CoNLL-2003 task

Documentation

|  |  |
| --- | --- |
| Upasana Parashar | May 19th, 2020 |

# **Data**

The CoNLL-2003 data contains a training file, a validation(development) file and a testing file. The development data has been used for parameter tuning. The testing file contains data which is completely unseen by the model. It has been used to assess the performance of the models.

## Pre-processing

The data supplied for this work had some linguistic pre-processing already done. A tokenizer, a part-of-speech tagger, a chunker were already applied. The Named Entity tagging was also done for all the three datasets.

## Data Format

The data files contain one line per word. Each document starts with the line “-DOCSTART- -X- -X- O”. Each empty line implies the end of a sentence. Each line contains four values, i.e. the word, the part-of-speech tag, the chunk tag and the name d entity tag. More information can be found in reference paper.

# **Data Preparation**

The raw files could not be used as is for modelling purposes. The data from each data file was converted to 4 separate lists of sentences.

Example:

['EU', 'rejects', 'German', 'call', 'to', 'boycott', 'British', 'lamb', '.']

['NNP', 'VBZ', 'JJ', 'NN', 'TO', 'VB', 'JJ', 'NN', '.']

['I-NP', 'I-VP', 'I-NP', 'I-NP', 'I-VP', 'I-VP', 'I-NP', 'I-NP', 'O']

['I-ORG', 'O', 'I-MISC', 'O', 'O', 'O', 'I-MISC', 'O', 'O']

EU NNP I-NP I-ORG

rejects VBZ I-VP O

German JJ I-NP I-MISC

call NN I-NP O

to TO I-VP O

boycott VB I-VP O

British JJ I-NP I-MISC

lamb NN I-NP O

. . O O

## Vocabulary

A dictionary was created with all the vocabulary present in the training data. Each word and each tag were assigned unique integer values.

Example: Below is the POS dictionary.

{'padding': 0,

'I-VP': 1,

'I-SBAR': 2,

'I-CONJP': 3,

'I-INTJ': 4,

'B-PP': 5,

'I-PP': 6,

'B-ADVP': 7,

'I-ADVP': 8,

'B-NP': 9,

'I-PRT': 10,

'O': 11,

'I-LST': 12,

'B-ADJP': 13,

'I-NP': 14,

'B-VP': 15,

'B-SBAR': 16,

'I-ADJP': 17}

**Note:** A key called “padding” has been added to the above vocabulary to tag the padding. Apart from this, a key called “unknown” has been added to the vocabulary of words. This has been done to ensure that if there’s a word in the development or testing set which is not present in the training data, it can be tagged as well.

## Vectorisation

Each word in the list of sentences is vectorised using the word vocabulary. Similarly, each tag ins the list of tags corresponding to the sentences are vectorised using the tag vocabulary.

Example:

['EU', 'rejects', 'German', 'call', 'to', 'boycott', 'British', 'lamb', '.']

Transformed array:

[18542, 17535, 7077, 14281, 1493, 19995, 2871, 970, 11006]

Similarly, the respective tags were also vectorised.

Example:

['I-NP', 'I-VP', 'I-NP', 'I-NP', 'I-VP', 'I-VP', 'I-NP', 'I-NP', 'O']

Transformed array:

[14, 1, 14, 14, 1, 1, 14, 14, 11]

## Padding

To prepare the data for modelling using Keras, the length of all sentences should be equal. To ensure that, the average length of sentences was calculated for the training set. Majority of the sentences have the length of around 50 or lower.

Therefore, all the sentences which were of length lower than 50 were padded with zeroes (or the word “padding”) and those which were longer were trimmed to length 50. The same was done to the sequences of tags as well.

## One-hot encoding

One hot encoding has been done on the padded sequences of tags as the final step to prepare the data for modelling

# **Model Architecture**

A Bi-LSTM model has been used for modelling.

* An embedding layer has been added to learn the embeddings along with the model training. These embeddings are specifically trained for the training data considered.
* A Bi directional LSTM layer has been added to the model to preserve the information from the previous and well as next units.
* This is followed by a time-distributed fully connected layer. This layer picks the corresponding class for the sequence.

Adam has been used as an optimizer for this model as it tends to perform well. The loss function used for this model is “categorical cross entropy”, as this is a classification model.,

# **Model Evaluation**

The model has been trained using the prepared training data, and it has been modified using the processed development set. While the training accuracy is extremely high, a lot of it can be attributed to the padding. So, for final model evaluation, padding has been removed.

The above steps have been followed for all the three tags coupled with the word sequences.

For model evaluation, the following metrics were used:

1. Precision
2. Recall
3. F1 score
4. Accuracy

The model diagnostics can be found along with the code in the uploaded BitBucket repository.

The Results section has been intentionally skipped in this document.

# **Discussions and Future Works**

Here are some of those points which could also be explored

1. The words have not been converted to lower case before creating a vocabulary. So, if a word is present in a different case in the training data from that of the test data, the word would be considered as “unknown”. A system where either the dictionary is created with all words in lower case, or where if a word is not present in the vocabulary, other cases of the word can be looked for in the dictionary before marking it as “unknown” can be explored.
2. Intuitively, if the number of classes could be reduced or if some classes could be merged, the performance of the model for each individual class could be improved.
3. There is a strong problem of class imbalance in the data. This could be addressed to improve the performance of individual classes.
4. Other model architectures can also be explored
5. Pre-trained embeddings could be used instead of training the word embeddings along with the data. This could also improve the model.
6. Pre-trained models could also be used classification of tags.

# **References**

Data: <https://github.com/patverga/torch-ner-nlp-from-scratch/tree/master/data/conll2003>

1. Sang, Erik F. Tjong Kim & Meulder, Fien De, “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”, Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003
2. Collobert, Ronan & Weston, Jason & Bottou, Leon & Karlen, Michael & Kavukcuoglu, Koray & Kuksa, Pavel. (2011). Natural Language Processing (Almost) from Scratch. Computing Research Repository - CORR. 12.
3. “An overview of gradient descent optimization algorithms”, by Sebastian Ruder, as on May 19th, 2020.   
   Link: <https://ruder.io/optimizing-gradient-descent/index.html#adam>
4. “Build a POS tagger with an LSTM using Keras” by bogdani, as on May 19th, 2020.   
   Link: <https://nlpforhackers.io/lstm-pos-tagger-keras/>
5. “Part-of-Speech tagging tutorial with the Keras Deep Learning library” by Axel Bellec, as on May 19th, 2020.   
   Link: <https://becominghuman.ai/part-of-speech-tagging-tutorial-with-the-keras-deep-learning-library-d7f93fa05537>