Part-of-Speech Tagging in English using Contextual Word Embeddings

Juhani Dickinson
jdickin9@uwo.ca

Paul Moore
email address or ORCID

***Abstract*—ABSTRACT**

***Index Terms*—pos tagging, part of speech, nlp, word embedding, contextual word embedding**

I. Introduction

A. *Parts of Speech*

In a sentence, each word has a syntactic role to play. In the sentence “The brown dog runs quickly towards food.”, “dog” denotes a thing, “brown” modifies or describes “dog”, “runs” denotes an action, and so on. These syntactic roles are commonly referred to as “parts of speech”, and are mapped to words using part-of-speech tags. However, this is not a one-to-one mapping. Many words have multiple meanings, and in many cases, these different meanings play different syntactic roles, mapping them to different parts of speech. For example, the word “orange”, spelled and pronounced identically, can be either a noun or an adjective, as seen in sentences (1) and (2).

“I often eat an orange after working out.” (Noun) (1)

“The orange car has arrived.” (Adjective) (2)

Figuring out what part of speech a word has is not an unsolvable task, however. Where a word is in the sentence and what words surround it play a major role in disambiguating between syntactic roles. In (1), the word after “orange” is “after”, which is an adverb. “Orange” is either an adjective or a noun, and since the following word is an adverb, it cannot be an adjective. In (2), the words surrounding “orange” are “the” and “car”, which are a determiner and a noun respectively. In this position, “orange” describes “car”, making it an adjective.

B. *Part-of-Speech (POS) Tagging*

The task of POS tagging (often simply called tagging) is as follows: given a sentence, label each word in the sentence with the POS tag that describes its syntactic role, seen below for sentence (3).

(3)	Can	you	spot	the	large	spot	on	your	nose	?
	AUX	D	VB	D	ADJ	N	PP	D	N	.

What makes the task of tagging challenging is the contextual requirement: without it, both instances of “spot” in sentence (3) would be tagged as either VB or N. Because of this, non-naive tagger implementations leverage some level of context to perform their task. Simple implementations like n-gram taggers assign tags to words based on both the word itself and the tags of the preceding n words.

C. *Word Embeddings*

A common issue with simple tagging implementations is resistance to parallelisation. Any operation below the level of sentence must be linear, as the tag of the current word depends on the tag(s) of the previous words. To get around the linear-time requirement, some taggers use word embeddings, which represent words as vectors. This embedding function takes into account the surrounding words, so that the meaning and context of each word is stored in the embedding. Word embeddings are classifier models usually trained using one of two methods: predict target word from context (Continuous Bag of Words, CBOW) and predict context from target word (Skipgram). In CBOW, the words surrounding the target word are mapped to one-hot vectors, which are each then projected down to a dense vector by the classifier model and averaged together. This averaged vector is then used to predict the target word. In Skipgram, the target word is mapped to a one-hot vector and then projected down to a dense vector by the classifier. This is done for multiple target words, the dense vectors of which are averaged together and used to predict the context in which the target words appear. Using embeddings, tagging can be parallelised at the level of individual words.

1) Static Word Embeddings: While word embeddings take context into account by design, static word embeddings take the naive approach to context. These mapping functions collect all contexts for a given word and uses them to generate a single embedding per word. Often, this approach fails to take into account that words can have different meanings (“river bank” vs. “bank account”) or different syntactic roles. This means that a singular word embedding must hold information on every possible context for any given word, which causes problems when the different meanings of a word are not syntactically or semantically close to one another.

2) Contextual Word Embeddings: Contextual word embeddings solve the syntactic and semantic problems that static word embeddings have by moving from word embeddings to word token embeddings. Now, words with multiple meanings or syntactic roles are split into multiple word tokens, one for each use. Contextual embeddings map these tokens to vectors, preserving specific meaning. With contextual embeddings, a word like “spot” is first split into two tokens, such as “spot–noun” and “spot–verb”, and these tokens are embedded separately. While contextual embedding systems are more computationally demanding to train and use, their contextaware nature often leads to increased performance in context-sensitive tasks such as POS tagging.

II. Motivations

POS tags are useful and sometimes key pieces of information in many natural language processing (NLP) tasks such as text-to-speech (TTS), word-sense disambiguation, marking word order, and named entity recognition (¡ref¿). In TTS applications, the pronunciation of a word can change depending on what its syntactic role is or sometimes even what tense it is. The word "resume" is pronounced differently based on whether it is meant as "continue doing" or as "a document detailing professional experience", which can be easily disambiguated based on whether it acts as a verb or a noun. This disambiguation-by-tag is also used in word-sense disambiguation, like in sentence (3). In a search-engine setting, understanding whether "spot" refers to an activity or a visual marker allows for more accurate search results.

A. Benefits for NLP in Low-Resource Languages

While high-resource languages like English no longer use POS tagging in some popular NLP applications like AI assistants by making use of the massive collections of curated data now available, most languages do not have the luxury of large, high-quality datasets. In these low-resource languages, POS tags are still critical for NLP tasks. For a task like machine translation, (¡ref¿) found that using linguistic features derived from POS tags improved the translation performance not only between the low-resource language pair of Thai and Myanmar but also between Thai and English and Myanmar and English. While the highest accuracy was found on English-Thai translations, likely due to both languages having a Subject-Verb-Object sentence structure, the English-Myanmar and Thai-Myanmar pairs both saw improvements over the baseline.

While some low-resource languages have POS taggers or have the resources available for one to be reasonably made, many such languages do not. In such cases, POS taggers for high-resource languages are still quite useful. In low-resource languages where training data is scarce, one established method of finding training data is to use cross-linguistic transfer to automatically generate POS training data. This is done by lining up texts that appear in both a high-resource language and the target low-resource language and running POS taggers and parsers for the high-resource language (¡ref¿). The results from these taggers and parsers are then projected onto the low-resource text. With an advanced projection method such as Graph Label Propagation (GLP) proposed by (¡ref¿) to transfer tags from only English, tagging accuracy on a variety of languages averaged 80.6%. Notably, methods such as GLP can leverage multiple parallel texts from high-resource languages to increase performance even further, with tests on the same variety of languages achieving an accuracy of 84.0%. However, leveraging multiple languages simultaneously is not always possible for certain low-resource languages due to the lack of available parallel data.

As well as being useful, the efficacy of cross-linguistic transfer is predictable. Reference (¡ref¿) found that language families and morphological structures have a major impact on the performance of cross-linguistic performance. Specifically, performance was much higher when the source model for cross-linguistic transfer was in the same language family as the target language. This was especially true of morphologically-rich languages. Even outside of the same family, cross-linguistic transfer had better performance when moving between morphologically-rich languages, contrasting with morphologically-poor languages experiencing steeper drops in performance in similar cross-family settings. Importantly, no data from the target language was used for training or fine-tuning in these tests, which indicates that a target language being low-resource does not impact the efficacy of cross-linguistic transfer.

pass

B. Benefits of an English POS tagger

Finding a good POS tagger for English, a high-resource language, has clear benefits for linguistically-related low-resource languages due to the success of cross-linguistic transfer. One example is Scots, a language in the Anglic family currently considered "vulnerable" by the UNESCO Atlas of the World's Languages in Danger (¡ref¿). Research suggests that NLP can support and help revitalise endangered languages (¡ref¿) through machine translation, TTS, and integration into learning materials. With competent machine translation, more materials can be translated into Scots, which encourages learning and expands which facets of life can be interacted with in Scots. TTS has applications when paired with translation and learning tools, and brings one facet of accessibility into the language as well. Finally, machine-based language learning apps like Duolingo have made use of NLP and AI to provide more language learning content (¡ref¿), and members of the Scots speaker community can make use of POS tagging directly to help with learning Scots linguistics.

To this end, we will test multiple prominent word embedding models on the same English dataset in order to determine the best model. Knowing the best model ¡–FINISH THIS–¿

III. Related Work

A. Existing Reviews of POS Taggers

¡–INTRO SENTENCE–¿. References (¡ref¿) and (¡ref¿) conduct literature reviews of many POS tagging papers published between 2017 and 2023, looking at a variety of deep learning and machine learning approaches. The studies chosen for these reviews span a wide range of both high-resource and low-resource languages, ¡¿ ¡¿

A common issue with most of the studies is the lack of a standardised dataset for training or testing on. As a result, the findings from the studies reviewed cannot be directly compared, even between papers analysing the same language. This serves as part of the motivation for our work here. Testing various word embeddings on a standard dataset provides a ¡¿

Although many languages were represented in the studies selected for review, there was very little representation of the English language among the selection. Contemporary comparisons of POS taggers on English focus more on specific applications and more niche datasets. ¡¿

B. Studies on Cross-Linguistic Transfer

Discussed above in II, (¡ref¿) provides crucial insights into the efficacy and patterns of cross-linguistic transfer. In this study, source-target language pairs were created from the chosen languages, and the XLM-RoBERTa multilingual transformer model was fine-tuned on each source language to serve as the POS tagger. Each pair was then analysed in terms of model performance, both quantitatively through precision, recall, and F1 scores, and qualitatively, through examining the areas where errors occurred to find commonly-appearing linguistic features such as morphology. With this work as inspiration, we will consider cross-linguistic transfer over a more diverse set of word embeddings rather than only a BERT variant. While we do not conduct any qualitative analyses on our results, ¡¿

Like (¡ref.prev¿), (¡ref¿) uses word embeddings generated by XLM-RoBERTa, ¡¿

C. Word Embeddings Used

IV. Methods

A. Hypotheses

We hypothesize that high quality embeddings trained on a high-resource language can result in good POS tagging performance in a related low-resource language. We also hypothesize that the powerful transformer based embeddings (citations) will result in superior performance on such tasks over more classical embedding methods (citations). Essentially, we will find the best embedding model for use in POS tagging of low-resource languages that are related to English.

B. Datasets

In order to meaningfully compare embeddings head to head, the classifiers using them to tag sentences must be trained and tested on the same datasets.

Each of the embedding models evaluated in this paper are trained on massive amounts of English data, so we initially test them on the Universal Dependencies English Web Treebank dataset (UD_ENGLISH) (citation), which contains sentences sourced from various forms of media. The Universal Dependencies English Web Treebank contains 16622 sentences that are split into three partitions: train, dev, and test, as illustrated in Table I.

In order to test the models' effectiveness at embedding text from low resource languages that are related to English, we use the Norwegian Dependency Treebank - Bokmål (UD_NORWEGIAN) (citation). However, Norwegian is not a low-resource language. Therefore we downsample the dataset to just 5% of it's original size, resulting in a corpus containing 1002 sentences partitioned as illustrated in Table I. This simulates a scenario where the data available for the target language is scarce, similar to how it is for real low-resource languages.

TABLE I
Distribution of Sentence Data

	English	**Norwegian**
Train	12544	785
Dev	2001	120
Test	2077	97

The Universal Dependencies Treebank datasets are used for multiple reasons, including their diverse data sources and manageable size for local hardware. However, their most important feature is that the annotations are hand corrected by humans to ensure 100% label accuracy.

C. Evaluation Metrics

To evaluate the taggers, we calculate macro precision, macro recall, and micro F1 score. Macro precision is the average precision of each of the N classes, weighted equally:

$$\text{Macro Precision} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i}$$

Macro recall is the average recall of each class weighted equally:

$$\text{Macro Recall} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i}$$

Macro precision and recall measure a model's performance on rare classes. Micro F1 score is actually equivalent to the average accuracy in multi-class classification tasks:

$$\text{Micro F1} = \frac{2\sum_{i=1}^{N} TP_i}{2(\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i + \sum_{i=1}^{N} FN_i)}$$

Micro F1 score is used as a measure of overall performance.

D. Classifier Model Architecture

In order to isolate the embeddings in our tests, we use the same architecture to classify the embeddings from each embedding model. It is also critical that the performance of each model can be attributed to the embeddings rather than to the classifier itself, so a simple linear classifier is used. In each case, the linear classifier takes as input the embeddings and outputs a vector of size 17—one feature for each tag in the UPOS standard. Locked dropout with probability 0.5 is applied to the embeddings before classification.

E. Model Training

Each model is trained using the Adam optimizer (citation) and cross-entropy loss. The learning rate is tailored to each model, and adjusted automatically with a learning rate scheduler to ensure fast convergence without instability or overfitting. For the transformer (citation) based embedding models, the final layer of the embedding model itself is fine-tuned along side the linear classifier.

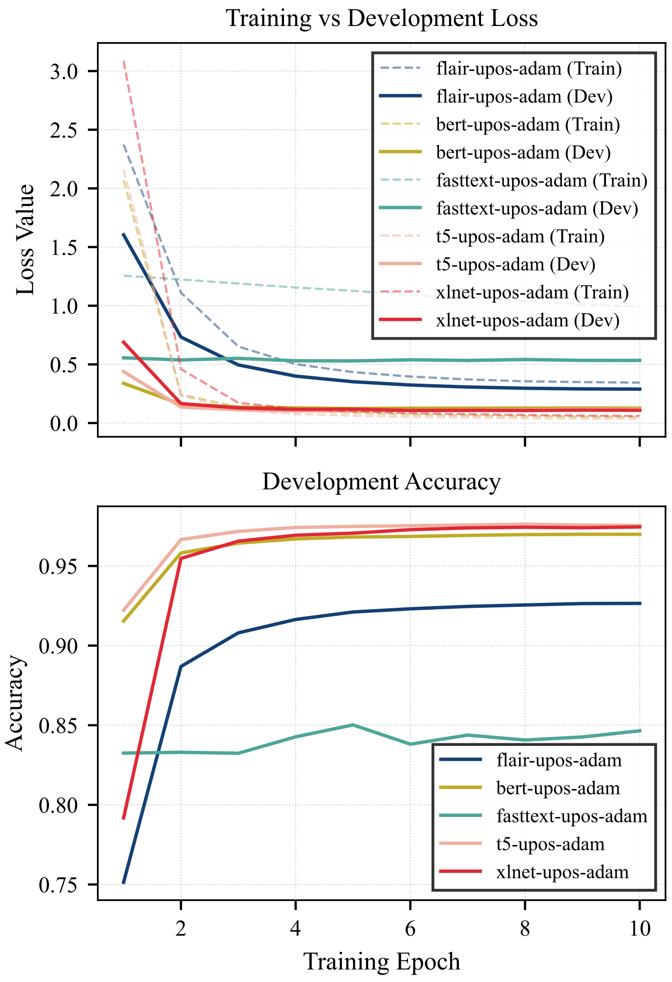

Fig. 1. Training vs. development loss and development accuracy per epoch on English data.

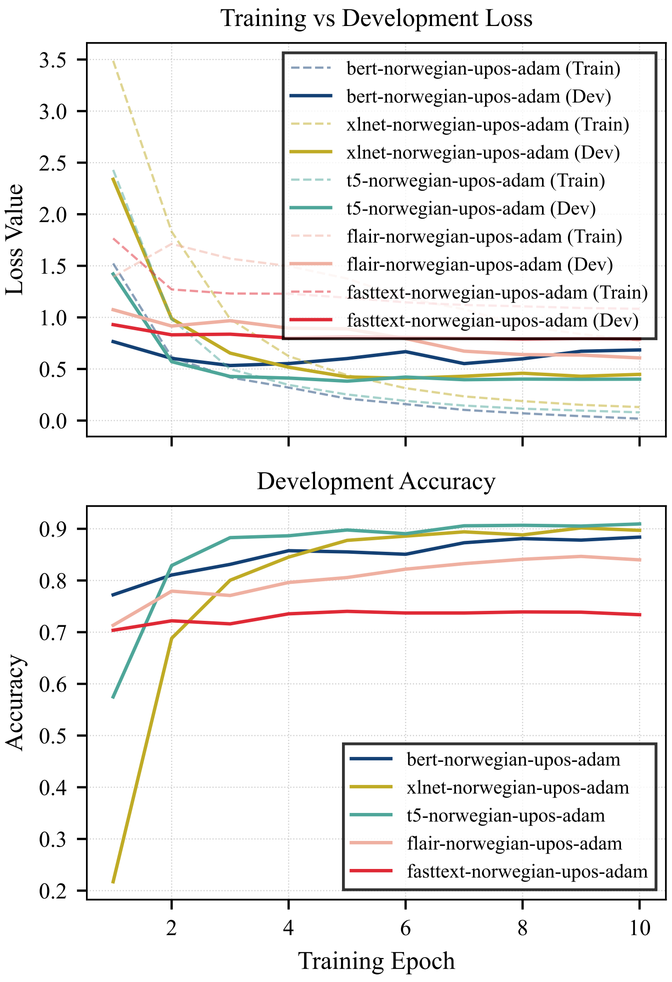

Fig. 2. Training vs. development loss and development accuracy per epoch on Norwegian data.

1) Framework: (flair citation) released a comprehensive NLP framework that leverages PyTorch (citation) alongside their contextual string embeddings that are evaluated in this paper. The Flair framework is used to implement, train, and evaluate each of the embedding models. All training and evaluation was executed locally using CUDA on an NVIDIA RTX4060-mobile GPU.

2) Training on English Data: As a baseline for measuring the performance of each embedding model, we train over the English dataset. Each model is trained for 10 epochs with min-batch size of 16. Longer training stints could be considered, but given the relatively small quantity of data, too much could result in over-fitting. Moreover, as illustrated in Fig. 2, each model converges rather quickly (after only 4-6 epochs). This demonstrates that additional training would have diminishing returns. Figure 2 also indicates that the baseline static embedding model—FastText, struggles to converge. This is likely due to its inability to capture contextual information.

3) Training on Norwegian Data: Next, to compare performance in the high-resource language to the performance in the target language, each model is trained on the downsampled Norwegian dataset from the same starting point as the English models. Again, the models are trained for 10 epochs with min-batch size of 16, to avoid over-fitting. Figure 1 shows that each model converges nicely (apart from the static embedding model), as they do when trained on English data. Unsurprisingly, the development loss and accuracy do not reach the same levels as in Fig. 2.

4) Fine-Tuning English Models on Norwegian Data: Finally, while most of the heavy lifting of the models comes from the embeddings, we are interested to see if pre-training the simple classifier head on English data can give another performance boost. Therefore, we fine-tune the English taggers on the Norwegian dataset for an additional 5 epochs with a min-batch size of 16. The convergence of loss and accuracy can be seen in Figure 3.

V. Results

A. English Model Performance

Rather unsurprisingly, the FastText embeddings yielded the worst performance, followed by the Flair embeddings, and then the three transformer based models performed similarly,

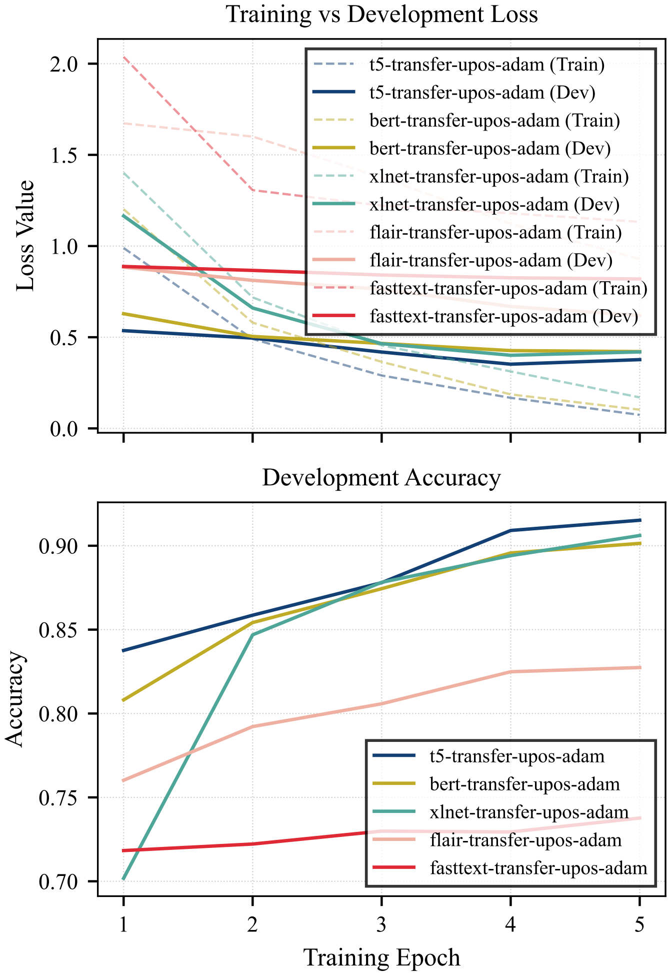

Fig. 3. Training vs. development loss and development accuracy per epoch during the fine-tuning of English models on Norwegian data.

each achieving an F1 score $>$ 97%. The macro precision, macro recall, and micro F1 score are shown for each model in Table II. The transformer models also perform quite well on the rare classes, as indicated by high macro precision and recall.

TABLE II
PERFORMANCE OF TAGGERS TRAINED ON ENGLISH

Embeddings	**Precision**	**Recall**	**F1 Score**
FastText	0.7944	0.7506	0.8471
Flair	0.8698	0.8079	0.9300
BERT	**0.9499**	0.9163	0.9714
XLNet	0.9486	**0.9177**	0.9745
T5	0.9116	0.9152	**0.9764**

B. *Norwegian Model Performance*

Table III shows the results of training the taggers on Norwegian directly. Despite never being trained on Norwegian text apart from the limited training done in this study, both XLNet and T5 performed quite well on the Norwegian dataset. T5 was the most robust of the models, losing only approximately 0.06 on its F1 score. This indicates a strong ability to generalize to similar languages. All models also saw large drops in macro precision and recall, which can be attributed to the lack of rare class representation in a smaller dataset.

TABLE III
PERFORMANCE OF TAGGERS TRAINED ON NORWEGIAN

Embeddings	**Precision**	**Recall**	**F1 Score**
FastText	0.7104	0.6533	0.7162
Flair	0.8796	0.8678	0.8600
BERT	0.8408	0.8354	0.8894
XLNet	**0.8440**	**0.8507**	0.9070
T5	0.7565	0.7606	**0.9163**

C. *Fine-Tuned English Model Performance*

Finally, Table IV illustrates the results of fine-tuning the English taggers on the Norwegian dataset. Interestingly, despite its simplicity, pre-training the classifier head does appear to have a small performance benefit. Both T5 and XLNet see increases to their F1 score, 0.49% and 1.22% respectively. FastText also saw a large boost in performance of 3.59%.

TABLE IV
PERFORMANCE OF TAGGERS FINE-TUNED ON NORWEGIAN

Embeddings	**Precision**	**Recall**	**F1 Score**
FastText	0.8226	0.7021	0.7521
Flair	0.7755	0.7168	0.8264
BERT	0.8364	0.8481	0.8815
XLNet	0.7614	0.7652	0.9192
T5	**0.8710**	**0.8629**	**0.9212**

VI. DISCUSSION

A. *Embedding Quality*

In our first experiment, we found that the transformer embeddings BERT, XLNet, and T5 were of the highest quality. When tested on an even playing field (The same classifier, training procedures, etc.), they significantly outperformed the competition. This indicates that their embeddings are richer, and encode a deeper understanding of the underlying patterns of the English language.

B. *Generalizability*

In our second experiment, we found that higher quality embeddings, and therefore higher POS tagging performance on a high-resource language results in higher performance on a related low-resource language. Models using T5, XLNet, and BERT, who achieved very high accuracy on the English dataset, performed similarly relative to each other and the other models on the downsampled Norwegian dataset. Indicating that the transformer embeddings are capable of generalizing to languages that are related to English, including low-resource languages.

C. Pre-Training and Fine-Tuning

While not the focus of this paper, our third experiment revealed that pre-training the classifier head of the tagging model before fine-tuning the whole model can result in a performance increase even with the simplest possible classifier. It would not be crazy to assume that the benefits of such pre-training would be amplified by a more sophisticated classification model like many of the state-of-the-art taggers, who use a biLSTM.

D. Which Embeddings Should Be Used?

It is clear that static embeddings should not be used for POS tagging in low-resource languages related to English, or really any POS tagger in general. Unless there are significant time, or compute restrictions in place preventing the use of larger models like T5, static word embeddings are overshadowed. Even so, many large models like T5, XLNet, and BERT were released with alternatives with fewer parameters.

For POS tagging in English, any of the transformer based models can provide extremely rich embeddings capable of very high performance when partnered with a sufficient classifier. Even with the simplest possible classifier, the transformer embeddings reach near perfect performance (Table II).

While T5, XLNet, and BERT demonstrated very similar performance when tested on English, BERT fell behind when evaluated on the simulated low-resource related language. Between T5 and XLNet, T5 was marginally more capable of generalizing to the new language.

VII. Conclusion

pass

VIII. Acknowledgements

¡–LLM USE YES/NO–¿

References

[1] Pass