Final Project - NLP

We have been assigned to complete a Methodology Description and Results docs in addition to our ipynb notebook. This document outlines every step we took to complete project number 4.

Objective

The objective was to examine whether the addition of syntactic information (POS Tags) to the Word Embedding improves the performance of sentiment classification in Movie reviews.

Steps

Data loading was performed using the **SST2** (Stanford Sentiment Treebank) dataset, which contains short texts with binary sentiment tagging (positive/negative).
Preprocessing involved tokenization, lemmatization, and the removal of stopwords. Additionally, each word was assigned a POS tag.

For **word embedding**, two versions were used. The regular version calculated the average of embeddings based on words only. The second version merged the words and POS tag for each word (example for what we did is: "love_VERB"), and then used the embedding based on the new words.

Two types of models were employed. A basic model,Simple MLP. And a more advanced model, LSTM (Long Short-Term Memory), operated on a sequence of words, with or without the addition of POS tags.

Evaluation metrics included Accuracy, Precision, and F1 Score.

Results

Model	Accuracy	F1 score	Precision
Simple MLP Without pos	0.88047	0.8942	0.88
Simple MLP with POS	0.871	0.891	0.84
LSTM without POS	0.894	0.905363	0.905544
LSTM with POS	0.894135	0.904141	

Conclusion

The addition of POS information did not contribute to improving model performance. However, there is potential for an LSTM model, which processes sequences, to utilize complex syntactic structures better, but the POS did not add anything to it.

The reasons mentioned in the internet said that:

1. Pos tagging might be noisy on informal texts - making the results worse.

2. Pretrained embedding already capture syntactic roles.

3. So basically the pos embedding might help in other cases in formal text etc... but not in our case.

Algorithm / Quality Metric Improvement Proposal

Proposal

We propose using a model that explicitly integrates POS tags as an additional embedding, rather than solely through text processing.

Rationale

Instead of adding the tag as text (e.g., "love_VERB"), it is possible to create a separate embedding for POS tags. This POS embedding can then be combined with the word embedding (for example, through concatenation) and fed into a model such as BiLSTM (Bidirectional LSTM) or an attention model. This approach would allow the model to understand the syntactic role of a word independently of its semantic content, potentially improving classification capabilities.

Hyperparameter Change Results Comparison

We examined how changing the learning rate in the LSTM model affects performance.

0	0.001	0.894506	0.906396	0.894320	0.892947	0.905455
1	0.005	0.894655	0.906675	0.893223	0.894135	0.905887
2	0.010	0.884113	0.896023	0.893821	0.883148	0.895485
3	0.050	0.714402	0.758491	0.715673	0.700148	0.757839
4	0.100	0.612769	0.668489	0.637764	0.517298	0.505476
5	0.250	0.515516	0.557859	0.566144	0.460431	0.264075

Insight

Best performance happens at lr 0.001 and 0.005.

- Accuracy and F1 are both highest for `lr=0.005`, with and without POS tags.
- The addition of POS tags don’t give a large boost, but is slightly helpful at these low learning rates.

Use **learning rate = 0.005** for best tradeoff between speed and accuracy, and consider using POS tags if training is stable.