

---

# Comparative Analysis of Natural Language Methods for Part-of-Speech Tagging

---

**Abdulrahman A. Alabdulkareem**  
arkareem@mit.edu

**Edvard Ronglan**  
eronglan@mit.edu

**Will Sussman**  
wsussman@mit.edu

## Abstract

This project is a comparative analysis between large language models (LLMs) and more traditional methods for part of speech tagging of natural language using a tagged subset of the widely used Penn Treebank dataset and a domain-specific dataset from the BioNLP Shared Task community challenge. LLMs, such as GPT-3 and BERT-style models, have shown remarkable performance in various NLP tasks, including part of speech tagging, and may offer advantages in terms of their ability to capture contextual information and handle long-range dependencies compared to more traditional methods. On the other hand, more traditional models have been widely used for part of speech tagging, exemplified by parsers such as the Stanford Parser, and the part of speech taggers available in the Natural Language Toolkit (NLTK) library in Python. We evaluate the accuracy of several models, and will provide insights into the strengths and weaknesses of each approach for part of speech tagging. This information will inform the choice of modeling technique for similar applications and contribute to the understanding of the trade-offs between LLMs and PCFGs in part of speech tagging of natural language text.

<https://github.com/Ar-Kareem/Penn-Tree-Bank-Project>

# 1 Introduction

Assigning a grammatical category, such as noun, verb, or adjective, to each word in a sentence is the fundamental task of part-of-speech (POS) tagging in NLP. It is an essential step in many NLP applications, such as machine translation, text-to-speech synthesis, and sentiment analysis. The most common approaches for solving this problem is rule-based, probabilistic and artificial intelligence methods (1; 2), with rule-based methods the most traditional approach. Rule-based methods utilize corpora of rules based on linguistic properties of the language (3), which often have been "hand-crafted" by linguistic experts. Since this is a very tedious exercise and prone to errors, simple rule-based methods have not been widely used by themselves in later years.

Probabilistic methods include approaches such as Probabilistic Context-Free Grammars (PCFGs) and Markov models (4; 5; 6). PCFGs define a set of context-free grammar rules and assign probabilities to these rules based on their likelihood of occurring in a given language. Markov models also use probability to determine the POS tagging, but the probability of a tag sequence is based on the probabilities of the previous tags in the sequence. A special case of Markov models for POS tagging is N-gram models, which takes the previous N tags into consideration when determining the probability of a tag sequence.

Even though the probabilistic methods give good results, researchers nowadays tend to favor deep learning-based POS tagging approaches over traditional machine learning methods due to their ability to efficiently learn from large, unlabeled text corpora. This is highlighted by the fact that almost 68% of the proposed POS tagging solutions were deep learning (DL) based methods, with LSTM, RNN, and BiLSTM being the three topmost frequently used DL algorithms (1). In recent years, transformer-based models have gained significant attention in the field of Natural Language Processing (NLP). These models have proven to be highly effective in various NLP tasks, including machine translation, text generation, and sentiment analysis. The transformer-based architecture, introduced by the "Attention is All You Need" paper (7), has revolutionized the way NLP models are designed by utilizing self-attention mechanisms. Transformer-based models such as BERT (8) and GPT-3 (9) have set new state-of-the-art benchmarks for various NLP tasks. Although these models have not been widely used for part-of-speech tagging, their exceptional performance on other NLP tasks makes them a promising approach for this task as well.

Based on the lack of work on using the most recent transformer-based models for POS tagging, we use in this project a large language model (LLM), more specifically a model based on sBERT, to do POS tagging on a subset of the Penn Treebank dataset and a domain-specific dataset. However, considering the computational time requirements and complexity of such a model, we also compare its performance with simpler probabilistic models, including the Stanford Parser, N-gram models and a maximum entropy model. This project compares the accuracy, measured in the number of correctly labeled words, to see if the increase in performance of deep learning methods is really worth the extra complexity. By informing the choice of modeling technique for similar applications, we hope this information enhances the understanding of the trade-offs between LLMs and more traditional methods in part of speech tagging of natural language text.

Additionally we compare the different models on a text dataset available online as part of the BioNLP Shared Task, which is a series of evaluations and workshops aimed towards a domain-specific task in biology. The reason we focus on this dataset alongside the Penn Treebank dataset is due to the domain-specific nature of the data that is substantially different from the Penn Treebank dataset which only includes general non-domain specific language. We hope this analysis provides better guidance on the capabilities of these different models to transfer to different domains.

## 2 Datasets

To evaluate the performance of the models on part-of-speech tagging, we used two different datasets. The first dataset is a tagged subset of the well-known Penn Treebank dataset, provided by NLTK and described in section 2.1. This dataset is widely used for comparing POS taggers and is featured in

online leaderboards (10). Additionally, we wanted to assess our models on a domain-specific dataset. Hence, we selected the dataset generated by the BioNLP Shared Task, as outlined in section 2.2. This dataset contains text from the biology domain, including numerous novel and compound words unfamiliar to the models. By using two datasets, we could train our models on one and test them on the other, enabling evaluation of their out-of-distribution prediction performance.

## 2.1 Penn Treebank subset

We utilized a tagged fragment of the Penn Treebank dataset, provided by NLTK, which represents approximately 5% of the entire dataset. This fragment consists of 3,914 tagged sentences. To create training and testing sets, we performed an 80/20 split, resulting in 3,131 sentences for training and 783 sentences for testing.

In Figure 1, we visually presented the most frequent words in the dataset. It is worth noting that along with expected tokens, we observed some unexpected tokens, e.g. tokens including "\*". We anticipate that these unexpected tokens may adversely affect our Machine Learning Approach since they deviate from what these large models typically encounter. Additionally, Figure 3 displays the word counts for the entire dataset.

The test set includes four POS tags that are not generated by the Stanford Parser: *-NONE-*, *-LRB-*, and *-RRB-*. As the Stanford Parser cannot match these tags, we evaluated all models on the test set both with the original tags (unmodified) and without these tags (modified).

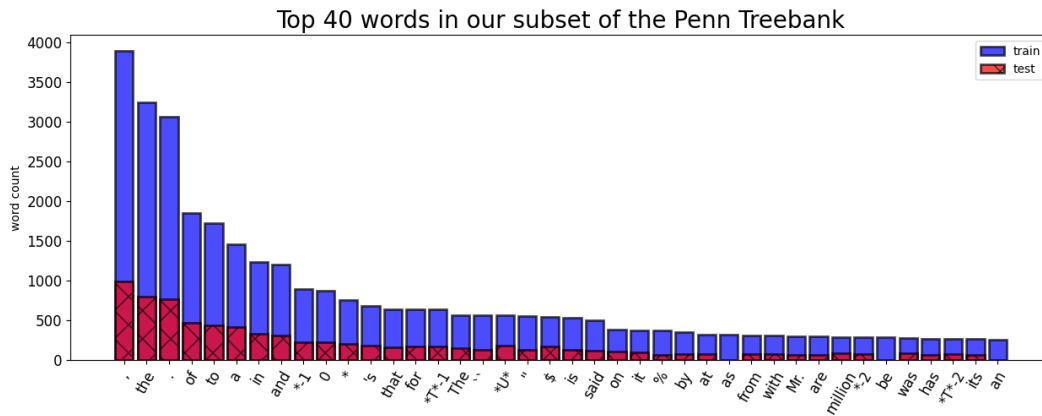


Figure 1: The counts of the 40 most common words in our subset of the Penn Treebank. Some are expected stop words like the words "the", "of", and "and", while others are unexpected symbols like "\*-1" or "\*U\*"

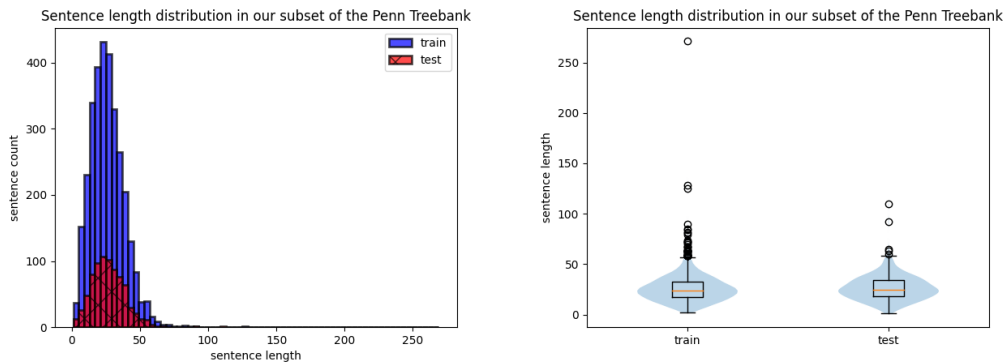


Figure 2: The distribution of sentence lengths in both the train and the test split on our subset of the Penn Treebank.

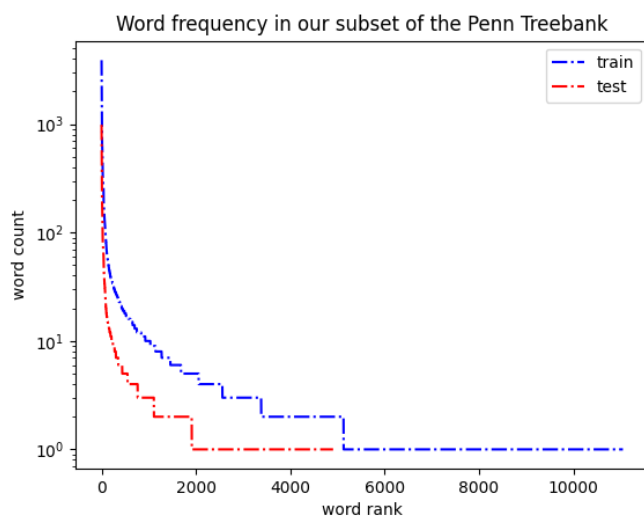


Figure 3: The counts of the words in the entire dataset (y-axis in logscale) on our subset of the Penn Treebank

## 2.2 BioNLP Shared Task dataset

The BioNLP Shared Task was a community challenge in the field of biomedical natural language processing (NLP), that aimed to advance the development of NLP techniques for processing biomedical literature. The dataset provided includes a variety of biomedical texts, such as scientific articles, abstracts, and full papers, which was complimented with POS tags. Figures 4, 5(a), 5(b) and 6 show the word and sentence length distribution of the dataset.

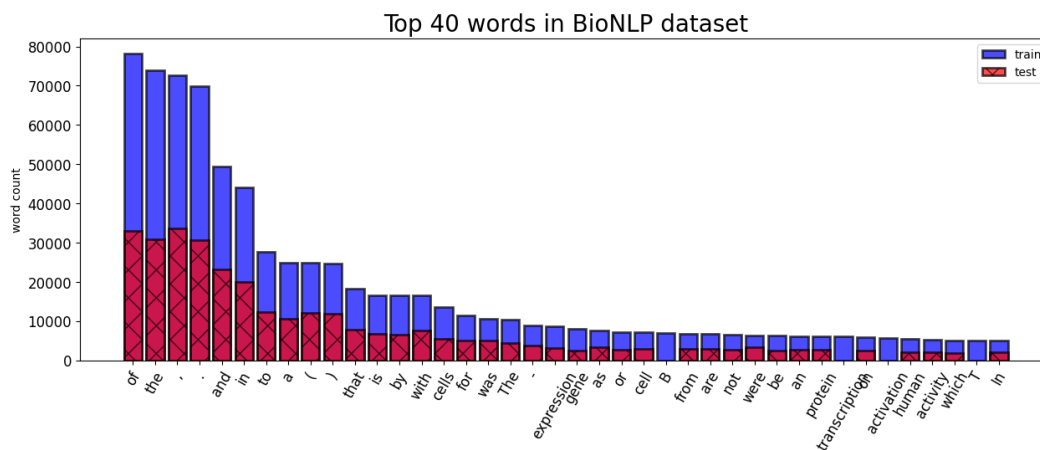


Figure 4: The counts of the 40 most common words in the BioNLP dataset. Some are expected stop words like the words "the", "of", and "and", while others are domain specific (in biology) words like "gene", "activation" or "protein"

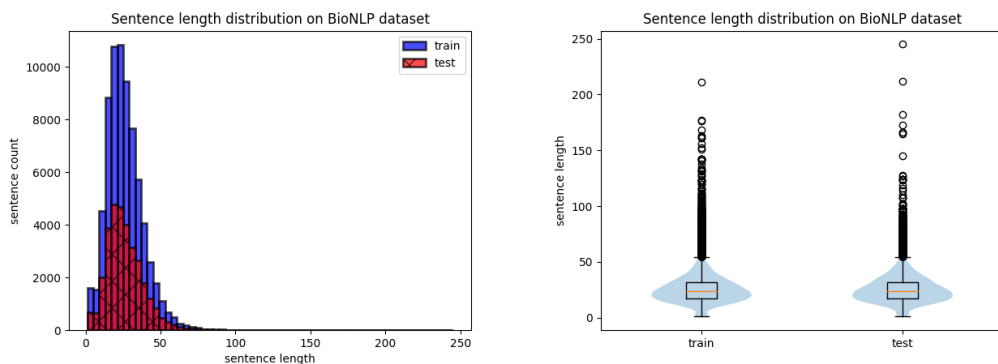


Figure 5: The distribution of sentence lengths in both the train and the test split in the BioNLP dataset.

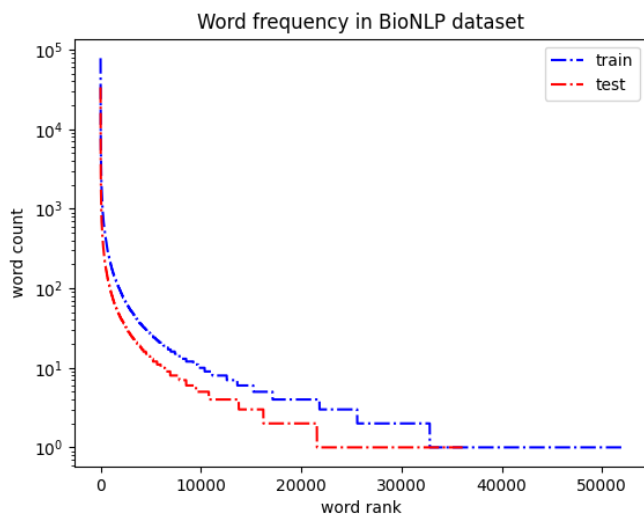


Figure 6: The counts of the words in the entire dataset (y-axis in logscale) in the BioNLP dataset

We note the diverse words in the BioNLP dataset as can be seen in 4 where even the most common words in the dataset contain words such as "Gene", "Transcription", and "activation" which are relatively rare English words when compared with average non-domain specific text such as 1.

We also note the diverse amounts of words provided in the BioNLP dataset as shown in Figure 6 when compared with Figure 3.

Additionally, the BioNLP dataset contains a slightly wider range of sentence lengths (heavier tails) as seen in Figures 5(a) and 5(b) when compared with Figures 2(a) and 2(b).

### 3 Methods

This section presents the methods we compare for part-of-speech tagging: the large language model (section 3.1), N-gram models (section 3.2), the maximum entropy model (section 3.3), and the Stanford Parser (section 3.4).

### 3.1 Machine Learning Approach

We use a frozen BERT-style model that was trained not trained on POS tagging and only trained on question answering (SBERT<sup>1</sup>). We use the model to encode each word in the sentences as a 768-dimensional vector.

Each sentence  $s$  in the dataset is composed of a list of words  $s_i = \{w_1, w_2, \dots, w_k\}$ . We must pass the sentences to a tokenizer  $\mathcal{T}(\cdot)$  that converts every word into one or more tokens based on whether that word is in the tokenizer’s dictionary or not. The tokenization step is required since the BERT-style model only understands token ids that were outputted by the tokenizer. Thus

$$\mathcal{T}(s_i) = \mathcal{T}(\{w_1, w_2, \dots, w_k\}) = \{t_1, t_2, \dots, t_{k'}\} \quad (1)$$

Each token  $t_j$  is an integer that the BERT-style model can consume, out of vocabulary words will be split into multiple tokens that represent parts of the word constructed from in-vocabulary tokens. The significance of this step will be important when getting the output of the BERT-style model. Now we pass the sentence to the BERT-style model  $\mathcal{B}$  to get the token embeddings

$$\mathcal{B}(\mathcal{T}(s_i)) = \mathcal{B}(\{t_1, t_2, \dots, t_{k'}\}) = \{e_1^{(t)}, e_2^{(t)}, \dots, e_{k'}^{(t)}\} \quad (2)$$

Where  $e_j^{(t)} \in \mathbb{R}^{768}$  is the embedding vector of token  $t_j$  considering the entire sentence. However, for our task we need embeddings for words  $w_i, \forall i \in \{1, \dots, k\}$  and not the embeddings we have which are for tokens  $t_j, \forall j \in \{1, \dots, k'\}$ . Retrieving an embedding for word  $w_i$  needs a little work as we need to inverse the transformation done by the tokenizer to figure out which set of tokens  $\{t_j, \dots, t_{j+\Delta j}\}$  represent the tokens for word  $w_i$ . For in-vocabulary words,  $\Delta j = 0$  while  $j$  is the next token in the sentence not related to any of the previous words  $\{w_1, \dots, w_{i-1}\}$ , we develop a simple script that reverses this mapping from words to tokens.

Once we obtain the values for  $j$  and  $\Delta j$  for all words  $w_i$  in the sentence, we construct the word embedding by taking the mean of all those embedding

$$e_i^{(w)} = \frac{1}{\Delta j} \sum_{k=0}^{\Delta j} e_{j+k}^{(t)} \quad (3)$$

Finally, we obtain the word embeddings for a single sentence from our BERT-style model. After encoding both the training and testing sentences we train a single hidden-layer Multi-Layer-Perceptron (MLP) on the embeddings of each word in the training set to predict the POS.

#### 3.1.1 BERT Pretrained on POS

For our second BERT-style model we use a frozen BERT-style model that was specifically trained on POS tagging (POS-BERT<sup>2</sup>) as provided by (11) to encode each word in the sentences as a 768-dimensional vector. We do note that this model has been trained on the Penn Treebank so it would be unfair to compare it to our other models that have not seen the Penn Treebank dataset.

### 3.2 N-gram models

We use the uni- and tri-gram models from the NLTK library and train/evaluate them on the same data as described in section 2. This includes both the dataset where all POS tags are included and the dataset where the POS tags the Stanford Parser can’t generate are excluded (“-NONE-”, “#”, “-LRB-”, and “-RRB-”). We also test them on the BioNLP Shared Task dataset, described in section 2.2, to evaluate how it performs on datasets of different distributions.

Uni-gram and tri-gram part-of-speech (POS) taggers are statistical models used for assigning POS tags to words in natural language processing. A uni-gram POS tagger operates by examining each

<sup>1</sup>huggingface model: sentence-transformers/multi-qa-mpnet-base-dot-v1

<sup>2</sup><https://huggingface.co/QCRI/bert-base-multilingual-cased-pos-english>

word in isolation and assigning a tag based solely on the word itself. It relies on the assumption that the probability of a word’s POS tag is independent of the surrounding context. In contrast, a tri-gram POS tagger takes into account the two previous words along with the current word to determine the most likely POS tag. By considering trigrams, which are sequences of three consecutive words, the tri-gram POS tagger captures the contextual information and dependencies between words. During training, the models learn the statistical patterns and co-occurrence frequencies of words and POS tags. When tagging new sentences, the uni-gram and tri-gram taggers calculate the probabilities of different POS tags given the current word or the current trigram, respectively, and assign the tag with the highest probability. While uni-gram taggers are simpler and faster, tri-gram taggers can capture more context and often yield higher accuracy in POS tagging tasks.

### 3.3 Maximum entropy model

The maximum entropy modeling framework, also known as the maximum entropy principle or MaxEnt, is a probabilistic modeling approach used in natural language processing and other fields. The principle behind maximum entropy is to choose the probability distribution that maximizes the entropy while being consistent with the available information or constraints. In the context of part-of-speech tagging, the maximum entropy modeling framework considers the task as a classification problem where each word is assigned a specific part-of-speech tag. The goal is to find the most probable distribution of tags given the observed features or attributes of the words, which is done by taking into account all the features and constraints provided by the training data, and it selects the probability distribution with the highest entropy that satisfies these constraints. The entropy represents the level of uncertainty or randomness in the distribution. By maximizing the entropy, the model avoids making unnecessary assumptions and ensures a more flexible and unbiased representation of the data.

### 3.4 The Stanford Parser

We used the Stanford Parser Version 4.2.0, released November 17, 2020 (12). It is designed to analyze the grammatical structure of sentences and generate syntactic parse trees. The input text is divided into individual tokens, such as words and punctuation marks. This is important because a sentence like “Hello, world!” would naively be split into two pieces, “Hello,” and “world!”, neither of which is in the vocabulary. We must split it into four pieces, “Hello”, “,”, “world”, and “!”. Then the parser uses a set of production rules defined in a context-free grammar (CFG) to generate parse trees. The leaves of these trees are (terminal, part-of-speech) pairs. We extract the parts of speech and compare against the ground-truth.

For testing the model, we used both datasets outlined in section 2. However, since the Stanford parser does not need training, we only tested it on the test set for both datasets to compare performance. Fortunately these sentences were pre-tokenized, so our results reflect the accuracy of parsing and *not* the accuracy of tokenizing. In addition to applying introducing a *per-token* accuracy, we also analyzed the *per-sentence*, where a sentence was considered correctly tagged if and only if all of its predicted POS tags were correct.

## 4 Results

In this section the results of evaluating the methods are included. The results of comparing the POS tagging performance of the different models are separated into two sections: section 4.1 for the Penn treebank dataset and section 4.2 for the BioNLP Shared Task dataset. Finally, in section 4.3 we evaluate our definition of sentence correctness for the Stanford Parser and show why we did not apply it to other methods.

### 4.1 Penn Treebank subset

The results obtained from running the models on the modified and unmodified Penn Treebank subset, as described in section 2.1, are presented in table 1. The reason for the accuracy not being given for the maximum entropy model and the NN using POS-BERT embeddings is that they were pretrained on the Penn treebank dataset and they would therefore have an unfair advantage over the other models.

When evaluating only the results for the unmodified test set, it is observed that the NN model utilizing sBERT embeddings performs the best, achieving an accuracy of 92.75%. It is closely followed by the tri- and uni-gram models, which achieve accuracies of 90.2% and 88.3%, respectively. We believe that the superiority of the sBERT model over the N-gram models is attributed to its pretraining on a large corpus of text, enabling it to consider more words when predicting the Part of Speech (POS) of a given word. Finally, we observe that the Stanford parser exhibits the poorest performance on this dataset. The N-gram models and the sBERT-based model outperform the Stanford parser due to their ability to capture unexpected tags such as "-NONE-", "#", "-LRB-", and "-RRB-" during training, which the Stanford parser cannot do as it is not trainable. This is highlighted by the fact that the Stanford parser achieves the best performance when the mentioned POS tags are excluded in the modified dataset. For the N-gram and the NN model based on sBERT the performance decrease slightly when going from the unmodified to the modified dataset. This is because the removed tags are easy to predict after optimizing the models on the training data, i.e. *\*-I* is always given POS label *None*.

When diving deeper into the results of the N-gram models, we observe that the tri-gram model outperforms the uni-gram model in terms of accuracy on both the modified and the unmodified dataset. This was expected since tri-gram taggers can capture more context by evaluating three words instead of one. However, the difference is not large, so depending on the application and requirement to accuracy the uni-gram tagger is a good alternative if speed of computation important.

	Unmodified testset	Modified testset
<b>Tri-gram model</b>	90.2%	89.8%
<b>Uni-gram model</b>	88.3%	87.7%
<b>Maximum entropy</b>	-	-
<b>NN using SBERT embeds</b>	92.75%	91.8%
<b>NN using POS-BERT embeds</b>	-	-
<b>Stanford Parser</b>	88.5%	94.9%

Table 1: Results from evaluating the models on the dataset with all POS tags (unmodified) and on the dataset where "-NONE-", "#", "-LRB-", and "-RRB-" are excluded (modified)

## 4.2 BioNLP Shared Task dataset

Table 2 summarizes the results for the BioNLP dataset. The tri-gram model, uni-gram model, NN using SBERT embeds, and NN using POS-BERT embeds were all trained by us, while the maximum entropy model and the Stanford Parser were pre-trained. Where applicable, the first column (Penn treebank training) shows the accuracy when trained on the Penn Treebank training set and tested on the BioNLP testset; the second column (BioNLP Shared Task training) shows the accuracy when trained on the BioNLP training set and tested on the BioNLP testset; and the third column (Models without training) shows the accuracy when tested on the BioNLP testset out of the box.

Across all training procedures, the best accuracy was achieved with the NN using POS-BERT embeddings trained on the BioNLP training set (96.0%). This makes sense, since POS-BERT is designed specifically for POS tagging, and it is able to catch up domain specific terms when finetuned on the BioNLP training set. Further, we see that the S-BERT model rank second of the models trained on the BioNLP training set, with 2.7% and 4.0% down to the tri- and uni-gram model, respectively. Similarly to the Penn treebank dataset, the reason for the superior performance is believed to be the pretraining of sBERT and the ability to take more words into consideration during prediction.

The first thing to notice when evaluating the models trained on the Penn treebank dataset, is the major decrease in accuracy of the N-gram models compared to when they are trained BioNLP training set. The reason for this is that the Penn Treebank subset training set is limited in size, so the models are overfitted to that dataset when only trained on that. Another interesting observation is that while the accuracy of both NN models decrease, the NN model based on sBERT experience a much more drastic decrease than the NN model based on POS-BERT. The reason for this can be attributed to the fact that the POS-BERT model is fine-tuned on the entire Penn treebank dataset, and will therefore be able to generalize better to other distributions than the sBERT model that is not finetuned to any specific task before implemented in our model.



The results of the maximum entropy model and the Stanford parser, which is not trained by us, achieve an accuracy of 82.7% and 82.4%, respectively. When comparing to the performance of the other models that are trained on the Penn treebank dataset, one can see that they outperform all models, except the NN model based on POS-BERT. One of the reasons for this is that the other models are trained on just a small part of the Penn treebank dataset, and therefore are not able of generalizing to new distributions.

	<b>Penn treebank training</b>	<b>BioNLP Shared Task training</b>	<b>Models without training</b>
<b>Tri-gram model</b>	56.5%	91.2%	-
<b>Uni-gram model</b>	55.6%	89.9%	-
<b>Maximum entropy model</b>	-	-	82.7%
<b>NN using SBERT embeds</b>	70.7%	93.9%	-
<b>NN using POS-BERT embeds</b>	86.1%	96.0%	-
<b>Stanford Parser</b>	-	-	82.4%

Table 2: Results from training the models on different datasets and testing them on the BioNLP Shared Task testset.

### 4.3 Per-sentence evaluation with the Stanford Parser

When evaluating per-token, the Stanford Parser was 88.5% accurate for the test set before excluding unknown parts of speech. After excluding, the accuracy jumped to 94.9%. When evaluating per-sentence, it was 0.0% accurate for the test set before excluding unknown parts of speech. In other words, every sentence in the test set included one of { -NONE-, #, -LRB-, -RRB- }. After excluding these tokens (but keeping the rest of the sentence), the accuracy jumped to 33.6%.

	<b>BioNLP per-sentence</b>	<b>Penn Treebank per-sentence</b>	
		<b>unmodified</b>	<b>modified</b>
<b>correct</b>	2,549 (8.0%)	0 (0.0%)	263 (33.6%)

Table 3: Stanford Parser per-sentence results.

We conclude that our per-sentence definition is not very useful. When we exclude unknown tags, only about a third of sentences were correct despite about 95% of individual tags being correct. For this reason we did not apply the definition to the other methods.

## 5 Garden-path sentences

To test the capabilities of our models on harder sentences, we compile a short list of 5 common Garden-path sentences.

For the choice of models we simply picked the two best models in section 4.1 which are the Stanford parser and the NN using POS-BERT embedding.

Additionally, we test our Garden-path sentences on ChatGPT to compare its language capabilities to our different models. We specify how exactly we prompt ChatGPT in Appendix 7 and we list all of the corresponding outputs in Appendix 7.

- G1: **The old man the boat.** (Identified as correct if man is tagged as a verb)
- G2: **The man whistling tunes pianos.** (Identified as correct if tunes is tagged as a verb)
- G3: **I convinced her children are noisy.** (Identified as correct if "her" is tagged as a personal pronoun (PRP) instead of possessive pronoun PRP\$)
- G4: **Without her contributions would be impossible.** (Identified as correct if "her" is tagged as a personal pronoun (PRP) instead of possessive pronoun PRP\$)

G5: **Fat people eat accumulates.** (Identified as correct if "Fat" is tagged as a noun instead of an Adjective. And "accumulates" is tagged as a verb instead of a Noun)

Garden path sentence	BERT	Stanford parser	ChatGPT
G1	X	X	X
G2	✓	✓	X
G3	X	X	X
G4	X	✓	X
G5	X	X	X

Table 4: Results for the garden path sentences

We list our results in Table 4. Interestingly we note that sentences *G1*, *G3*, and *G5* were identified incorrectly by all three different methods we tested. While the Stanford parser was the only method able to correctly parse *G4*. And that *G2* was correctly parsed by both the BERT-based model and the Stanford parser while only ChatGPT failed on it.

We note that all of ChatGPT’s answers are incorrect as it identifies all the sentences as ungrammatical. This is slightly surprising as ChatGPT was trained on large subsets of online data which inevitably includes mentions of Garden Path Sentences.

Interestingly, we can see that despite the incredibly high accuracies obtained when parsing regular sentences in 4.1, the models still struggle with correctly parsing Garden Path Sentences.

## 6 Online access to our Bert-based model and code

We have developed a page to test our NN using POS-BERT embeddings model, the page is available at <http://somerset.csail.mit.edu:47316/>.

Additionally, all of our code can be found at <https://github.com/Ar-Kareem/Penn-Tree-Bank-Project>

## 7 Conclusion

In this report we have compared the performance of six NLP models on POS tagging. Two models were based on the large language model BERT, and the remaining models were more traditional methods. We first evaluated the models on a tagged subset of the well-known Penn Treebank dataset, provided by NLTK. We then perform more evaluations on a text dataset in the BioNLP Shared Task which contains biological and highly domain-specific text which tests the capabilities of these models to transfer to other domains. Finally, we contrast the success of these models with the struggle faced when parsing garden-path sentences.

The results show that all models achieve similar accuracy when both trained and evaluated on the same data, which is true for both datasets in this project. This indicates that the traditional methods are still competitive when it comes to POS tagging, especially when computational speed and low complexity are important. Another interesting observation was the big difference when training the models on the NLTK Penn Treebank training set and evaluating them on the BioNLP test set, compared when training and evaluating on the same dataset. This highlight the the importance of a large training set to be able to generalize to new distributions and the difficulty of capturing domain specific terms in POS tagging.

## References

- [1] A. Chiche, B. Yitagesu, Part of speech tagging: a systematic review of deep learning and machine learning approaches 9 (1) 10. doi:10.1186/s40537-022-00561-y.  
URL <https://doi.org/10.1186/s40537-022-00561-y>
- [2] C. Lv, H. Liu, Y. Dong, Y. Chen, Corpus based part-of-speech tagging 19 (3) 647–654. doi:10.1007/s10772-016-9356-2.  
URL <https://doi.org/10.1007/s10772-016-9356-2>
- [3] B. E., Rule-based part of speech, in: Proc. third Conf. Appl. Nat. Lang. Process, 1992.
- [4] F. Pisceldo, M. Adriani, R. Manurung, Probabilistic part of speech tagging for bahasa indonesia.
- [5] J. Kupiec, Robust part-of-speech tagging using a hidden markov model 6 (3) 225–242. doi:10.1016/0885-2308(92)90019-Z.  
URL <https://www.sciencedirect.com/science/article/pii/088523089290019Z>
- [6] F. Jelinek, J. D. Lafferty, R. L. Mercer, Basic methods of probabilistic context free grammars, Springer, 1992.
- [7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).
- [8] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).
- [9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.
- [10] Papers with code - penn treebank benchmark (part-of-speech tagging).  
URL <https://paperswithcode.com/sota/part-of-speech-tagging-on-penn-treebank>
- [11] F. D. F. A. A. R. K. Hassan Sajjad, Nadir Durrani, J. Xu, Analyzing encoded concepts in transformer language models, in: North American Chapter of the Association of Computational Linguistics: Human Language Technologies (NAACL), NAACL '22, Seattle, 2022.
- [12] T. S. N. L. P. Group, Stanford parser, Available at <https://nlp.stanford.edu/software/lex-parser.shtml>.

## Appendix

### ChatGPT Prompt

To evaluate ChatGPT on garden path sentences, we use the following prompt:

For the following input sentence, identify if it is ‘Grammatical’ or ‘Not Grammatical’, and if you identify it as ‘Grammatical’ then label each word in the sentence with the correct Penn Part of Speech Tags. Input sentence: *Garden Path Sentence inserted here.*

### ChatGPT Output

Here we list the output for all 5 garden path sentences from ChatGPT:

```
G1: Not Grammatical
Explanation: This sentence lacks clarity and is grammatically
incorrect. It could be interpreted in multiple ways, and none of
them make complete sense. It needs restructuring or additional
information to be grammatical.
```

G2: Not Grammatical

Explanation: The sentence is not grammatically correct as the verb is missing. It could be rewritten as "The man who whistles tunes pianos."

G3: Not Grammatical

The sentence is not grammatical because the verb "convinced" requires an object. It can be fixed by adding an object to the sentence, for example: "I convinced her that children are noisy."

G4: Not Grammatical

Explanation: The sentence is missing a subject. It can be corrected by adding a subject, such as "Without her, it would be impossible."

G5: Not Grammatical

The input sentence "Fat people eat accumulates" is Not Grammatical because it lacks subject-verb agreement and proper word usage.

If we were to correct the sentence to "Fat people eat accumulations," the Penn Part of Speech tags for each word would be:

Fat (JJ): adjective

people (NNS): plural noun

eat (VB): verb

accumulations (NNS): plural noun

We note that all of ChatGPT's answers are incorrect as it identifies all the sentences as ungrammatical.