Conference handbook

Natural Language Processing (2022)

PG01: Project BAGEL

This project extended the baseline tagger/parser system introduced in labs three and four. This was done by adding two additional features to the system. The first additional feature was 'valency' in which the number of right and left arcs to a each word was counted and included in the prediction process. The second feature was 'distance' which is the distance between the arced words in the input sentence. The model were then tested on Arabic, English, Swedish and Finish treebanks in order to encapsulate a variety of language families. For each treebank, we measured the results of the powerset of the two features on top of the baseline. For evaluation accuracy, unlabeled attachment score (UAS) and unlabeled exact match (UEM) score were used to measure the system's performance and allow for further evaluation of results. The results showed that valency had the largest impact of increasing UAS, and the increase given by distance followed relatively closely. Combining the two features gave a further increase in UAS, however this was only a slight increase when to the UAS given by only the valency feature. We found that Germanic languages English and Swedish performaned significantly better Arabic (Semitic) and Finish (Uralic).

PG02: A Customized BERT Model for Dependency Parsing

For our project we replaced the syntactic parser in the baseline system based on lab 3 and 4, with a fine-tuned pretrained BERT-model, combined with a biaffine attention layer on top of the BERT model. This architecture was based on a paper by Glavaš and Vulić [1]. This was done to customize the BERT model for the task of dependency parsing. More specifically, we first tokenized and passed sentences to the BERT model. Then, to get word-level embeddings, we averaged subword embeddings of the output from the pretrained model for words that were divided during tokenization. Subsequently, we divided the embeddings into two parts X and X´. X consists of the embeddings for all words and X´ consists of the embeddings for all words plus the root embedding. One can see X as the dependents and X´ as the heads. These were then passed into a biaffine attention layer which gave the predictions of heads for each

word in the sentence. With the baseline system we got a tagger accuracy of 0.8944 and a parser UAS of 0.6612. With the extended parser we got a UAS of 0.7817.

PG03: Feature Engineering for a Projective Treebank Dependency Parser

The performance of a tagger and parser can be impacted and increased by adding different features. Our project developed several features to improve the baseline which is a combination of the tagger from lab 3 and the parser from lab 4. The features were developed and tested in isolation of other new features to determine the impact each individual feature had on the baseline. Further, the features that improved the baseline were combined and tested together to see the final improvement of the UAS measured on the development set created from the english web treebank. The combined new features resulted in an increase of UAS from 67.3% to 76.3%. For the parser we found the biggest impact from a new feature to be the word form of the first child to the left of the first word on the stack. For the tagger the endings of the surrounding words helped out-of vocabulary examples, and increased the accuracy the most.

PG04: Transfer Learning – An Effective Shortcut in Machine Translations?

The project aims to explore and make use of transfer learning as a tool to bridge machine translations of a low resource language pair. To achieve this, a series of bilingual and a series of monolingual sequence-to-sequence models were created and trained on the Europarl and IATE datasets. To produce synthetic low resource language data, of progressively increasing quality, a large dataset was reduced by a fixed percentage of the large dataset as training data. The bilingual model was trained on a full dataset of a high resource language together with the synthetically produced low resource language. The monolingual model was trained purely on the synthetically produced low resource language. The models were then used to produce a translation of the WMT-news dataset, and the performance of the models were compared by the BLEU scores of the thereby produced translations. In the experiment, it is shown that the transfer model performs worse on all synthetically produced datasets when compared to the monolingual model. Furthermore, it is shown that a bilingual model performs better on closely related languages.

PG05: Dependency Parsing using Bidirectional LSTM

The project changed the dependency parser in the baseline from a fixed window model to a bidirectional LSTM (biLSTM). First, the words and their respective part-of-speech tags are embedded and then concatenated. After this, the words are encoded to create the deep biLSTM vectors, regarding each word to its context. The features for each move are picked out. This includes the next buffer word, topmost-, second topmost-, and third topmost word in the stack. These vectors are concatenated and sent to a multi-layer perceptron(MLP). The MLP consists of three linear layers that reduce the size of the vector to three. Which represents the scores for each of the three moves the parser can perform. These scores in conjunction with the oracle for the valid moves determine which move is performed. The results from our pipeline were not as good as expected. The final results were not better than the original results from the baseline and we, unfortunately, did not find the cause of this problem.

PG06: Syntactic Parser with Extended Features

The purpose of the project work was to first implement the tagger-parser pipeline and extend the features for the parser to include the set of unique dependency tags for each given sentence. The parser would then be trained on retagged data provided by the tagger and the UAS score was thereafter evaluated and compared to the baseline parser. During the project work the group also implemented valency into the list of extended features to the parser. Though both features were originally implemented in models that did not make use of neural networks, they were altered in such a way that made it possible to implement into the current parser model. Experiments were conducted on several languages, where each language was run on the parser 10 consecutive times where each individual run consisted of all combinations of the newly implemented features. The languages in question were: English, German, Italien, Spanish and Japanese. One experiment on transfer learning was also conducted, where the parser was first trained on Italian and then was evaluated on the Spanish dataset. In general, the results show that every language had an UAS-increase for valency and dependency set individually but a lower UAS-increase than expected when both features were used.

PG07: Sentiment of COVID-19 Vaccine

In this project, we used a NLP model to determine if a tweet is positive, negative or neutral with regards to the COVID-19 vaccine. At the time of writing and executing

this project, we are at the end of the restrictions period of the ongoing pandemic so this is a highly relevant topic. At first, we collected tweets mentioning the vaccine using the Twitter developer API which we then annotated with the three sentiments positive, negative or neutral. Our model was based on a BERT model imported from the Hugging Face library. The BERT model consists of multiple encoders which makes it "understand" text data. The BERT model was pre-trained on large amounts of English text data using masked language modelling and next sentence prediction. To make the model perform sentiment classification on COVID-19 vaccine tweets we fine-tuned the model using our hand annotated data. Our baseline implementation was a naive Bayes classifier which achieved an accuracy of 68,9%. Our BERT model achieved an accuracy score of 78,4%.

PG08: Model Training Efficiency Analysis

With deep learning taking the world by storm, the demand for highly parallel computing has increased drastically. This is because the time consuming task of training a model can be alleviated through the parallelization of the task. However, even with the use of GPUs and TPUs, training is still time consuming. Also, it requires a lot of energy, which can affect the environment poorly. For this project, we compared two Reverse Dictionaries, both based on using multiple channels for evaluation, the first is utilising a BiLSTM, while the other utilises a RoBERTa. With these two models, we compared how quickly they increased their top 1, 10, and 100 accuracies during training, and their power consumption during the process. We used random samples corresponding to 2% of a training and a development dataset during training. For testing, we used 100% of seen, unseen, and human description datasets. These datasets were provided by the projects related to the models. We found that training the RoBERTa model on only 2% of the data gave us similar results to training a BiLSTM model on 100% of the data. Also, the returns for training RoBERTa with more than 2% of the data were quickly diminishing.

PG09: Increasing UAS using Best-First Search and Error States in Transition-Based Parsing

In our project, we use the tagger-parser system the same as in the lab sessions. However, in the greedy arc-standard parser we implemented in the labs, we only consider a single path, as each decision for a transition is irremediable. In our project, we use the best-first search and thus explore all the paths that could have been taken with

the three different transitions at each node. To reach a higher accuracy, we add a new class called error state, which provides the probability that an error has occurred previously and that the current state belongs to an incorrect derivation path. We use a priority queue to store the score of each state as the product of probabilities of each move along the movement tree. The highest scoring state is chosen so the transition can be driven. Process is repeated without clearing the queue (unexplored new states) until the final state. In our initial experiment, the model without error state has an unlabelled attachment score of 66.49% on the English Web Treebank development dataset. With the help of error state, the modified model can reach a slightly higher accuracy of 66.9%.

PG10: Unsupervised Grammatical Error Correction Using Language Models

Grammatical Error Correction (GEC), the task of correcting errors in text, is a central problem in natural language processing (NLP). It is a useful tool for avoiding ambiguity and making written communication better and more efficient. A few years ago, the main approaches of this problem were statistical machine translation (SMT) and classifier-based approaches, but they rely heavily on large volumes of labeled training data, requiring a substantial amount of manual work for any new application area. In recent times, however, Language Model (LM) based approaches, with practically no required labeled data, have gathered more interest. This project investigates the performance and trade-offs of different language models in the context of GEC. We investigate the performance of the basic model fixed window N-Gram, and we also investigate the effectiveness of the more complex pre-trained transformer model DistilBert. We compare our result on a true and tested, freely available baseline model KenLM. Furthermore, we evaluate how different sizes of the models affect the performance. Evaluations are done using M2 F0.5, the "de facto standard of GEC evaluation".

PG11: Feature Comparison for a Transition-Based Dependency Parser

In this project, we enhanced our baseline system by adding additional features to the dependency parser. The features added was, distance, left and right arc valency, unigrams and third order. These features was presented in Transition-based Dependency Parsing with Rich Non-local Features (Zhang, Nivre, 2011), we used paper as a guide for the implementation of our features into our baseline. The features was evaluated

based on unlabeled attachment score (uas) and unlabeled exact match (uem). We ran 20 evaluations for each feature to minimize the impact of the random initial values and get an accurate view of the features impact on the baseline. Furthermore, the parser trained on both the English and Chinese treebank, and both saw improvements in both uas and uem respectively. The combined result of the features improved upon the uas of the baseline by 11% (to a uas score of 78%) and the uem by 31% (to a uem score of 44%).

PG12: Deterministic Parser with a Dynamic Oracle using Arc-Hybrid

A pipeline was created with a part of speech tagger and a dependency parser using neural networks. The dependency parser was first trained using a static oracle and the arc-standard and arc-hybrid algorithms respectively. Afterwards, the parser was altered to use a dynamic oracle and the arc-hybrid algorithm. The static oracle had golden arcs and a configuration as parameters and returned a single transition that, for the given transition, would not eliminate the creation of arcs from the golden data. The dynamic oracle instead returns all transitions that do not result in eliminations of golden arcs. The pipeline was tested on English, Swedish, Chinese and Latin. For the English and Latin languages there was a significant increase in switching from arc-standard to arc-hybrid and for the other languages there were negligible results. Using a dynamic oracle decreased or barely changed the performance but the training time became five times longer.

PG14: Sentence Parsing using Arc-Hybrid and Dynamic Oracles

The goal of this project is to see how a transition-based syntactic parser could be improved by switching out of static to a dynamic oracle for training by comparing them and also employing alternative transition systems. By employing a dynamic oracle to see the impacts of teacher forcing, we train the arc-hybrid parser without using teacher imitation learning. The arc-standard algorithm is used to create dependency trees, which take a sentence as input and return a dependency tree in the form of a series of directed arcs linking a word to another word. Goldberg and Nirve's algorithms were used as the base for training with exploration. The evaluation of the transition system had done by using UAS and LAS scores. The dynamic oracle could not give a big performance in most languages as well there was a big difference in the results between languages, but the hybrid parsers perform better results than standard parsers.

PG15: Baseline UAS Improvement by Adding New Features

Our project was an extension of the standard project which consists on a tagger-parser pipeline. The tagger first assign a predicted Part Of Speech tag for each word, and then the parser predicts the heads of all words in the sentence with the predicted tags. The purpose of our project was to find new ways or new features that could improve the UAS score. We tested our program on ten different UD treebanks in English, Chinese and French. To improve the program we tried to add three different types of new features : the distance between a head and its dependent, the valency (which corresponds to the number of dependents for a given head), and combine the unique dependency labels with the other features. If the distance and the valency had a negligible impact on the unlabeled attachment score, we got a surprisingly very high score for using dependency labels for the two topmost words on stack and the next word in queue. Indeed, for a tagger accuracy of around 89.5% on average on all the treebanks we tested, we got an average UAS of 86.6% for the English data, 87% for the Chinese data and 81.2% for French, against respectively 63.7%, 59% and 64.1% with the initial parser with basic features. We also wanted to implement a neural network along with a beam-search algorithm and error states, but we did not have the time to make it functional. Using pre-trained embeddings as well did not improve the program accuracy / UAS. Finally, we tried various combinations of the initial parser features with the new ones, but the best result we get was always the 9-features implementation : three for the tag, word form and dependency label for the three considered words previously mentioned.

PG16: Automated Selection of Quotes using RNN

In Computer Science Education qualitative methods are sometimes utilized in research. These are often labour intensive as they can include collecting relevant quotes from transcribed interviews. This project evaluates the viability of using machine learning to aid researchers in the selection of relevant quotes. In this project we evaluate a system based on word embeddings generated through skip-gram model with negative sampling. We used the first 1M sentences from Swedish Wikipedia for this. These embeddings were then used to initialize an embedding layer in an RNN-model. The RNN-model further consists of an LSTM-layer followed by a linear layer that reduces the hidden dimension to a binary prediction of whether to include a sentence or not. This RNN-model is trained using half the interviews. All of the interviews used to train the model were conducted by two of the three interviewers at one of the two universities. We then evaluated the model by feeding it interviews conducted by

the third interviewer at the other institution. Our evaluation indicates that the system based on the RNN-model outperforms the baseline, particularly in that it produces no false negatives. Not missing relevant data is crucial in this application.