Conference handbook

Natural Language Processing (2021)

G01

The project compares three different stack-based systems for dependency parsing: arc-standard with a static oracle (baseline), arc-eager with a static oracle, and arc-eager with a dynamic oracle. To compare the systems, tests were run on nine different Indo-European languages. The languages were chosen since they all belonged to different branches in the Indo-European language tree. The reason for only using Indo-European languages was to see if the defining characteristics of a language family would correlate to the performance of a parser. The tests measured the unlabeled attachment score (UAS) for each system over each language. The UAS was then used when attempting to find correlations between the performance of a system and the structure of the language. The results showed that there were no clear patterns between the performance of a given parser for each language family. Arc-eager with a dynamic oracle outperformed the arc-eager with a static oracle in all cases but one. In addition, arc-eager with a dynamic oracle showed comparable results to the baseline system, of which, arc-eager with a dynamic oracle outperformed the baseline on five out of the nine languages.

G02

The dependency parser baseline was extended to include labels for the dependency arcs. A variant parser using the arc-eager algorithm rather than the arc-standard was also implemented, and a comparison was made between these two parsers for both labeled and unlabeled arcs. The comparison included the number of transitions made and the unlabeled attachment scores for both the labeled and the unlabeled arcs, as well as the labeled attachment scores and the label accuracy for the labeled arcs. In the end both implementations give very similar results both on unlabeled and labeled arcs, but the arc-eager implementation makes noticeably fewer transitions than the arc-standard variant.

G03

Machine Reading Comprehension (MRC) is a central problem in natural language understanding. In this paper, we study the multi-choice version of this problem, where the machine is given a passage of text, and is tasked with selecting one of several options that best answers a given question. The current state of the art solutions are based on pre-trained language models. We use the BERT pretrained model as a base, and implement three different final stages used for classification; Dual Multi-head Attention (DUMA), Multistep Attention (MA), and a simple linear classifier (LC) as a baseline. We use the RACE dataset for training and validation, and we additionally evaluate the model performance on reading comprehension questions from Högskoleprovet (Swedish SAT equivalent) in both Swedish (LÄS) and English (ELF). To handle Swedish texts, we use the multilingual BERT model fine-tuned on the RACE dataset. Our best model for each dataset achieved 64% accuracy on the RACE dataset, 54% accuracy on ELF, and 31% on LÄS, respectively. For reference, the average human accuracy on both ELF and LÄS are around 48%. Our results indicate that multilingual zero-shot transfer learning might be possible, but a more extensive model or training regime is most likely required.

G05

Our project is an extension of lab 3 and lab 4 where we added labelling to our dependency parser. We evaluated our implementation by comparing the UAS and LAS on different features and languages. The languages we tested were English, Swedish, German and Finnish. The results we observed in our data was that 3 words and 3 tags were close to the optimal number of features to use in this implementation. Adding more lookback features only negligibly improved the accuracy and removing features drastically reduced the accuracy. We could not draw any clear conclusions regarding the reasons for the observed differences among the languages.

G06

In this project, we have implemented a neural conversational model using sequence to sequence learning. Conversational models have many applications, e.g. chatbots, auto-completing sentences and data aggregation. We base our project on the 2015 paper "A Neural Conversational Model" in which an encoder-decoder architecture with LSTM cells is used. Due to time constraints and limited computational power, we were unable to replicate these results. By instead using a shallower model trained on a smaller dataset, with various modifications such as attention and bidirectionality,

we achieve a model that converses relatively well. Responses tend to be of a more general nature, but can in many cases display relevance to the input sentence from the user. The replies from the model are coherent and mostly grammatically correct. One difficulty was evaluating the results of a model, as there is no single correct answer in a conversation. We mostly relied on human evaluation, though most times it seemed to correlate well with the perplexity and loss calculated with respect to the target answers. We also included BLEURT as an evaluation metric which is often used while evaluating translations.

G07

In our project on Language translation using the transformer model, we have managed to do the following implementations: preprocessing the Europarl SV-EN dataset; Positional Encoding - to encode position of word in a sentence along with the word embedding; Scaled dot product for multi head attention (to get the context and attention score); Multihead attention that takes the splitted projection of the Query, key and values; apply the scale dot product and FFN layer; Implements the self Attention mechanism; Next, we define the encoder architecture for input english sentence. And similarly, we also create the decoder architecture for our target swedish sentence along with padding & look ahead mask; Finally, we combine the whole architecture to build a transformer model with our hyperparameters; TODO...Training, evaluation and testing.

G08

The relationship between words in a text is crucial to gain an understanding of the text's meaning. To do this we've created a pipeline that first tags each word in the sentence with a part-of-speech-tag and then predicts the relationship between the words through those tags. Our parsing component is based on transition-based parsing. To improve performance we've followed Vaswani & Sagae's (2016) work on a similar project. On top of the basic baseline we started with, we added expressive features for our parser and error states for our classifier. We train our local classifier on the error states class which are then used for global scoring just like in the research literature. The error states provide information about wrong path derivations. Using best-first search in our decoder and neural networks for our classifier, we produced an average UAS score of 73.28%. Our result suggests that the error states not only improve the search accuracy, but also improve the accuracy of the local classifier itself, and the expressive feature increases the UAS significantly. The best first search is

slightly worse than beam search with proper beam size on our dataset, which gives rise to interesting discussions on why.

G09

Mapping a sentence to a formal representation of its syntactic structure is an interesting task. One of the approaches for solving this task can be transition-based dependency parsing. In this research project this type of parser was used together with part-of-speech-tagger, a beam-search algorithm and a neural network to predict the dependency tree. By introducing a new parser state for incorrect partial dependency trees, it was possible to add information about inaccurate derivation paths. This is often not taken into account in locally-trained models, but necessary when decoding in a global manner using beam-search. Together with new model input features, the parser scored approximately 74% accuracy for transition-based dependency parsing.

G10

We have extended a transition-based parser with error states and beam search. In addition to training the model to choose correct moves from certain features we also trained the model to recognize when it has made an incorrect action. This means that the model learns to associate features as error after a faulty move, according to gold standard. The model calculates separate losses for the gold standard and the error states and are then weighted according to their importance using a hyperparameter alpha: To generate the model with the highest Unlabeled Attachment Score (UAS), different beam widths between one and eight have been tested. We saw minor improvements when increasing the beam width. The largest increase was when the beam width was increased from one to two. When increasing the beam width further than four we saw a minimal increase in most cases, but in certain scenarios we had a worse UAS. We achieved the best UAS when using alpha equals 0.95 and a beam width of 4. This extended parser resulted in a ~5% UAS increase.

G11

We present a parser-tagger pipeline for dependency parsing, mapping a sentence to a formal representation of its syntactic structure. The Universal Dependencies english GUM treebank is used for training and evaluation. Part-of-speech is shown to be an important feature for dependency parsing. Hence, the part-of-speech tagger plays an essential part in the pipeline. The tagger model is built with stacked Bidirectional

LSTMs with pre-trained word embeddings. The presented tagger achieves a part-of-speech accuracy of 91.5%. The parser builds upon a transition-based approach, using neural networks based on BiLSTM as local classifiers. Sentence tokens are identified with a BiLSTM vector characterizing tokens in their context. By combining some of the Bi-LSTM vectors, feature vectors are constructed. Furthermore, error states are included in local training, adding information about faulty derivation paths. Beam search is applied to minimize the downsides of greedy local search. The combined tagger-parser model achieves an accuracy of 79.1%.

G12

In the project we have implemented additional features that is used by the fixed-window parser subsystem of the baseline. We call this new and improved system "Prosper". The new features are Distance, Left and Right Valency, Base Word, Left and Right Unigram and Number. Quite some effort was put into streamlining the process of adding a new feature and specifying its usage (embedding specifications etc.). As just implementing new features was not necessarily sufficient for the project's scope, we also evaluated all our new features on several different languages, some within the same language family (e.g., Germanic) and some in different families. Our result was that Left Valency provided the greatest relative improvement for almost all the languages we evaluated Prosper on. Left Unigram also made an improvement. The Base Word feature got us worse results for some languages and never provided a notable improvement. For Czech, Right Valency and Right Unigram was better than their left counterparts, being an exception to the other languages. Live long and Prosper.

G13

Before the introduction of Transformers, most state-of-the-art NLP systems relied on gated recurrent neural networks (RNNs), such as LSTMs and gated recurrent units (GRUs), with added attention mechanisms. The Transformer built on these attention technologies without using an RNN structure, highlighting the fact that the attention mechanisms alone, without recurrent sequential processing, are powerful enough to achieve the performance of RNNs with attention. In our project we implemented a Transformer model as a baseline. We trained six different translators with OPUS datasets. We showed that this baseline already achieves higher performance than the RNN that was trained during the lab sessions. To improve the Transformer model, we implemented two different methods: TeaForN (Teacher-Forcing with N-

Grams) and MUTE (Transformers with multiple parallel units). In TeaForN models, N decoders are stacked so that the Transformer can not only learn from the ground truth but also from previous predictions. We achieved an improvement on some of the datasets. MUTE models use multiple parallel units in the encoder layers. This method improved the performance on big datasets. In summary, both methods show potential to improve the Transformer but further research needs to be done.

G14

The aim of this project was to study a transition-based dependency parser using an arc-standard algorithm. In particular we studied how we can improve the unlabelled attachment score by extending the feature representation of our baseline. Our baseline consisted of a part-of-speech tagger that was pipelined together with the transition-based dependency parser. This project extends on previous work on adding additional features to a dependency parser using an arc-eager approach (Zhang, Nivre, 2011). In this project we show that we can improve the unlabelled attachment score by adding additional features to the baseline system. Using the English treebank from the Universal dependencies project we improved the score from 63.25% to 64.24% using additional features. Using the Chinese treebank bank we improved the score from 55.7% to 59.2%.

G15

In this project we performed a comparative study of parsing accuracies of our baseline parser, arc standard algorithm and various algorithm of maltparser package on Swedish and English treebanks. Malt parser is a system of several parsing algorithms based on data-driven dependency parsing. We tried several combinations of parsing and learning algorithms available in the malt parser package along with features optimization. Algorithms in the Malt parser package gave significantly higher accuracy even in the default mode than the baseline parser. We selected two algorithms with highest accuracy score from the malt parser package and performed feature optimization to get better accuracy score. We got a slight improvement in accuracy of about 0.5% in the both the Swedish and English databanks.

G16

In this project, we implement a model for dependency parsing using bidirectional LSTMs, also called BiLSTSMs. Initial input vectors to the BiLSTM are made by

creating embeddings for words and their respective part-of-speech tags and then concatenating these. The words are then encoded in regard to its sentential context by using a BiLSTM and representing them as deep BiLSTM vectors. Feature vectors are then assembled by concatenating a selection of these BiLSTM vectors which are fed into our linear model, a multilayer perceptron (MLP). The output scores from the MLP are then used to determine the next configuration. A greedy transition-based parser is used that we train on datasets in English, Arabic, Spanish and Swedish. The results from this model are significantly improved from the original baseline which utilizes a fixed window parsing model.

G17

The purpose of dependency parsing is to find the asymmetric relation between words in a sequence, often represented as dependency trees. In the lab portion of the course, we have implemented a parser that extracts the arcs of these dependency trees. In our work, we expand the implemented parser, to also be able to show the grammatical relation labels between the words connected by each arc. We present results from two different implementation approaches to adding these grammatical labels; the first where additional moves corresponding to the combination of type of arc and grammatical relation gets added to the parser, and the second where an additional classifier is added to predict the grammatical relation of the already parsed arcs. Our experiments show that both approaches are able to produce acceptable UAS of 0.68 and 0.72 respectively. However, our implementation of the first approach struggles to achieve LAS scores higher than 0.13, while the second approach is able to produce a much more convincing LAS of 0.62. Our presentation will provide a discussion on the different approaches, and what might have led to the discrepancy of the results.

G18

This project extended the baseline implementation of lab4 to optimize the UAS-score on the English Web Treebank (EWT) and then compare how well the parser performs on 6 other treebanks. We extended the earlier implementation by including more syntactic information from the training data and including the new data in the featurize- and oracle function. Previously researched features with increased performance were implemented. These are distance between a word and it's head (if there's an arc-action), unigram-information as the word form of a parentwords´ left- and rightmost children, the lemma and dependency relation of the first word on the buffer and top two words on the stack and the previous action taken by the parser. The

parser achieved an 85.71% UAS-score with 1 epoch compared to increasing epochs to 4 which achieves 84,48% UAS-score. It's difficult to say how well the parser is generalizing as it is dependent on information from the treebanks. Missing data results in very sparse feature vectors which in turn adds noise to often smaller datasets than EWT. The highest achieved UAS-score of the scandinavian languages was the Swedish score at 82,08% while Danish (small treebank with the baseline information) scored 60.57%.

G19

In our project, we are aiming to extend the toolkit of NLP methods that are capable of detecting social inequalities in textual data. This project seeks to enhance the analytical power of word embeddings with dependency relations among words in order to classify sentences. Applying this idea to the case of gender biases, we want to test the following hypothesis: Men are more likely to appear as subjects in a sentence, while women are more likely to be sentence's objects. To test this hypothesis, we have solved two crucial tasks: First, we have created a gold-standard dataset from scratch and annotated 450 Wikipedia biographies with gender labels. Second, we have trained an LSTM-based sentiment classifier on this dataset so that we are able to predict the genders of unseen text. The baseline classifier, developed by Komninos and Manandhar (2016) has been modified to perform better on the task at hand and makes use of GloVe embeddings. Our experiments show that the classifier reaches up to 84% accuracy (combined) on the validation dataset and is able to predict genders across various text forms reliably, which we have cross-validated on a dataset of State of the Union texts.