an active area of research in the domain of natural language processing (NLP). The task of biomedical IE can be divided into a number of sub-tasks, each dependent on the previous sub-task. These sub-tasks are Named Entity Recognition (NER), Named Entity Linking (NEL), and Relation Extraction (RE). NER involves categorizing words or phrases of interest in a given text. In the biomedical context, this entails labelling ***entities*** (words or phrases) with ***tags*** such as genes, proteins, diseases, and species. NEL involves the linking of entities tagged by NER to a unique identifier in external biomedical databases. Finally, RE involves the detection of relations between entities. For example, recognizing that the expression of gene A has an inhibitory effect on the expression of protein B is an RE task. Because the sub-task of Biomedical Named-Entity Recognition (BioNER) continues to be an active field of research and the dependence of NEL and RE on BioNER, my undergraduate thesis research focuses on the problem of Named Entity Recognition for biomedical literature.

1.2 Problem Statement

BioNER has been approached by a number of ”traditional” machine learning and probabilistic models as well as more ”modern” deep learning techniques, with varied success. This is due to a number of difficulties unique to BioNER.

Genes, proteins, diseases, and species in biomedical literature can often be long compound-word phrases. As such, acronyms are commonly used to abbreviate these long phrases (i.e. TCF). However, there are many inconsistencies in the use of acronyms across biomedical literature [2]. For example, TCF can refer to ”T-cell Factor” or ”Tissue Culture Fluid” depending on the context of the biomedical paper [2]. As such, for any BioNER model to be successful, it must have some degree of context driven tagging. This limits or complicates potential solutions such as rule-based tagging.

Because BioNER is a relatively niche research area, the limited labelled corpora available for model training and evaluation (in the case of deep learning) is another important challenging factor to this problem. Modern supervised deep learning approaches require massive amounts of labelled data. As such, despite the promising performance of recent neural network solutions in this field, they have been shown to exhibit over-fitting and limited model generalization, suggesting that the current models are not yet ready for de-

ployment in practical settings [3]. This is likely due to the limited availability of labelled biomedical corpora during model training.

With these difficulties in mind, my research will focus on exploring and evaluating the success of a somewhat new approach to BioNER, transfer learning. This technique involves the fine-tuning of an existing deep learning language model for the specific purposes of BioNER. Although researchers in the past have found it difficult to successfully implement transfer learning in the field of NLP, transfer learning is a well-known and established technique in deep learning applications for computer vision. My undergraduate thesis will explore the success of different methods for fine-tuning a language model, BERT, for the purposes of BioNER. This approach is especially promising for BioNER because it seems to have the potential to address some if not all of the difficulties outlined above. By utilizing an existing language model that has already ”learned” the basics of language, we will only need to slightly modify the network to ”learn” the specific conventions of ”biomedical language”.

CHAPTER 2

EXISTING BIONER MODELS

2.1 Dictionary Models

Early solutions in BioNER involved generating massive dictionary look-up tables [4]. Methods from [4], depict the extraction of symbols, alias names, and full names, of genes and proteins, from prominent gene and protein databases of the time (HUGO, TREMBL, and Swiss-Prot). This data was compiled into a dictionary object that was then curated to resolve ambiguities, redundancies, and irrelevant synonyms [4]. Thus, when presented with a new biomedical abstract, each word was tokenized following a search through the constructed dictionary [4]. Although they have a relatively simple architecture, dictionary models are limited in their ability to discern acronym ambiguity.

2.2 Machine Learning Models

The machine learning models depicted in this section all involve some form of supervised learning, where a cost or prediction computed from a labelled corpus is minimized. However, unlike deep learning models, these machine learning models use hand-selected features to extract relevant information from tokens, before training or performing tag prediction. This dependence on manual feature selection can be effective when there is a limited amount of labelled data, as is the case for BioNER. As such, some of these models have been relatively successful in the field of BioNER in the past.

2.2.1 Support Vector Machines (SVMs)

Support Vector Machines (SVMs) have been explored as a possible approach to BioNER [5]. Tokens are first mapped to vectors in a hyper-parameter space using hand-picked features, such as the index position of word in a predefined vocabulary, prefixes/suffixes, and the presence of sub-strings or root words. SVMs then attempt to determine hyper-planes that separate these vector points into regions, according to their token labels. The optimal hyper-plane is derived by maximizing the margin of the hyper-plane. The margin is defined as the distance between the nearest data points and the hyper-plane boundary. Maximizing this distance allows the hyper-plane to find the optimal ”middle” boundary separation between token classification. The model then predicts the tag of a new token from its vector mapping in the hyper-parameter space.

2.2.2 Maximum Entropy Models

Taking a more probabilistic approach to the problem, maximum entropy models attempt to predict the tag classification likeliness for a token, given a set of features of that token. The conditional probabilities are generated from a labelled training corpus, while the prediction or evaluation of the model involves the following equation, as depicted in [6]:

$$p(o|h) = \frac{1}{Z(h)} \prod_i \alpha_i^{f_i(h,o)}$$

where $p(o|h)$ is the probability of tag o given the prior information derived from labelled training corpora, h. This is computed by taking each predefined feature f_i and learning its weight α_i given h and o. $Z(h)$ is the normalization term and takes the form:

$$Z(h) = \sum_0 \prod_i \alpha_i^{f_i(h,o)}$$

The prediction of a tag is made by taking the tag with the highest probability. For [6], features were hand-picked and included a context window, providing the two previous and the two next tokens, capitalization, digit information, and the presence of special characters (i.e. dashes and commas).

2.2.3 Hidden Markov Models (HMMs)

When given a sequence of n tokens, $\mathbf{w} = \{w_1, w_2, ...w_n\}$, Hidden Markov Models (HMM) predict a sequence of n tags, $\mathbf{t} = \{t_1, t_2, ..., t_n\}$. Denoting $\mathbf{T}$ as the set of all possible sequences of $\mathbf{t}$, an example second order HMM formula from [7], predict a sequence $\hat{\mathbf{t}}$ such that,

$$\hat{\mathbf{t}} = \underset{\mathbf{t} \in \mathbf{T}}{\text{argmax}} [logP(t_1) + logP(t_2|t_1) + \sum_{i=3}^{n} logP(t_i|t_{i-1}, t_{i-2}) + \sum_{i=1}^{n} logP(w_i|t_i)]$$

HMMs are similar to Maximum Entropy Models in the sense that they also take a probabilistic approach to the problem. However, HMMs have a stronger emphasis on incorporating context into its probabilistic prediction. Instead of predicting the tag of each token individually, HMMs maximize the probability of a tag sequence. Like the Maximum Entropy Models, the conditional probabilities in the equation above would be derived from a labelled training corpus. Although for HMMs, this process is probably computationally heavier due to the incorporated conditional dependence of past tag predictions and tokens. Moreover, HMMs also require feature extraction for probability calculations. For most BioNER HMMs, these features are generally hand-picked [7], [8], [9].

2.2.4 Conditional Random Fields (CRFs)

Conditional Random Fields (CRFs) are a more generalized form of Hidden Markov Models. To see this, suppose the model is again considering a sequence of n tokens, $\mathbf{w} = \{w_1, w_2, ...w_n\}$ and predicting a sequence of n tags, $\mathbf{t} = \{t_1, t_2, ..., t_n\}$. Let $s_j(\mathbf{t}) = \{t_{j-k+1}, ...t_j\}$, where k is a chosen number of previous tags to consider. For CRFs, m number of feature functions, $f(s_j(\mathbf{t}), w_j)$, are defined. Each of these feature functions mathematically captures some association between the j^{th} token and the previous k tags up until and includ-

ing the j^{th} tag. Moreover, let λ_i be the learned weight of the i^{th} feature function. Again, denoting $\mathbf{T}$ as the set of all possible sequences of $\mathbf{t}$, as explained by [10], CRFs predict a sequence $\hat{\mathbf{t}}$ such that,

$$\hat{\mathbf{t}} = \underset{\mathbf{t}\in\mathbf{T}}{\operatorname{argmax}}\, exp\left(\sum_{j=1}^{n}\sum_{i=1}^{m}\lambda_i f_i(s_j(\mathbf{t}), w_j)\right)$$

By defining suitable features and weights, it is possible to show that HMMs are simply a subset of CRFs.

Similar to both Maximum Entropy Models and Hidden Markov Models, CRFs also generally involve the hand-selection of a set of features. However, CRFs seem to be much more generalized and include a training process for weighting these features to maximize performance. CRFs have been depicted as models that reduce the problem of NER to selecting an appropriate set of features [11]. Although it should be noted that manual feature selection is not exactly a trivial task. For many CRFs, features may include orthographic features (spelling, hyphenation, capitalization, etc.), semantic features (prefixes, suffixes, etc.), and induction features (dependence on previous tags or tokens) [11], [10], [12].

2.3 Recurrent Neural Network Models (RNNs)

A class of artificial neural networks, Recurrent Neural Networks (RNNs) are heavily used in NLP [13]. At its simplest, RNNs have an input, output, and hidden layer. The hidden layer "recurrently" feeds back into itself allowing the network to store information in its hidden layers, to be used at a later time. This lends itself very well to processing data such as language, where there is a strong temporal dependence between words.

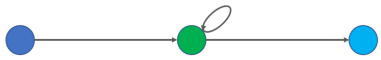

Figure 2.1: RNN Unit

[13]

2.3.1 Word Embeddings

Similar to the machine learning models discussed earlier, RNNs models also require their word inputs to be in some vector-form representation. These word representations are then used as input data for training the actual models. For machine learning, these representations are typically hand-picked features. However, for deep learning models, these vector representations are typically learned models that were trained on massive data sets. For example, Word2vec, a prominent word embedding model, was trained using a 1.6 billion word data set [14]. Having access to large amounts of labelled data,

sufficient processing speeds, and an appropriately designed model architecture, word embeddings seem to be much more effective at translating words into a vector space representation, compared to traditional hand-picked features [14], [15].

2.3.2 Long Short-term Memory (LSTM)

A specific type of RNNs, Long Short-term Memory (LSTM) Units, demonstrate much better performance than basic RNN models due to their complex model structure. This structure includes gated control of the input, internal state, and output. Consider a sequence of n input vectors $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}$, each being the word embedding of a token from an input sentence. Let $\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_n}$ be the output sequence of the LSTM. The following formulas were adapted from [16]. $\mathbf{W}$ are weight matrices and $\mathbf{b}$ are bias vectors, learned during the training phase of the model.

At step t, $\mathbf{i_t}$ is computed from a linear combination of $\mathbf{x_t}$ (the new input vector), $\mathbf{h_{t-1}}$ (the output vector of the previous iteration), and $\mathbf{c_{t-1}}$ (the memory vector from the previous iteration):

$$\mathbf{i_t} = \sigma(\mathbf{W_{xi}x_t} + \mathbf{W_{hi}h_{t-1}} + \mathbf{W_{ci}c_{t-1}} + \mathbf{b_i})$$

This acts as a "filter", where only relevant information is allowed to pass into the network. With $\mathbf{i_t}$, the model computes $\mathbf{c_t}$ (the new memory vector) and $\mathbf{h_t}$ (the new output vector):

$$\mathbf{c_t} = (\mathbf{1} - \mathbf{i_t}) \odot \mathbf{c_{t-1}} + \mathbf{i_t} \odot \mathbf{tanh}(\mathbf{W_{xc}x_t} + \mathbf{W_{hc}h_{t-1}} + \mathbf{b_c})$$

$$\mathbf{o_t} = \sigma(\mathbf{W_{xo}x_t} + \mathbf{W_{ho}h_{t-1}} + \mathbf{W_{co}c_{t-1}} + \mathbf{b_o})$$

$$\mathbf{h_t} = \mathbf{o_t} \odot \mathbf{tanh}(\mathbf{c_t})$$

where $\odot$ is element-wise multiplication.

Here, the internal memory of the network, $\mathbf{c_t}$ is computed by combining the new information ($\mathbf{x_t}$ and $\mathbf{i_t}$) and states from the previous iteration ($\mathbf{c_{t-1}}$ and $\mathbf{h_{t-1}}$). The output of the model at iteration t, $\mathbf{h_t}$, involves the current memory state $\mathbf{c_t}$, as well as the previous memory state $\mathbf{c_{t-1}}$, computed as part of $\mathbf{o_t}$.

This depicted method of "gating/filtering" allows information to be retained for much longer periods of time inside the network [17] compared to most other RNN structures, that may experience issues such as the "vanishing gradient problem" [18]. The vanishing gradient problem is when input data has a limited effect on the output because too many iterations of the RNN have passed by [18]. This can be problematic when earlier data

may be required to provide context in understanding new data [18]. As such, words from the beginning of a sentence providing context to the end of the sentence may no longer be preserved in a typical RNN model. LSTMs are powerful models that can mitigate these issues. They are used quite prevalently for NLP tasks [13].

2.3.3 Bi-LSTM-CRF

Currently one of the dominant model architectures in BioNER, Bi-LSTM-CRF models combine word embeddings, LSTM models, and CRFs to form a complex network [3]. The model uses two LSTM models. Given an input sentence, $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}$, the first LSTM considers the sentence in order $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}$, producing an output $\overrightarrow{h_t}$ at each time step, while the second LSTM considers the sentence in reverse order, $\mathbf{x_n}, \mathbf{x_{n-1}}, ..., \mathbf{x_1}$, producing an output $\overleftarrow{h_t}$ [16]. This is to capture contextual information, which may have forward or backward dependence. Concatenating the output of the LSTMs at each time step, we get $\mathbf{h_t} = |\overrightarrow{h_t}; \overleftarrow{h_t}|$. Interestingly, this Bi-LSTM model is used as the feature function for a CRF model. By combining a probabilistic CRF model with a complex Bi-LSTM feature extraction function, Bi-LSTM-CRF models have achieved the current state of the art results for BioNER.

Researchers have explored variations of this model to improve performance, including the use of word embeddings and/or character embeddings before feeding into the Bi-LSTM network [16], [2], [19], as depicted in Figure 2.2. Moreover, adding a fully connected layer after the Bi-LSTM layer and before the CRF layer has been shown to improve performance [2]. Different training methods have also been explored such as multi-task learning [19] and techniques to overcome the limited data sets available for BioNER [2], showing that the training process can also have an effect on model performance.

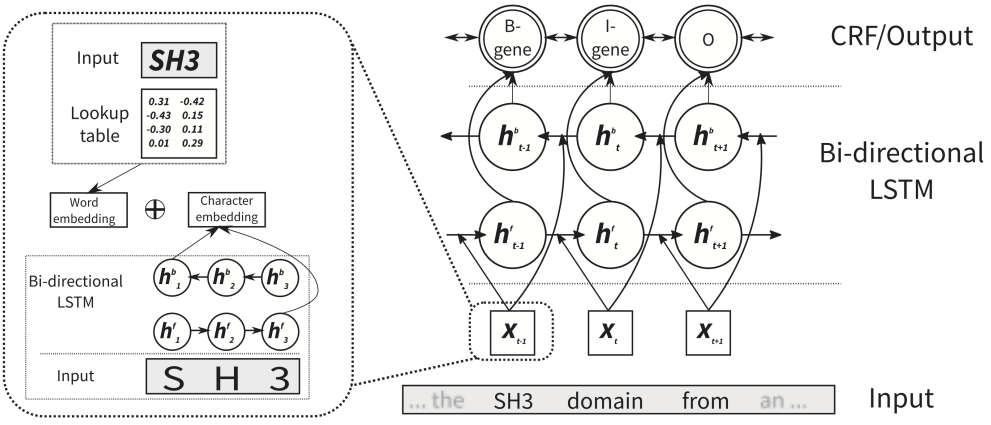

Figure 2.2: Bi-LSTM-CRF with Character and Word Embeddings

[16]

2.4 Current State-Of-The-Art BioNER Performance

Table 2.1 presents results comparing the F_1 scores of current state-of-the-art BioNER models for a number of corpora. The *Literature Best* column will compared against the preliminary results of the new model proposed in this report.

Entity	Corpus	Crichton *et al.* (2017)	Habibi *et al.* (2017)	Wang *et al.* (2018)	Giorgi and Bader (2018)	Giorgi and Bader (2019)	Literature Best
Chemicals	BC4CHEM	82.95%	86.62%	**89.37%**		88.46%	**89.37%**
	BC5CDR	89.22%	91.05%		91.64%	**92.82%**	**92.82%**
	CRAFT	80.00%				**84.98%**	**84.98%**
Diseases	BC5CDR	80.46%	83.49%		82.32%	**84.49%**	**84.49%**
	NCBI-Disease	80.46%	84.64%	86.14%	84.72%	**87.01%**	**87.01%**
	Variome		**86.05%**		85.45%	85.75%	**86.05%**
Species	CRAFT	**97.74%**				96.28%	**97.74%**
	Linnaeus	83.98%	93.40%		**93.54%**	89.44%	**93.54%**
	S800		72.10%		**74.98%**	72.75%	**74.98%**
Genes/proteins	BC2GM	73.04%	78.57%	80.74%	78.66%	**81.48%**	**81.48%**
	CRAFT	75.16%				**84.46%**	**84.46%**
	JNLPBA	69.73%	77.25%			**80.92%**	**80.92%**

Table 2.1: State-of-the-art BioNER F_1 Performance

[3]

CHAPTER 3

LANGUAGE MODELS, TRANSFER LEARNING, AND BERT

This section provides background to the BioNER approach that is explored for this undergraduate thesis.

3.1 Language Models

Language models are an important task in the field of NLP [13]. Language models take a sequence of words from a sentence as its input and attempts to predict the next word of the sequence [13]. Early language modelling involved probabilistic approaches, while more recently, neural networks have shown promise in this field [13].

3.2 Transfer Learning

Despite being a well-established technique in the field of computer vision, implementing transfer learning in the context of NLP has not been nearly as wide-spread [20]. Instead of training a new neural network model from scratch (i.e. random initialization of model weights), transfer learning aims to transfer a portion of learned weights from an already trained model over to this new untrained model [21]. This can significantly reduce the training time required for this new model [21]. Moreover, this technique can reduce the amount of labelled data required to train the new model [21].

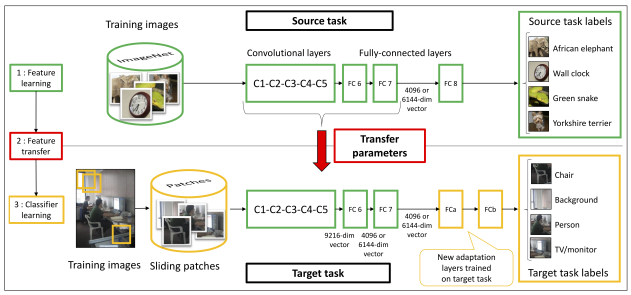

Figure 3.1: Transfer Learning for CNNs

[21]

An example of successful transfer learning in computer vision is the reuse of convolution layers of a convolutional neural network (CNN) for image recognition [21], as depicted in Figure 3.1. The first few convolution layers of a CNN are typically associated with general feature detection (i.e. recognizing lines, shapes, etc.) [21]. As such, when training a new image recognition model, it would likely also need the same capacity for general feature detection. This explains why "transferring" certain pre-trained layers will work and can significantly reduce the training time of a new network.

There are also some examples of transfer learning in NLP. One such example that has already discussed is the use of word embeddings [20]. These are essentially pre-trained

models that generate vector space representations of words that are added to the first layers of many NLP models [20], such as Bi-LSTM-CRF. Despite word embeddings being used in most state-of-the-art NLP models, the main tasks of these NLP models must still be trained from scratch [20].

3.3 BERT

The BERT model, or Bidirectional Encoder Representations from Transformers, ties together many of the concepts discussed earlier. BERT is a pre-trained language model with the versatility to be re-purposed for a wide range of language tasks [22]. A few relevant neural network structures will be explored before presenting BERT's architecture.

3.3.1 RNN Encoder-Decoder

RNN encoder-decoder architectures were initially developed for machine translation, translating a sentence from one language to another [23]. These models typically involved two RNNs. The first RNN would encode the source sentence $\mathbf{x} = \{x_1, x_2, ..., x_n\}$, where x_i is a vector representation of each input token. At each step, t, the *encoder* RNN output would be h_t^e. The following equations were adapted from [23] to maintain consistent notation in this report. Figure 3.2 is a visual representation to help guide the equations.

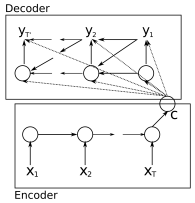

Figure 3.2: RNN Encoder-Decoder

[24]

The encoder network computes:

$$h_t^e = f^e(x_t, h_{t-1}^e)$$

Letting $c = h_n^e$, c can be expressed as some function q of $\{h_1^e, ..., h_n^e\}$:

$$c = q(\{h_1^e, ..., h_n^e\})$$

The second RNN is the *decoder* RNN. It proceeds to "unravel" c, which is a single vector composed from the entire input sequence $\{x_1, x_2, ..., x_n\}$. The hidden state of the decoder at step t is:

$$h_t^d = f^d(h_{t-1}^d, y_{t-1}, c)$$

where

$$P(y_t|y_{t-1}, y_{t-2}, ..., y_1, C) = g(h_t^d, y_{t-1}, c)$$

y_t is computed by taking the output with the maximum probability.

As outlined in [24], the RNN Encoder-Decoder is trained by maximizing the conditional log-likelihood:

$$\underset{\theta}{\text{argmax}} \frac{1}{N} \sum_{i=1}^{N} log p_\theta(y_i|x_i)$$

where θ are the model parameters of the two RNNs.

3.3.2 Attention

Although encoder-decoder models are able to collapse variable length sequences $\{x_1, x_2, ..., x_n\}$ into a single vector c, this can be problematic when trying to "unravelling" a sequence of information $\{y_1, y_2, ..., y_n\}$ from a single vector [23]. To mitigate this, a bi-directional RNN is used in the encoder, similar to the bi-directional LSTM used in the Bi-LSTM-CRF model, in addition to the implementation of an *attention* mechanism in the decoder [23].

As described in [23], the attention mechanism introduces a slight difference in the decoder RNN from the previous section. This is seen in the probability computation of the output at step t:

$$P(y_t|y_{t-1}, y_{t-2}, ..., y_1, c_t) = g(h_t^d, y_{t-1}, c_t)$$

Instead of a single c vector, used to decode the entire output sequence, attention introduces a c_t unique to step t, defined as a linear combination of the hidden units of the encoder:

$$c_t = \sum_{i=1}^{n} \alpha_{t,i} h_i^e$$

where each output of the encoder h_i^e is weighted by $\alpha_{t,i}$ such that:

$$\alpha_{t,i} = \frac{exp(e_{t,i})}{\sum_{j=1}^{n} exp(e_{t,j})}$$

and

$$e_{t,i} = a(h_{t-1}^d, h_i^e)$$

The function a is the alignment model, which scores how well the input vectors near position i match to the output vectors near the position t [23]. One can imagine for language translation, corresponding words might show up in different parts of a sentence due to their different grammar structures. For example, ”blue car” in English is translated to ”voiture bleue” in French, where the noun and adjective ordering is reversed. The function a tries to capture this alignment between the two languages

Figure 3.3 provides a visual representation of the attention mechanism. Note that the notation in the figure is different than presented in this report.

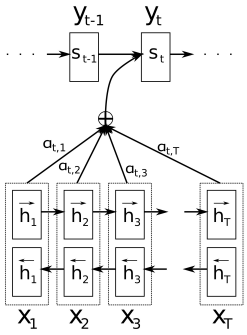

Figure 3.3: Attention Mechanism

[23]

3.3.3 Transformers

Also initially developed for machine translation, the general idea of the transformer model architecture is to stack attention mechanisms and fully connected layers [25]. The transformer structure is depicted in Figure 3.4. The encoder and decoder being the left and right halves of the transformer, are composed of repeating modules [25]. For the encoder, each module is composed of an attention and a fully connected layer [25]. For the decoder, each module is composed of an attention and fully connected layer as well as an attention layer over the output of the encoder stack [25]. Unlike encoder-decoder attention models, transformers only use attention and fully connected layers [25]. Without any recursive components, transformers have significant advantages over the previous RNN models [25]. Having a purely feed-forward architecture significantly reduces the computational complexity compared to training an unrolled RNN [25]. Moreover, transformers allow for more parallelization during training because outputs are no longer dependent on previous results, as is the case in RNN models [25]. Finally, transformers eliminate the vanishing gradient problem because the entire network is feed-forward. This also allows for long-range dependencies of input vectors [25].

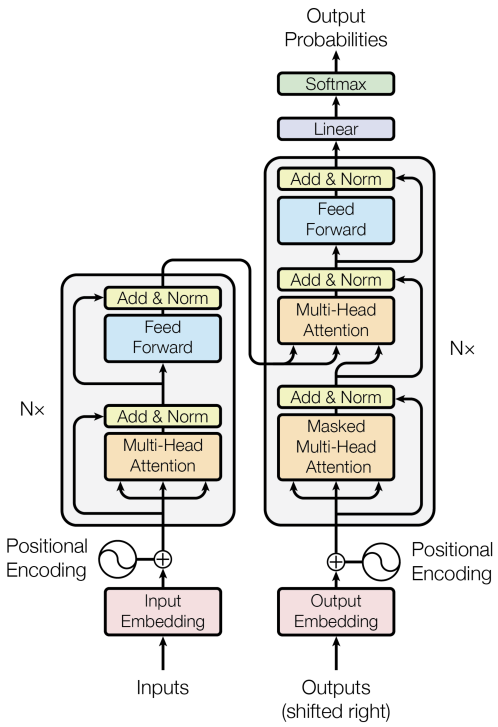

Figure 3.4: The Transformer Architecture

[25]

3.3.4 BERT Architecture

BERT's model architecture is constructed from multiple layers of bi-direcitonal transformer encoders [22].The BERT model is depicted in Figure 3.5. By adding an additional output layer to BERT to define a new model with a specific language task, the previously pre-trained weights of BERT can be transferred over to the new model [22]. As such, one would only need to fine-tune the added output layers to the specific task [22]. This is a ground-breaking application of transfer learning in NLP.

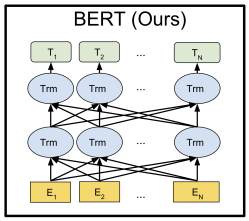

Figure 3.5: BERT Model Architecture

[22]

CHAPTER 4

THE BERT-BIO-NER MODEL

Due to the promising application of BERT for a number of different language tasks [22], we explored the idea of fine-tuning BERT for the task of BioNER.

4.1 Model Architecture

As described in their paper, by feeding BERT's final hidden representation into a classification layer trained over a labelled NER corpus, BERT was successfully fine-tuned to out-perform state-of-the-art models for a number of NER tasks [22]. By restructuring code from a PyTorch implementation of Google's BERT [26], we mimicked the architecture modifications depicted in the BERT paper, to approach NER tasks. The new BERT-Bio-NER model, having a core BERT model with pre-trained weights, was then trained with labelled BioNER corpora. The below figure depicts the model architecture.

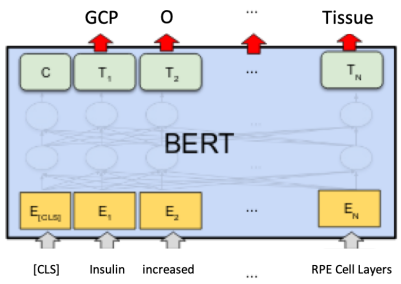

Figure 4.1: BERT-Bio-NER Model Architecture

[26]

4.2 K-Fold Cross-Validation Training

To achieve preliminary "proof-of-concept" results for BERT-Bio-NER, K-fold cross-validation was used to train and evaluate the model's performance.

4.2.1 Training Method

The K-fold cross validation method is a well-established training technique in machine learning. Unlike the traditional train, test, and validation data split, this technique involves first splitting the data set into k equal partitions [27]. The model is then trained on $k-1$ partitions of the data and tested on the k^{th} partition [27]. This is done k times, so that each partition is left out once and tested on the model that was trained on the $k-1$ other partitions [27].

This method is generally much better at getting more realistic representations of a model's performance [27]. This is because in the event that one partition might be "easier" or "harder" to get good results on, this result will be averaged out across the performance

of the other partitions [27]. On the other hand, when training on a single train, test, and valid split, by chance the split could generate better or worse performance, simply depending on how the data was split. This can be mitigated with cross validation.

For BERT-Bio-NER, each corpus was partitioned into 5 folds for cross validation. Averaging the performance across these 5 folds provided a much better representation of the model performance.

4.2.2 Performance Results

Without any hyper-parameter optimization or special training techniques, these results were gathered to gauge the potential of the BERT-Bio-NER model. Figure 4.2 compares the F_1 scores of BERT-Bio-NER against current state of the art BioNER performance. The performance of BERT-Bio-NER was acquired using 5-fold cross validation.

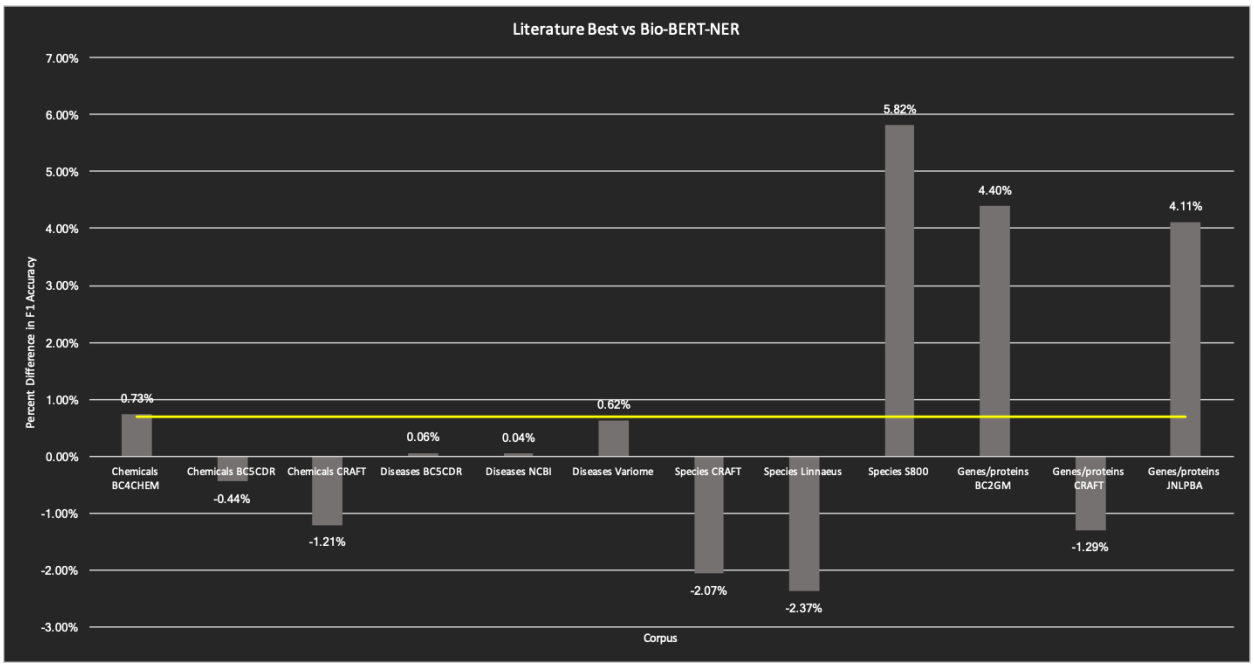

Figure 4.2: State-of-the-art vs. BERT-Bio-NER F_1 performance

[3]

The percentage difference was computed with the following formula:

$$\frac{F_1^{BERT} - F_1^{Lit-Best}}{F_1^{Lit-Best}} \times 100\%$$

An average 0.70% increase in F_1 score across the available corpora suggests that BERT-Bio-NER is indeed a promising approach to BioNER. Please see Table 6.1 for a full breakdown of the performance results.

4.3 Multi-Task Learning

It is expected that the performance from the previous section will continue to improve with future revisions of the BERT-Bio-NER model. In particular, model generalizability is an important concern when considering potential real world deployment of the model. As described in [3], current state-of-the-art BioNER models are unable to achieve high performance outside of the corpus that they were trained on. As such, we explored the generalizability of BERT-Bio-NER. Due to the core BERT model being pre-trained on massive data sets, we expected the BERT-Bio-NER model to be much better with respect to out-of-coprus performance.

As discussed in [3], using multi-task learning seems to significantly boost out-of-corpus performance. As such, we decided to use this training method to explore the out-of-corpus performance of BERT-Bio-NER.

4.3.1 Training Method

The intuition behind multi-task learning is to train a single model for a number of different but related tasks. [28]. This technique helps the models generalize in their tasks, by exposing them to domain information from the related tasks [28].

For our purposes, these different but related tasks will be the different biomedical corpora. The idea is to share the same core BERT model across our BERT-Bio-NER models for each corpus. In this way, BERT will be exposed to more biomedical text during the training process to help it learn signals specific to biomedical literature. Figure 4.3 presents a comparison of the previous single-tasked learning approach vs. the multi-task learning used in this section, for BERT-Bio-NER.

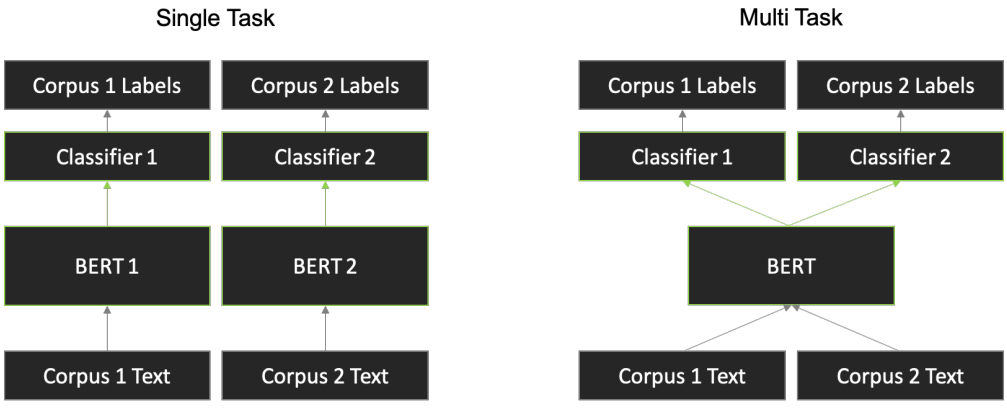

Figure 4.3: Multi-Task Learning for BERT-Bio-NER

Given the limited amount of data available for training BioNER models, multi-task learning tries to solve this problem by combining the data of multiple corpora and training it on a single BERT model. With more data, the model is much less likely to over-fit

to a specific corpus. Moreover, for a smaller corpus, multi-task learning will likely help with the performance because the model will be able to learn domain knowledge from a larger related corpus.

The models were trained using the multi-task training technique with corpora of the same entity type. Given that there are three corpora available for each entity type (chemicals, diseases, species, and genes/proteins), each model was trained on two of the three corpora. The model was then evaluated on these two corpora for in-corpus performance and also evaluated on the third corpora for out-of-corpus performance. This method of evaluation was borrowed from [3].

4.3.2 Performance Results

Figure 4.4 compares the F_1 scores of BERT-Bio-NER single task learning against BERT-Bio-NER multi-task learning, out-of-corpus performance.

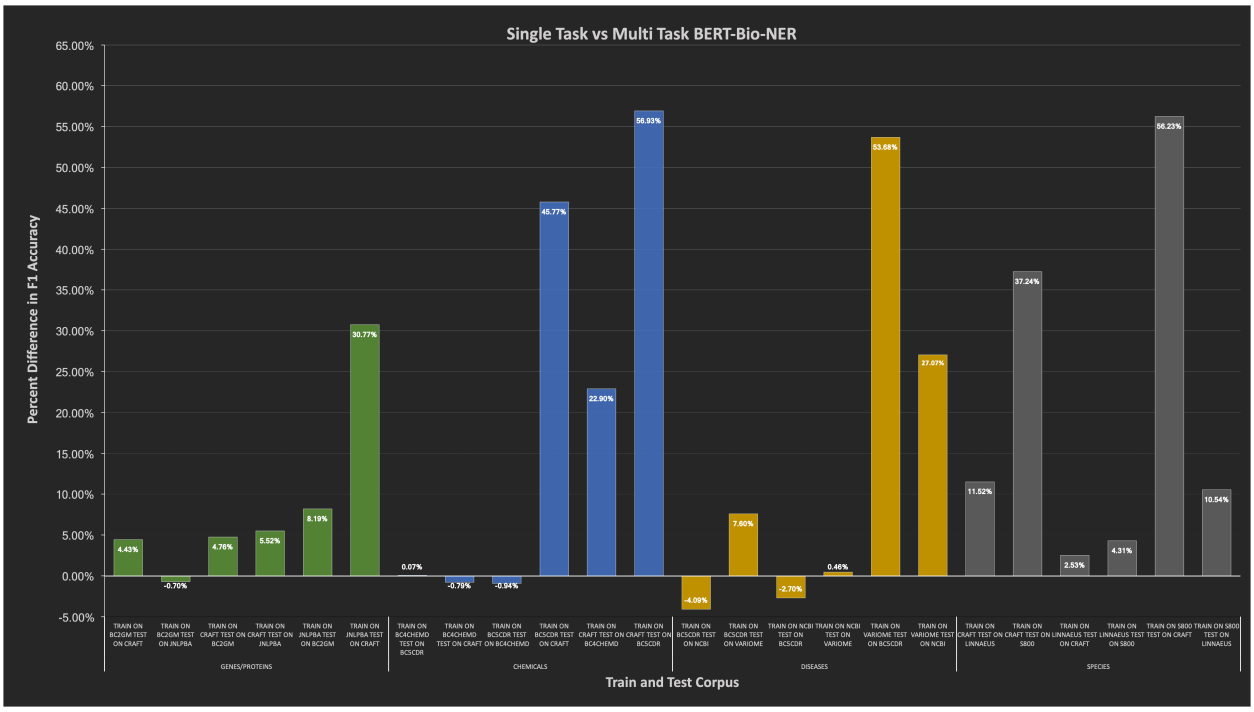

Figure 4.4: Single Task Learning vs Multi Task Learning for BERT-Bio-NER

Again, the percentage difference was computed with the formula:

$$\frac{F_1^{MultiTask} - F_1^{SingleTask}}{F_1^{SingleTask}} \times 100\%$$

There was an average 15.89% increase in F_1 score for out-of-corpus performance after

using multi-task learning. For most models, out-of-corpus performance either stayed the same or increased. This suggests that multi-task learning is quite effective at boosting out-of-corpus performance. Moreover, multi-task learning did not seem to negatively impact in corpus performance at all. Performance results showed that in corpus performance barely changed after applying the multi-task learning technique. Please see Table 6.2 for a full breakdown of the performance results.

CHAPTER 5

DISCUSSIONS AND CONCLUSIONS

5.1 BERT-Bio-NER is a Promising Approach

The results presented in this report suggest that the BERT-Bio-NER model is a very promising approach for future research in the field of biomedical named-entity recognition and biomedical information extraction as a whole. With 5-Fold Cross-Validation Training, we have shown that a ”vanilla” implementation of BERT-Bio-NER without hyper-parameter optimization or special training techniques can achieve state-of-the-art performance across most BioNER corpora. With more fine-tuning, it is very likely that we can significantly improve performance of this model. Moreover, these models were trained for about 10 epochs, whereas training Bi-LSTM-CRF models typically took up to 50 epochs. This reduction in training time itself presents BERT-Bio-NER as an extremely attractive alternative to previous state-of-the-art model architectures.

Out-of-corpus performance has long been a crippling drawback to most models in the field of bio-NER. [3] has shown that many current state-of-the-art models suffer from an inability to generalize. This has been one of the main reasons why these models have not been deployed for real-world applications. However, our results concerning out-of-corpus performance of BERT-Bio-NER have proven to be very promising in this respect. With a 15.89% increase in out-of-corpus F_1 performance, multi-task learning for BERT-Bio-NER seems to be an effective method of preventing over fitting and maintaining model generalization.

Interpreting Figure 4.4, we can see some interesting trends. For example, when using single task learning on smaller corpora, the out-of-corpus performance of these models is generally quite poor. This is understandable because the lack of training data would make it more difficult for the model to generalize. However, with multi-task training, the core BERT model is trained on a second corpora in addition to the smaller corpora. This provides parts of the model with more data during training and as such, a better ability to generalize. An example of this is for ”Train on S800 Test on CRAFT” under the Species entity. Because S800 is a relatively small corpora, using multi-task learning seems to significantly boost (56.23%) out of corpus performance F_1 accuracy. The model became exposed to more training data during multi task learning and as such, was less likely to over fit to the small corpora.

5.2 Future Work

These BERT-Bio-NER results show exciting potential that we are exploring a promising solution to BioNER. However, there are still many unexplored research directions that are likely to improve the current model’s performance. Moreover, there are many other

exciting applications of BERT in the field of biomedical information extraction that have yet to be explored.

5.2.1 Hyper-parameter Tuning

These BERT-Bio-NER results were trained without any hyper-parameter tuning. These hyper-parameters could include the learning rate, the architecture of the output NER layer, the batch size, and the number of training epochs. With hyper-parameter tuning, it is likely that we will be able to further improve the performance of BERT-Bio-NER.

5.2.2 Relation Extraction

Given the promising results of BERT-Bio-NER, it may be possible to approach the problem of relation extraction in a similar manner of transfer learning. It would be very exciting to explore the possibility of appending and fine-tuning an additional "relation extraction" layer to the output of the a high performance BERT-Bio-NER model.

There are other research groups that are also currently exploring BERT for BioNER and relation extraction [29]. However in terms of relation extraction, current approaches are relatively naive and only identify binary relationships (i.e. a single cause and a single effect). Moreover, they do not indicate relationship direction. For example, with gene-gene relationships, the currently proposed BERT RE models are unable to differentiate whether a specific gene entity is the cause or the effect in the gene-gene relationship. Thus, more sophisticated relation extraction models would definitely be an interesting research direction to pursue.

5.3 Comparing RNNs and BERT

It might initially seem quite intuitive that language data has a strong uni-directional time dependence. We certainly speak to each other through sound waves, a time series signal. We also read and write in a time-series fashion, one word after the next. As such, there seems to be some built-in time-based nature to language. It is therefore, not surprising that most NLP research has focused on time-series models such as HMMs and RNNs.

BERT is a model that seems to accommodate beyond the uni-directional temporal dependence of RNN models. I believe that interpreting BERT's fully feed forward non-recurrent approach to language processing begins with viewing language (English, French, etc.) as merely a tool for expressing our ideas. Although this tool that we use to express our ideas, language, may seem uni-directional and time-series in nature, ideas themselves are not time-based in the same way.

Figure 5.1 is a high-level comparison of how LSTMs and BERT's attention layers might be exposed to a sentence. Although in English, the sentence in Figure 5.1 translates to "the small blue car" where both adjectives are before the noun "car", the French sentence

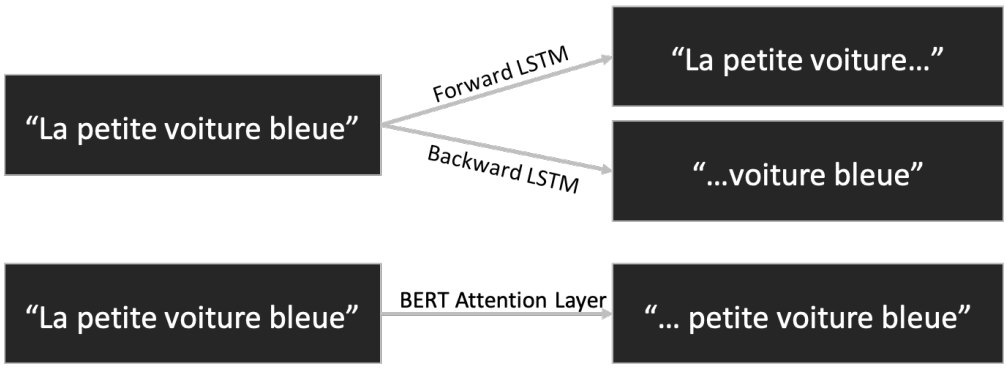

Figure 5.1: Comparing LSTM and Attention-based Model Approaches

has adjectives ”petite” and ”bleue” on either side of the noun, ”voiture”. This example shows that it is often necessary to read an entire sentence before understanding the complete idea that is being expressed. Being a fully feed forward model that takes an entire sentence as input at once, BERT’s attention layers probably take this sort of approach to language. LSTM models seem to have a stronger focus on directional (either forward or backward) dependence between entities. Thus, the LSTMs in Figure 5.1 will probably have a harder time realizing that the adjective-noun dependence between ”petite voiture” and ”voiture bleue” are the same relationships. Whereas BERT, which has no pre-defined directional bias between entities probably has an easier time identifying that both ”petite” and ”bleue” are adjectives of ”voiture”. The idea that is in this French sentence is the same as in the corresponding English sentence, but the time-series ordering of the actual words is different in each language. This should not affect a model’s interpretation because the ordering is merely due to grammatical differences in French and English. This would explain why BERT and its feed forward approach to language processing is much more effective than the traditional uni-directional and time-series approaches to NLP.

BERT is an exciting new advance in NLP research. It has provided a framework for transfer learning that was once nearly non-existent in the field of NLP. It will be exciting to see how future research work will continue to build on BERT in the field of biomedical information extraction.

CHAPTER 6
APPENDICES

6.1 Appendix A: K-Fold Cross Validation Performance

Entity	Corpus	Literature Best	BERT-Bio-NER	Percentage Difference
Chemicals	BC4CHEM	89.37%	90.03%	0.73%
	BC5CDR	92.82%	92.41%	-0.44%
	CRAFT	84.98%	83.95%	-1.21%
Diseases	BC5CDR	84.49%	84.54%	0.06%
	NCBI-Disease	87.01%	87.05%	0.04%
	Variome	86.05%	86.59%	0.62%
Species	CRAFT	97.74%	95.72%	-2.07%
	Linnaeus	93.54%	91.32%	-2.37%
	S800	74.98%	79.34%	5.82%
Genes/proteins	BC2GM	81.48%	85.36%	4.40%
	CRAFT	84.46%	83.37%	-1.29%
	JNLPBA	80/92%	84.24%	4.11%

Table 6.1: State-of-the-art vs. BERT-Bio-NER F_1 performance

[3]

6.2 Appendix B: Multi-Task Learning Performance

Entity	Train Corpus*	Test Corpus	F1 Accuracy				Percentage Difference	
			Single Task Learning		Multi Task Learning			
			In Corpus Validation	Out of Corpus**	In Corpus Validation	Out of Corpus**	In Corpus Validation	Out of Corpus
Chemicals	BC4CHEMD (CRAFT)	BC5CDR	90.41%	89.11%	90.52%	89.17%	0.13%	0.07%
	BC4CHEMD (BC5CDR)	CRAFT	90.18%	44.87%	90.12%	44.52%	-0.07%	-0.79%
	BC5CDR (CRAFT)	BC4CHEMD	93.95%	69.46%	93.90%	68.81%	-0.05%	-0.94%
	BC5CDR (BC4CHEMD)	CRAFT	94.28%	27.80%	95.00%	40.52%	0.77%	45.77%
	CRAFT (BC5CDR)	BC4CHEMD	88.20%	36.28%	88.94%	44.58%	0.83%	22.90%
	CRAFT (BC4CHEMD)	BC5CDR	87.95%	40.52%	90.75%	63.59%	3.19%	56.93%
Diseases	BC5CDR (Variome)	NCBI	90.15%	76.78%	90.57%	73.64%	0.46%	-4.09%
	BC5CDR (NCBI)	Variome	90.41%	68.74%	90.49%	73.97%	0.09%	7.60%
	NCBI (Variome)	BC5CDR	89.70%	70.61%	89.40%	68.71%	-0.33%	-2.70%
	NCBI (BC5CDR)	Variome	89.90%	73.11%	90.45%	73.45%	0.61%	0.46%
	Variome (NCBI)	BC5CDR	89.69%	40.70%	91.70%	62.55%	2.25%	53.68%
	Variome (BC5CDR)	NCBI	90.12%	46.01%	90.19%	58.46%	0.08%	27.07%
Species	CRAFT (S800)	Linnaeus	96.80%	53.13%	97.20%	59.25%	0.41%	11.52%
	CRAFT (Linnaeus)	S800	97.93%	35.53%	97.36%	48.77%	-0.58%	37.24%
	Linnaeus (S800)	CRAFT	95.69%	80.69%	95.36%	82.74%	-0.34%	2.53%
	Linnaeus (CRAFT)	S800	95.89%	61.21%	95.78%	63.85%	-0.12%	4.31%
	S800 (Linnaeus)	CRAFT	75.07%	52.64%	74.11%	82.24%	-1.27%	56.23%
	S800 (CRAFT)	Linnaeus	75.24%	59.81%	72.73%	66.12%	-3.34%	10.54%
Genes/proteins	BC2GM (JNLPBA)	CRAFT	86.32%	53.09%	85.37%	55.45%	-1.09%	4.43%
	BC2GM (CRAFT)	JNLPBA	86.56%	68.64%	85.38%	68.16%	-1.36%	-0.70%
	CRAFT (JNLPBA)	BC2GM	79.98%	43.93%	78.31%	46.02%	-2.09%	4.76%
	CRAFT (BC2GM)	JNLPBA	74.31%	52.63%	75.32%	55.53%	1.36%	5.52%
	JNLPBA (CRAFT)	BC2GM	84.58%	56.08%	84.80%	60.67%	0.26%	8.19%
	JNLPBA (BC2GM)	CRAFT	84.71%	39.43%	84.62%	51.56%	-0.10%	30.77%

Table 6.2: Single Tasked Learning vs Multi Tasked Learning for BERT-Bio-NER

* For multi task learning, Corpus1(Corpus2) indicates that the model was trained on both Corpus1 and Corpus 2, but the test corpus was evaluated on the Corpus1 model. For single task learning, the model was only trained on Corpus1.

** The "In Corpus Validation" reports the performance of the epoch with the best results for the validation data across all epochs during training. The "Out of Corpus performance" is the performance of the model on the test corpus during the epoch that was the best performing validation data epoch. It is not the best Out of Corpus performance across all training epochs. This is to simulate true out of corpus performance, where the model would only be tuned to the validation set performance.

6.3 Appendix C: BERT-Bio-NER Code

The code for this project can be found via the git repository:

`https://github.com/BaderLab/saber`

BIBLIOGRAPHY

[1] "Pubmed." https://www.ncbi.nlm.nih.gov/pubmed/, 2019. [Online; accessed 27-January-2019].

[2] J. Giorgi and G. Bader, "Transfer learning for biomedical named entity recognition with neural networks," *Bioinformatics*, 2018.

[3] J. Giorgi and G. Bader, "Towards reliable named entity recognition in the biomedical domain," *bioRxiv*, 2019.

[4] D. Hanisch, J. Fluck, H.-T. Mevissen, and R. Zimmer, "Playing biology's name game: identifying protein names in scientific text," in *Biocomputing 2003*, pp. 403–414, World Scientific, 2002.

[5] J. Kazama, T. Makino, Y. Ohta, and J. Tsujii, "Tuning support vector machines for biomedical named entity recognition," in *Proceedings of the ACL-02 workshop on Natural language processing in the biomedical domain-Volume 3*, pp. 1–8, Association for Computational Linguistics, 2002.

[6] S. K. Saha, S. Sarkar, and P. Mitra, "Feature selection techniques for maximum entropy based biomedical named entity recognition," *Journal of biomedical informatics*, vol. 42, no. 5, pp. 905–911, 2009.

[7] N. Ponomareva, F. Pla, A. Molina, and P. Rosso, "Biomedical named entity recognition: a poor knowledge hmm-based approach," in *International Conference on Application of Natural Language to Information Systems*, pp. 382–387, Springer, 2007.

[8] N. Collier, C. Nobata, and J.-i. Tsujii, "Extracting the names of genes and gene products with a hidden markov model," in *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pp. 201–207, Association for Computational Linguistics, 2000.

[9] D. Shen, J. Zhang, G. Zhou, J. Su, and C.-L. Tan, "Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain," in *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine-Volume 13*, pp. 49–56, Association for Computational Linguistics, 2003.

[10] R. McDonald and F. Pereira, "Identifying gene and protein mentions in text using conditional random fields," *BMC bioinformatics*, vol. 6, no. 1, p. S6, 2005.

[11] B. Settles, "Biomedical named entity recognition using conditional random fields and rich feature sets," in *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications*, pp. 104–107, Association for Computational Linguistics, 2004.

[12] C. M. Friedrich, T. Revillion, M. Hofmann, and J. Fluck, "Biomedical and chemical named entity recognition with conditional random fields: The advantage of dictionary features," in *Proceedings of the Second International Symposium on Semantic Mining in Biomedicine (SMBM 2006)*, vol. 7, pp. 85–89, BioMed Central Ltd, London UK, 2006.

[13] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning in natural language processing," *arXiv preprint arXiv:1807.10854*, 2018.

[14] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[15] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[16] M. Habibi, 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.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[19] X. Wang, Y. 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," *arXiv preprint arXiv:1801.09851*, 2018.

[20] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 328–339, 2018.

[21] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1717–1724, 2014.

[22] 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.

[23] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[24] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

[26] Tobias Sterbak, "Named entity recognition with bert." `https://www.depends-on-the-definition.com/named-entity-recognition-with-bert/`, 2019. [Online; accessed 28-January-2019].

[27] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, pp. 1137–1145, Montreal, Canada, 1995.

[28] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[29] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "Biobert: pre-trained biomedical language representation model for biomedical text mining," *arXiv preprint arXiv:1901.08746*, 2019.