Introduction:

Implemented the sentiment classification using custom built Convolutional Neural Network and Long Short-Term Memory models for the yelp restaurant review dataset using word2vec embeddings.

Data preprocessing:

To convert data in a form that can be easily read by the python code we load the dataset in JSON format and convert it to a Pandas data frame and export the first 100,000 rows of the data frame to a CSV file. Strings without whitespace characters from each line and loaded into a Pandas data frame. Finally, the first 100,000 rows of the data frame are exported to a CSV file located in an OUTPUT_FOLDER directory. Since the dataset is large, we use a few chunks of it to train and test our models by selecting the top 10000 sentiments of each category which are defined based on the stars of each one of them. For the selected dataset we preprocess it by removing stop words as they don't add any meaning to the text for model training.

Once the stop words are removed, we proceed to tokenization to split the text into array of words which gives advantage for transforming each word separately. Stemming is performed to reduce the number of words to its root word.

After the preprocessing is completed, we divide the dataset into the train set and test set using the train test function ().

Helper functions:

- make_word2vec_model-

This function creates a Word2Vec model from a dataset of text which takes various parameters such as size of the vectors, minimum count of a word, and size of the window to determine no of words to be included in each context. The function also allows for the option of adding padding tokens to the text data to ensure all sequences have the same length.

- make_word2vec_vector:

This function returns the list of numbers as a tensor which is created by converting each word in the sentence to a number by looking up that word in a pre-trained Word2Vec model. If the word isn't in the Word2Vec model, it sets that word's number to 0.

Implementation of CNN model:

The CnnTextClassifier loads a pre-trained Word2Vec model from a file and creates an embedding using the weights and initialized the convolutional and fully connected layers.

The forward method applies the convolutional layer and max-pooling operations for each window size. After the pooling operation, the results of each convolutional layer are concatenated together and passed through a fully connected layer to produce the final output and passing it through softmax function for generating probabilities.

Implementation of LSTM model:

The LstmTextClassifier defines embedding layer to convert text into vectors, LSTM layer to extract relevant features, and a fully connected layer for classifying.

Like the implementation of forward method of CNN model, we process the text data through embedding layers whose output is processed through LSTM layer. The output of LSTM layer is processed through fully connected layer to classify the text data.

Results/Findings:

As we increase the number of epochs, the loss in the classification of the text data is decreased.