Introduction to NLP (CS7.401)

Spring 2022, IIIT Hyderabad
Project Report

Abhinav S Menon (2020114001)
Pratyaksh Gautam (2020114002)
Shashwat Singh (2020114016)

Introduction

We have implemented a graph-based dependency parser according to Stanford's submission for the 2017 CoNLL shared task. It uses a POS tagger and a biaffine edge scorer and edge labeller.

Word Embeddings

The word embeddings are created by summing up three vectors for each word: a pre-trained embedding, an ordinary trained embedding, and a character-level embedding.

The pre-trained embeddings are taken from GloVe's 100-dimensional embeddings.

The trained embeddings are formed from a randomly initialised matrix.

The character-level embedding uses an LSTM and computes attention over the sequence of hidden states, followed by concatenation to the cell state and linear transformation to the word embedding space.

POS Tagger

The POS tagger uses the above word embeddings passed through an LSTM, with an affine layer applied to the hidden states.

The embeddings for the tags (required for the parser) are extracted from the weight matrix of the affine layer.

Parser

The parser consists of two biaffine classifiers: an edge scorer and an edge labeller. Both of these take as input the outputs of a BiLSTM, run on the embeddings described above.

The edge scorer takes a sentence and returns, for each word w_j, a probability distribution $P(w_i) =$ the probability that w_i is the head of w_j. The head of the root word is taken to be the `<BOS>` token.

The edge labeller takes a pair of words (a head and a dependent) and classifies the edge from the head to the dependent into one of the universal dependency relations.

During training, the heads are found by simply taking the highest-probability head for each word. However, this may not always result in a valid parse. For final testing, the MST algorithm described in Jurafsky & Martin, Chapter 14 is used, which iteratively identifies and removes cycles.

Results

POS Tagger

The results of the POS tagger, trained and tested on the English Atis datasets, are as follows.

English

Hidden Size of 50×2

```
Overall
              precision    recall  f1-score   support

         ADJ       0.92      0.97      0.95       220
         ADP       0.98      1.00      0.99      1434
         ADV       0.98      0.71      0.82        76
         AUX       0.99      0.99      0.99       256
       CCONJ       1.00      0.99      1.00       109
         DET       1.00      0.99      0.99       512
        INTJ       1.00      1.00      1.00        36
        NOUN       0.95      0.99      0.97       995
        NULL       1.00      1.00      1.00     13344
         NUM       0.97      0.84      0.90       127
        PART       0.98      0.96      0.97        56
        PRON       0.98      1.00      0.99       392
       PROPN       0.99      0.99      0.99      1738
        VERB       0.99      0.94      0.96       629

    accuracy                           0.99     19924
   macro avg       0.98      0.96      0.97     19924
weighted avg       0.99      0.99      0.99     19924
```

Hidden Size of 200×2 (note: this is as prescribed by the paper)

```
Overall
              precision    recall  f1-score   support

         ADJ       0.89      0.96      0.93       220
         ADP       0.99      1.00      0.99      1434
         ADV       0.94      0.76      0.84        76
         AUX       0.99      0.98      0.99       256
       CCONJ       1.00      0.99      1.00       109
         DET       0.99      0.98      0.99       512
        INTJ       0.97      1.00      0.99        36
        NOUN       0.96      0.99      0.97       995
        NULL       1.00      1.00      1.00     13344
         NUM       0.96      0.83      0.89       127
        PART       0.98      0.93      0.95        56
        PRON       0.98      0.99      0.99       392
       PROPN       0.99      0.99      0.99      1738
        VERB       0.99      0.95      0.97       629

    accuracy                           0.99     19924
   macro avg       0.97      0.95      0.96     19924
weighted avg       0.99      0.99      0.99     19924
```

Hindi

```
Overall
              precision    recall  f1-score   support

         ADJ       0.92      0.86      0.89      2043
         ADP       0.99      1.00      0.99      7544
         ADV       0.89      0.86      0.87       304
         AUX       0.97      0.98      0.97      2596
       CCONJ       0.98      1.00      0.99       635
         DET       0.95      0.96      0.96       745
        NOUN       0.91      0.91      0.91      8036
        NULL       1.00      1.00      1.00     77398
         NUM       0.96      0.85      0.90       693
        PART       0.99      0.96      0.98       677
        PRON       0.98      0.97      0.97      1372
       PROPN       0.84      0.88      0.86      4438
       PUNCT       1.00      1.00      1.00      2420
       SCONJ       0.98      0.99      0.99       655
        VERB       0.96      0.94      0.95      3263
           X       0.21      0.33      0.26         9

    accuracy                           0.98    112828
   macro avg       0.91      0.90      0.91    112828
```

```
weighted avg       0.98      0.98      0.98    112828
```

Sanskrit

```
Overall
              precision    recall  f1-score   support

         ADJ       0.44      0.33      0.37       870
         ADV       0.92      0.87      0.90      1084
         AUX       0.80      0.44      0.57        90
       CCONJ       0.99      0.96      0.98       152
         DET       0.88      0.15      0.25        48
        NOUN       0.77      0.74      0.75      3074
        NULL       1.00      1.00      1.00    226008
         NUM       0.86      0.28      0.42        89
        PART       0.99      0.99      0.99       785
        PRON       0.92      0.88      0.90      1443
       SCONJ       0.86      0.82      0.84        97
        VERB       0.62      0.82      0.71      1940

    accuracy                           0.99    235680
   macro avg       0.84      0.69      0.72    235680
weighted avg       0.99      0.99      0.99    235680
```

Parser

English

LAS: 0.9492089925062448 UAS: 0.9685863874345549

Hindi

LAS: 0.9090909090909091 UAS: 0.9162303664921466

Sanskrit

LAS: 0.4508990318118949 UAS: 0.5780141843971631

Analysis

POS Tagger

English

We note from both English POS taggers that the recall of adverbs and numbers is low. This means that the model was unable to identify a significant fraction of adverbs and numbers.

One factor common to both these parts of speech is the relatively small number of samples available – 76 for adverbs and 127 for numbers (as compared to, say, 995 for nouns and 1434 for adpositions). However, interjections and particles also have comparably low counts.

One possible reason for this might be that interjections and particles can be decided *lexically*. In other words, the model can decide if a word is an interjection or a particle from the word itself, with no knowledge of the context, which makes it easy. For example, `oh` is always an interjection, regardless of the surrounding words.
On the other hand, adverbs cannot be decided so easily. For example, *good* and *early* may be adverbs or adjectives depending on how they are used. Now the number of samples of adjectives is greater, which may bias the model towards classifying ambiguous cases as adjectives.

As a side effect of this, we expect the precision of adjectives to drop (due to the adverbs wrongly classified as adjectives). This is in fact what we observe, with the precision of adverbs being the next lowest value in the above table.

Cross-Linguistic

The Hindi POS tagger (compared to English) achieves comparable, but not equal results. Many parts of speech have low recall – adverbs, numbers and adjectives among them. In addition, proper nouns have low precision as well.

The Sanskrit POS tagger performs poorly on all parts of speech except precision of conjunctions, determiners, particles and pronouns, and the recall of conjunctions, particles and pronouns.
All these parts of speech constitute function words, which means that lexical cues enable us to identify them.
Sanskrit's morphology is such that a single word may double as an adverb, adjective, or noun. Thus, these parts of speech are hard to identify. The model seems to be biased towards marking them as adverbs, giving them high precision. The lack of word-order cues may also be a factor in the poor tagging.

Parser

The model performs the best on English, achieving an attachment score of approximately 95% (both labelled and unlabelled). We hypothesis that this is because of English's fixed word order and lack of morphological complexity.

Our justification for this explanation lies in the attachment scores of Hindi (which has a slightly freer word order and greater morphological complexity) and Sanskrit (which has a much freer word order and greater morphological complexity). The model achieves approximately 90% attachment in Hindi, but is unable to cross 60% (unlabelled) in Sanskrit.

The model trained on Mandarin, notably, performs equally poorly, achieving no more than 59% unlabelled attachment. We were unable to explain this drop in performance on a language even more fixed word-order and isolating than English.

Overall Model

In an attempt towards interpretability, we attempted to find how important the POS tagger was to the overall model. We partially trained the POS tagger and then recorded the unlabelled attachment score of an edge-scorer trained on the “bad” POS tagger. Our results are as follows:

POS Tagger Accuracy	UAS	Num_Epochs
15.2%	90.8%	20
52.7%	90.2%	24
81.1%	89.6%	24
86.8%	90.4%	16
92%	90.3%	11
96.5%	90.9%	14

Surprisingly, the accuracy of the POS tagger does not significantly affect the accuracy of the edge scorer. However, it does seem to affect how fast it trains.

A natural question to ask now is whether the inaccurate POS tagger’s outputs are even used by the model somehow, or completely ignored. To test this, we carried out two experiments: we ran the edge scorer *without* a POS tagger, and we replaced the bad POS tagger of a trained edge scorer with a good one.

The edge scorer without a POS tagger achieved 90.9% UAS. This led us to believe that the inaccurate POS taggers’ results are being ignored by the previous edge scorers.
However, an edge scorer trained with a 13.2%-accurate POS tagger achieved 91.5% accuracy. When this POS tagger was replaced with a 98.6%-accurate one, its performance dropped to 68.1%.

We were then forced to the conclusion that the model somehow learns to make use of the POS tagger’s poor predictions – in a sense, the POS tagger has its own syntactic “theory”, which allows it to achieve comparable results to ours in terms of dependency parsing.