News Classification using Word Embeddings and LSTM

Shreyas Suhas Yeolekar
School of Computing
National College of Ireland
Dublin, Ireland
Id: 18182674

Karan Mahendra Singh
School of Computing
National College of Ireland
Dublin, Ireland
Id:18192726

Nandita Sharma
School of Computing
National College of Ireland
Dublin, Ireland
Id:18185100

Monisha Laxme Gowda
School of Computing
National College of Ireland
Dublin, Ireland
Id:18195261

***Abstract*—The Internet is a vast source of information and news. With the ease of creating a website and adding information, sources of news have become ever increasing, also leading to the spread of fake news. As the amount of information available to the internet users is humungous, there is a need to segregate it into categories for ease of access and reading so that users can find information according to their preferences. In this project, we have made use of OneHot Encoding to convert the text-based categories into numerical categories and subsequently, applied LSTM to build the model and classify news into respective categories. We have presented how accuracy was affected by the number of categories used in the dataset. We have achieved a competent level of accuracy with the model and have seen that with removal of categories difficult to place, the model has achieved higher accuracy in training as well as validation.**

***Index Terms*—classification, LSTM, One-Hot Encoding**

I. Introduction

Due to rapid development of internet technology and there is a steady movement of society towards informatization. The Internet has become a source of information, education and news. It provides massive amounts of information and allows people to easily access various news text from a variety of devices. News contain information relating to different sectors, understanding and processing news text is particularly important. Classification of news text is an essential technology to produce news content [10]. It helps in efficiently organizing content and easy identification of information categories according to user needs and/or preferences. News stories can be categorized by marking news texts so readers can pick what they are interested in. Automatic categorization will help users to read and access news based on their interests. Additionally, it will help the news websites in suggesting the same or similar form of news to attract more attention via each reader's reading record. However, varying lengths of news and their headlines make it a challenging task to categorize the news [21].

Few researchers have classified the news with the help of short texts with semantics and ensemble methods. The TF-IDF vectorizer with word embedding word2vec have been employed for feature engineering to conserve the semantics of text in order to accomplish higher performance. The KNN, SVM, Naïve Bayes and the Gradient Boosting classifiers are presented in their research work for comparison. Multinomial Naïve Bayes performed the best classifier among these by yielding a higher accuracy with 90.12% and recall with 90% compared to other classifiers. As only three categories were considered for classifying, this proved to be a limitation of this research work [21].

Apart from machine learning, deep learning neural networks have been a common choice for classification of text. RCNNs have a shorter memory due to which passing the information through the sequence of steps. Thus, RNNs could be difficult to train and gather crucial information.

Therefore, in this research, we aim to apply LSTM which have internal memories in the form of gates to hold vital information to form knowledge. Our research question is as follows:

"*How well can the news text data be classified into appropriate categories using multilayer LSTM?*"

The LSTM better handles problems like gradient vanishing and exploding [22].The following sections are as follows: Related Work, Methodology,Implementation, Results, Conclusion and References.

II. Related Work

The internet has become one of the most important sources of news. It has a major impact on our day to day lives. [1] used deep learning along with Bi-Gated recurrent unit combined in the model to achieve good accuracy. LSTM is used to prevent the long-term build on and constant modification of the load in the self-loop by adding of the input gate. Words are segmented to categorize the news. LSTM is made by blending the gates to make double layer of LSTM, so that accuracy can increase [2] [1] [3] [4]. The accuracy obtained for is 73%. It is also mentioned that better accuracy can be achieved by using word vector [1].

The conventional method of classification is unable to yield good accuracy, because of scarce data. [5] conducted research to classify the news by using Word Embeddings with Sentence-LDA. Classification was done using word vector in embedding model and SVM for classification. To improve quality, topic vector was also incorporated. This research can be expanded by task-specific vector for better accuracy.

Machine learning methods are not suitable for text classification as they require several automatically trained data. Hence, [6]. used Bi-LSTM-CNN for recognizing the class of the text. They used word-based classification. RNN is used for extracting the expressions of the text. As a result, combination of LSTM and CNN amplified the accuracy by 0.84%. This research could be improved by generalizing.

[7] conducted the research to classify news headings. It was based on negative and positive headlines. In this paper, machine learning methods like Naïve Bayes and Support Vector machine are used. Naïve Bayes outperformed SVM with 70% accuracy.

Text classification is an essential task in performing Natural language processing. [8] attempted to combine the Convolution and RNN to exploit the benefits of both the neural network which enhance the extraction of features. The text classifier is made using the word vector. In this process, LSTM is used to extract the features of the text. On extracting features, the accuracy can be increased. Superior accuracy was achieved by the proposed model of about 98%. This research can be extended by applying the mechanism of attention on the text so that the model can be used for real time classification.

Depending upon the dataset machine learning methods can also be used to classify the news. Support Vector Machine, Naïve Bayes, and decision tree have been used in the paper for classification of Indonesian News. Among all the machine learning models SVM surpassed all other models with 93% accuracy [9].. This research is limited to small dataset.

Large datasets require well-built models for classifying data. [10]. used Gradient boosting, KNN, SVM, Naïve Bayes model for the study. Word vector is used to classify the text. By comparing all the models, multinomial Naïve Bayes accuracy was the best with 90%. The only limitation of this paper is classification done by only three categories.

Many of the researches were conducted using the machine learning approaches like Random Forest, SVM, Logistic Regression, Naïve Bayes etc [11] [12] [12]. conducted the research using both machine and neural networks. Word vector is used for classification. The ensemble models of neural networks provided the accuracy about 68%. The research can be expanded to character-based classification to get more accuracy. Similarly, in the other paper, all machine and deep learning models are used as baseline models. Then A2Text-net models is built with three Hypothesis layers. This A2Text-net model is used to mimic the interaction speech, which concentrates on parts of speech, grammar of the speech, punctuations, smiles with expressions etc. This model consists of three major layers like, Hypothesis layer, Feature processing layer, Neural Network layer. This model is mainly proposed to identify the sarcastic phase of the speech, make end users to understand the language clearly, additional features has been proposed based on the statistical hypothesis to improve the detection of the sarcastic phases. As a result, this intermediate layer is very useful and adoptable for different neural network models [11]..

RNN and CNN are the major methods which are used for text classification [13] [3]. Along with CNN and RNN, Hierarchical approach is built by using the Bi-direction LSTM in the third layer. This model is used to represent the 2 vectors direction. In this paper character level classification is done. The major drawback of this approach is that it is not suitable for a smaller number of data [13].

Apart from the RNN, and CNN, Capsule networks are also applied for classification. Capsule network is based to use in image classification but can also be used in non-image data. By using this model disadvantages of neural network can be solved. This model is more robust as it uses dynamic routing which separates the model from traditional approaches. Word2vec and Glove is used to classify the text in categories. A smaller number of neurons are trained with better performance [14]. LSTM plays an important role of encoding for text sequence [14] [11] [15] [2]. As it provides good accuracy, Capsule model can used instead of CNN. This model can also be used in the hierarchical classification of text [14].

By analysing the different research papers, SVM model is very frequently used in most of the papers [11] [16] [7] [10]. SVM and ANN models are applied to compare the model for text classification. After evaluating the obtained results, it can be explained that SVM model is less complex than ANN and SVM is better for text classifying small documents [16].

Most of the researches are based on word classification. But new model was made to classify the text using the character at small level. The accuracy of the model depends on the number of hidden layers. Hence, conducted the research by using the 29 layers of convolutions to improve the accuracy. As several hidden layer increases, accuracy of the model increases. As a result, good accuracy was accomplished due to deep convolution layers [17].

By combining both CNN and LSTM a unique approach is developed to classify the long sentences. Along with the CNN-LSTM model word vector is used for classification. This proposed method was useful to increase the accuracy of classifying the text [18] [19].

By analysing the different papers, it is understandable that the classification can be done in two various types: Character and Word level. Character is used as it does not need any history about the language. Both CNN and LSTM are used for this kind of classification. But this idea is only applicable for small text classification [20].

The previous methods for classifying text have no timely and reliable knowledge from the large quantities of information available. The new method KNN algorithm is applied in this research work which is based on correlation analysis and rough set. In training set, text vector are divided into uncertain and certain areas. The category can be directly judged for certain areas. The Text classification KNN algorithm determine the text vector types using correlation analysis. Since the limit of the defined and uncertain area is unclear, there is need to explore a more efficient and rational way to determine their boundaries as part of future work which states the limitation of this research [25].

In [25] Latent Dirichlet Allocation approach was used for text classification. Dimensionality reduction and feature extraction techniques were applied to reduce the high dimension news text using the topic model. In this paper, softmax algorithm is used and model is applied to the real news dataset. The news text is converted to the topic vector from VSM, then these vectors are converted to low dimension from high dimension by topic model. The result was effectively achieved using softmax regression. The research is limited to selection on the topic model parameters.

III. Methodology

A. Dataset

The dataset used in this project [24] contained over 2,00,000 news between the years 2012 and 2018 and was obtained from Kaggle. This dataset contained news from around 40 categories along with columns containing headlines, authors, links, short descriptions and date.

The dataset was first preprocessed. Out of all the columns, only relevant columns were selected to carry out successful research. Thereafter, some columns were combined to obtain all words in a row relating to the respective category. Subsequently, there were many categories with similar contents under different names leading to ambiguity in the data. For instance, 'Arts & Culture', 'Culture & Arts' and 'Arts' have same meaning. Thus, they were renamed to a single label 'Arts'. We followed the same approach for other categories. Furthermore, we removed categories having less article counts and also those like 'Weird News' and 'Good News' considering their low popularity. The final dataset consisted of 15 categories and around 150,000 articles.

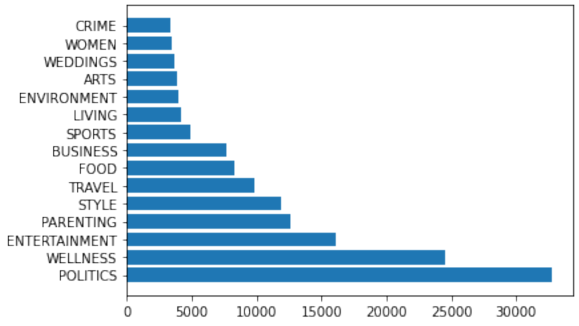
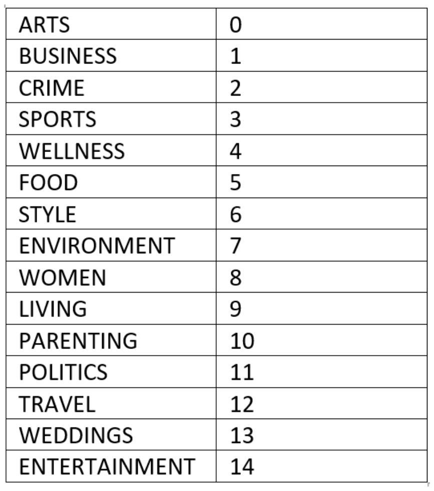


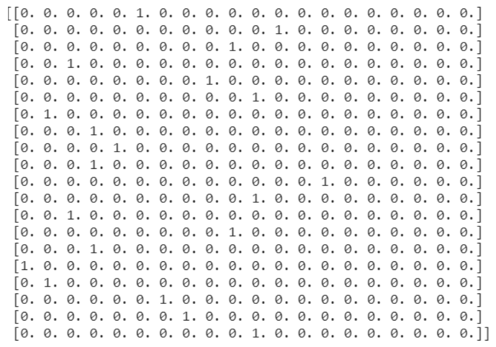
Fig. 1. Article counts

Further, as the categories contained varying numbers of news, in aiming to have a uniform frequency of news under all categories for training the model, contents of all categories were made into the same frequency. As the lowest frequency was 3400, only a subset of rows of size 3400 from each label was used. Thus, the model got equal numbers of news under all categories. The rows under each label were shuffled. Shuffling helps reduce variance and ensure that the models do not over-fit. Thereafter, One-Hot Encoding was employed for labeling the categories as the model could not work directly with the textual-categorical data. This technique was used as there was no ordinal relationship amongst the categories in the dataset. One-Hot Encoding also known as Dummy Coding introduces dummy variables 0 and 1 for absence and presence of the category respectively.



ARTS	0
BUSINESS	1
CRIME	2
SPORTS	3
WELLNESS	4
FOOD	5
STYLE	6
ENVIRONMENT	7
WOMEN	8
LIVING	9
PARENTING	10
POLITICS	11
TRAVEL	12
WEDDINGS	13
ENTERTAINMENT	14

Fig. 2. Category Labels



```
[[0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

Fig. 3. One Hot Encoding

Following which, we selected 18,000 most common words to turns into tokens and learn word embeddings. Finally, we built the model using LSTM with the Softmax activation function.

B. KDD approach

The KDD methodology of discovering knowledge was used in during the process. The dataset obtained from Kaggle contained some columns which were irrelevant to the scope of the project which were hence removed. The book [26], was referred for KDD.

(i) Data cleaning: There were no null or NA values in the dataset and thus, no cleaning of such values was required.
(ii) Data integration: This step too was not required as all required columns and data available in the same dataset and no joining of other datasets was needed.

(iii) Data selection:Originally, the dataset contained some columns which were of no use in answering the research question. Therefore, they were removed from the dataframe and only columns named: category, headline and short_description were selected.

(iv) Data transformation: The columns headline and short_description were combined into a column named text. Thereafter, a good number of categories existed under more than one name, for instance, columns Arts and Culture & Arts. Such categories were combined under one category to get rid of ambiguity. Thereafter, punctuations used in the headlines as well as the news text were removed and unique tokens were formed.

(v) Data mining: After transformation of the data, the categories were labeled using OneHotEncoding to get numerical categories against textual ones and subsequently, the text was tokenized. The data was split into 80%-20% for training and testing, respectively and finally, the model was built using LSTM and Softmax activation function.

(vi) Pattern Evaluation: The output thus obtained was evaluated using a confusion matrix which shows the performance of the model against various categories in the dataset.

(vii) Knowledge presentation: The knowledge so obtained is represented using plots and confusion matrix and is discussed in detail in the Results section.

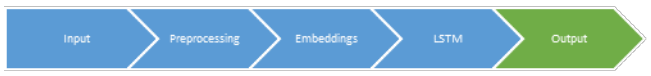


Fig. 4. Process Diagram.

IV. Implementation

A. Word Embeddings

Word embeddings are vector representations of words that are semantically and syntactically correspondent. One-to-One word to vector mapping is done and learning is done in a way that mimics a neural network. Thre are many techniques which can be used for word embedding such as, Embedding Layer, Word2Vec and GloVe. For this project we have used an embedding layer of Keras. The embedding layer requires the input to be cleaned and one-hot encoded. So our data satisfies the parameters for embedding layer. The vector space size needs to be specified for the model. Here, we used 100 dimensions.

B. Algorithm

a) RNN: We used Recurrent Neural Networks(RNN) which are a class of Neural Networks. A Recurrent Neural Network is generally used to process sequences while preserving a previous state of the sequences. The present output acts as an input for the next step. The model also considers everything that it rembembers from the previous input. The fig. 5 shows the basic model of a RNN.

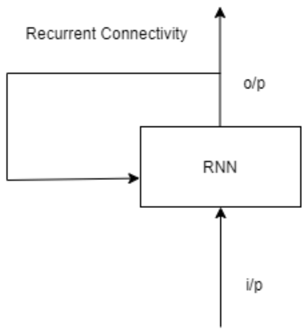


Fig. 5. A basic RNN with a loop

Long Short Term Memory(LSTM) networks are a type of RNN which are suitable for learnings involving long term dependencies [23]. To overcome the problem of short term memory faced by basic RNN, LSTM's posed as a solution. LSTM's have gates which are used for regulation of information flow.

b) Forget Gate: The forget gate is used to decide whether the information from the previous step should be discarded or kept.

c) Input Gate: The input gate is used to add only relevant information to the next step.

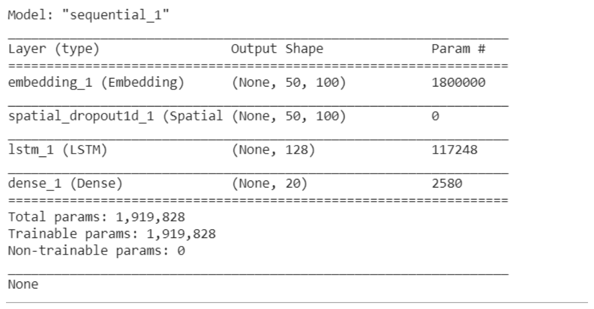
d) Output Gate: The output gate deals with deciding the next hidden state which is used for further predictions.

TABLE I
MODEL HYPERPARAMETERS

Name	Values
Epochs	30
Embedding Dimensions	100
Batch Size	50
Spatial Dropout	0.7
Dropout	0.7
LSTM units	128
Dense layer	20

The hyperparameters used for training the model are specified in the table I. The model was tested by tweaking the hyperparameters but the best results were achieved using the ones given in the table.

The softmax activation function was used for this model. Softmax activations converts scores into probabilities which total to one. This function is used here as the problem is multi-class. After compiling the model the summary was generated. Refer fig. 6.



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 50, 100)           1800000
_________________________________________________________________
spatial_dropout1d_1 (Spatial (None, 50, 100)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               117248
_________________________________________________________________
dense_1 (Dense)              (None, 20)                2580
=================================================================
Total params: 1,919,828
Trainable params: 1,919,828
Non-trainable params: 0
_________________________________________________________________
None
```

Fig. 6. Model Summary

V. Results

First we used 5 labels for the analysis, fig. 7. The model was fit using 9000 most common words and a max_length of 50 words.

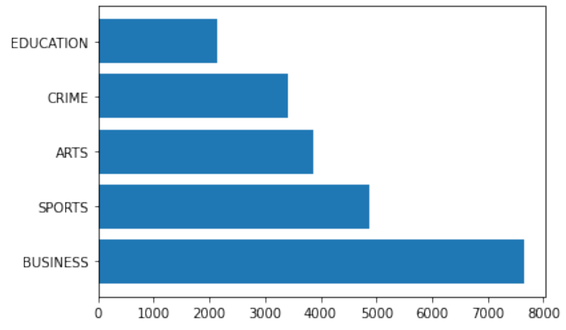
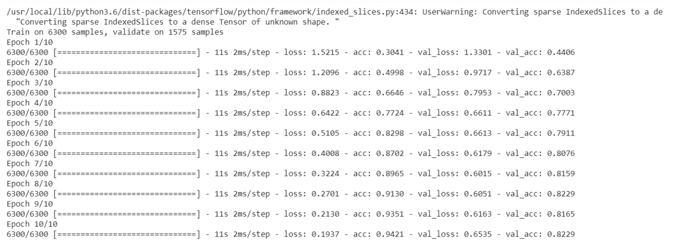


Fig. 7. Article count for 5 categories



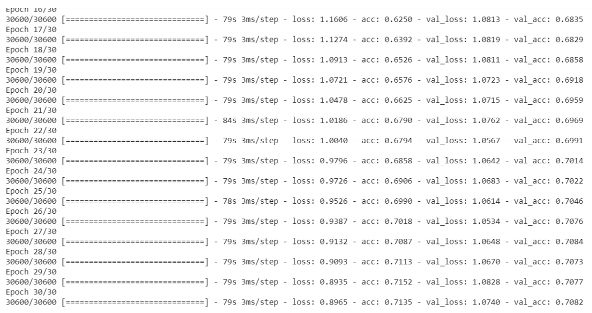
```
/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/indexed_slices.py:434: UserWarning: Converting sparse IndexedSlices to a de
  "Converting sparse IndexedSlices to a dense Tensor of unknown shape. "
Train on 6300 samples, validate on 1575 samples
Epoch 1/10
6300/6300 [==============================] - 11s 2ms/step - loss: 1.5215 - acc: 0.3041 - val_loss: 1.3301 - val_acc: 0.4406
Epoch 2/10
6300/6300 [==============================] - 11s 2ms/step - loss: 1.2096 - acc: 0.4998 - val_loss: 0.9717 - val_acc: 0.6387
Epoch 3/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.8823 - acc: 0.6646 - val_loss: 0.7953 - val_acc: 0.7003
Epoch 4/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.6422 - acc: 0.7724 - val_loss: 0.6611 - val_acc: 0.7771
Epoch 5/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.5105 - acc: 0.8298 - val_loss: 0.6613 - val_acc: 0.7911
Epoch 6/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.4008 - acc: 0.8702 - val_loss: 0.6179 - val_acc: 0.8076
Epoch 7/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.3224 - acc: 0.8965 - val_loss: 0.6015 - val_acc: 0.8159
Epoch 8/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.2701 - acc: 0.9130 - val_loss: 0.6051 - val_acc: 0.8229
Epoch 9/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.2130 - acc: 0.9351 - val_loss: 0.6163 - val_acc: 0.8165
Epoch 10/10
6300/6300 [==============================] - 11s 2ms/step - loss: 0.1937 - acc: 0.9421 - val_loss: 0.6535 - val_acc: 0.8229
```

Fig. 8. Training 5 categories

Here, the lowest value count was of the label Education(approximately 2000). So 2000 was the maximum subset size of other labels. We followed the shuffle and merge approach for each label. We found 33186 tokens after tokenizing the data.

After training the model we got training accuracy of 95.2% and testing accuracy of 80.7%. The training phase is shown in fig. 8.

In the next step we considered 15 labels. Here the lowest value count was around 3400. So we set the subset size for all other labels as 3400. After shuffling and merging the data it was given as input to the tokenizer where it was converted into sequences. The final output returned 90660 tokens. The model yielded training accuracy of 71.35% and testing accuracy of 71.6% fig. 9



```
Epoch 16/30
30600/30600 [==============================] - 79s 3ms/step - loss: 1.1606 - acc: 0.6250 - val_loss: 1.0813 - val_acc: 0.6835
Epoch 17/30
30600/30600 [==============================] - 79s 3ms/step - loss: 1.1274 - acc: 0.6392 - val_loss: 1.0819 - val_acc: 0.6829
Epoch 18/30
30600/30600 [==============================] - 79s 3ms/step - loss: 1.0913 - acc: 0.6526 - val_loss: 1.0811 - val_acc: 0.6858
Epoch 19/30
30600/30600 [==============================] - 79s 3ms/step - loss: 1.0721 - acc: 0.6576 - val_loss: 1.0723 - val_acc: 0.6918
Epoch 20/30
30600/30600 [==============================] - 79s 3ms/step - loss: 1.0478 - acc: 0.6625 - val_loss: 1.0715 - val_acc: 0.6959
Epoch 21/30
30600/30600 [==============================] - 84s 3ms/step - loss: 1.0186 - acc: 0.6790 - val_loss: 1.0762 - val_acc: 0.6969
Epoch 22/30
30600/30600 [==============================] - 79s 3ms/step - loss: 1.0040 - acc: 0.6794 - val_loss: 1.0567 - val_acc: 0.6991
Epoch 23/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.9796 - acc: 0.6858 - val_loss: 1.0642 - val_acc: 0.7014
Epoch 24/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.9726 - acc: 0.6906 - val_loss: 1.0683 - val_acc: 0.7022
Epoch 25/30
30600/30600 [==============================] - 78s 3ms/step - loss: 0.9526 - acc: 0.6990 - val_loss: 1.0614 - val_acc: 0.7046
Epoch 26/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.9387 - acc: 0.7018 - val_loss: 1.0534 - val_acc: 0.7076
Epoch 27/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.9132 - acc: 0.7087 - val_loss: 1.0648 - val_acc: 0.7084
Epoch 28/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.9093 - acc: 0.7113 - val_loss: 1.0670 - val_acc: 0.7073
Epoch 29/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.8935 - acc: 0.7152 - val_loss: 1.0828 - val_acc: 0.7077
Epoch 30/30
30600/30600 [==============================] - 79s 3ms/step - loss: 0.8965 - acc: 0.7135 - val_loss: 1.0740 - val_acc: 0.7082
```

Fig. 9. Training 15 categories

We applied the model on 20 news categories and achieved an accuracy of 63% for the test set. The model was fit using 15,000 most common words. A maximum length of 50 words were considered in the text column. The subset size was set to 2000 as the lowest value count was approximately 2000. refer fig. 10. The tokenized output returned 78012 unique tokens.

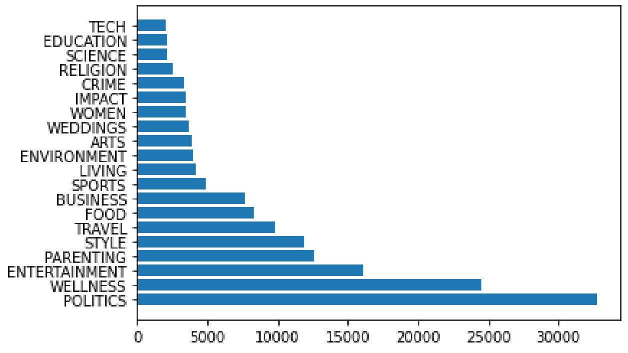
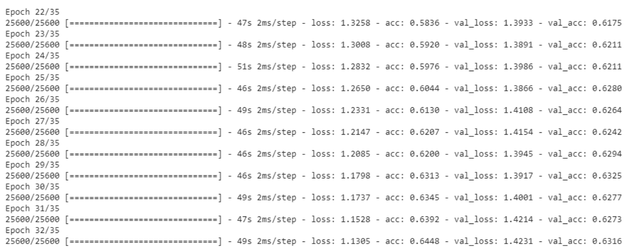


Fig. 10. Article count for 20 categories



```
Epoch 22/35
25600/25600 [==============================] - 47s 2ms/step - loss: 1.3258 - acc: 0.5836 - val_loss: 1.3933 - val_acc: 0.6175
Epoch 23/35
25600/25600 [==============================] - 48s 2ms/step - loss: 1.3008 - acc: 0.5920 - val_loss: 1.3891 - val_acc: 0.6211
Epoch 24/35
25600/25600 [==============================] - 51s 2ms/step - loss: 1.2832 - acc: 0.5976 - val_loss: 1.3986 - val_acc: 0.6211
Epoch 25/35
25600/25600 [==============================] - 46s 2ms/step - loss: 1.2650 - acc: 0.6044 - val_loss: 1.3866 - val_acc: 0.6280
Epoch 26/35
25600/25600 [==============================] - 49s 2ms/step - loss: 1.2331 - acc: 0.6130 - val_loss: 1.4108 - val_acc: 0.6264
Epoch 27/35
25600/25600 [==============================] - 46s 2ms/step - loss: 1.2147 - acc: 0.6207 - val_loss: 1.4154 - val_acc: 0.6242
Epoch 28/35
25600/25600 [==============================] - 46s 2ms/step - loss: 1.2085 - acc: 0.6200 - val_loss: 1.3945 - val_acc: 0.6294
Epoch 29/35
25600/25600 [==============================] - 46s 2ms/step - loss: 1.1798 - acc: 0.6313 - val_loss: 1.3917 - val_acc: 0.6325
Epoch 30/35
25600/25600 [==============================] - 49s 2ms/step - loss: 1.1737 - acc: 0.6345 - val_loss: 1.4001 - val_acc: 0.6277
Epoch 31/35
25600/25600 [==============================] - 47s 2ms/step - loss: 1.1528 - acc: 0.6392 - val_loss: 1.4214 - val_acc: 0.6273
Epoch 32/35
25600/25600 [==============================] - 49s 2ms/step - loss: 1.1305 - acc: 0.6448 - val_loss: 1.4231 - val_acc: 0.6316
```

Fig. 11. Training 20 categories

EarlyStopping was used as a callback function with the monitor on validation loss which led to its termination at Epoch: 32. Fig. shows Training and validation accuracy as well as Training and validation loss. Convergence of accuracy can be seen at around 27th Epoch.

Thereafter, with the aim of increasing the accuracy of the model, some categories: Education, Tech, Impact, Religion and

Science were removed and the model was trained again using 30 Epochs. This made a huge difference leading to an accuracy of 70% with convergence of training and validation accuracy seen on the 27th Epoch.

A. *Loss and Accuracy*

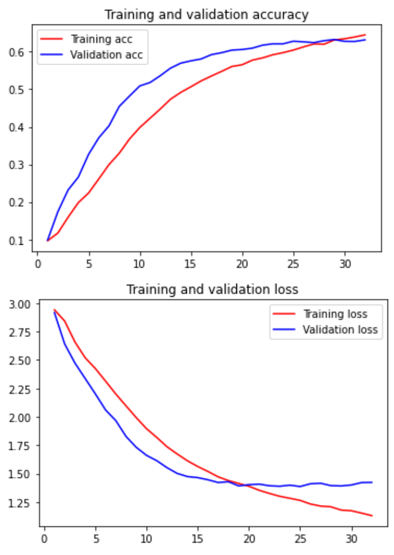


Fig. 12. Loss and Accuracy(20 categories)

The graph in figure 12 shows that there is hardly any difference between the accuracy of training and validation. The loss value for training and validation converges at 1.50

Figures 13 and 14 shows negligible or no variance in the accuracy of training and validation set. 1.05 is the converging point for the loss function of both validation and test set.

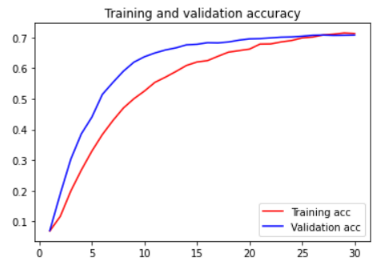


Fig. 13. Accuracy(15 categories)

Figure 15 however shows some variance in the accuracy of training and validation sets. This indicates that the model might be an over-fit. There is a noticeable difference in the loss values of training and validation as well.

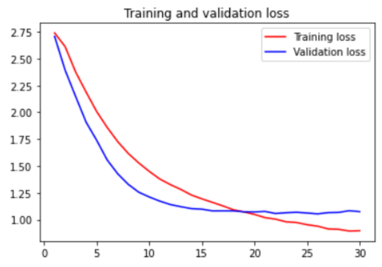


Fig. 14. Loss(15 categories)

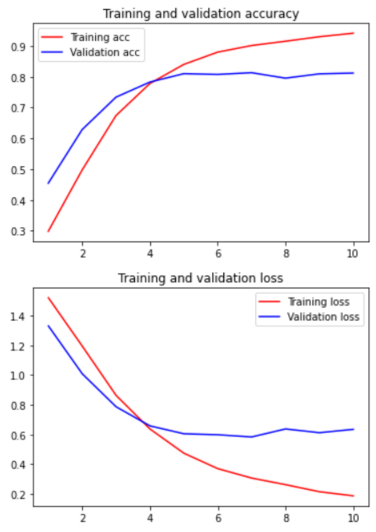
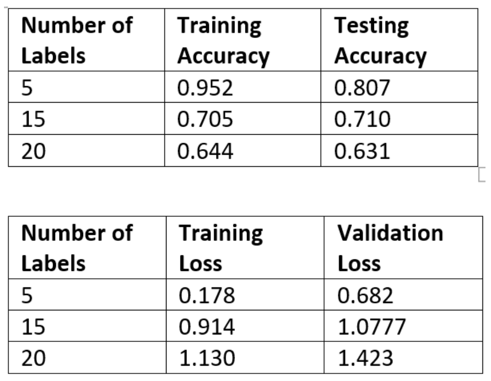


Fig. 15. Loss and Accuracy(5 categories)

B. *Evaluation Metrics*

The evaluation metrics loss and accuracy were used for this analysis. The results can be seen in figure 16.



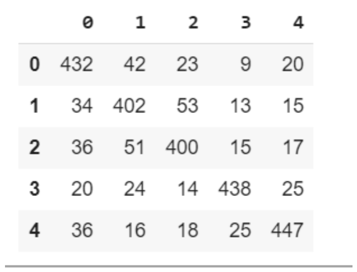
Number of Labels	Training Accuracy	Testing Accuracy
5	0.952	0.807
15	0.705	0.710
20	0.644	0.631

Number of Labels	Training Loss	Validation Loss
5	0.178	0.682
15	0.914	1.0777
20	1.130	1.423

Fig. 16. Loss and Accuracy

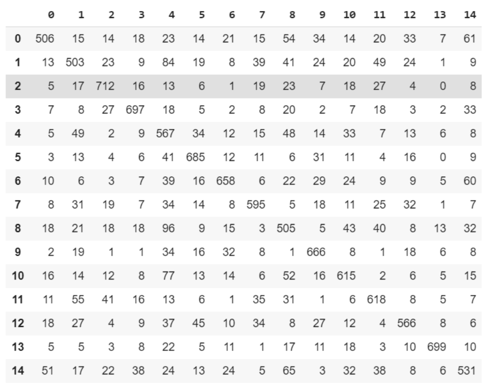
C. *Confusion Matrix*

The confusion matrix for the results with 5,15 and 20 categories is given in figures 17, 18 and 19 respectively.



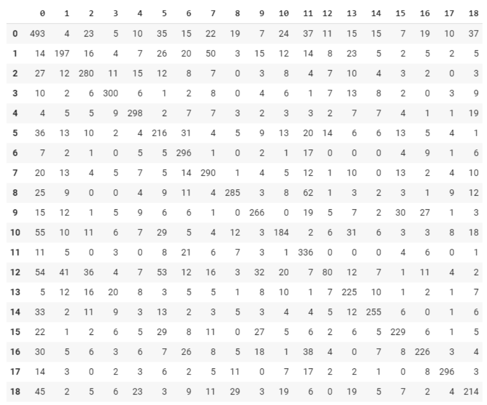
	0	1	2	3	4
0	432	42	23	9	20
1	34	402	53	13	15
2	36	51	400	15	17
3	20	24	14	438	25
4	36	16	18	25	447

Fig. 17. Confusion Matrix(5 categories)



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	506	15	14	18	23	14	21	15	54	34	14	20	33	7	61
1	13	503	23	9	84	19	8	39	41	24	20	49	24	1	9
2	5	17	712	16	13	6	1	19	23	7	18	27	4	0	8
3	7	8	27	697	18	5	2	8	20	2	7	18	3	2	33
4	5	49	2	9	567	34	12	15	48	14	33	7	13	6	8
5	3	13	4	6	41	685	12	11	6	31	11	4	16	0	9
6	10	6	3	7	39	16	658	6	22	29	24	9	9	5	60
7	8	31	19	7	34	14	8	595	5	18	11	25	32	1	7
8	18	21	18	18	96	9	15	3	505	5	43	40	8	13	32
9	2	19	1	1	34	16	32	8	1	666	8	1	18	6	8
10	16	14	12	8	77	13	14	6	52	16	615	2	6	5	15
11	11	55	41	16	13	6	1	35	31	1	6	618	8	5	7
12	18	27	4	9	37	45	10	34	8	27	12	4	566	8	6
13	5	5	3	8	22	5	11	1	17	11	18	3	10	699	10
14	51	17	22	38	24	13	24	5	65	3	32	38	8	6	531

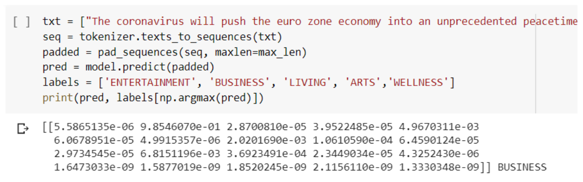
Fig. 18. Confusion Matrix(15 categories)



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0	493	4	23	5	10	35	15	22	19	7	24	37	11	15	15	7	19	10	37
1	14	197	16	4	7	26	20	50	3	15	12	14	8	23	5	2	5	2	5
2	27	12	280	11	15	12	8	7	0	3	8	4	7	10	4	3	2	0	3
3	10	2	6	300	6	1	2	8	0	4	6	1	7	13	8	2	0	3	9
4	4	5	5	9	298	2	7	7	3	2	3	3	2	7	7	4	1	1	19
5	36	13	10	2	4	216	31	4	5	9	13	20	14	6	6	13	5	4	1
6	7	2	1	0	5	5	296	1	0	2	1	17	0	0	0	4	9	1	6
7	20	13	4	5	7	5	14	290	1	4	5	12	1	10	0	13	2	4	10
8	25	9	0	0	4	9	11	4	285	3	8	62	1	3	2	3	1	9	12
9	15	12	1	5	9	6	6	1	0	266	0	19	5	7	2	30	27	1	3
10	55	10	11	6	7	29	5	4	12	3	184	2	6	31	6	3	3	8	18
11	11	5	0	3	0	8	21	6	7	3	1	336	0	0	0	4	6	0	1
12	54	41	36	4	7	53	12	16	3	32	20	7	80	12	7	1	11	4	2
13	5	12	16	20	8	3	5	5	1	8	10	1	7	225	10	1	2	1	7
14	33	2	11	9	3	13	2	3	5	3	4	4	5	12	255	6	0	1	6
15	22	1	2	6	5	29	8	11	0	27	5	6	2	6	5	229	6	1	5
16	30	5	6	3	6	7	26	8	5	18	1	38	4	0	7	8	226	3	4
17	14	3	0	2	3	6	2	5	11	0	7	17	2	2	1	0	8	296	3
18	45	2	5	6	23	3	9	11	29	3	19	6	0	19	5	7	2	4	214

Fig. 19. Confusion Matrix(20 categories)

All three confusion matrices show that large number of predicted values lie in the diagonal. This indicates that the model is a good fit for the data. We also tested new data apart from training and testing set and the model predicted the class correctly for the data. The result is given in fig. 20.



```
[ ]  txt = ["The coronavirus will push the euro zone economy into an unprecedented peacetime
     seq = tokenizer.texts_to_sequences(txt)
     padded = pad_sequences(seq, maxlen=max_len)
     pred = model.predict(padded)
     labels = ['ENTERTAINMENT', 'BUSINESS', 'LIVING', 'ARTS','WELLNESS']
     print(pred, labels[np.argmax(pred)])
```

```
[[5.5865135e-06 9.8546070e-01 2.8700810e-05 3.9522485e-05 4.9670311e-03
  6.0678951e-05 4.9915357e-06 2.0201690e-03 1.0610590e-04 6.4590124e-05
  2.9734545e-05 6.8151196e-03 3.6923491e-04 2.3449034e-05 4.3252430e-06
  1.6473033e-09 1.5877019e-09 1.8520245e-09 2.1156110e-09 1.3330348e-09]] BUSINESS
```

Fig. 20. Testing New Data

VI. Conclusion and Future Work

Thus, the application of LSTM for classification of news into categories has yielded competent accuracy scores. As shown in figure 16, the model with only 5 labels achieved the highest testing accuracy of 71% however, it showed a training accuracy of 95% thereby, displaying a high difference in the two. Thereafter, the model when trained with 15 labels displayed testing and training accuracies of 71% and 70%, respectively and thus the difference was much lesser than the machine with 5 labels. Finally, when 20 labels were used, the training and testing accuracies were 64% and 63% respectively. Thus, although the difference was same as the second model, the accuracy obtained was lower. Therefore, the model performed the best with 15 labels.
In future, character-based approaches, stemming of words and TF-IDF could be tried for achieving better accuracy. Multilayer LSTM can thus be applied in the above way for obtaining a decent accuracy.

References

[1] Z. Wang and B. Song, "Research on hot news classification algorithm based on deep learning," Proc. 2019 IEEE 3rd Inf. Technol. Networking, Electron. Autom. Control Conf. ITNEC 2019, no. Itnec, pp. 2376–2380, 2019, doi: 10.1109/ITNEC.2019.8729020.

[2] X. Bai, "Text classification based on LSTM and attb2ention," in 2018 13th International Conference on Digital Information Management, ICDIM 2018, 2018, pp. 29–32, doi: 10.1109/ICDIM.2018.8847061.

[3] K. Borna and R. Ghanbari, "Hierarchical LSTM network for text classification," SN Appl. Sci., vol. 1, no. 9, p. 1124, Sep. 2019, doi: 10.1007/s42452-019-1165-1.

[4] Y. Zhu, X. Gao, W. Zhang, S. Liu, and Y. Zhang, "A bi-directional LSTM-CNN model with attention for Aspect-level text classification," Futur. Internet, no. 12, 2018, doi: 10.3390/fi10120116.

[5] F. Zhang, W. Gao, and Y. Fang, "News title classification based on sentence-LDA model and word embedding," Proc. - 2019 Int. Conf. Mach. Learn. Big Data Bus. Intell. MLBDBI 2019, pp. 237–240, 2019, doi: 10.1109/MLBDBI48998.2019.00053.

[6] C. Li, G. Zhan, and Z. Li, "News Text Classification Based on Improved Bi-LSTM-CNN," Proc. - 9th Int. Conf. Inf. Technol. Med. Educ. ITME 2018, pp. 890–893, 2018, doi: 10.1109/ITME.2018.00199.

[7] K. Bergen and L. Gilpin, "Negative News No More : Classifying News Article Headlines," 2012.

[8] R. Wang, Z. Li, J. Cao, T. Chen, and L. Wang, "Convolutional Recurrent Neural Networks for Text Classification," Proc. Int. Jt. Conf. Neural Networks, vol. 2019-July, no. 2018, pp. 1–6, 2019, doi: 10.1109/IJCNN.2019.8852406.

[9] G. L. Yovellia Londo, D. H. Kartawijaya, H. T. Ivariyani, P. W. P. Yohanes Sigit, A. P. Muhammad Rafi, and D. Ariyandi, "A Study of Text Classification for Indonesian News Article," Proceeding - 2019 Int. Conf. Artif. Intell. Inf. Technol. ICAIIT 2019, pp. 205–208, 2019, doi: 10.1109/ICAIIT.2019.8834611.

[10] R. Bogery, N. Al Babtain, N. Aslam, N. Alkabour, Y. Al Hashim, and I. U. Khan, “Automatic semantic categorization of news headlines using ensemble machine learning: A comparative study,” Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 11, pp. 689–696, 2019, doi: 10.14569/IJACSA.2019.0101190.

[11] L. Liu, J. L. Priestley, Y. Zhou, H. E. Ray, and M. Han, “A2Text-net: A novel deep neural network for sarcasm detection,” Proc. - 2019 IEEE 1st Int. Conf. Cogn. Mach. Intell. CogMI 2019, pp. 118–126, 2019, doi: 10.1109/CogMI48466.2019.00025.

[12] O. Fuks, “Classification of News Dataset,” pp. 1–6, 2018.

[13] R. Jing, “A Self-attention Based LSTM Network for Text Classification,” in Journal of Physics: Conference Series, 2019, vol. 1207, no. 1, doi: 10.1088/1742-6596/1207/1/012008.

[14] R. Katarya and Y. Arora, “Study on Text Classification using Capsule Networks,” 2019 5th Int. Conf. Adv. Comput. Commun. Syst. ICACCS 2019, pp. 501–505, 2019, doi: 10.1109/ICACCS.2019.8728394.

[15] Y. Du, X. Zhao, and B. Pei, “Short Text Sentiment Classification Based on CNN-LSTM Model,” Beijing Gongye Daxue Xuebao/Journal Beijing Univ. Technol., vol. 45, no. 7, pp. 662–670, 2019, doi: 10.11936/bjutxb2017120035.

[16] A. Basu, C. Walters, and M. Shepherd, “Support vector machines for text categorization,” Proc. 36th Annu. Hawaii Int. Conf. Syst. Sci. HICSS 2003, pp. 1–7, 2003, doi: 10.1109/HICSS.2003.1174243.

[17] A. Conneau, H. Schwenk, Y. Le Cun, and L. Barrault, “Very deep convolutional networks for text classification,” 15th Conf. Eur. Chapter Assoc. Comput. Linguist. EACL 2017 - Proc. Conf., vol. 1, no. 2001, pp. 1107–1116, 2017, doi: 10.18653/v1/e17-1104.

[18] X. She and D. Zhang, “Text Classification Based on Hybrid CNN-LSTM Hybrid Model,” in Proceedings - 2018 11th International Symposium on Computational Intelligence and Design, ISCID 2018, 2018, vol. 2, pp. 185–189, doi: 10.1109/ISCID.2018.10144.

[19] J. Wang, L. C. Yu, K. R. Lai, and X. Zhang, “Dimensional sentiment analysis using a regional CNN-LSTM model,” 54th Annu. Meet. Assoc. Comput. Linguist. ACL 2016 - Short Pap., pp. 225–230, 2016, doi: 10.18653/v1/p16-2037.

[20] J. D. Prusa and T. M. Khoshgoftaar, “Deep neural network architecture for character-level learning on short text,” FLAIRS 2017 - Proc. 30th Int. Florida Artif. Intell. Res. Soc. Conf., pp. 353–358, 2017.

[21] F. Miao, P. Zhang, L. Jin and H. Wu, ”Chinese News Text Classification Based on Machine Learning Algorithm,” 2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, 2018, pp. 48-51.

[22] D. D. Monner and J. A. Reggia, ”Recurrent Neural Collective Classification,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 24, no. 12, pp. 1932-1943, Dec. 2013.

[23] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Neural Comput. 9, 8 (November 1997), 1735–1780. DOI:https://doi.org/10.1162/neco.1997.9.8.1735

[24] Misra, Rishabh. (2018). News Category Dataset. 10.13140/RG.2.2.20331.18729.

[25] Z. Li, W. Shang and M. Yan, ”News text classification model based on topic model,” 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, 2016, pp. 1-5.

[26] J. Han, M. Kamber, and J. Pei, “Advanced Pattern Mining,” Data Mining, pp. 279–325, 2012.