I. 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 over-fitting. For the transformer (citation) based embedding models, the final layer of the embedding model itself is fine-tuned along side the linear classifier.

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

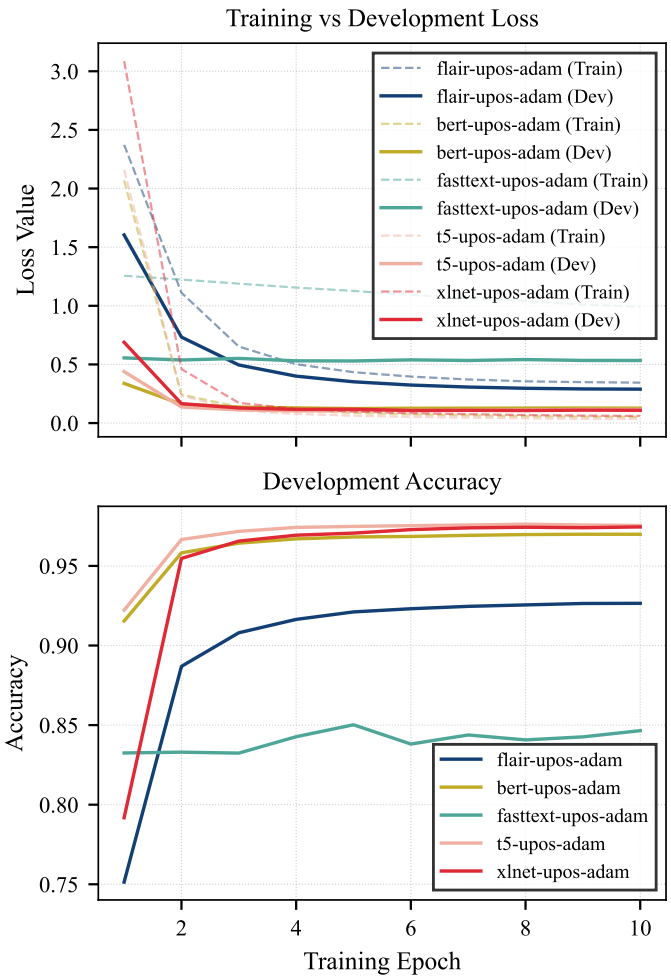

Fig. 1. Training vs. development loss and development accuracy per epoch on English data.

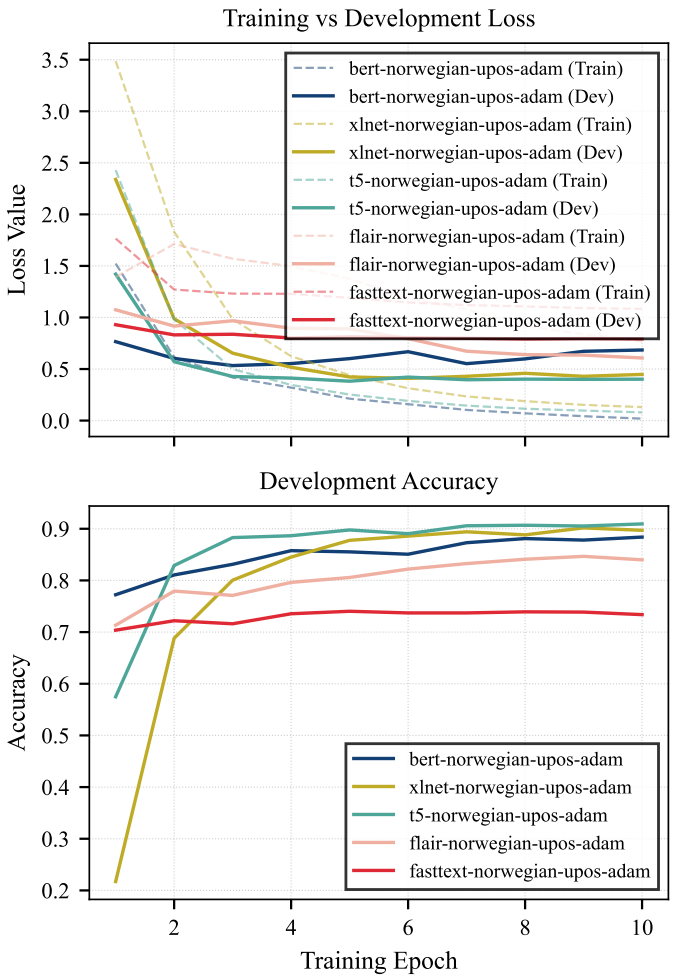

Fig. 2. Training vs. development loss and development accuracy per epoch on Norwegian data.

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.

II. 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, 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.

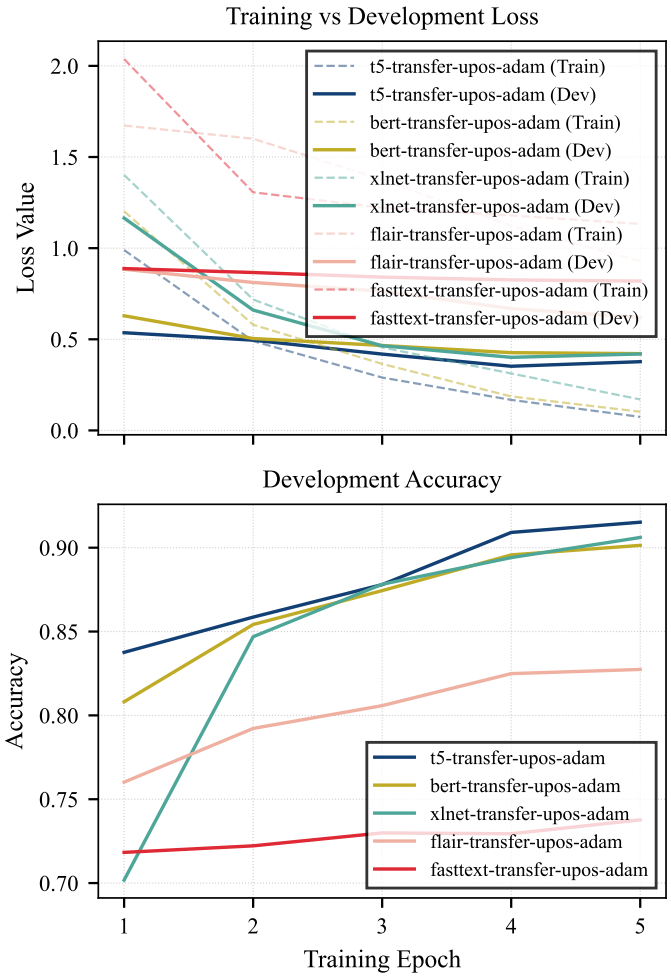

Fig. 3. Training vs. development loss and development accuracy per epoch during the fine-tuning of English models on Norwegian data.

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%.

III. Discussion

A. Embedding Quality

In our first experiment, we found that the transformer embeddings BERT, XLNet, and T5 were of the highest quality.

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**

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.