L665 ML for NLP Spring 2018
Carlos Sathler (cssathler@gmail.com)

Assignment 4

LSTM and NLP based Sentence and Text Classification

Task b – Sentence level classification

Since I implemented a LSTM for Assignment 3, I will repeat it again here, but this time with a different dataset. I will the same dataset I'll be using for my class project. The new dataset is described in the paper "PubMed 200k RCT: a Dataset for Sequential Sentence Classification in Medical Abstracts" [1].

Dataset description:
The "PubMed" dataset [1] is delivered in two versions. I used the smaller version, which contains 20k records (the larger contains 200k). Medical abstract sentences are the single feature in the dataset; each row of data contains one sentence. We want to predict sentence class. Classes are "objective", "background", "method", "result" and "conclusions". The dataset was created to support research of randomized control trial (RTC) articles, which are a key resource in evidence-based medicine (EBM). The dataset is available in three partitions: train, dev (validation) and test, containing 80/10/10 percent of the data in that order. Notebook "Task1_Create_Dataset.ipynb" reads combines the 3 partitions into a single file.

Architecture and implementation:
Notebook "Task1_LSTM.ipynb" implements a neural net with a LSTM layer followed by 2 perceptron layers. I used pre-trained Glove word embeddings of dimension = 25, for fast performance. Sequence length was set to 50. This architecture does not match the architectures described in [1], but for the purposes of this assignment it offers a baseline for comparison with the extended model (Task c), which uses distributional information in the text. I tried a few variations of cell numbers and ended up setting with 200 for the LSTM layer, 25, and 5 for the subsequent layers.

Training procedure:
I used the dev partition delivered with the dataset for validation. I used dropout regularization (0.5) for the 2 perceptron layers and L1 (0.01) and L2 (0.01) regularizations for the output layer. Without regularization the model was overfitting.

The loss function used during training was Keras' "binary-crossentropy". Optimizer was Adam. Activation functions were "relu" for perceptron layers, "tanh" and "hard sigmoid" for LSTM layer and "softmax" for output layer. My final implementations use batch size = 256.

Evaluation methods and results:
I used the accuracy metric to evaluate my results. The final accuracy on the test set was 0.759.

Code:
https://github.com/csathler/IU_MSDS/blob/master/NLP/assignment4/Task1_Create_Dataset.ipynb
https://github.com/csathler/IU_MSDS/blob/master/NLP/assignment4/Task1_LSTM.ipynb

L665 ML for NLP Spring 2018
Carlos Sathler (cssathler@gmail.com)

Task c – Extending model (DLSTM)

In their paper "Deep LSTM based feature mapping for query classification" [2], Shi, Yangyang, et al. propose a model of stacked LSTM layers to perform convolution, and therefore capture distributional features in text, for query classification tasks. I used their architecture to extend the model I created for task b of this assignment.

Architecture and implementation:

Notebook "Task2_DLSTM.ipynb" implements architecture described in [2]. I created 3 LSTM layers per the paper. Layer 2 processes output of layer 1 and layer 3 processes output of layer 2. The output of each of the 3 layers is then concatenated and average pooling is used to create a final representation of each sentence. The tensor thus obtained is fed to the output layer. See summary of the architecture below. Here again I used pre-trained Glove word embeddings as input to the network. Sequence length is the same as the baseline.

Layer (type)	Output Shape	Param #	Connected to
input_24 (InputLayer)	(None, 50)	0	
embedding_24 (Embedding)	(None, 50, 25)	1683925	input_24[0][0]
lstm_69 (LSTM)	(None, 50, 200)	180800	embedding_24[0][0]
lstm_70 (LSTM)	(None, 50, 200)	320800	lstm_69[0][0]
lstm_71 (LSTM)	(None, 50, 200)	320800	lstm_70[0][0]
concatenate_20 (Concatenate)	(None, 50, 600)	0	lstm_69[0][0] lstm_70[0][0] lstm_71[0][0]
average_pooling1d_8 (AveragePoo	(None, 1, 600)	0	concatenate_20[0][0]
flatten_2 (Flatten)	(None, 600)	0	average_pooling1d_8[0][0]
dense_6 (Dense)	(None, 5)	3005	flatten_2[0][0]

After a few attempts with a small set of varying node counts, I settled with node = 200 for all LSTM layers.

Training procedure:

I used the dev partition delivered with the dataset for validation. I used L1 (0.01) and L2 (0.01) regularizations for the output layer. The loss function used during training was Keras' "binary-crossentropy". Optimizer was Adam. Activation functions were "tanh" and "hard sigmoid" for LSTM layers and "softmax" for output layer. My final implementations use batch size = 256.

Evaluation methods and results:

Like in the previous task, I used the accuracy metric to evaluate my results. The final accuracy of the extended model on the test set was 0.795, which is roughly a 5% increase over the previous implementation.

Code: https://github.com/csathler/IU_MSDS/blob/master/NLP/assignment4/Task2_DLSTM.ipynb

References:

[1] Dernoncourt, Franck, and Ji Young Lee. "PubMed 200k RCT: a Dataset for Sequential Sentence Classification in Medical Abstracts." *arXiv preprint arXiv:1710.06071* (2017).

[2] Shi, Yangyang, et al. "Deep LSTM based feature mapping for query classification." Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016.