

Artificial Intelligence and Machine Learning.

6CS012

News Category Classification Using LSTM, RNN and Word2Vec

|  |  |
| --- | --- |
| Student Id  Student Name  Group  Tutor | : 2358277  : Himani Shrestha  : L6CG22  : Mr. Aatiz Ghimire |

# **Abstract**

This project deals with the area of news category classification , i.e. discovering whether a news is related to a category or not. To cope with this, three deep learning models were tested. A Simple RNN, a LSTM with trainable ‘embeddings’, and a LSTM with pre-trained Word2Vec ‘embeddings.

All three models were trained, and a comparison was made. The Simple RNN learnt at a fast rate but overfitted and performed badly on unseen data. The LSTM was better but could not cope with positive sentiments correctly. The best of them was the LSTM model that works with the pretrained Word2Vec embeddings, which allowed it to understand the language of finance better and to make predictions more balanced and accurate.

In conclusion, and based on the results, the use of pretrained embeddings enhanced the understanding and classifying news category, making this option the best pick for this classification work.

**Table of Contents**

[**Abstract** 2](#_Toc198635259)

[**Table of Figures** 4](#_Toc198635260)

[**1.** **Introduction** 1](#_Toc198635261)

[**2.** **Dataset** 2](#_Toc198635262)

[**2.1.** **Dataset Description** 2](#_Toc198635263)

[**2.2.** **Preprocessing** 2](#_Toc198635264)

[**3.** **Methodology** 3](#_Toc198635265)

[**3.1.** **Text Preprocessing** 3](#_Toc198635266)

[**3.2.** **Data Splitting** 3](#_Toc198635267)

[**3.3.** **Handling Class Imbalance** 4](#_Toc198635268)

[**3.4.** **Model Architectures** 4](#_Toc198635269)

[**3.5.** **Model Compilation** 5](#_Toc198635270)

[**4.** **Experiments and Results** 5](#_Toc198635271)

[**5.** **Model Evaluation** 10](#_Toc198635272)

[**6.** **Conclusion and Future Work** 11](#_Toc198635273)

**Table of Figures**

[Figure 1: Labels Count 2](#_Toc198635282)

[Figure 2: Confusion Matirx Simple RNN 6](#_Toc198635283)

[Figure 3: Classification Report of RNN 6](#_Toc198635284)

[Figure 4: Model Accuracy and Loss of Simple RNN 7](#_Toc198635285)

[Figure 5: Confusion Matrix 7](#_Toc198635286)

[Figure 6: Classification Report of LSTM 8](#_Toc198635287)

[Figure 7: Accuracy and Loss graph of LSTM 8](#_Toc198635288)

[Figure 8: Confusion Matrix 9](#_Toc198635289)

[Figure 9: Accuracy and Loss Graph of LSTM+Word2Vec 9](#_Toc198635290)

[Figure 10: Classification Report of LSTM+Word2Vec 10](#_Toc198635291)

[Figure 14: Confusion Matrix of RNN, LSTM (pre-trained) 10](#_Toc198635292)

# **Introduction**

This project is aimed at classifying the news category into 5 different classes. They are Culture & Arts, Education, Money, Sports and Tech .

To overcome this, the project employs deep learning models intended to operate with sequential data. Three models are used at an implementation and comparison level:

• A Recurrent Neural Network with an embeddable layer

• A trainable embedding LSTM network Description

• A pretrained word2vec LSTM model that improves words representation.

The purpose is to compare the results of the work of each model and whether the usage of pretrained embeddings has an impact. The project also includes steps for preprocessing data before training, evaluation of models, and creation of a simple interface for making real time sentiment predictions.

1. **Dataset**

The dataset for this project is a news category dataset named news\_category.csv. It was provided as a college project and consists of news category with labels.

## **Dataset Description**

Source: Academic use custom dataset provided.

Task: Multi-class news category classification

Classes:

* SPORTS 5077
* TECH 2104
* MONEY 1756
* CULTURE & ARTS 1074
* EDUCATION 1014

Total samples: 11025

The Imbalance dataset is present in the dataset, as there are more Sports news than other category news. Such an imbalance might influence training and performance of models, and in particular in cases of minority class predictions.

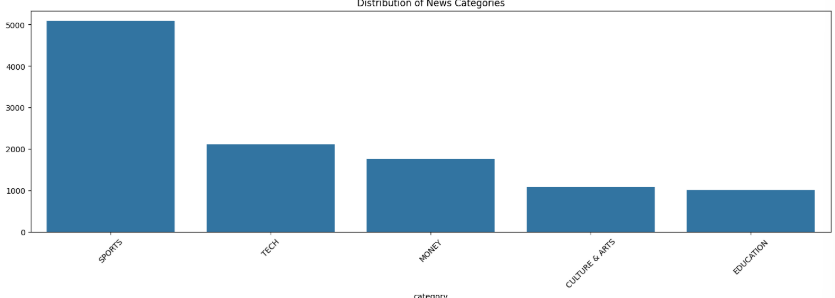


Figure : Labels Count

## **Preprocessing**

In order to prepare the models for training, there were the following steps taken on the text data; Cleaning and standardization as follows:

1. Lowercasing: Text was converted into lower case throughout keeping all texts consistent.

2. Removal of unwanted elements: To minimize noise in the data, URLs, user mentions (e.g., @username), hashtags, numbers, punctuation marks were cleaned.

3. Stopword removal: Stop words in the NLTK stopword list were eliminated (common English words that do not convey significant meaning – is, the, and etc).

4. Lemmatization: WordNet Lemmatizer was used to lemmatize words that is to convert them to their base or dictionary form. For example, “running” becomes “run”.

1. **Methodology**

In order to carry out sentiment classification, three different deep learning models were implemented, a Simple RNN, LSTM and LSTM, with pretrained word2Vec-embeddings. Below we describe the steps followed in preprocessing, training, and evaluation.

## **Text Preprocessing**

To prepare text data an object-oriented custom preprocessing function was used. The function performed lowercasing, eliminated unwelcome components, including URLs and special characters, and carried out stopword removal as well as lemmatization. Then the text was tokenized and padded using the cleaned text.

* **Tokenization:** Text was transformed to sequences of integers through Keras Tokenizer.
* **Padding:** Sequences were padded to equal lengths so that all inputs into the models are of the same magnitude.

## **Data Splitting**

The dataset after preprocessing was further split into training set, and testing set. 80% of the data was used for training and 20% was saved for testing. This split was implemented by use of train\_test\_split from Scikit-learn. In this way the dividing the data to ensures the model is evaluated on data it has not seen during training which is helpful in the checking of generalization performance.

## **Handling Class Imbalance**

The dataset is unevenly distributed regarding the classes of sentiments with the neutral class being represented by a much larger number of samples than the positive and the negative classes. To overcome the effect of this imbalance, class weights were applied during training. These weights assign higher importance to underrepresented classes in the loss function, helping the model learn more balanced predictions across all classes. This method helps reduce bias toward the majority class.

## **Model Architectures**

All models utilized Adam optimizer which adjusts learning rate automatically in training. As follows are details of each model:

**1. Simple RNN with Trainable Embeddings**

* Embedding layer with 128 dimensions (trainable)
* SimpleRNN layer with 64 units
* Dense output layer having 3 units with softmax activation
* **Total Parameters**: 543,363 (all trainable)

**2. LSTM with Trainable Embeddings**

* Embedding layer with 128 dimensions (trainable)
* LSTM layer with 64 units
* Dense: 2(128, 256) with Dropout in between
* Final Dense with 3 units for classification.
* **Total Parameters**: 622,339 (all trainable)

**3. LSTM with Pretrained Word2Vec Embeddings**

* GloVe vector based embedding layer (50 dimensions, non-trainable)
* LSTM layer with 128 units
* Final Dense layer with 3 units
* **Total Parameters**: 299,385
  + **Trainable**: 92,035
  + **Non-trainable**: 207,350

## **Model Compilation**

The same settings were used to compile all the three models. Whenever we are dealing with a multi class classification problem like this one where the output can be positive or negative or neutral the loss function used was categorical crossentropy. The Adam optimizer was preferred, as it is efficient while tuning the learning rate automatically for efficient and stable training process. During training, the dominant evaluation metric was accuracy, which defines the number of correct predictions.

* 1. **Hyperparameters**

In the training process, a batch size of 32 was used: that is, the model viewed 32 samples at a time before adjusting its weights. The models were trained for as many as 20 epochs; however deep learning based early stopping of training was used, to terminate the learning if the validation loss failed to improve in a couple of rounds. This helps avoid overfitting. 20%, of the training data was used for validation, during training. This set of validation data was used in monitoring the model's performance and for determining when the model should be stopped from learning.

1. **Experiments and Results**

This section presents analysis of three different models i.e. a simple RNN, LSTM, and an LSTM with pretrained Word2Vec embeddings. These models were tested in a financial sentiment classification assignment. The goal was comparison of performance in terms of accuracy, generalization, training behavior and computational efficiency.

* 1. **RNN Performance**

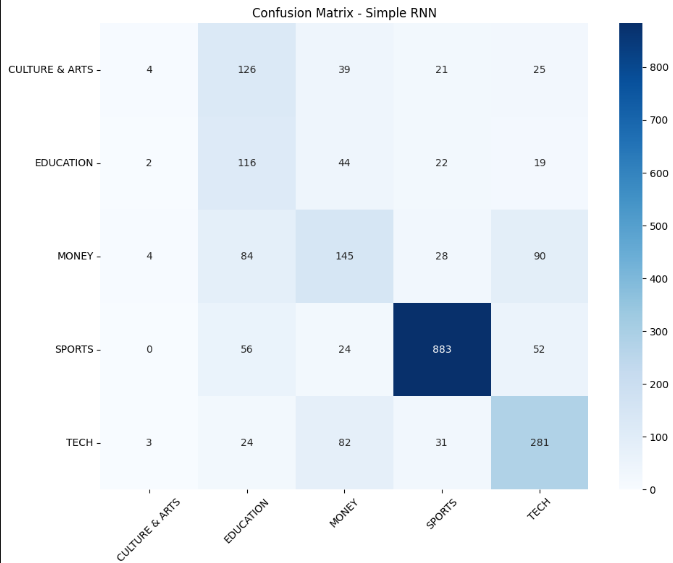


Figure : Confusion Matirx Simple RNN

A screenshot of a computer

AI-generated content may be incorrect.

Figure : Classification Report of RNN

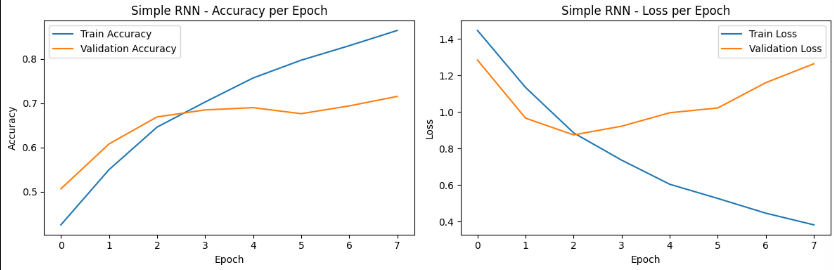


Figure : Model Accuracy and Loss of Simple RNN

Here the RNN model give the Accuracy of 65% after running for the 70 epoch. The graph shows the training accuracy of the 90% and Validation accuracy of 70% which shows the model is overfitting.

4.2 LSTM Performance

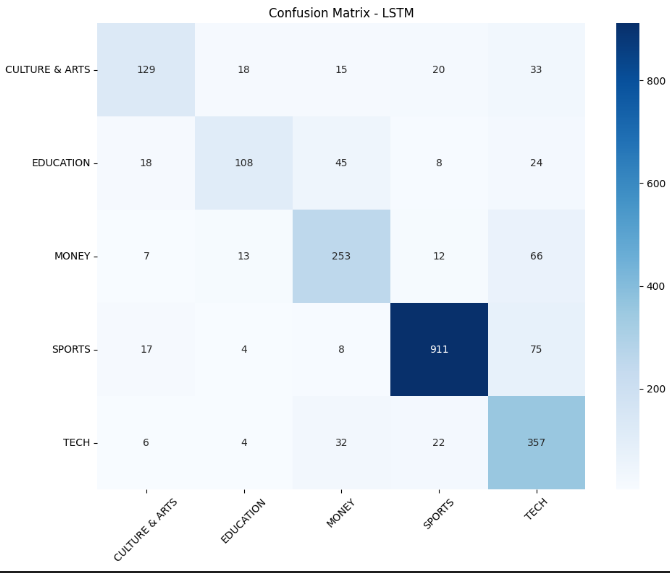


Figure : Confusion Matrix

A screenshot of a computer

AI-generated content may be incorrect.

Figure : Classification Report of LSTM

A graph of a line

AI-generated content may be incorrect.

Figure : Accuracy and Loss graph of LSTM

Here the LSTM model achieved the accuracy of 64% after running for the 70 epoch. The graph shows the training accuracy of the 90% and Validation accuracy of 80% which shows the model is overfitting.

4.3 LSTM+ Word2

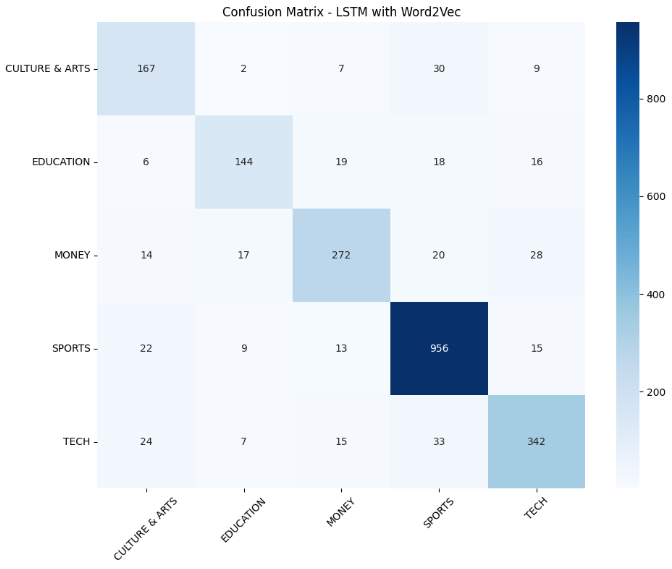


Figure : Confusion Matrix

A graph of a graph of a graph

AI-generated content may be incorrect.

Figure : Accuracy and Loss Graph of LSTM+Word2Vec

A screenshot of a computer

AI-generated content may be incorrect.

Figure : Classification Report of LSTM+Word2Vec

1. **Model Evaluation**

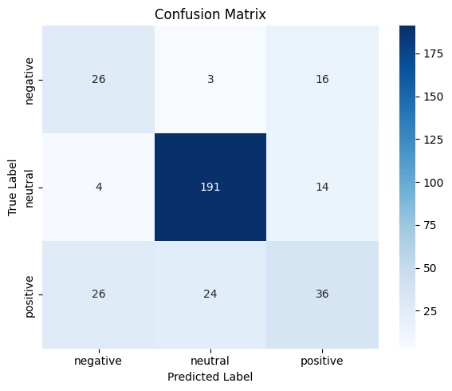
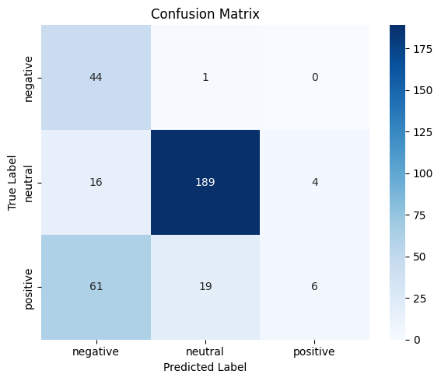
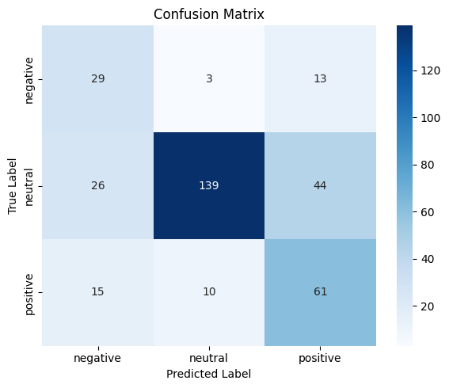


Figure : Confusion Matrix of RNN, LSTM (pre-trained)

To evaluate the models several metrics were used: accuracy, confusion matrix, and precision-recall-F1 scores. Although accuracy gives a rough representation of model performance it can be misleading, particularly in imbalanced or multi class problems such as the case for financial sentiment classification. That’s why, along with class-wise precision, recall and F1-scores, we separately assessed the performance for each class.

LSTM+Word2vec model resulted in test accuracy of about 85% and worked fairly well in news category classification.

# **Conclusion and Future Work**

The results indicated that the Simple RNN, while quick in training, showed considerable overfitting and poor generalization on unseen data. The LSTM with trainable embeddings showed moderate enhancement however continued to face challenges, especially in addressing class imbalance and maintaining accuracy across all categories. The LSTM model using pretrained Word2Vec embeddings delivered the superior performance, significantly exceeding the alternatives with an accuracy of roughly 85%. This model exhibited superior generalization and equitable predictions owing to its capacity of understanding word semantics via pretrained vectors.