EDAN20 Language Technology
-Lab 4.

Nils Romanus

September 2022

1 Extracting syntactic groups using machine-learning techniques

Synthetic groups can be extracted using a neural network architecture consisting of one embedding layer, one LTSM module finished of by a dense logistic classifier.

1.1 Creating Embeddings and Features

The embedding matrix can be created unioning the predetermined glove embeddings with randomly initialized embeddings for the words that are not present in the glove embeddings dictionary. The input words can subsequently be mapped to numbers by simply creating creating an index where every word is mapped to a number. The same goes for the chunks where every chunk tag is transformed to a number. The sequences can subsequently be padded with zeros in order to fit the network architecture. Furthermore one may use the convention of letting a one correspond to an unknown word. An embedding matrix can finally be created by assigning each index of the matrix its corresponding glove embedding or a random initialization for the out-of-sample embeddings.

1.2 Network Architecture

The chosen network architecture consists of one embedding layer, one recurrent network module (or an improved counterpart) and a final dense layer with a standard multiclass softmax activation. The reccurrent module can be an RNN or an LTSM set up uni- or bidirectionally. All of the architectures that were tried, coupled with the architecture used in the paper, are listed in figure 1. A bidirectional LTSM, using 256 units with a 0.5 dropout, a batch size of 32, and that was trained for 15 epochs, performed the best as shown below. Unidirectional RNN and LTSM architecture was also tried however these performed worse in terms of chunker F1 score.

Method	Parameters	Score
Baseline	selecting the chunk tag which was most frequently associated with the current part-of-speech tag	0.771
RNN	hidden_units=256, dropout=0.5	0.832
RNN	hidden_units=64, dropout=0.5	0.785
LSTM	hidden_units=256, dropout=0.5	0.859
LSTM	hidden_units=64, dropout=0.5	0.811
BiLSTM	hidden_units=256, dropout=0.5	0.91
Akbik et al.	BiLSTM	0.967

Figure 1: Test chunker F1 score from experiments using different architectures.

2 Contextual String Embeddings for Sequence Labeling

The architecure in the paper is, in a sense, similar to the one that was used in the lab. It also uses an embedding layer whose output is fed to a recurrent bidirectional LTSM module. The main difference is that their embedding is not constructed from using pre-trained glove embedding unioned with random ones. Instead the embedding layer consists of a pre-trained LSTM structure with the characters as input. The pretrained LSTM structure constitutes a mapping that is used as embeddings for the subsequent BiLSTM-CRF sequence labeler fulfilling the same function as our glove embeddings but with increased performance. The main advantage of using this custom character based embedding is that the model handles pre- and suffixes in a more advantageous way. As shown in the paper they also choose to concatenate several types of different embeddings, including constant ones similar to our glove embedding. Using multiple concatenated embeddings is probably increases robustness and generates superior results as shown in table 2 in the paper. The bidirectional LSTM-CRF also differs from our simple bidirectional LSTM. Our LTSM module does not include a conditional random field (CRF) layer. The CRF component adds a Bayesian component on top of the LTSM which enables it to better leverage the intrinsic relationship between the sequence of outputs of the LTSM.