CSE 582: Natural Language Processing **Salika Dave**

Homework 2 sad6003@psu.edu

04/09/2022 **PSU ID: 909604058**

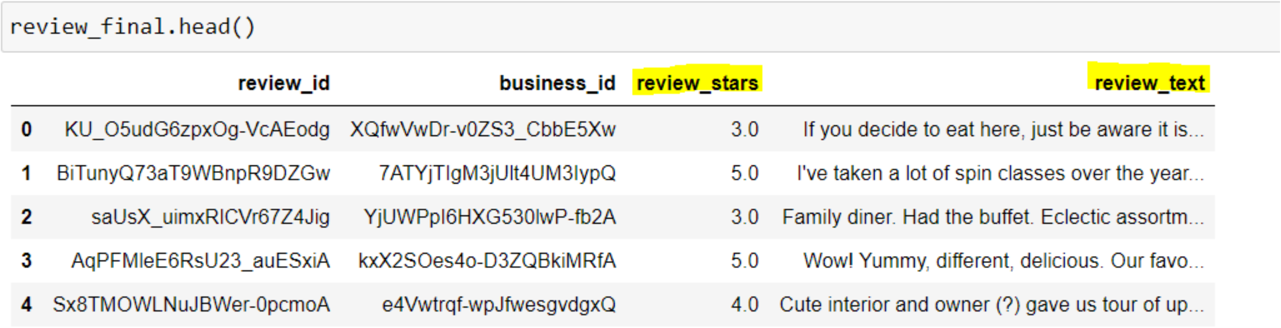
Task Description: Use Convolutional Neural Network (CNN) and Long-Short-Term Memory (LSTM) for sentiment classification.

Code: https://github.com/salikadave/cse582-hw2

Libraries used:
sklearn, pytorch, gensim

Dataset:
We are working with a subset of Yelp dataset that contains multiple reviews written by users from restaurants across multiple businesses.
Size of the dataset: 4.9GB
No of rows available for reviews: 6990280

Dataset Preprocessing:



```
review_final.head()
```

	review_id	business_id	review_stars	review_text
0	KU_O5udG6zpxOg-VcAEodg	XQfwVwDr-v0ZS3_CbbE5Xw	3.0	If you decide to eat here, just be aware it is...
1	BiTunyQ73aT9WBnpR9DZGw	7ATYjTIgM3jUlt4UM3IypQ	5.0	I've taken a lot of spin classes over the year...
2	saUsX_uimxRICVr67Z4Jig	YjUWPpI6HXG530lwP-fb2A	3.0	Family diner. Had the buffet. Eclectic assortm...
3	AqPFMleE6RsU23_auESxiA	kxX2SOes4o-D3ZQBkiMRfA	5.0	Wow! Yummy, different, delicious. Our favo...
4	Sx8TMOWLNuJBWer-0pcmoA	e4Vwtrqf-wpJfwesgvdgxQ	4.0	Cute interior and owner (?) gave us tour of up...

Explored two methods of sentiment classification using data available about "stars" given by the user with the review text.
Method 1: (Used this in CNN)
Three classes: "negative" (stars < 2) ; "neutral" (stars == 3); "negative" (stars > 4);
Method 2: (Used this in LSTM)
Two classes: "negative" (stars < =2); "positive" (stars > 2);
For the purpose of this homework, the dataset of reviews was reduced to 10000 reviews for each of the classes.
Dataset split >> Training (75%) & Validation (25%)

Data Cleaning:

1. Removed newline characters
2. Remove punctuations and special characters
3. Convert all words to lowercase
4. Perform tokenization
5. Generate stemming tokens (used Porter Stemmer in CNN)

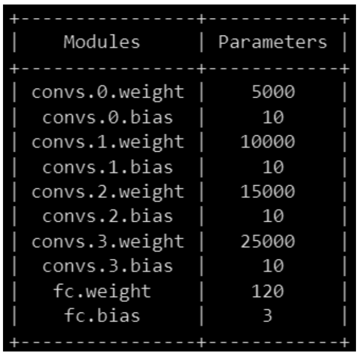
Embedding Vectors:
Used the Gensim word2vec embeddings for both CNN and LSTM.

Training Details:
CNN Training Time for 1 Epoch ~ 1m 8s
Embedding size = 500
# of filters = 10
window sizes = 1,2,3,5
Activation function in CNN: tanh. relu, sigmoid
Loss function: Cross Entropy Loss

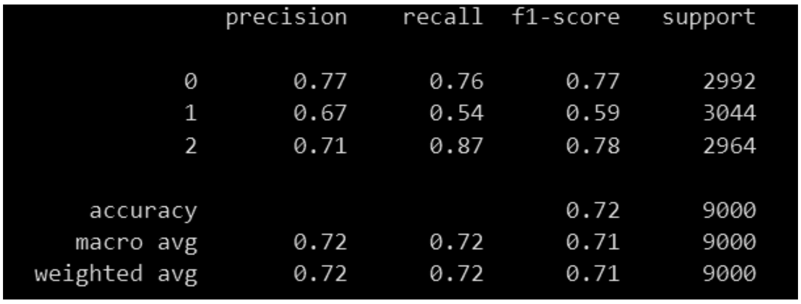
LSTM Training Time for 1 Epoch ~ 1m 20s
Due to time constraints experimentation is still in progress with the following:
clipping_quotient = 5   # set this value to avoid gradient explosion issues
num_epochs = 50
layers_count = 2
embedding_dimensions = 500
output_dimensions = 1
hidden_dimensions = 256
Activation function in LSTM: tanh. Sigmoid
Loss function: Binary Cross Entropy Loss

Overall, training time for CNN could be quicker than LSTM over 30 epochs.

Results:
CNN accuracy: 72%;



```
+---------------+-----------+
|    Modules    | Parameters |
+---------------+-----------+
| convs.0.weight |    5000   |
|  convs.0.bias  |     10    |
| convs.1.weight |   10000   |
|  convs.1.bias  |     10    |
| convs.2.weight |   15000   |
|  convs.2.bias  |     10    |
| convs.3.weight |   25000   |
|  convs.3.bias  |     10    |
|   fc.weight    |    120    |
|    fc.bias     |     3     |
+---------------+-----------+
```



```
              precision    recall  f1-score   support

           0       0.77      0.76      0.77      2992
           1       0.67      0.54      0.59      3044
           2       0.71      0.87      0.78      2964

    accuracy                           0.72      9000
   macro avg       0.72      0.72      0.71      9000
weighted avg       0.72      0.72      0.71      9000
```

LSTM accuracy: 69%

Accuracy comparison

CNN	tanh	72.66
	relu	72.85
LSTM	tanh	69.01

	relu	68.87

Key Learning outcomes:

- CNN & LSTM implementation for sentiment analysis
- LSTM architecture implementation
- Usage of pytorch for building models
- Exploration of activation functions for binary and multi-class classification

Future Scope:

Experiment with more efficient pretrained models for word vector embeddings
Experiment with more layers with

References:

1. https://towardsdatascience.com/sentiment-classification-using-cnn-in-pytorch-fba3c6840430
2. Building LSTM classes for Hebrew text classification: https://galhever.medium.com/sentiment-analysis-with-pytorch-part-4-lstm-bilstm-model-84447f6c4525
3. Understanding LSTM from scratch: https://medium.com/hackernoon/understanding-architecture-of-lstm-cell-from-scratch-with-code-8da40f0b71f4