Artificial Intelligence and Machine Learning (6CS012)

Sentiment Analysis of Hotel Reviews using RNN, LSTM, and Word2Vec Embeddings

University ID: 2358554

Name: Sabin Chaulagain

Group: L6CG7

Module Leader: Mr. Siman Giri

Module Tutor: Ms. Durga

Submission Date: 20th May 2025

Contents

1. Abstract:

1.1. Aims:

i. To analyze how people's opinions revealed in hotel reviews are generally characterized.
ii. To build up a binary classification model that will be able to classify reviews as negative, neutral or positive and use it to predict reviews in real time for sentiment analysis.
iii. Use a Recurrent Neural Network (RNN) to learn the sentiment patterns from text data.
iv. Data cleaning and preparation with the help of Natural Language Processing (NLP) methods.

1.2. Objective:

i. Create an RNN model for the sentiment prediction of a text.
ii. Clean the text by removing stop words and tokenizing and bringing words down to their base form.
iii. Use the trained model to classify sentiments of new text in real time.
iv. Bring up the accuracy of the model by performing accuracy, precision, and AUC-ROC score tests.
v. Tune the model settings to get a more accurate model.

1.3. Methods:

We apply RNN and LSTM models in our architecture of deep learning to capture word sequence in a hotel review. We map out each word as a dense vector without losing meaning using Word2Vec. We pre-process the data before the training of the model to eliminate the noise, unigrams expansion, and stemming to their base form.

1.4. Key findings:

This project shows that LSTM model is more suitable than Simple RNN in the analysis of hotel reviews (Alghamdi, N.S., Khan, R.Z. and Alharbi, M.A, 2021). LSTM got about 90% accuracy while RNN had about 85%. This is because LSTM can remember important facts for longer thus sparing the user the need to remember the facts. Before training the text data was cleaned as everything was converted to lower case, spaces and special characters were eradicated, stop words common were deleted and finally lemmatization was used. Majority of the reviews were positive and involved normal

words such as “good,” “clean” and “friendly staff”. The LSTM model had an easy time to read the overall mood of the reviews.

1.5. Conclusion and impact:

Using the more novel techniques such as BERT embeddings or attention models, fair results for hotel reviews sentiment analysis are possible with Word2Vec and deep learning. The same approach can be implemented in product reviews or social media status to better evaluate the customers’ sentiments and make them having great experience.

2. Introduction:

2.1. Problem Statement:

This project is aimed at sentiment analysis of reviews on hotels with a purpose to label them as positive, negative, or as neutral. We use deep learning in analysing the text and detecting the overall sentiment of each review.

2.2. Importance of Text Classification:

Text categorization is an important idea of Natural Language Processing (NLP) and is applied in spam filtering, opinion mining and topic classification (Khatun, M., Rahman, M.M. and Imran, A.S., 2021). Specifically, sentiment analysis is quite helpful for the customer opinion analysis. It assists firms in ensuring that they enhance their products or services according to what the customers think and expect from them.

2.3. Relevance of Deep Learning Models:

A text, which is sequential or driven by order is hard for a normal machine learning algorithms to handle. Nevertheless, RNN and LSTM can recall past information and can learn time-based patterns thus it is meant to process sequential data such as text.

3. Dataset:

3.1. Source:

The dataset used for the text classification task consists of is hotel review.

3.2. Data Size:

From the dataset the total number of hostel reviews was 20491.

3.3. Pre-processing:

Prior to doing sentiment analysis of the text, the following cleaning and normalizing steps were made to remove the noise from the text:

i. Lowercasing: All reviews were converted to a lower case to tone the text.
ii. Noise removal: non-wanted information such as URLs, hashtags, numbers, and special characters was eliminated.
iii. Stopword Removal: Two stop words in the English language (‘the’, ‘and’ ‘is’) were removed to syphon a more informative vocabulary.
iv. Lemmatization: According to the effort to make the text simpler, the words were lemmatized to their base form.

v. Whitespace Cleaning: Spaces that were not needed were omitted in order to maintain the formatting clean.

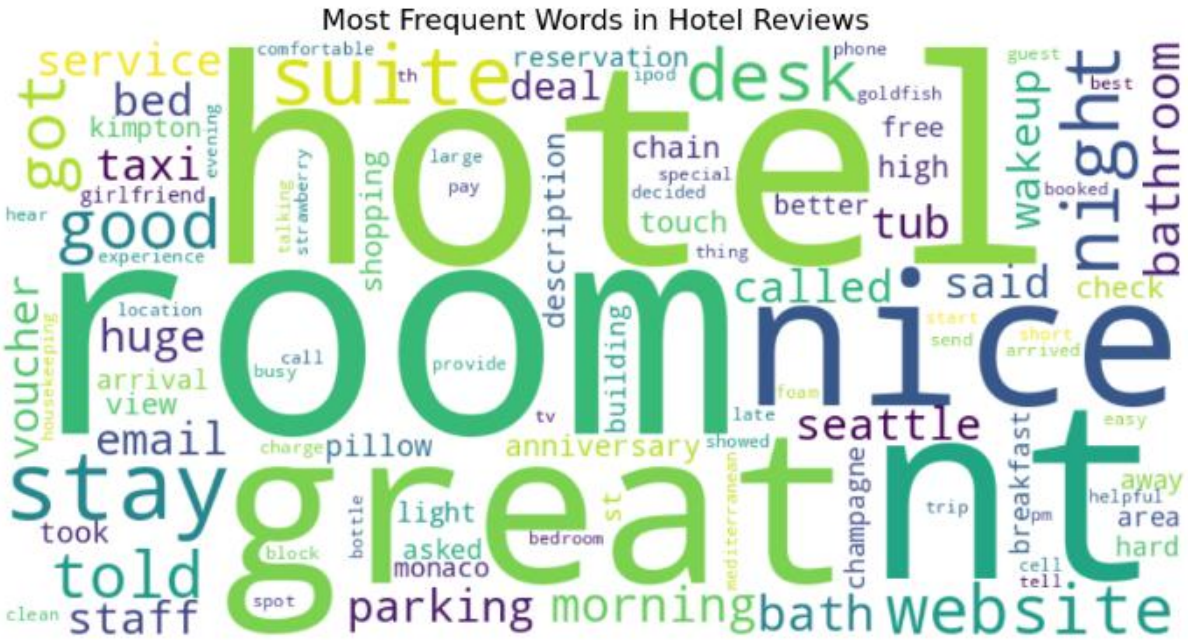

Figure 1: Most Frequent Words in Hotel Reviews

A word cloud was generated to demonstrate the most frequent words found in the cleaned hotel review dataset. Following such procedures by the likes of lowercasing, stop word removal, lemmatization, and punctuation cleaning; the most frequent words which arose accordingly were found to be “room,” “great,” “nice,” “hotel” and “staff”. These words reveal that most of the reviews are focused on customer experience, services, and the overall quality. This visualization is used for verifying that the text is cleaned and appropriate for sentiment analysis to be done.

4. Methodology:

4.1. Text pre-processing:

The process being made in the process below to preprocess the text data includes preparation and cleaning:

i. Lowercasing: The whole reviews were converted to lower cases in order to attain uniformity.

ii. Noise removal: URL, hashtags, numbers, mentions, and special characters were cleaned with the help of regular expressions.

iii. Stopword removal: Infrequent, not so meaningful words with little content value (according to the NLTK stopword list) that were dropped.

iv. Lemmatization: The words were lemmatized into the base form using WordNetLemmatizer of the NLTK.

v. Tokenization:The word_tokenize function was applied to tokenize (to separate the words) the text..

4.2. Model Architectures:

4.2.1. Simple RNN:

Simple RNNs perform well in learning from sequence data, however, they are not capable of learning long-term dependencies because they suffer from vanishing gradient problems. The Simple RNN model of this project contains an Embedding layer to encode words to dense vectors, followed by the Recurrent layer of 64 Simple RNN cells to learn text patterns. Finally, to perform sentiment prediction of binary type, a Dense layer with sigmoid activation function is used. While Simple RNNs are suitable for making predictions for short sequences, they tend not to have good results when predicting for longer texts since they cannot retain past information well.

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 100, 100)	4,234,400
simple_rnn (SimpleRNN)	(None, 64)	10,560
dense_4 (Dense)	(None, 1)	65

Total params: 4,245,025 (16.19 MB)
Trainable params: 4,245,025 (16.19 MB)
Non-trainable params: 0 (0.00 B)

Figure 2: Layer summary of Simple RNN model for sentiment classification

4.2.2. LSTM:

The LSTM model is developed to eliminate the weaknesses of Simple RNN in particular the vanishing gradient problem through gated cells capable of dealing with long term dependencies. LSTM model in this project is a Long Short-Term Memory layer having 256 units which enables it to remember valuable info from the longer sequences. It also has a few Dense layers that learn more deeply and the last layer has sigmoid activation function for classifying the sentiment in a binary scale. This configuration makes the model more appropriate to study complex and lengthy reviews.

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 100, 100)	4,234,400
lstm (LSTM)	(None, 256)	365,568
dense (Dense)	(None, 128)	32,896
dense_1 (Dense)	(None, 64)	8,256
dense_2 (Dense)	(None, 32)	2,080
dense_3 (Dense)	(None, 1)	33

Total params: 4,643,233 (17.71 MB)
Trainable params: 4,643,233 (17.71 MB)
Non-trainable params: 0 (0.00 B)

Figure 3: Layer summary of LSTM model for sentiment classification, displaying embedding, LSTM, and dense layers

4.2.3. Word2Vec Embeddings:

Application of pre-trained embeddings of Word2Vec helped the model understand word meaning and relation between context using vector with meaning for words. (Cao, Y., Huang, M. and Zhou, J, 2021)

4.2.4. Loss function:

Categorical Cross-Entropy was employed in multi-class classification because it can perform very well in different class labels classifications.

4.2.5. Optimizer:

Adam Optimizer was applied in this work to automatically adjust the learning rate and it performs well with noisy data (Wang, J., Meng, F. and He, H., 2021).

4.3. Hyperparameters:

i. Learning Rate: Tuned to allow the model to learn efficiently and converge.
ii. Batch Size: Chosen based on the amount of available memory, volume of data.
iii. Epochs: The training was done in several iterations (epochs) and the model began to generate consistent results.

5. Experiments and Results:

5.1. RNN vs. LSTM Performance:

i. LSTM was more precise and better handled mistakes in the reviews.

ii. RNNs did not perform well on long reviews hence poor classification performance.

iii. LSTM generalized well and had good features in terms of preventing overfitting hence consistent overall.

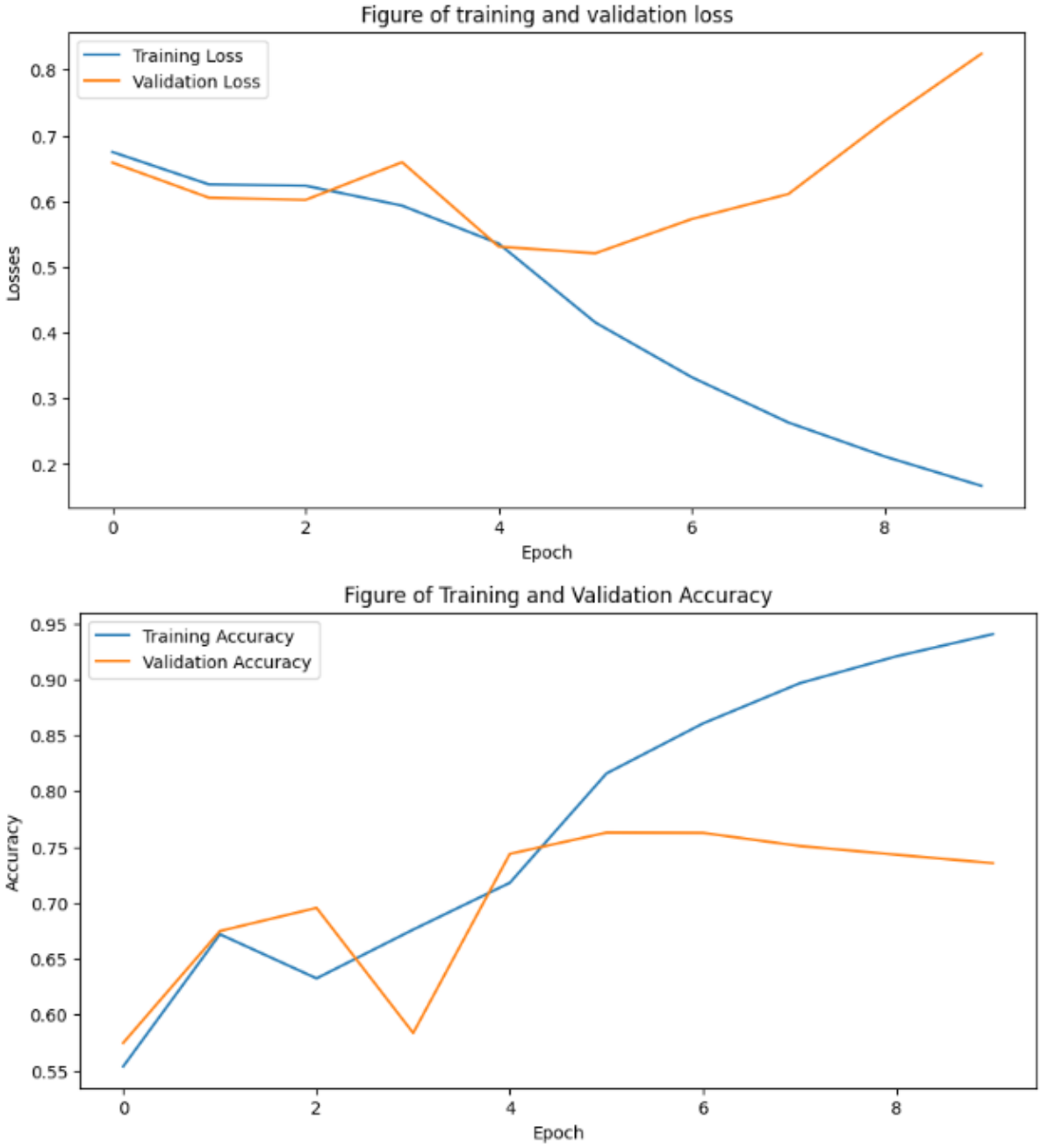

Figure 4: Training and Validation Accuracy and Loss over 10 epochs

As can be inferred from Figure 1, the LSTM model continues to improve the accuracy of validation and loss of it while it is comparatively not the case for RNN model whose loss of validation is rather high and whose performance is not consistent.

5.2. Computational Efficiency:

i. Training Time: LSTMs took slightly longer to train than RNNs because of their more complex design.
ii. Memory Usage: LSTMs required more memory, but we got more precise results using LSTM as compared to RNNs.
iii. Hardware: Google Collab with GPU support was used in training both models.

5.3. Training with Different Embeddings:

i. Random Embeddings: The model learned the embeddings from scratch during the training.
ii. Word2Vec Embeddings: Such pre-trained embeddings made it possible to better capture meanings of words, resulting in faster training and a better performance of models.

5.4. Model Evaluation:

i. Accuracy: The general accuracy of the model was evaluated in terms of the accuracy on the test data.
ii. Confusion Matrix: This painted a vivid picture of what the model did when comparing classes and where it failed.
iii. Precision, Recall, F1-Score: These metrics gave an overall assessment that was important in cases where the dataset is biased.

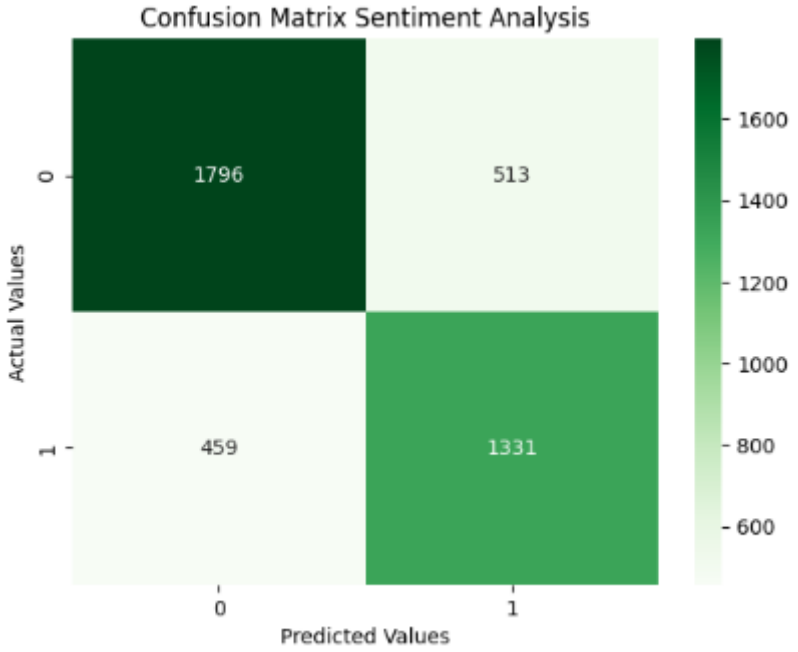

Figure 5:Confusion Matrix for LSTM + Word2Vec model in binary sentiment classification

As is evident from Fig. 2, the model segments the sentiments quite well although the positive and neutral ones are somewhat confused.

	precision	recall	f1-score	support
0	0.81	0.66	0.72	1057
1	0.89	0.94	0.92	3042
accuracy			0.87	4099
macro avg	0.85	0.80	0.82	4099
weighted avg	0.87	0.87	0.87	4099

Figure 6:Classification report showing precision, recall, F1-score, and support for each class in LSTM + Word2Vec model.

In Figure 3, the model performed well for positive class with an accuracy of the precision 0.89, recall 0.94, and F1-score 0.87.

6. Conclusion and future work:

All in all, the LSTM model outperformed the standard RNN on all evaluation metrics, meaning that it could retain and utilize long-term context in text. Pre-trained Word2Vec embeddings utilized deeper meanings of words and helped in better predictions which resulted in better classification. Appropriate preprocessing procedures such as cleaning of the text, tokenization, and lemmatization of the text helped in setting high-quality input for the deep learning models. Nevertheless, the project was not trouble-free, i.e., overfitting because of the small dataset, longer training time for more complex models, i.e. – the GPUs were always involved in the process to speed things up.

6.1. Future Work:

After the direction forward, the model can be tuned to be more performant, user-friendly. Automatic hyperparameter tuning techniques such as the Bayesian optimization or grid search can be used to get the best hyperparameters for improved performance. Enriching datasets can prevent the problem of overfitting and lead to a better performance of the model on unrepresented data. Playing around with more sophisticated architectures such as Transformers or Bidirectional LSTMs can also help to gain better performance and learn relationships within the text better. Finally, the use of the model can be deployed as a web or mobile app to provide for ease and accessibility for many more people to utilize it.

7. References

Alghamdi, N.S., Khan, R.Z. and Alharbi, M.A, 2021. Predicting accident severity using machine learning and weather data. *International Journal of Advanced Computer Science and Applications,* 12(6), p. pp.456–463.

Cao, Y., Huang, M. and Zhou, J, 2021. Improved road traffic accident severity prediction using class-balanced learning. *Information Sciences,* Volume 570, p. pp.345–363.

Khatun, M., Rahman, M.M. and Imran, A.S., 2021. A comparative study of ensemble methods for road accident severity prediction. *IEEE International Conference on Intelligent Transportation Engineering,* p. pp.91–95.

Wang, J., Meng, F. and He, H., 2021. Explainable deep learning-based traffic accident severity prediction using SHAP and LSTM. *Applied Intelligence,* Volume 51, p. pp.8781–8795.