Deep Neural Networks for Text Classification

Problem Statement:

Text classification is one of the most important Natural Language Processing & Supervised Machine Learning tasks in different business problems. Our task is to identify the type of news based on their headlines.

To achieve this, we took a huge collection of news headings belonging to 20 different news categories and trained CNN, RNN and HAN models on this data to identify tags for untracked news articles.

Along with this, we also compared the performance of these 3 different types of deep neural networks for the above-mentioned text classification task.

Approach:

Data preprocessing:

- Our initial dataset had around 41 different types of news headings and we preprocessed this data to merge the related categories to finally cut down the total number to 20 most effective news classes.

- We have used word vectors generated by Google's GloVe as an underlying data model for obtaining vector representations for words. GloVe stands for "Global Vectors for Word Representation" and it is a popular embedding technique based on factorizing a matrix of word co-occurrence statistics.

- We converted all text samples in the dataset into sequences of word indices. A "word index" would simply be an integer representation for the word. We set the max length of any sentence to the average size of news headlines in the corpus (truncating or padding with zeroes if necessary).
  - We use keras tokenizer API and feed it with our corpus to fit our text data.
  - Using this model, each news headline of our corpus is sequence encoded into integers.
  - The list of unique words is obtained from the tokenizer object created in the previous step and then the embedding matrix is created by mapping the index with Glove vector representation.

- The embedding matrix is used to create the embedding layer, the first layer for all our neural networks.

- The sequence encoded training data set is then fed into the neural networks for training the models.

- For output categories, we convert each of the 20 categories into a vector form of 20-dimension using one hot encoding technique.

Trained DNN Models:

CNN:

- Convolutional layers are used to identify special patterns in text data.

- Each input will pass through a series of convolution layers with filters (kernels), pooling, and dense layers with softmax activation function to classify an object with probabilistic values between 0 and 1 corresponding to each category in the 20-dimensional output (multi-class text classification).

RNN:

- A recurrent neural network (RNN), unlike a feed forward network, is a variant of a recursive artificial neural network in which connections between neurons make a directed cycle where output depends not only on the present inputs but also on the previous step's neuron state.
- We used Spatial Dropout layer to perform variational dropout in text models and LSTM layer to retain the last output in RNN. RNN overcomes shortcoming of traditional NN in dealing with sequence data by integrating lexical and semantic information.
- Each input will pass through LSTM, spatial dropout, dense with softmax activation layers.

HAN:

- The main idea behind this model is that words make sentences and sentences make documents and each document correspond to a news heading. Hence, we preprocess all our headings in a different way and construct a 3-D matrix to cater to the needs of HAN architecture.
- The first dimension represents the total number of documents, the second one represents each sentence in a document and the last one represents each word in a sentence.
- We built bidirectional LSTM layer to incorporate contextual information and used time distributed layer to apply a layer to every temporal slice of input.
- On top of these layers, we add a new layer called "Attention Layer" to apply attention mechanism at the word level as well as sentence level. Thus, enabling it to attend differentially to more and less important content when constructing the document representation. This will be followed by the dense layers with softmax activation to classify the document.

Results and Conclusion:

Performance: HAN (74.05 %) > RNN (67.01 %) > CNN (64.75 %)

However, CNN model has outperformed the other two models (RNN & HAN) in terms of training time.