© 2021 Jatin Arora

DOMAIN-AGNOSTIC NAMED ENTITY RECOGNITION ON UNSTRUCTURED TEXT

BY

JATIN ARORA

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois Urbana-Champaign, 2021

Urbana, Illinois

Adviser:

Professor Jiawei Han

ABSTRACT

Named Entity Recognition (NER) is the task of extracting informing entities belonging to predefined semantic classes from raw text. These semantic classes could be general purpose like person, location or domain-specific like genes, protein names in biomedical texts. NER has widespread applications in natural language processing (NLP) and serves as the foundation for applications like question answering, information retrieval and machine translation. Recently, the NER task has got a lot of traction in the research community with the advent of deep learning models like BERT which are able to capture textual semantics very well.

In this work, we present a detailed study approaching the NER task from three different perspectives, namely, sequence labeling, question answering (QA), and span-based classification. We propose a simple span detection and classification pipeline that first detects all mention spans irrespective of entity type and then feeds each mention span as input to a model and expects entity type as output. This setup is the reverse of a traditional QA-based NER system where we feed entity type as input and expect mention spans as output. We also introduce explicit pattern embeddings which compliment character embeddings to learn better word representations even with less training data. Experimental results demonstrate the effectiveness of our proposed domain-agnostic techniques on multiple datasets. We set the new state-of-the-art for `BioNLP13CG` and give competitive performance on `CoNLL 2003` and `JNLPBA` datasets. Additionally, we probe into the BERT model and show that mere concatenation of external feature vectors with BERT outputs may not train effectively at the recommended low learning rates for BERT. More sophisticated feature fusion is essential.

To my parents, for their love and support.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Prof. Jiawei Han for his continuous guidance and support. Prof. Han's teachings, his principles and dedication to work has always been a source of inspiration for me and has encouraged me to pursue my research interests relentlessly. With Prof. Han, I can confidently say that I have been treated not just as his student, but as his family. His care extends far beyond the call of duty and I am fortunate to have got a chance of being his student.

I would also like to thank all the members of our Data Mining Group. We are one big family and it has been a pleasure to work with such a talented group of peers and share ideas. Especially, I would like to thank Xuan Wang and Jiaming Shen for their guidance throughout my course of study.

I am also very grateful to Dr. Youngja Park, my mentor from IBM Research, without whom this work would not have been possible. Thank you so much Youngja for always being there for me.

Besides, I thank all my friends, well wishers and family members for always having faith in me and for their motivation and support in times of need.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

As quoted by Forbes[1], 2.5 quintillion bytes of new data is generated every day and this number is fast growing. In fact, 90 percent of all the data has been generated just in the last 2 years. Quite rightly said, it is indeed a *data-driven world*, and the future is going to be no different. This vast multitude of raw data is a gold mine with an abundance of knowledge hidden inside it. A lot of it is in the form of unstructured text including books, research papers, news articles, blog posts, customer reviews, and tweets. However, the pace of data growth has made it humanly impossible to manually browse through everything and get the required knowledge thus motivating research towards automated information extraction and knowledge discovery.

In essence, the ultimate goal is to acquire knowledge from raw data which in-turn helps guide decision making. This can be viewed as a step-by-step process in which the first step is to parse large volumes of raw text and extract informing entities along with their inter-relationships. Next is to organize this in the form of a knowledge graph preserving the interconnections. Finally, given a plain text query from the user, convert it into some graph operations to retrieve and return the relevant results.

In this thesis we focus on the first step, that is, information extraction (IE) which itself covers a diverse range of tasks. Named Entity Recognition (NER) deals with the extraction of important entities of interest from text. Relation extraction is the process of extracting inter-relationships among informing entities. Sentiment analysis deals with classifying the overall sentiment conveyed in given text. Question answering is the study of extracting an answer for a given question from a given input text. We primarily focus on the named entity recognition task here and next look at some of its applications.

Industry Documents. Most modern-day organizations have to deal with lots of documents. These include annual reports, purchase invoices, salary slips, client contracts, resumes, and emails. This makes up a vast and diverse pool of content-rich data which is persisted from a regulatory perspective and otherwise. Named entity recognition on this data can help draw important insights from past decisions and further improve the current business model. However, many such documents contain protected and sensitive information interspersed within. For example, employee records contain their address, date of birth, phone number, and social security number (SSN). which is private information protected by laws like CCPA(California Consumer Privacy Act). Another application of NER, from an information security perspective, is to identify and redact such content and prevent sensitive

[1]https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/?sh=7cf99c3f60ba

information leakage.

Scientific Literature. The research community is ever flourishing, and it has become increasingly difficult for any researcher to remain up to date with all the latest developments even in their domain itself. As quoted in the first ever machine generated book of Chemistry published in Springer Nature[1], even in the niche research domain of Li-ion batteries, there were more than 53,000 research papers released in just the past 3 years. It is hence essential to have information extraction and summarization tools to filter out important content and NER becomes a vital step in this process.

Web Data. In this age of the internet, the amount of online content has been exploding. There are around 500 million tweets everyday[2]. E-commerce has been blooming on websites like Amazon[3]. According to some stats[4], Amazon ships around 1.6 million packages per day. After making purchases, people voice their opinions and compare products and their features by writing reviews. Around 79% of customers check posted reviews before making a purchase. These vast volumes of reviews have lots of valuable information embedded including product names, their features, and opinion predicates, making it a trending NER application.

NER has been a popular research area for long and has seen transitions in parallel, with advances in the field of machine learning. Broadly, NER techniques can be categorized into rule-based, unsupervised, feature engineered and more recently, deep learning-based techniques. For NER, it is important to be able to understand the semantics of the sentence and how different entities are woven together by the rules of grammar and interacting with each other. In recent years, deep learning techniques like recurrent neural networks, LSTMs[2] and now transformer-based[3] architectures like BERT[4] have made great progress in language modeling and capturing these inter-token dependencies which has aided the performance of named entity recognition.

In this work, we develop several variants for NER on top of the vanilla BERT architecture making sure that the setup is domain-agnostic, that is, does not making any assumptions about the nature of entities in the underlying dataset. We present a thorough study and compare and contrast the NER task from three different perspectives, as a sequence labeling problem, a question answering task and span detection and classification task. The contributions of our work are:

- We introduce a novel pattern modeling approach which converts sparse character space to dense pattern space for more effective training of alphanumeric entities even with

[2]https://www.internetlivestats.com/twitter-statistics/
[3]https://www.amazon.com/
[4]https://landingcube.com/amazon-statistics/

minimal training samples.

- We propose a span detection and classification framework for NER which first queries on the input text to detect mention spans. It then does a novel reverse question answering wherein instead of querying to extract all mentions of a given entity type, we feed the detected mention spans as input and query for their entity type.
- Our approach is easy to apply and achieves competitive and even better results on general English as well as scientific and biomedical NER domains. We also set a new state-of-the-art for `BioNLP13CG` corpus consisting of 16 fine-grained named entity types.
- We present a study showing that mere concatenation of additional semantic features with BERT outputs may not able to effectively train and reach its full potential at the recommended low learning rates. More sophisticated feature fusion is essential.

CHAPTER 2: PRELIMINARIES

2.1 TASK FORMULATION

Formal Definition. Consider $\mathcal{T}$ as a set of entity types that the user is interested in extracting. Given a sentence $\mathcal{S}$ as a N-length sequence of tokens, $\mathcal{S} = \langle w_1, w_2 \dots w_N \rangle$, named entity recognition (NER) is defined as the task of extracting a list of tuples $\langle s, e, t \rangle$ such that $s \in [1, N]$ is the *start* index, $e \in [1, N]$ is the *end* index and $t \in \mathcal{T}$ is the *entity type*. Corresponding to each such tuple, the n-gram, $w_{s\,..\,e}$ is called a *mention* of entity type t.

Example. Let $\mathcal{T} = \{$`Person`, `Location`$\}$. Consider a tokenized input sentence $\mathcal{S} = \langle$*Emily*, *lives*, *in*, *United*, *States*$\rangle$. NER task is to output tuples $\langle 1, 1,$ `Person`$\rangle$ and $\langle 4, 5,$ `Location`$\rangle$. Here, *Emily* is a mention of entity `Person` and *United States* is a mention of entity `Location`.

2.2 NATURE OF ENTITIES

Language dynamics and style of writing varies from domain-to-domain and so does the nature of informing entities in corresponding sentences. This affects the approaches one should use for extracting entities and the expected performance. Some differentiating characteristics of entities are:

- **N-gram length**. Scientific literature include `chemicals`, `diseases`, `phenomena` which are all generally long and wordy while news articles include `person`, `organization`, and `location` which are comparatively smaller.

- **Intrinsic properties**. Common entities in news articles like `person`, `location` are wordy and alphabetic. They are called **word-based entities**. Frequently occurring ones can be learnt or else can be inferred from surrounding context. However there are entities like `driving license` and `telephone number` that have semantics encoded in the form of alphanumeric and special symbol patterns (like `telephone number` has form `xxx-xxx-xxxx` with a area code in the beginning). Similarly, in biomedical domain, there are `chemical` and `gene` abbreviations/formulas like *MgCl2* and *mutCK1delta gene*. Their semantics lies in the intrinsic language of chemistry where *Mg* is *Magnesium* and *Cl* is *Chlorine*. We call these **non-word entities**.

- **Diversity**. Some entities are *broad spectrum* with lots of diverse variety of mentions. For example, `chemicals` are sometimes written in words like *potassium permanganate* or as formula *KMnO4*, sometimes with intricate special symbols like, *Ca(2+)*. Other entities may be *narrow spectrum* following some distinct patterns, like `phone numbers`.
- **Hierarchy**. Depending on the use-case, not all entities to be extracted may be at the same level when organized in a concept taxonomy. For example, in computer science literature, we may be interested in extracting mentions related to `cybersecurity`, `RNN`, `CNN`, `transformers`. Here, `RNN`, `CNN`, `transformers` are all niche entity classes which come under the broad umbrella of machine learning and are at a lower level in hierarchy than `cybersecurity`.
- **Nesting**. Some entity mentions may be embedded within bigger mentions thus developing a nested structure. For example, *Goldman Sachs London* is an `Organization` while *London* is a `Location`. Several such examples can be found in the biomedical domain in `GENIA` corpus[5] and news article domain in `ACE 2004`[6] and `ACE 2005`[7] corpora.
- **Mention density**. This characteristic may have some influence of the *diversity* and *hierarchy* concepts described above. Depending on the domain, not all labeled entities may have an equable representation in the data. Some may have lots of examples. We call such entities as **high-resource entities** while others may have only a few and are called **low-resource entities**.

2.3 EVALUATION METRICS

All the results reported in this work are calculated on predefined test set data unless otherwise specified. We report *micro-averaged F1* score using *exact-match* evaluation over *entity spans.*

Definition 2.1. An entity mention tuple $\langle s, e, t \rangle$ extracted by NER system on a given sentence is considered correct iff both the boundary positions, s and e, along with the entity type t are correct as per ground truth.

Based on Definition 2.1, we count:

- True Positives (TP): Count of mention tuples returned by NER system which also appear in ground truth.

- False Positives (FP): Count of mention tuples returned by NER system which do *not* appear in ground truth.
- False Negatives (FN): Count of mention tuples which appear in ground truth but are not extracted by the NER system.

Precision refers to percentage of retrieved NER results which are correct. Recall refers to the percentage of ground truth entities correctly retrieved by our system. F_1 score is the harmonic mean of precision and recall. Note that here we are counting mention tuples as a whole and not separately for each entity type. So the values calculated are known as *micro-averaged* values over all entity types.

$$\text{Micro-Precision} = \frac{TP}{TP + FP} \qquad \text{Micro-Recall} = \frac{TP}{TP + FN}$$

$$\text{Micro-F}_1 = 2 \times \frac{\text{Micro-Precision} \times \text{Micro-Recall}}{\text{Micro-Precision} + \text{Micro-Recall}}$$

2.4 TAGGING SCHEME

In order to capture the boundaries of entity mentions correctly, it is a standard practice to assign a label to each token as per a tagging scheme. Some tagging schemes popular in NER literature are:

- `SingleTag`: Every token of an entity mention $\langle s, e, \texttt{Tag} \rangle$ is assigned label `Tag`. All other tokens are assigned a special label `O`.
- `BIO`: For every entity mention of the form $\langle s, e, \texttt{Tag} \rangle$, token at position s is assigned label `B-Tag` (`B` stands for *begin*). All other tokens in the mention are assigned `I-Tag` (`I` stands for *intermediate*). Remaining tokens, not part of any mention, are assigned `O` label.
- `BIOE`: Apart from the `B`, `I` and `O` labels in `BIO` scheme, here the token at position e is assigned `E-Tag` (`E` stands for *end*). Unigram mentions are labeled with `B-Tag`.
- `BIOES`: Additionally over `BIOE` scheme, unigram mentions are labeled `S-Tag` (`S` stands for *singleton*).

2.5 DATASETS

In this study we work with multiple English language datasets belonging to news and biomedical domains. Table 2.1 gives a summary of the datasets and nature of entities they posses. We define non-word entities, high/low resource, nesting in Section 2.2.

Dataset	**Domain**	**#Entities**	**Non-Word**	**High/Low Resource**	**Nesting**
`CoNLL 2003`	News	4	No	High	Flat
`OntoNotes 5.0`	News	18	Yes	High + Low	Flat
`BioNLP13CG`	Biomedical	16	Yes	High + Low	Nested
`JNLPBA`	Biomedical	4	Yes	High	Flat

Table 2.1: Datasets Summary

2.5.1 CoNLL 2003 Dataset

The `CoNLL 2003`[8] corpus is a collection of news wire articles from the Reuters corpus manually annotated with 4 entity types: `PER` (Person), `ORG` (Organization), `LOC` (Location) and `MISC` (Miscellaneous). We obtain the dataset from `datasets`[1] package. Around 16.8% of tokens in the dataset are part of some entity mention. From diversity perspective, the `MISC` class is more diverse than the other 3 classes. Only 4.6% tokens belonging to entity mentions are non-word. Hence all named entities are predominantly word-based. The corpus has no nesting and as per the distribution of entities shown in Table 2.2a, none of the entities are low-resource. Average sentence length is 14.5 tokens. Table 2.2b shows the `Train/Dev/Test` splits.

Entity	**Count**
`LOC`	10645
`PER`	10059
`ORG`	9323
`MISC`	5062

(a) Entity Distribution

Split	**# Sentences**
`Train`	14041
`Dev`	3250
`Test`	3453

(b) Data Split

Table 2.2: CoNLL 2003 Dataset Stats

[1]https://huggingface.co/datasets/conll2003

2.5.2 OntoNotes 5.0 Dataset

`OntoNotes 5.0`[9] is a large corpus consisting of text from various genres (news, weblogs, talk shows, broadcast, and conversational telephone speech) labeled with structural information (syntax) and shallow semantics (word sense linked to an ontology and coreference). There are 5 versions, from Release 1.0 to Release 5.0. We work with Release 5.0 and focus on named entity labels on English data. There are 18 entity types including common ones like `PERSON`, `ORG` and entities having numerical semantics like `DATE` and `MONEY`. We obtain the dataset from GitHub[2]. Around 11.0% of tokens in the dataset are part of some entity mention. There are some entities like `GPE` (geo-political entity) and `LOCATION` which have very subtle difference in their semantics making them challenging to disambiguate. Corpus is flat-labeled and around 19.3% entity-mention tokens are non-word. As per Table 2.3a, entities like `PERSON`, `DATE` are high-resource while `LANGUAGE`, `LAW` are low-resource. Average sentence length is 19.0 tokens. Table 2.3b shows the `Train/Dev/Test` splits.

Entity	**Count**
`ORG`	29963
`GPE`	28133
`PERSON`	27332
`DATE`	23786
`CARDINAL`	13626
`NORP`	11608
`MONEY`	6425
`PERCENT`	4866
`ORDINAL`	2737
`LOC`	2691
`TIME`	2289
`WORK_OF_ART`	1650
`QUANTITY`	1583
`FAC`	1440
`PRODUCT`	1296
`EVENT`	1273
`LAW`	568
`LANGUAGE`	412

(a) Entity Distribution

Split	**# Sentences**
`Train`	115812
`Dev`	15680
`Test`	12217

(b) Data Split

Table 2.3: OntoNotes 5.0 English NER Dataset Stats

[2]https://github.com/yuchenlin/OntoNotes-5.0-NER-BIO

2.5.3 JNLPBA Dataset

The JNLPBA[10] dataset comes from GENIA[5] corpus (version 3.2) and contains abstracts of papers taken from MEDLINE database. The GENIA corpus consists of 36 fine-grained nested named entity types. For preparing JNLPBA, some of these entity classes are combined to a higher-level entity class and others are ignored. In all, JNLPBA has 5 flat-labeled entity types: `protein`, `DNA`, `RNA`, `cell_line`, `cell_type`. We obtain the dataset available from GitHub[3]. Around 21.7% of tokens in the dataset are part of some entity mention. Around 31.1% of entity mention tokens are non-word. As per Table 2.4a, most entities are high resource although representation of `RNA` is comparatively less. Average sentence length is 26.5 tokens. Table 2.4b shows the `Train/Dev/Test` splits.

Entity	**Count**
protein	35336
DNA	10589
cell_type	8639
cell_line	4330
RNA	1069

(a) Entity Distribution

Split	**# Sentences**
Train	16807
Dev	1739
Test	3856

(b) Data Split

Table 2.4: JNLPBA Dataset Stats

2.5.4 BioNLP13CG Dataset

The BioNLP13CG[11] (Cancer Genetics) dataset comes from BioNLP Shared Task 2013. The text belongs to the theme of biological processes relating to the development and progression of cancer. It consists of 16 entity types with a mix of high-resource and low-resource ones. We obtain the dataset available on the shared task website[4] and process it into `tsv` format. For most part of NER study when using this dataset we focus on flat-annotated entity mentions and ignore the small percentage (around 1%) of nested entities both from training and evaluation. The flat-annotated corpus is consistent with the one available on GitHub[5]. Around 23.5% of tokens in the dataset are part of some entity mention. Around 24.9% of entity mention tokens are non-word. Average sentence length is 27.6 tokens. Table 2.5a shows the distribution of entities and Table 2.5b gives the `Train/Dev/Test` splits.

[3]https://github.com/cambridgeltl/MTL-Bioinformatics-2016/tree/master/data/JNLPBA
[4]http://2013.bionlp-st.org/tasks/cancer-genetics
[5]https://github.com/cambridgeltl/MTL-Bioinformatics-2016/tree/master/data/BioNLP13CG-IOB

Entity	**Count**
Gene_or_gene_product	7908
Cell	3492
Cancer	2582
Simple_chemical	2270
Organism	1715
Multi-tissue_structure	857
Tissue	587
Cellular_component	569
Organ	421
Organism_substance	283
Pathological_formation	228
Amino_acid	135
Immaterial_anatomical_entity	102
Organism_subdivision	98
Anatomical_system	41
Developing_anatomical_structure	35

(a) Entity Distribution

Split	**# Sentences**
Train	3033
Dev	1003
Test	1906

(b) Data Split

Table 2.5: BioNLP13CG Dataset Stats

2.6 EXPERIMENTAL SETUP

We report all our results on the test sets after taking the model checkpoint corresponding to the best micro-averaged F1-score on development set. The development set evaluation takes place at steps of 0.5 training epochs. We train the models for 300 epochs at learning rate 10^{-5} unless otherwise specified.

We use `transformers`[6] python library by HuggingFace and `pytorch` for implementation and fix random seed to 42 for replication. For general English corpora like `CoNLL 2003` and `OntoNotes 5.0`, by default, we use the pretrained `bert-base-uncased`[7] model. For biomedical datasets, `BioNLP13CG` and `JNLPBA`, we use `BioBERT-Base`[8] model. Note that in all our experiments, we only use the BERT-Base architecture which has around 110M trainable parameters. We use `Nvidia GeForce GTX 1080` and `Nvidia Tesla V100` gpus for model training and evaluation.

We use cross entropy loss during training unless otherwise specified. The training data is randomly shuffled and a batch size of 16 is used with post-padding. For BERT-based models, we fix maximum sequence length to 256 for `BioNLP13CG`, `CoNLL 2003`, `JNLPBA` datasets and

[6]https://github.com/huggingface/transformers
[7]https://huggingface.co/bert-base-uncased
[8]https://github.com/dmis-lab/biobert#download

512 for `OntoNotes 5.0` data. Unless otherwise specified, the BERT-based models output entity labels for each sub-token (as per WordPiece tokenization) of an existing token in the dataset. As a heuristic, we take the label of first sub-token as the label for the corresponding token during our evaluations.

CHAPTER 3: METHODOLOGY

We approach the named entity recognition problem from different perspectives, as a sequence labeling task, a question answering task and a span detection and classification task. In this chapter, we look at all these in detail and compare and contract them with one-another. Primarily, we develop on top of pretrained BERT model. We look at the effect of loss function, tagging schemes, and providing additional semantic information on the overall task. We also study how much guidance the model derives from any additional information fed with the input. Experiments are conducted on multiple datasets and we simultaneously compare our models with previously published state-of-the-art results as well.

3.1 SEQUENCE LABELING

Definition 3.1. Given sentence $\mathcal{S}$ as a N-length sequence of tokens, $\mathcal{S} = \langle w_1, w_2 \ldots w_N \rangle$, sequence labeling is the task of forming output sequence $\mathcal{L} = \langle l_1, l_2 \ldots l_N \rangle$ where l_i is the label assigned to token w_i.

For NER, we extract entity mentions by making output labels follow a tagging scheme (Section 2.4). For example, with `BIO` scheme, each output label is of the form `B-Tag`, `I-Tag` or `O` with `Tag` $\in \mathcal{T}$ where $\mathcal{T}$ is the set of all possible entity types. Figure 3.1 gives a diagrammatic representation of our setup. In all our sequence labeling experiments, we make use of `BIO` tagging scheme and train the following variants:

- `BERT`: Proposed by [4], BERT is a bidirectional encoder transformer[3]. It applies WordPiece[12] tokenization on input sentence which is then passed through several encoder layers with multiple attention heads capturing sentence semantics and inter-token relationships well. The model outputs contextualized embeddings for each sub-token in the sentence. We take the last hidden layer outputs from BERT model and pass it to a fully connected layer. The outputs are converted to a probability distribution over labels space. Model parameters are initialized from a pretrained model and fine-tuned on our NER task.

- `BERT-Freeze`: To understand how much semantic information is already captured in a pretrained BERT model, we use the exact same architecture as `BERT` model above but freeze the BERT model parameters. So, the only trainable parameters remain from the fully connected layer. For this setting, we use learning rate as 0.005.

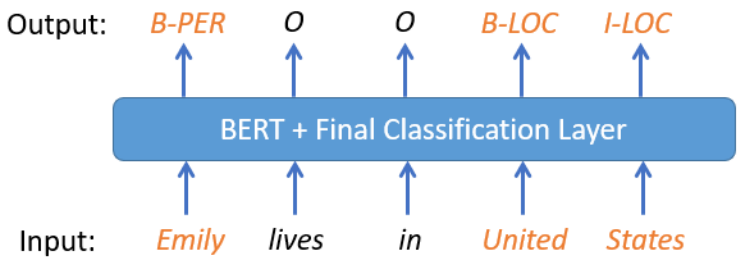
Output: B-PER O O B-LOC I-LOC
BERT + Final Classification Layer
Input: Emily lives in United States

Figure 3.1: Sequence Labeling Setup with `BIO` tagging scheme (colored tokens are the gold entity mentions and expected output labels)

Model	BioNLP13CG	JNLPBA	CoNLL 2003	OntoNotes 5.0
`BERT-Freeze`	75.42	55.93	82.79	67.35
`BERT`	**85.99**	**74.35**	**91.36**	**83.39**

Table 3.1: Results: Sequence Labeling (Test set Micro-F1 in %)

3.1.1 Observations

Based on results summarized in Table 3.1 we see that BERT-Freeze serves as a naive yet strong baseline. Nevertheless, as expected, after fine-tuning the BERT model we get much better results since the representative power of the model increases many fold.

3.2 QUESTION ANSWERING

In recent years there has been a trend of formulating NER problems as question answering(QA) tasks. [13, 14] model relation extraction as QA tasks and [15] propose a QA-based multi-task learning setup.

For NER using a QA setup, we feed a question to the model asking it to extract all mentions of a given entity type from the supplied text. Since there can be multiple entities of interest, each combination of entity question and input sentence is fed to the model. [16, 17] show the effectiveness of such a setup using BERT over multiple general English and Chinese news datasets and output candidate spans (start and end indices) where the entity in question is present. This setup has advantages over sequence labeling since it can handle nested entities as well.

However, for learning spans (start and end indices), the classification layer has to do $\mathcal{O}(n^2)$ computations where n is the number of tokens in the input sentence. Such a computation can be expensive for large sentences. To mitigate this, [18] propose an approach in the middle of question answering and sequence labeling. They return `B`, `I` and `O` labels for each token to mark the presence of the entity in question. Hence, their problem complexity becomes

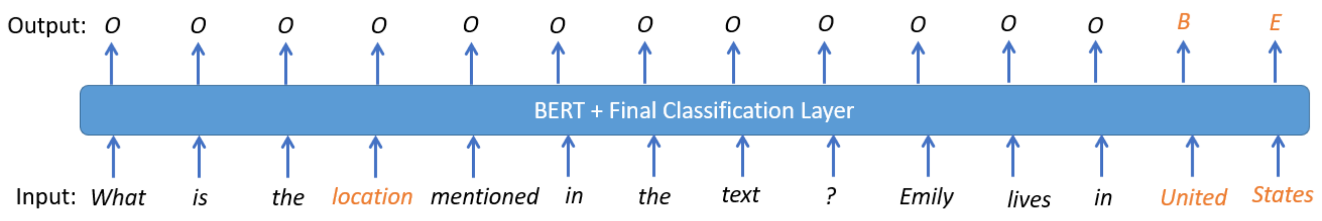
Output: O O O O O O O O O O O O B E
BERT + Final Classification Layer
Input: What is the location mentioned in the text ? Emily lives in United States

Figure 3.2: Question Answering Setup with `BIOE` scheme and *What* as question word (colored tokens depict the entity type in question and gold entity mention with expected output labels)

$\mathcal{O}(n)$ since they output one label for each token. We implement this framework (shown in Figure 3.2) and study the effect of several factors:

- **Tagging Scheme**: Classify each token using `BIO` or `BIOE` scheme. In both these models, we ask questions of the form, *What is the person mentioned in the text?*.

- **Question Formulation**: In QA setup, extraction from a sentence is highly dependent on the question semantics. We study the importance of the question word being used (`What` or `Where`). Our sample question is of the form, ***What*|*Where* is the person mentioned in the text?** In both these models, we follow the `BIOE` output tagging scheme.

- **Entity Scrambling**: In the question, *What is the **entity** mentioned in the text?*, we study the effect of the entity keyword used. We replace each entity with some scrambled English letters, for example, `Person` becomes `xyz12qqr`. So, during training, we ask questions like *What is the **xyz12qqr** mentioned in the text?* but give the right `person` mentions as gold labels to the model. Since our task here is to probe the model and develop an understanding of what it focuses on, so we conduct this experiment on a single dataset, `BioNLP13CG`. Table 3.2 shows some example scrambled entity keywords. We compare it with the unscrambled model in the same `BIOE` tagging scheme setting.

Original Entity Name	**Scrambled Entity Name**
Cancer	OUYOFhok
Amino_acid	DJHkjh KJDSjh
Organ	UQUIhkjsndf
Cell	OIFoisjf

Table 3.2: Entity Scrambling Examples

Model	BioNLP13CG	JNLPBA	CoNLL 2003
`BERT-QA(BIO)`	86.15	74.81	91.06
`BERT-QA(BIOE)`	**86.45**	**74.92**	**91.17**

Table 3.3: Results: Question Answering (Tagging Scheme) (Test set Micro-F1 in %)

Model	BioNLP13CG	JNLPBA	CoNLL 2003
`BERT-QA(What)`	86.45	**74.81**	91.17
`BERT-QA(Where)`	**86.83**	74.64	**91.82**

Table 3.4: Results: Question Answering (Question Formulation) (Test set Micro-F1 in %)

Model	BioNLP13CG
`Original`	**86.45**
`Scrambled`	85.83

Table 3.5: Question Answering (Entity Scrambling) (Test set Micro-F1 in %)

3.2.1 Observations

- From Table 3.3, `BIOE` tagging is able to better capture the entity boundaries as compared to `BIO` tagging scheme.
- From Table 3.4, we observe that the model is sensitive to slight changes in question semantics. Since the ultimate goal of the model is to identify mention boundaries, asking `Where` works better than `What`.
- From Table 3.5, we observe that as expected, `Original` keyword model works better than `Scrambled`. However the contribution of entity keyword is very minimal. It is possible to form some logical groups of entity mentions with unknown group name and the model should still be able to distinguish and differentiate the group well. We use this finding in Section 3.8.

3.3 SPAN DETECTION AND CLASSIFICATION PIPELINE

Previous approaches like sequence labeling and question answering (QA) treat the NER problem as a whole. One single model must take a sentence as input and return mention tuples with correct boundaries and correct entity type. Another possibility is to have a division of labor. We break down the NER problem into a two-step pipelined process. In the first step (**Span Detector**), we detect all mention spans in a given sentence. In the next step (**Span Classifier**), we classify these spans into their corresponding entity type. Now,

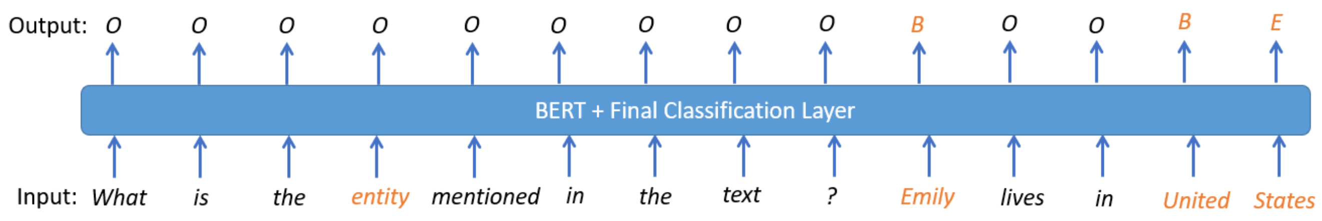
Output: O O O O O O O O O B O O B E
BERT + Final Classification Layer
Input: What is the entity mentioned in the text ? Emily lives in United States

Figure 3.3: Span Detection Setup with `BIOE` scheme and *What* as question word (colored tokens depict the generic entity type in question and gold entity mentions with expected output labels)

we can train two separate models independently which specialize in their own sub-tasks and together solve the NER problem. We borrow the basic intuitions of QA model to solve both our sub-tasks.

3.3.1 Span Detection

Given a sentence $\mathcal{S}$ as a N-length sequence of tokens, $\mathcal{S} = \langle w_1, w_2 \dots w_N \rangle$, the goal of this module is to output a list of spans (mention tuples) $\langle s, e \rangle$ where $s \in [1, N]$ is the *start* index, $e \in [1, N]$ is the *end* index. Note that here the mention tuples are not associated with an entity type.

We formulate this as a question answering task asking the model to identify all entity spans in a given sentence. For example, the sentence, *Emily*[`PERSON`] *lives in United States*[`LOCATION`], is converted to the input, *What is the* ***`entity`*** *mentioned in the text? Emily lives in United States*. This is fed to BERT model which outputs labels for each token following the `BIOE` scheme. In this example, we expect two spans, *Emily* and *United States*. Figure 3.3 shows our span detection setup.

3.3.2 Span Classification

Here, we are given a sentence $\mathcal{S}$ as a N-length sequence of tokens, $\mathcal{S} = \langle w_1, w_2 \dots w_N \rangle$ and a span $\langle s, e \rangle$ where $s \in [1, N]$ is the *start* index, $e \in [1, N]$ is the *end* index. The goal is to output a label t for the span such that $t \in \mathcal{T}$, where $\mathcal{T}$ is the set of all entity types.

This is modeled as the reverse of QA model for NER described in Section 3.2. For every gold entity mention (E.g. *United States*) in a training set sentence, *Emily*[`PERSON`] *lives in United States*[`LOCATION`], we form a sample input, *Emily lives in United States. What is United States?* The sentence is fed to a BERT model where we do sequence classification. The pooled sequence embedding returned by BERT is fed to a fully connected layer and converted to a probability distribution over possible entity types. In this example, the model is expected to assign maximum probability to `LOCATION`. Figure 3.4 shows our span classification setup.

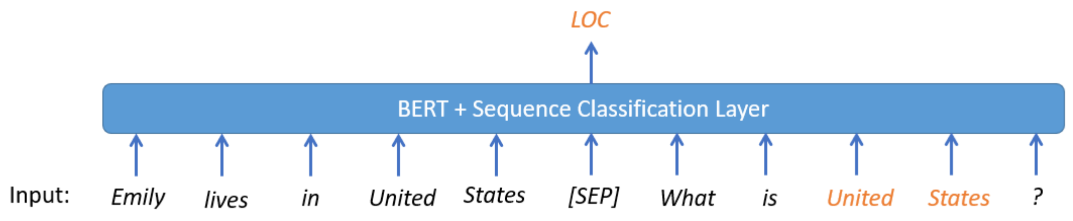

Figure 3.4: Span Classification Setup (colored tokens depict the entity mention in question with expected output entity label)

3.3.3 Pipeline

Both the models can be trained independently. The pipeline structure comes during the inference time. Here, every unlabeled sentence is first passed through Span Detector and for each output span, we convert to an input sample for Span Classifier.

3.3.4 Salient Features

- Compared to sequence labeling and question answering approach, this span-based approach has more representative power. This is because here we have two BERT models each working on their own sub-tasks and contributing towards better NER while the other approaches just have a single model.

- Even though we are training two BERT models, they can be trained independently, in parallel. Only at inference time, we need to maintain the sequential nature.

- If we have T entities of interest, then standard question answering approach creates T samples for each input sentence both at train and inference time. Considering that each sentence on an average has much lesser than T entity mentions, there is a lot of redundancy in this approach.

- Our span-based approach removes QA model redundancy even though inherently we have a QA-based setup. Span Detector only sees an input sentence once and identifies all mention spans. The span classifier will work on only these identified mention spans and classify them into an entity type.

- Nevertheless, our approach has a pipeline-based structure and hence errors made by span detector propagate to the classifier. Sequence labeling and question answering approaches do not face this concern.

- Our span-based approach shows the effectiveness of *reverse question answering*. For a sentence, *Emily lives in United States*, rather than asking a question of the form, *"What is the* ***`Person`*** *mentioned in the text?"*, we ask, *"What is* ***`Emily`****?"*. This opens up prospects for more intuitive forms of approaching NER, taking us closer to human understanding and interpretations.

- Comparable and even improved performance of this span-based approach compared to the general QA NER setup (results in Table 3.6) shows that boundary detection of mentions has less correlation with the entity type it belongs to.

	BioNLP13CG	**JNLPBA**	**CoNLL 2003**
`Span Detection`	90.12	78.35	95.23
`Span Classification`	94.06	95.08	94.50
`Pipeline`	85.89	**75.01**	**91.64**
`BERT-QA`	**86.45**	74.81	91.17

Table 3.6: Results: Span Pipeline (Test set Micro-F1 in %)

3.3.5 Observations

Table 3.6 reports the results of the pipelined span detection and classification procedure and compares it with simple BERT QA setup. We present this comparison since QA model serves as the primary backbone of our span-based approach. All models here use `BIOE` tagging scheme and use *What* as the question word in question formulation.

- **Span Detection**: Detecting all mention spans together without classification is a simpler problem for the model than full NER and we get better performance on this sub-task compared to complete NER task in QA setup.

- **Span Classification**: Given that spans are pre-identified, classifying them to an entity type is a relatively simple task for the BERT model. On all datasets, we see around 95% Micro-F1 on test set.

- **Pipeline**: The pipelined procedure gives comparable and even better performance than standard QA NER model on all datasets demonstrating the effectiveness of this division of labor.

- Since out `Pipeline` results are comparable to `BERT-QA` model, we conclude that internally `BERT-QA` model also tries to logically segregate boundary detection and classification as separate tasks.

- The results of span pipeline are limited by the performance of the span detector part. Since this procedure is pipelined, errors in this first step propagate to the next step. Boundary detection serves as the primary challenge in Span Detector and has a large scope for improvement on biomedical datasets.

- Qualitative analysis reveals that both `BERT-QA` and `Span Detector` share very similar boundary detection issues as highlighted in Section 3.5.1.

3.4 LEARNING OBJECTIVE VARIATION

An ML algorithm learns by optimizing its learning objective (loss function). In this section, we use some standard popular loss functions and also design our new ones and study their effect on model performance. In all our experiments here, we use sequence labeling setup with `BIO` tagging scheme. Without loss of generality, we use the binary classification setting while illustrating the different loss functions. Let $\mathcal{X}$ be the set of all training samples such that a sample $x_i \in \mathcal{X}$ is associated with ground truth label, $y_i = [y_{i0}, y_{i1}]$ where $y_{i0}, y_{i1} \in \{0, 1\}$. Let $p_i = [p_{i0}, p_{i1}]$ be model prediction probabilities such that $p_{i0}, p_{i1} \in [0, 1]$ and $p_{i0} + p_{i1} = 1$. Let N be the batch size.

3.4.1 Cross Entropy (CE) Loss

As can be seen from Equation (3.1), Cross Entropy loss penalizes misclassifications. When the classification is correct, it pushes the model to output the correct result with a probability of 1. Hence, Cross Entropy loss models accuracy. However, during evaluation in NER setting, we calculate F1-Score. This difference between training objective and evaluation metric may lead to sub-optimal performance in some situations.

$$CE = -\frac{1}{N} \sum_{i} \sum_{j \in \{0,1\}} y_{ij} \log p_{ij} \tag{3.1}$$

3.4.2 Weighted CE Loss

Not all entity types may have an equable distribution in the labeled dataset. Generally negative samples outweigh positive samples in NER datasets. Only 23.5% tokens in `BioNLP13CG` dataset belong to some entity, rest are labeled `O`. This may introduce bias in the model which can be countered by giving different importance/weights to different entity types. In our case, we want to reduce the dominant influence of `O` labels, hence, any token with a gold label `O` is given a weight of 0.5 while all others have a weight of 1. This can be further extended to give different weights to high and low resource entity types. However, this approach makes the model very sensitive to the assigned weights and it can rather easily develop a bias towards low-resource entities[19].

3.4.3 Punctuation Weighted CE Loss

From qualitative analysis of misclassified samples (Section 3.5.1) in standard CE loss setup we notice that the model is not able to learn good representations for special symbols like parenthesis, hyphen, period, or slash. Hence, we emphasize these symbols by penalizing the model twice if the misclassified token is a punctuation/special symbol.

3.4.4 Dice Loss

As detailed earlier, Cross Entropy is an accuracy-oriented objective while during evaluation, we calculate the F1-Score. This difference can lead to sub-optimal model training. To counteract, [17] make use of Sørensen-Dice coefficient(DSC)[20, 21] and Tversky index[22] which are F-Score oriented statistics. Given sets A and B, DSC is used to gauge similarity among two sets and is defined as,

$$DSC(A, B) = \frac{2\ |A \cap B|}{|A| + |B|} \tag{3.2}$$

Consider A as set of all positive samples predicted by a model and B as set of all ground truth positives. Then, by definition of true positive (TP), false positive (FP) and false negative (FN) from Section 2.3, we have,

$$TP = |A \cap B| \qquad |A| = TP + FP \qquad |B| = TP + FN$$

$$DSC(A,B) = \frac{2TP}{2TP + FP + FN} \quad = \frac{2\ \frac{TP}{TP+FN}\ \frac{TP}{TP+FP}}{\frac{TP}{TP+FN} + \frac{TP}{TP+FP}} = \frac{2\ Precision \times Recall}{Precision + Recall} \quad = F1$$

The dice coefficient gives equal importance to false-positives and false-negatives at training time and is more immune to data-imbalance issues[23, 24, 25]. The above formulation shows its equivalence to F1-score thus removing the discrepancy among training and evaluation metrics. From Equation (3.2), for an individual sample x_i, dice coefficient is defined as,

$$DSC(x_i) = \frac{2p_{i1}y_{i1} + \gamma}{p_{i1} + y_{i1} + \gamma}$$

where γ is the smoothing parameter. Then over all samples, dice loss (DL) is defined as:

$$DL = 1 - \frac{2\sum_i p_{i1}y_{i1} + \gamma}{\sum_i p_{i1} + \sum_i y_{i1} + \gamma} \tag{3.3}$$

3.4.5 Conditional Random Field

[26] show the effectiveness of a conditional random field (CRF) in modeling output label transitions in sequence labeling settings when added over a bidirectional LSTM. In this setup, we apply a similar CRF layer on top of BERT for assigning output labels to each token. For CRF implementation, we use `torchcrf` python package.

	BioNLP13CG	**CoNLL 2003**
`CE Loss`	85.99	91.36
`Weighted CE Loss`	85.93	91.26
`Punctuation CE Loss`	85.74	91.55
`Dice Loss`	86.35	90.76
`CRF`	86.20	91.23

Table 3.7: Results: Learning Objectives (Test set Micro-F1 in %)

3.4.6 Observations

- `Weighted CE Loss` and `Punctuation CE Loss` give mixed results. Their performance varies with the weights set for different misclassification cases. The same set of weights do not generalize across both datasets.

- In `BioNLP13CG` corpus, `Dice Loss` performs well as this corpus as several high and low resource entities (data imbalance). `Dice Loss` is not able to give its advantages with CoNLL 2003 corpus since here all 4 entity types have a comparable and high representation.

- From a sequence labeling perspective with BIO tagging scheme, we get $2K + 1$ output entity classes for K entity types. This means BioNLP13CG corpus with 16 entity types requires 33 output classes and CoNLL 2003 with 4 entities requires 9 output classes. A CRF primarily captures tag transitions and is found to be helpful when number of output labels is more, that is, on BioNLP13CG corpus. On CoNLL 2003 data, it gives comparable performance to the base model.

3.5 CAPTURING ADDITIONAL TOKEN SEMANTICS

After having run through the three major approaches for NER namely, sequence labeling, question answering and span detection and classification, in this section we present a qualitative error analysis. Next, we address the identified issues by either architectural modification or additional input features.

3.5.1 Qualitative Error Analysis

Different approaches have their own strengths and weaknesses highlighted later in Section 3.9. However there are several common errors which can be segregated into three major categories:

- **Out of Vocabulary Terms**: Both in news articles and research texts, it is common to coin new terms and abbreviations to describe latest events or new concepts. Such terms are pretty much localized to that article and rare otherwise. Additionally scientific texts also have chemical formulas which have numerals encoded within. Semantics of many such terms may not be captured well by generically pretrained BERT models. Although BERT uses sub-word semantics using WordPiece tokenizer but such sub-word combinations may still be rare. Hence we call these *out-of-vocabulary* terms. Table 3.8 shows some errors made by our models on BioNLP13CG corpus which fall in this category.

- **Special Symbols**: Several entity mentions have hyphens, periods, parenthesis within them. Pretrained embeddings do not capture their semantics well leading to boundary detection issues. Table 3.9 shows some errors of this type from BioNLP13CG dataset.

Entity	**Misclassification Examples**
`Gene_or_Gene_Product`	DPD, XhoI, mutCK1delta, FAS
`Simple_Chemical`	MnCl2, AglRhz, NO
`Cell`	LoVo, DeltaG45, BMSVTs
`Amino_Acid`	phosphoS727, Y705F

Table 3.8: Out-of-Vocabulary terms in `BioNLP13CG` corpus

- **Modifier Suffix/Prefix**: Apart from the root entity required to be extracted the gold labels sometimes expect a modifier term as well which occurs as a prefix/suffix. Missing these leads to boundary detection issues. Table 3.9 lists some error cases of this type from `BioNLP13CG` corpus.

Misclassification Category	**Gold**	**Predicted**
`Special Symbols`	L . Se (+) cells	L . Se
`Special Symbols`	Gs - IB (4 -) ion	Gs - IB (4
`Modifier Suffix/Prefix`	epicardial coronary artery	coronary artery
`Modifier Suffix/Prefix`	T140 analogs	T140

Table 3.9: Boundary detection issues in `BioNLP13CG` corpus

In general, we conclude that for a gold entity mention $\langle s, e, t \rangle$ where s and e represent the span boundaries and t represents entity type, the detection of accurate boundary indices (s and e) serves as a primary bottleneck of all current NER approaches discussed. Given a correct mention span, classifying it into an entity type t is relatively simpler, as observed in Section 3.3.

3.5.2 Experiment Details

To address the above mentioned issues, we provide additional inputs and infrastructure to the model to learn the underlying semantics better. In these experiments, we develop on top of the sequence labeling setup with `BIO` tagging scheme.

- **Special Symbol Features**: Before feeding to final classifier, concatenate BERT hidden layer output with a one-dimensional vector, set if the input token is a pure special symbol. This means we assign 1 for *hyphen*(`-`), *parenthesis*(`(` and `)`), and *comma*(`,`). Terms like `carbon`, `123`, `Ca(2+)`, and `AB-3` get assigned 0.

- **Word Type Features**: As an extension to special symbol features, here we associate each input token with a word type (shown in Table 3.10) which is converted into a one-hot vector concatenated with BERT output embeddings before feeding to the classifier layer.

Word Type	**Encoding**
`[CLS] token`	0
`[SEP] token`	1
`all lowercase`	2
`all caps`	3
`first letter caps, rest lowercase`	4
`all digits`	5
`all special symbols`	6
`alphabets + digits`	7
`all the rest`	8

Table 3.10: Word Type Encoding

- **Character and Pattern Features**: Chemical formulas and scientific terms generally follow a nomenclature convention or pattern. Similarly, out-of-vocabulary terms may have some intrinsic character-level information which is not well captured by the sub-words fed to the BERT model. [27] study this issue and propose a character-CNN instead of WordPiece tokenizer at the input stage to the BERT model. Motivated by the CNN-LSTM-CRF[26] model and this study, we do the following:

  Modeling characters. Each word is passed to BERT and simultaneously to five one-dimensional CNNs with kernel sizes of 1 to 5, each having 16 input and 16 output channels. Input character is indexed and mapped to a 16-dimensional embedding. Character-level outputs are max-pooled to get word representation. Outputs from multiple CNNs are concatenated and passed through a linear layer to get overall 768-dimensional output vector for each token.

  Modeling patterns. Each word is converted to a pattern (a regular expression or a denser space with smaller character set). For example, all uppercase letters are mapped to `U`, lowercase to `L`, and digits to `D`. These patterns are then fed to a Character-CNN (like the one described above) and then to a bidirectional LSTM to get contextual pattern token embeddings.

  Finally, these character and pattern embeddings are concatenated with BERT outputs and fed to final classifier layer.

- **Part-of-Speech and Dependency Parse Features**: Concatenate BERT embeddings with the one-hot part-of-speech and dependency parse features before feeding to final classifier layer. The part-of-speech and dependency parse tags are generated using `scispacy` for `BioNLP13CG` and `JNLPBA` datasets, `spacy` for `CoNLL 2003` and `OntoNotes 5.0` datasets.

- **Head Tokens**: BERT uses WordPiece tokenizer and may break a single input token into multiple sub-words. Instead of doing token classification on each of these sub-words and making sure everything is correct, it is simpler to take the BERT hidden layer output for the first (*head*) sub-word for each token. This technique is also used in the original BERT paper[4] for NER.
- **Highway Network**: Instead of directly concatenating character-based features with BERT, we create a highway network[28] similar to the one used in BiDAF[29] architecture and train a logic gate to control the inflow of information from BERT vectors and additional character-level semantics.

Model	**BioNLP13CG**	**CoNLL 2003**
`Vanilla BERT`	85.99	91.36
`Special Symbol`	86.63	91.67
`Word Type`	86.50	91.55
`Character/Patterns`	86.44	91.08
`Part-Of-Speech`	86.11	91.47
`Dependency Parse`	86.20	91.32
`Head Tokens`	86.17	91.49
`Special Symbol + Head Tokens`	86.36	91.68
`Highway Net`	85.77	91.53

Table 3.11: Results: Token Semantics (Test set Micro-F1 in %)

3.5.3 Observations

Our results are summarized in Table 3.11. We also experimented with combinations of the above described features together but omit the results from this report if not found to be significant. We make the following observations:

- Almost all of the proposed additional features are found to improve upon the `Vanilla BERT` model on both `BioNLP13CG` and `CoNLL 2003` datasets.
- Handling special symbols is a primary issue of `Vanilla BERT`. Explicitly handling it is found to be most helpful among all the other features across both datasets and on both models `Special Symbol` and `Special Symbol + Head Tokens`.

- Extending the special symbol features to word types gives mixed signals to the model. `Word Type` model hence gives the second best performance on both datasets.
- `Character/Patterns` modeling helps in the biomedical text setting since it helps understand semantics of chemical names. However, giving importance to intrinsic patterns gives conflicting signals for general English entities and hence gives lower performance on `CoNLL 2003` data.
- `Part-of-Speech` and `Dependency Parse` features give some additional insights to the model over `Vanilla BERT` and hence give comparable or even better performance.
- Considering only `Head Tokens` helps over `Vanilla BERT` on both datasets. `Special Symbol + Head Tokens` gives the best performance on `CoNLL 2003` data. On `BioNLP13CG`, the improvement is reduced compared to `Special Symbol`. We suspect this is because of the loss of intrinsic sub-word details which can be crucial for biomedical text since they have lots of out-of-vocabulary chemical/gene names.
- `Highway Net` rightly ignores confusing character-level information for `CoNLL 2003` data and hence gives a performance boost over `Character/Patterns`. For the `BioNLP13CG` data, this additional highway layer is unable to give improvements since it adds undesired complexity to the model.

3.6 TRAINING EFFECTIVENESS STUDY

In section 3.5, we looked at some limitations of the pretrained BERT model in capturing special symbols and rare word semantics. To counteract, we proposed feeding in some additional token semantics. In this section, we study how effectively the model is able to pick cues and learn from the supplied additional features. Our experiments are conducted in sequence labeling setting on `BioNLP13CG` corpus using `BIO` tagging scheme.

3.6.1 Feeding Answer as Input

To study the training effectiveness, we give the model the best ideal-case information, that is, for each token, we feed its gold label as a one-hot vector. This is concatenated with BERT outputs before feeding to final classifier. We study the following variants:

- `BERT(Freeze) + Answer`: We concatenate BERT outputs with answer vectors and freeze BERT model parameters during training. This effectively reduces the no. of trainable parameters to around $26,000$. We train this in two settings, low learning rate of 10^{-5} (recommended BioBERT learning rate[1]) and high learning rate of 0.005.

- `BERT + Answer`: This mimics the standard setting where the BERT model is fine-tuned with given additional information at a low learning rate of 10^{-5}.
- For comparison with the above variants, we also present the results for simple `BERT` (fine-tuned) and `BERT(Freeze)` models as well (without any answer inputs).

Model	**Learning Rate**	**BioNLP13CG**
`BERT`	10^{-5}	85.99
`BERT + Answer`	10^{-5}	86.35
`BERT(Freeze)`	0.005	75.42
`BERT(Freeze) + Answer`	0.005	100.00
`BERT(Freeze) + Answer`	10^{-5}	63.89

Table 3.12: Results: Training Effectiveness - Feed Answer as Input (Test set Micro-F1 in %)

From results summarized in Table 3.12, we make the following observations:

- From `BERT` and `BERT(Freeze)`, we conclude that fine-tuning the model definitely helps.
- From `BERT` and `BERT + Answer`, we observe that feeding the answer with input gives better performance. However at a low learning rate of 10^{-5}, the gain is very small.
- From `BERT(Freeze) + Answer` at learning rates 0.005 and 10^{-5}, we observe that at low learning rate the model is not able to effectively catch the provided cues. At high learning rate, we get perfect scores as expected.

3.6.2 What happens at high learning rate?

From the previous section, we see that model needs training at high learning rate for learning from additional input features. But with BERT fine-tuning, it is recommended[2] to use a low learning rate in the order of 10^{-5}. To study this, we pass gold mention spans as input to the model. Basically, for each token we add a one-dimensional vector which is set if the token is a part of some gold entity mention. This means effective the model just has to do span classification which is a relatively easy task as seen previously in Section 3.3. We experiment with the following variants:

[1]https://github.com/dmis-lab/biobert#named-entity-recognition-ner
[2]https://github.com/dmis-lab/biobert#named-entity-recognition-ner

- `BERT(Freeze) + Gold Span`: Concatenate last hidden layer output from BERT with gold-labeled span vectors. and freeze BERT parameters during training. This model is expected to perform better than `BERT(Freeze)`.
- `BERT + Gold Span`: Same as the above model but with BERT fine-tuning. We train this in two settings, with low learning rate of 10^{-5} and high learning rate of 0.005.
- For comparison, we also present the results for simple `BERT` (fine-tuned) and `BERT(Freeze)` models.

Model	**Learning Rate**	**BioNLP13CG**
`BERT(Freeze)`	0.005	75.42
`BERT(Freeze) + Gold Span`	0.005	79.18
`BERT`	10^{-5}	85.99
`BERT + Gold Span`	10^{-5}	85.78
`BERT + Gold Span`	0.005	0.0

Table 3.13: Results: Training Effectiveness - Feed Gold Span as Input (Test set Micro-F1 in %)

From Table 3.13, we make the following observations:

- From `BERT(Freeze)` and `BERT(Freeze) + Gold Span`, we observe that indeed giving gold span information helps the model.
- From `BERT` and `BERT + Gold Span` at learning rate 10^{-5}, we observe that with a low learning rate, the model is not able to focus on and effectively utilize the gold span information (similar to our previous findings with answer inputs).
- From `BERT + Gold Span` at learning rates 10^{-5} and 0.005, we observe that increasing the learning rate has a deteriorating effect on the pretrained BERT parameters and the rigorous push from a high learning rate pushes the model to an unsatisfactory local optima.

3.7 PRETRAINED MODEL VARIATION

In all the experiments done in this work, pretrained BERT model serves as our model backbone. However based on which dataset the pretrained model is trained on and what learning objective is used, there are several variants. We study the effect of this pretraining procedure on NER performance in sequence tagging setup through the following variants:

- `BERT-Base-Uncased`: Proposed by [4], this model is trained on English text from Wikipedia and BookCorpus[30] which totals around 16GB of uncompressed text. The model is uncased. It is trained on masked language modeling (MLM) and next sentence prediction objective. We use `bert-base-uncased` model provided by HuggingFace[3] for our `CoNLL 2003` and `OntoNotes 5.0` corpora.

- `RoBERTa-Base`: Proposed by [31], this model is trained on English text from 5 different datasets totalling around 160 GB of uncompressed text. The model is cased and trained on only the masked language modeling (MLM) objective. We use `roberta-base` model provided by HuggingFace[4] for our `CoNLL 2003` and `OntoNotes 5.0` corpora.

- `BioBERT-Base`: Proposed by [32], this model is trained on English biomedical literature including PubMed abstracts and PMC full text articles. The model is cased and trained on same MLM and next sentence prediction objectives proposed in standard BERT model. We use `BioBERT-Base v1.1` model provided on GitHub[5] and import it in HuggingFace as `dmis-lab/biobert-base-cased-v1.1`. We use this for `BioNLP13CG` and `JNLPBA` datasets, since models pretrained on biomedical and scientific texts are found to capture similar semantics more effectively than those trained on general English text.

- `SciBERT-Base-Uncased`: Proposed by [33], this model is trained on full texts of papers on Semantic Scholar[6]. The model is uncased and trained using the MLM and next sentence prediction objectives originally proposed by the BERT paper. However, this model uses `SciVocab`, a specially created WordPiece vocabulary for scientific texts. Just like BioBERT, we use this for `BioNLP13CG` and `JNLPBA` datasets.

Model	**BioNLP13CG**	**JNLPBA**
`BERT-Base-Uncased`	82.63	73.03
`BioBERT-Base`	85.99	74.35
`SciBERT-Base-Uncased`	86.01	74.68

Table 3.14: Results: Pretrained (Biomedical) Model Variation (Test set Micro-F1 in %)

In Tables 3.14 and 3.15, we present our results. We observe that:

[3]https://huggingface.co/bert-base-uncased
[4]https://huggingface.co/roberta-base
[5]https://github.com/dmis-lab/biobert#download
[6]https://www.semanticscholar.org/

Model	CoNLL 2003	OntoNotes 5.0
BERT-Base-Uncased	91.36	83.39
RoBERTa-Base	91.19	86.34

Table 3.15: Results: Pretrained (General) Model Variation (Test set Micro-F1 in %)

- On OntoNotes 5.0 data, RoBERTa-Base performs better than Bert-Base-Uncased. This gives us an insight that case of tokens plays an important role in general English news data. However, on CoNLL 2003, the performance is similar. On investigating further[7], we identify that original CoNLL 2003 data has some casing issues which people try to resolve by doing truecasing as a pre-processing step. It is because of this casing issue that RoBERTa-Base model is not able to show its advantages on CoNLL 2003 data.

- SciBERT-Base-Uncased performs marginally better than BioBERT-Base v1.1 on both biomedical datasets. As expected, both of these domain-specific pretrained models perform much better than BERT-Base-Uncased.

3.8 PARTITIONING DIVERSE ENTITIES

From the entity distribution in BioNLP13CG dataset (Table 2.5a), we see that entities like Cancer and Gene_or_gene_product are high-resource entity types. They may have high mention diversity and hence it may be difficult for a model to capture all their mention representations. Instead, it may be easier to break a heterogeneous entity class into relatively homogeneous sub-entity classes and work with them. At inference time, remap the sub-entities to the original entity and report F1-scores.

3.8.1 Experiment Details

We work with BioNLP13CG dataset using question answering setup on only 3 entities, Gene_or_gene_product, Cancer and Simple_chemical. All other entities are ignored. We partition Gene_or_gene_product (the largest entity class) into K sub-entities (K becomes a hyper-parameter). The procedure is described below:

- Collect all mentions of Gene_of_gene_product. Pass their corresponding sentences to pretrained BioBERT-Base-Uncased model. Calculate contextualized mention embedding as the concatenation of BERT outputs of first and last sub-words of the mention.

[7]https://github.com/google-research/bert/issues/223#issuecomment-649619302

- For multiple instances of same mention, take the mean of contextual mention embeddings across sentences.
- Reduce mention embeddings to 100-dimensional vectors using principal component analysis (PCA). Then take their tSNE[34] projections to convert each mention to a 2-dimensional vector representation.
- **Clustering**: Fix $K = 4$ (number of clusters). Randomly select K mention instances as cluster centers and apply K-Medoids clustering. Use euclidean distance among tSNE projections as the distance metric[8].
- After clustering, we relabel the corpus tagging each mention to its cluster. Each cluster is considered a sub-entity of `Gene_of_gene_product`. The sub-entities are named from `Gene_of_gene_product0` to `Gene_of_gene_product3`.
- Next we train a question answering NER model with this new labeled dataset. Note that in Section 3.2 we saw that the keyword used to represent an entity does not play a crucial role in extraction performance. So, our sub-entity naming convention should not be a problem.
- During inference time, the model mention with their sub-entity types. We post-process the labels and map each sub-entity output to `Gene_of_gene_product` and then calculate overall Micro-F1.

Model	**BioNLP13CG**
`BERT-QA`	87.97
`BERT-QA (With Partitioning)`	86.66

Table 3.16: Results: Partitioning Diverse Entities (Test set Micro-F1 in % on 3 high-resource entities)

3.8.2 Observations

From results in Table 3.16, we observe that entity partitioning technique gives slightly reduced performance compared to original setup. This is because our current clustering is on the basis of semantic similarity captured by BERT embeddings and not similarity among

[8]We evaluated cosine distance and KL-Divergence on tSNE and PCA vectors as distance metrics as well. However the presented setup is found to work the best.

word patterns. As an example, among `Simple_chemical`, we want formulas like *CaSO4* to be segregated from names like *calcium sulphate*. However, the BERT model clubs them into the same sub-entity since they are semantically same. Hence, our sub-entities still have high heterogeneity and not well separated, making it difficult for the model to train well. However, with better word-pattern oriented clusters, this technique has potential to give better results.

3.9 COMPARATIVE PRECISION/RECALL ANALYSIS

In this section, we take some major models described in the previous sections and compare and contrast them quantitatively by comparing their precision, recall and F1-score on multiple datasets.

- `BERT`: Represents sequence labeling NER setup with `BIO` tagging scheme.
- `Dice Loss`: Same as `BERT` model using dice loss instead of standard cross-entropy loss.
- `Special Symbol`: Same as `BERT` model with additional one-hot input feature to capture if the token is a special symbol like *hyphen*, *comma*, or *parenthesis*.
- `BERT-QA`: Represents question answering NER setup with `BIOE` tagging scheme and *What* as the question word.
- `BERT-QA(Where)`: Same as `BERT-QA` with *Where* as the question word.
- `Span-Based`: Uses the `BERT-QA` setup for span detection and QA-based sequence classification for span classification.

	BioNLP13CG			**CoNLL 2003**			**JNLPBA**		
	P	R	F1	P	R	F1	P	R	F1
`BERT`	86.17	85.82	85.99	91.24	91.48	91.36	70.97	78.07	74.35
`Dice Loss`	86.68	**86.03**	86.35	91.05	90.47	90.76	**72.05**	78.23	**75.01**
`Special Symbol`	87.62	85.66	86.63	91.75	91.60	91.67	70.42	78.47	74.23
`BERT-QA`	88.62	84.39	86.45	91.54	90.80	91.17	71.97	78.11	74.92
`BERT-QA(Where)`	**89.21**	84.58	**86.83**	**92.47**	91.19	**91.82**	71.45	78.11	74.64
`Span-Based`	86.33	85.47	85.89	91.46	**91.82**	91.64	71.14	**79.34**	**75.01**

Table 3.17: Results: Precision/Recall Comparison

From the results summarized in Table 3.17, we make the following observations:

- QA-based models give a much higher precision than sequence labeling or span-based methods. However, they compromise on the recall. This is especially true for BioNLP13CG data with 16 entity types and less prevalent in CoNLL 2003 and JNLPBA datasets with 4 and 5 entity types respectively. We suspect that this is because in QA setup, each sentence is fed K times (once with each entity type) during training/testing where K is the number of entity types. A sentence generally has mentions belonging to at most 3-4 entity types. So, the model receives positive samples corresponding to each of these and a negative sample for every other entity type. These large number of negative samples (especially in BioNLP13CG dataset) make the model very risk-averse. It outputs a mention only when it is highly confident (giving high precision) else considers it a negative sample and ignores the entity completely (giving low recall).

- Sequence labeling and span-based methods both have a low margin between their precision and recall values while QA setup gives a larger gap.

- Dice Loss is found to help improve recall in sequence labeling setup and Special Symbol helps improve precision (mention span boundary detection).

- In QA setup, asking *Where* gives a higher precision than asking *What*.

- Span-based technique is generally found to give the best recall values. This can be attributed to the entity-agnostic span detector which identifies all candidate entity spans irrespective of their type.

- The recall of span detector is inversely proportional to number of output entity types. With more entity types, the heterogeneity increases making span detector error-prone. This is the reason for low recall on BioNLP13CG corpus with 16 entity types compared to CoNLL 2003 and JNLPBA datasets with 4 and 5 types respectively.

- Due to its large advantage in precision, the BERT-QA(Where) model gives the best performance among the compared models.

3.10 ENTITY-WISE PERFORMANCE

Next, we deep dive into the BioNLP13CG dataset which has 16 entity types including several high and low-resource types. We compare the model performance at the entity type level for our 3 major NER approaches: sequence labeling, question answering and span-based pipeline. We compare our best performing model variants through entity-level and macro-averaged F1-scores. Let $\mathcal{T}$ be the set of all entity types and F1_t be the F1-score for individual entity type $t \in \mathcal{T}$. Then,

$$\text{Macro-F1} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \text{F1}_t$$

We present the comparison among the following models:

- `Dice Loss`: Sequence labeling NER approach over BERT model with `BIO` tagging scheme and dice loss instead of cross entropy.
- `Special Symbol`: Sequence labeling NER approach over BERT with `BIO` tagging scheme and additional one-hot input feature to capture if a token is a special symbol like *hyphen*, or *parenthesis*.
- `BERT-QA (Where)`: Question answering NER approach with `BIOE` tagging scheme and *Where* as the question word.
- `Span Based`: Pipelined approach which uses the QA setup with `BIOE` tagging scheme and *What* as question word for span detection and QA-based sequence classification for span classification.

Model	**Macro F1**
`Dice Loss`	70.51
`Special Symbol`	70.59
`BERT-QA (Where)`	72.00
`Span Based`	**72.43**

Table 3.18: Results: Test set Macro-F1 in %

From results in Table 3.18 and Figures 3.5 and 3.6, we make the following observations:

- `Span Based` pipelined approach reports the best overall macro-averaged F1. It is able to perform well even on low-resource entity types. This can be attributed to the span detector which is entity type agnostic and hence does not develop high-resource bias.
- From overall Macro-F1 scores we see that `BERT-QA (Where)` and `Span Based` both use question answering approach internally and perform better than sequence labeling methods.
- For high-resource entity types all variants perform comparably while for several low-resource entities, `Span Based` method takes the lead.

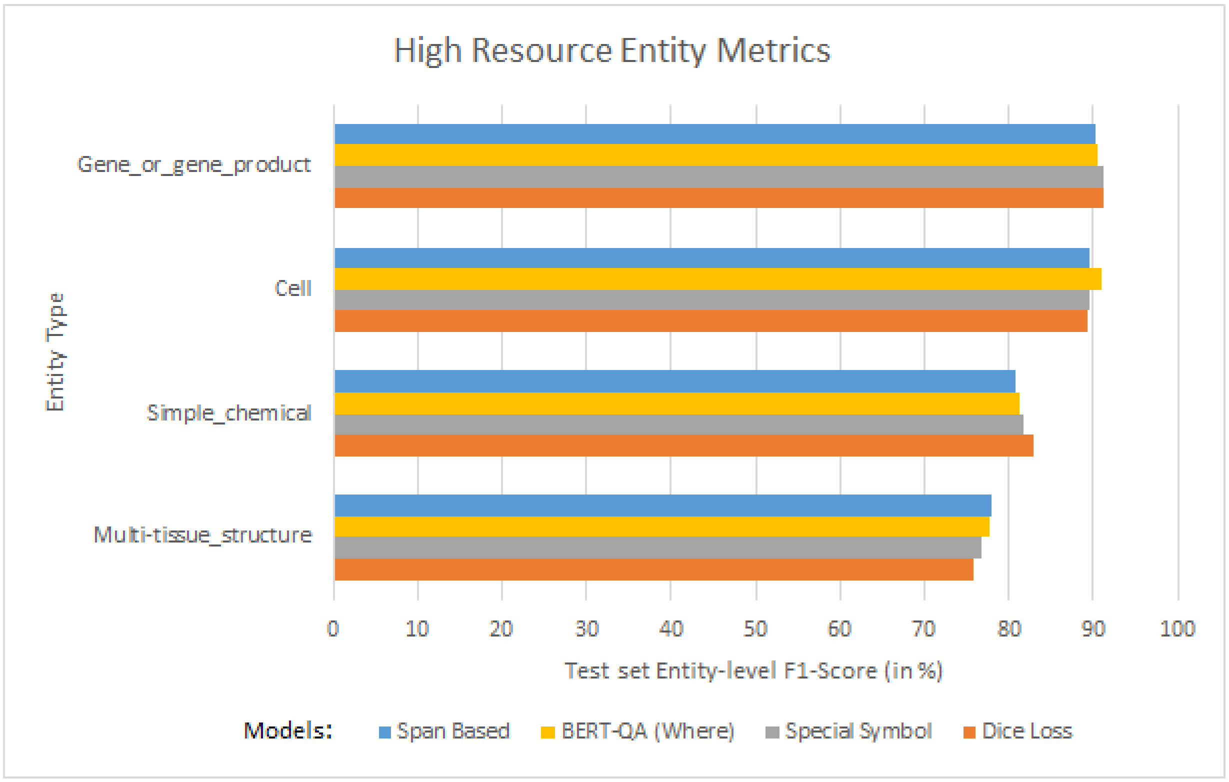

Figure 3.5: Test set Entity-level F1 scores for high resource entities in `BioNLP13CG` dataset

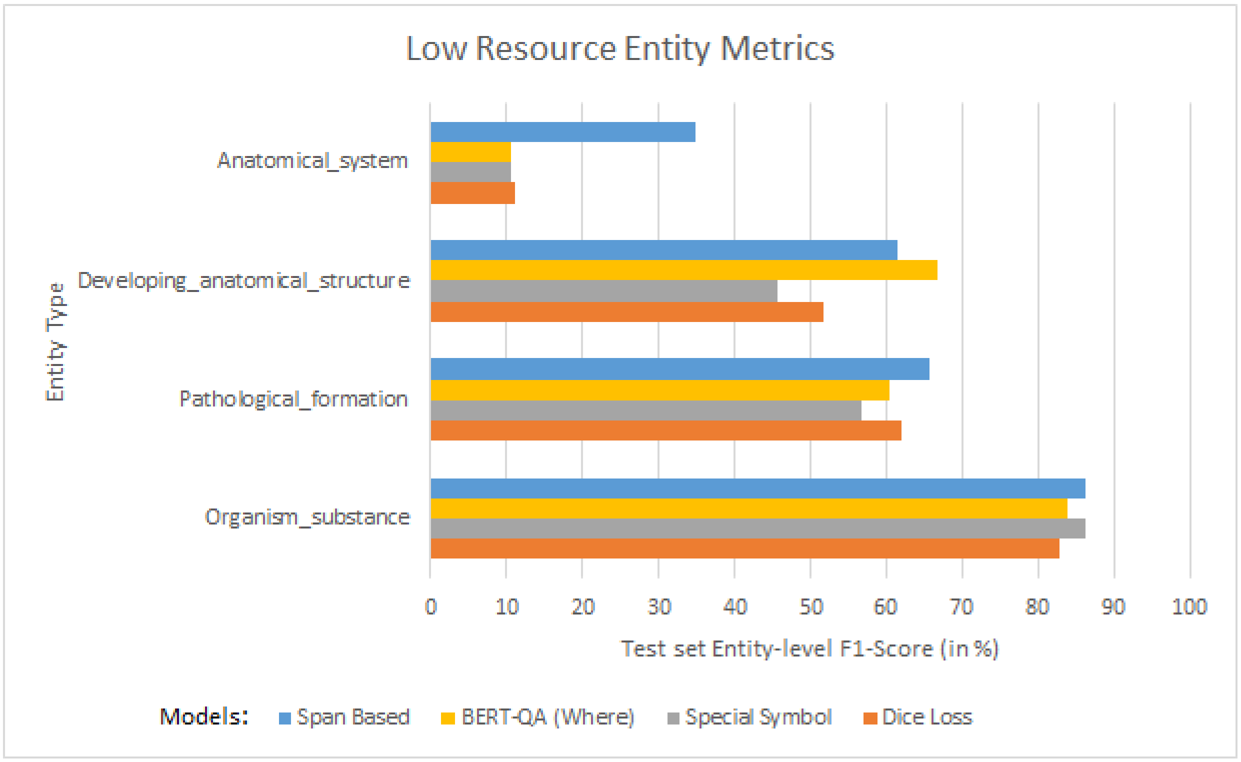

Figure 3.6: Test set Entity-level F1 scores for low resource entities in `BioNLP13CG` dataset

3.11 COMPARISON WITH OTHER WORKS

Based on our literature review in the NER domain, we find that different past works differ not only in their approach but also in the data they use and the output entities they focus on. These differences have an impact on the overall performance and make it difficult to have a one-to-one comparison. Hence, in this section we omit a tabular summary and instead give a detailed comparison presenting the results along with the intricacies of each work.

3.11.1 BioNLP13CG

[35] report 78.90 as test set micro-F1 in a multi-task learning setup and [36] report 77.60 using their SciSpacy system. BioBERT[32] authors does not state their performance on this dataset however [18] replicate the BioBERT model as a baseline and report 85.56. Our vanilla BioBERT model achieves 85.99. [18] report 82.39 when using the standard train/dev/test splits on their proposed system. When using a combined training corpus of 15 biomedical datasets they get 89.58, however this setting is not comparable to the other mentioned systems which use only the provided training data. Our `BERT-QA (Where)` model achieves 86.83 and `Span Based` model reports 85.89 which set the new state-of-the-art on this dataset.

3.11.2 CoNLL 2003

Doing NER on `CoNLL 2003` data has been a popular NLP task[9] for long. [37] set the current state-of-the-art result at 94.3 micro-F1 score on test set. They propose a new pre-training task learning entity representations and pretrain the RoBERTa model on a large corpus. Besides, they provide document context as input apart from each sample sentence both during training and inference. BERT[4] reports 92.4 using a case-preserving WordPiece model including maximal document context available. For each input token, they take the vector representation of the first sub-token and feed to a final classifier. The vector representation is created by concatenating the outputs of top four hidden layers of the BERT transformer. Additionally, as per some discussions on GitHub[10], researchers add additional CRF layer, bidirectional LSTMs along with truecasing and heuristic-based intelligent input sentence splitting as data pre-processing steps to achieve high results. Without these additional steps, the Vanilla BERT-Base model is found to achieve around 91.0 micro-F1 score.

[9] https://paperswithcode.com/sota/named-entity-recognition-ner-on-conll-2003
[10] https://github.com/google-research/bert/issues/223

Our experiments with vanilla BERT achieved 91.36. Using the same vanilla BERT model with our proposed span-based setup, we get 91.64. Our proposed approach is very generic, simple and efficient. It can be easily combined with popular pre-processing steps described above and custom pre-training done by [37] to further improve the state-of-the-art on this popular dataset.

3.11.3 JNLPBA

From literature review we find that there are two groups into which the past results can be segregated. All figures reported are micro-averaged F1 scores. The first group of works use the version of dataset provided by MTL-Bioinformatics[35] on GitHub[11]. For ease of comparison, we also fall into this category. The source paper[35] proposes a multi-task learning setup and reports 70.09. [38] also do multi-task modeling and combine training and development sets of multiple datasets to achieve 73.52. SciSpacy[36] reports 73.21. [18] report 73.63 for BioBERT model fine-tuned on JNLPBA. Our vanilla BioBERT setup achieves 74.35. [39] report 75.03 using an LSTM-based language model pretraining and fine-tuning procedure. To the best of our knowledge, this work serves as the state-of-the-art on this dataset. Our proposed span-based model reaches 75.01 which matches with the SOTA.

The second group of papers have parsed and pre-processed the JNLPBA data from scratch. This follows from [40] who identified that the MTL-Bioinformatics data version has sentence segmentation issues. [40] train on a combined data source consisting of multiple datasets and remove `Cell_type` entity class from consideration. They report 78.58 micro-averaged F1. BioBERT[32] achieves 77.59. SciBERT[33] reports 77.28 with a CRF used in the final classification step. They mark their results as a macro-F1 score however on cross-referencing with other papers, we suspect it to be a typo which should be micro-F1. Recently [41] report 81.29 which serves as a new state-of-the-art in this setting.

We conclude from the survey that our proposed span-based architecture gives equivalent results to the state-of-the-art model on JNLPBA dataset provided on GitHub by MTL-Bioinformatics[35].

[11]https://github.com/cambridgeltl/MTL-Bioinformatics-2016/

CHAPTER 4: RELATED WORK

NER has been a popular research topic in the field of natural language processing for long. The progress can be categorized into four classes namely, rule-based/dictionary-based techniques[42], unsupervised methods[43], feature-engineering approaches[44] and more recently deep learning-based approaches[45, 46]. Our focus here is more on the recent deep learning methods. [4, 47, 48] propose contextualized word/string representations to better model sentence semantics thus improving NER performance. Deep learning systems are however limited by the amount of labeled training data. Hence, there have been efforts to generate noisy labeled data as weak supervision signals for training. [49] propose AutoNER framework which uses distant supervision for generating noisy labels for training. [50] use regular expression patterns for artificial training data generation and train a LSTM model for entity extraction. [51] propose adversarial perturbations and use CNN-LSTM-CRF[26] architecture to train a robust model with limited gold data. [52, 53] train a task-aware language model from unlabeled data which guides NER. As a side note, sometimes gold labels in datasets may have some errors. [54] use a bootstrapping framework to correct such imperfect annotations.

In scientific and biomedical domain, NER has its own set of challenges. Entity mentions are long and may have alpha-numeric symbols and chemical formulas which may be hard for a language model to make sense of. Many entity types also low-resource with a shortage of labelled training data. [38] use a multi-task learning framework and combine labelled data from multiple corpora mapping entity tags across corpora to coherent classes. [55] use noisy distant supervision from domain-specific dictionaries. [56] form entity-type meta patterns for entity extraction. [57] make use of a setup similar to AutoNER[49] tailor-made for biomedical NER. [58, 59] apply weak or distant supervision for NER on COVID-19 literature.

Several named entities have numeric nature, for example, phone numbers and SSNs look very much alike and require understanding of number patterns to differentiate the two. NER systems dealing with such entities need some explicit numeral semantic handling. [60] propose a Bi-GRU framework for understanding numbers. [61] shows that standard contextual word embeddings like ELMo[47], BERT[4] have some good sense of numbers and character-level embeddings are more effective than word-level embeddings in capturing numeral semantics.

CHAPTER 5: CONCLUSIONS AND FUTURE WORK

In this work, we looked at the NER problem from three different perspectives, namely, sequence labeling, question answering and span-based approach. We compared and contrasted them with each other and studied their advantages and limitations. Taking inspiration from the QA setup, we proposed the span detection and classification pipeline which uses a reverse question formulation. Additionally, we also proposed to convert from a sparse character space to a dense pattern space through which we can learn meaningful intrinsic character patterns in alphanumeric and pattern-oriented entities. We demonstrated the effectiveness of our proposed domain-agnostic techniques on multiple datasets in general English and biomedical domains. We also presented a study depicting that trivial concatenation of external semantic vectors with BERT outputs may not train the model effectively at lower learning rates.

Our span-based setup opens up prospects for more intuitive and creative ways of approaching the NER problem. However, the pipelined nature of the approach currently serves as a bottleneck. It may be worthwhile to think of some ensemble-based approach where we train individual BERT models on some sub-problems and each of those models contributed its part to solve the overall NER problem in a majority-voting setup.

Our study on training effectiveness reveals that feeding additional external semantics while fine-tuning the BERT model is non-trivial. This again motivates future research on designing feature fusion techniques which are effective with a BERT (transformer-like) architecture.

From the qualitative analysis of the various approaches, we observe that boundary detection serves as a primary issue in NER. To alleviate this problem, we explicitly model word types and special symbols. However, there is still a wide margin to cover. We encourage the research community to design architectures or new training objectives tailor-made to handle mention boundaries effectively.

CHAPTER 6: REFERENCES

[1] B. Writer, "Lithium-ion batteries," *A Machine-Generated Summary of Current Research. Cham: Springer International Publishing*, 2019.

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

[3] 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.

[4] 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.

[5] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, "Genia corpus—a semantically annotated corpus for bio-textmining," *Bioinformatics*, vol. 19, no. suppl_1, pp. i180–i182, 2003.

[6] A. Mitchell, S. Strassel, S. Huang, and R. Zakhary, "Ace 2004 multilingual training corpus," *Linguistic Data Consortium, Philadelphia*, vol. 1, pp. 1–1, 2005.

[7] C. Walker, S. Strassel, J. Medero, and K. Maeda, "Ace 2005 multilingual training corpus," *Linguistic Data Consortium, Philadelphia*, vol. 57, p. 45, 2006.

[8] E. F. Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," *arXiv preprint cs/0306050*, 2003.

[9] R. Weischedel, M. Palmer, M. Marcus, E. Hovy, S. Pradhan, L. Ramshaw, N. Xue, A. Taylor, J. Kaufman, M. Franchini et al., "Ontonotes release 5.0 ldc2013t19," *Linguistic Data Consortium, Philadelphia, PA*, vol. 23, 2013.

[10] J.-D. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, and N. Collier, "Introduction to the bio-entity recognition task at jnlpba," in *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications.* Citeseer, 2004, pp. 70–75.

[11] S. Pyysalo, T. Ohta, R. Rak, A. Rowley, H.-W. Chun, S.-J. Jung, S.-P. Choi, J. Tsujii, and S. Ananiadou, "Overview of the cancer genetics and pathway curation tasks of bionlp shared task 2013," *BMC bioinformatics*, vol. 16, no. 10, pp. 1–19, 2015.

[12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[13] X. Li, F. Yin, Z. Sun, X. Li, A. Yuan, D. Chai, M. Zhou, and J. Li, "Entity-relation extraction as multi-turn question answering," *arXiv preprint arXiv:1905.05529*, 2019.

[14] O. Levy, M. Seo, E. Choi, and L. Zettlemoyer, “Zero-shot relation extraction via reading comprehension,” *arXiv preprint arXiv:1706.04115*, 2017.

[15] B. McCann, N. S. Keskar, C. Xiong, and R. Socher, “The natural language decathlon: Multitask learning as question answering,” *arXiv preprint arXiv:1806.08730*, 2018.

[16] X. Li, J. Feng, Y. Meng, Q. Han, F. Wu, and J. Li, “A unified mrc framework for named entity recognition,” *arXiv preprint arXiv:1910.11476*, 2019.

[17] X. Li, X. Sun, Y. Meng, J. Liang, F. Wu, and J. Li, “Dice loss for data-imbalanced nlp tasks,” *arXiv preprint arXiv:1911.02855*, 2019.

[18] P. Banerjee, K. K. Pal, M. Devarakonda, and C. Baral, “Knowledge guided named entity recognition for biomedical text,” *arXiv preprint arXiv:1911.03869*, 2019.

[19] S. Valverde, M. Cabezas, E. Roura, S. González-Villà, D. Pareto, J. C. Vilanova, L. Ramió-Torrentà, À. Rovira, A. Oliver, and X. Lladó, “Improving automated multiple sclerosis lesion segmentation with a cascaded 3d convolutional neural network approach,” *NeuroImage*, vol. 155, pp. 159–168, 2017.

[20] T. A. Sorensen, “A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons,” *Biol. Skar.*, vol. 5, pp. 1–34, 1948.

[21] L. R. Dice, “Measures of the amount of ecologic association between species,” *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[22] A. Tversky, “Features of similarity.” *Psychological review*, vol. 84, no. 4, p. 327, 1977.

[23] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, “Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations,” in *Deep learning in medical image analysis and multimodal learning for clinical decision support.* Springer, 2017, pp. 240–248.

[24] C. Shen, H. R. Roth, H. Oda, M. Oda, Y. Hayashi, K. Misawa, and K. Mori, “On the influence of dice loss function in multi-class organ segmentation of abdominal ct using 3d fully convolutional networks,” *arXiv preprint arXiv:1801.05912*, 2018.

[25] O. Kodym, M. Španěl, and A. Herout, “Segmentation of head and neck organs at risk using cnn with batch dice loss,” in *German conference on pattern recognition.* Springer, 2018, pp. 105–114.

[26] X. Ma and E. Hovy, “End-to-end sequence labeling via bi-directional lstm-cnns-crf,” *arXiv preprint arXiv:1603.01354*, 2016.

[27] H. E. Boukkouri, O. Ferret, T. Lavergne, H. Noji, P. Zweigenbaum, and J. Tsujii, “Characterbert: Reconciling elmo and bert for word-level open-vocabulary representations from characters,” *arXiv preprint arXiv:2010.10392*, 2020.

[28] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387*, 2015.

[29] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," *arXiv preprint arXiv:1611.01603*, 2016.

[30] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *arXiv preprint arXiv:1506.06724*, 2015.

[31] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[32] 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.

[33] I. Beltagy, K. Lo, and A. Cohan, "Scibert: A pretrained language model for scientific text," *arXiv preprint arXiv:1903.10676*, 2019.

[34] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[35] G. Crichton, S. Pyysalo, B. Chiu, and A. Korhonen, "A neural network multi-task learning approach to biomedical named entity recognition," *BMC bioinformatics*, vol. 18, no. 1, pp. 1–14, 2017.

[36] M. Neumann, D. King, I. Beltagy, and W. Ammar, "Scispacy: Fast and robust models for biomedical natural language processing," *arXiv preprint arXiv:1902.07669*, 2019.

[37] I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto, "Luke: deep contextualized entity representations with entity-aware self-attention," *arXiv preprint arXiv:2010.01057*, 2020.

[38] 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," *Bioinformatics*, vol. 35, no. 10, pp. 1745–1752, 2019.

[39] D. S. Sachan, P. Xie, M. Sachan, and E. P. Xing, "Effective use of bidirectional language modeling for transfer learning in biomedical named entity recognition," in *Machine learning for healthcare conference.* PMLR, 2018, pp. 383–402.

[40] W. Yoon, C. H. So, J. Lee, and J. Kang, "Collabonet: collaboration of deep neural networks for biomedical named entity recognition," *BMC bioinformatics*, vol. 20, no. 10, pp. 55–65, 2019.

[41] V. Kocaman and D. Talby, "Spark nlp: Natural language understanding at scale," *Software Impacts*, vol. 8, p. 100058, 2021.

[42] A. P. Quimbaya, A. S. Múnera, R. A. G. Rivera, J. C. D. Rodríguez, O. M. M. Velandia, A. A. G. Peña, and C. Labbé, “Named entity recognition over electronic health records through a combined dictionary-based approach,” *Procedia Computer Science*, vol. 100, pp. 55–61, 2016.

[43] S. Zhang and N. Elhadad, “Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts,” *Journal of biomedical informatics*, vol. 46, no. 6, pp. 1088–1098, 2013.

[44] P. McNamee and J. Mayfield, “Entity extraction without language-specific resources,” in *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[45] A. Torfi, R. A. Shirvani, Y. Keneshloo, N. Tavvaf, and E. A. Fox, “Natural language processing advancements by deep learning: A survey,” *arXiv preprint arXiv:2003.01200*, 2020.

[46] J. Li, A. Sun, J. Han, and C. Li, “A survey on deep learning for named entity recognition,” *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[47] 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.

[48] A. Akbik, D. Blythe, and R. Vollgraf, “Contextual string embeddings for sequence labeling,” in *Proceedings of the 27th international conference on computational linguistics*, 2018, pp. 1638–1649.

[49] J. Shang, L. Liu, X. Ren, X. Gu, T. Ren, and J. Han, “Learning named entity tagger using domain-specific dictionary,” *arXiv preprint arXiv:1809.03599*, 2018.

[50] J. Arora, S. Agrawal, P. Goyal, and S. Pathak, “Extracting entities of interest from comparative product reviews,” in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1975–1978.

[51] J. T. Zhou, H. Zhang, D. Jin, H. Zhu, M. Fang, R. S. M. Goh, and K. Kwok, “Dual adversarial neural transfer for low-resource named entity recognition,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3461–3471.

[52] L. Liu, J. Shang, X. Ren, F. Xu, H. Gui, J. Peng, and J. Han, “Empower sequence labeling with task-aware neural language model,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[53] L. Liu, X. Ren, J. Shang, J. Peng, and J. Han, “Efficient contextualized representation: Language model pruning for sequence labeling,” *arXiv preprint arXiv:1804.07827*, 2018.

[54] Z. Wang, J. Shang, L. Liu, L. Lu, J. Liu, and J. Han, “Crossweigh: Training named entity tagger from imperfect annotations,” *arXiv preprint arXiv:1909.01441*, 2019.

[55] X. Wang, Y. Guan, Y. Zhang, Q. Li, and J. Han, "Pattern-enhanced named entity recognition with distant supervision," in *2020 IEEE International Conference on Big Data (Big Data).* IEEE, 2020, pp. 818–827.

[56] X. Wang, Y. Zhang, Q. Li, C. H. Wu, and J. Han, "Penner: Pattern-enhanced nested named entity recognition in biomedical literature," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2018, pp. 540–547.

[57] X. Wang, Y. Zhang, Q. Li, X. Ren, J. Shang, and J. Han, "Distantly supervised biomedical named entity recognition with dictionary expansion," in *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2019, pp. 496–503.

[58] X. Wang, X. Song, B. Li, K. Zhou, Q. Li, and J. Han, "Fine-grained named entity recognition with distant supervision in covid-19 literature," in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2020, pp. 491–494.

[59] X. Wang, X. Song, B. Li, Y. Guan, and J. Han, "Comprehensive named entity recognition on cord-19 with distant or weak supervision," *arXiv preprint arXiv:2003.12218*, 2020.

[60] C.-C. Chen, H.-H. Huang, H. Takamura, and H.-H. Chen, "Numeracy-600k: learning numeracy for detecting exaggerated information in market comments," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 6307–6313.

[61] E. Wallace, Y. Wang, S. Li, S. Singh, and M. Gardner, "Do nlp models know numbers? probing numeracy in embeddings," *arXiv preprint arXiv:1909.07940*, 2019.