Name - Nihar Sudhanshu Limaye

Program – MCM

Email id – nihar.limaye3@mail.dcu.ie

Project title - Design of a Decision System for Topic Prediction from News Titles

Supervisor name – Dr. Yvette Graham

Design of a Decision System for Topic Prediction from News Titles

Nihar Sudhanshu Limaye
School of Computing,
Dublin City University,
Dublin, Ireland
nihar.limaye3@mail.dcu.ie

Abstract— The motivation of this research arises due to the widespread availability of news titles on the internet. The decision system is needed to classify the news automatically rather than doing it manually by humans. One of the most important tools offered by news aggregators is based on the classification of articles into a fixed set of categories. However, one of the significant problems with online news sets is the categorization of vast amounts of news and articles. To overcome this problem, machine learning models along with Natural Language Processing (NLP) is widely used for automatic news classification to categorize topics of untracked news. This research paper defines a novel comparative approach for predicting news categories from news titles. It compares the performance of traditional frequency-based embeddings like count vectorizer, TF-IDF along with machine learning classifiers to prediction based embeddings like Word2Vec along with its classifiers to predict the exact news category. Results show prediction based embedding is more successful in capturing semantics but produces lower precision/F1 score in classification.

Keywords—NLP, Word2Vec, semantics (keywords)

1.INTRODUCTION

Nowadays on the internet, many sources generate immense amounts of daily news. Furthermore, the demand for information by users has been growing continuously, so the news must be classified to allow users to acquire information of interest quickly and effectively. This way, the machine learning model for automated news classification could be used to identify topics of untracked news and/or offer individual suggestions based on the user's prior interests. Thus, the aim is to design models that take as input news headlines and output news categories.

The automatic categorization of the news corpus will profit society in several ways [17]. However, automatic categorization of the news headlines remains a challenging task as the length of news varies. Natural language processing (NLP) is primarily adopted to automatically categorize documents and speech by word count or frequency. Also, the relevant literature contains a vast number of effective solutions, including statistical models (like linear or logistic regression), data structures (like decision trees and random forests), and neural networks. Another branch of the relevant research comprises works focusing on the problem of feature extraction. Text classification includes many methods previously presented in the literature like Naïve Bayes. Naïve Bayes is a classification method that delivers high accuracy with direct calculation [2]. Moreover, other tools developed deep learning solutions based on convolutional and recurrent neural networks [5]. The traditional methods discussed in the literature are for Indonesian and Arabic news articles and consist of a count vectorizer with popular classifiers like Naïve Bayes, Logistic Regression [2] [7]. Moreover, some of the news articles might not classify properly as the machine learning algorithms are not trained to understand the context behind the news title [4]. Hence, to overcome this challenge Word2Vec and FastText embeddings are used in this research. This work aims to answer the two following research questions:

a. *How efficiently trained ML models can predict the news categories by analysing news titles?*
b. *Can prediction-based word embeddings improvise the result over traditional frequency-based embeddings?*

1.1 What is Word2Vec[1] embedding?

Word2vec is used to construct a vector representation of the words. This representation can catch the semantic similarity between words, and we can use these vectors to extract the classification features. The result utilizing this semantic feature improves the previous one that used the lexical feature from bag-of-words and TF-IDF [4].

1.2 What is FastText[2] embedding?

FastText is an open-source, free, lightweight library that allows users to learn text representations and text classifications. FastText, a fast and effective method to learn word representations and perform text classification. The primary objective of the FastText embeddings is to take into consideration the internal structure of words instead of learning word representations. This is remarkably effective for morphologically rich languages so that the representations for different morphological forms of words would be learned independently [15].

In this research, a comparative study is performed for news titles classification. Initially, the performance of traditional methods count vectorizer and TF-IDF along with ML classifiers is tested. The later introduction of prediction-based word embeddings along with classifiers to improvise the results of earlier methods.
The remainder of the paper consists of a literature review (section 2) which comments about previous studies done to find news categories, a piece of information on the dataset (section 3), the experimental setup of the system (section 4)

[1] https://code.google.com/archive/p/word2vec/

[2] https://fasttext.cc/

with an explanation to every block present in it. Evaluation and results (section 5) mention the working of a system and results. A final summary of this paper in the conclusion section and future enhancements.

2. LITERATURE REVIEW

This section presents a literature review of the most significant studies related to the research topic. These topics are feature engineering, such as word embedding and machine learning classifiers for NLP analysis.

2.1 Feature Engineering

Feature engineering is an essential part of building an intelligent system. As Andrew Ng says [19]:

"Coming up with features is difficult, time-consuming, and requires expert knowledge.' Applied machine learning' is feature engineering."

One of the essential steps in NLP is applying feature engineering for the dataset, first to preserve the context of the text and secondly to reduce the vector's dimensionality of the text [6]. The word embeddings and semantic networks like Word2Vec can help in preserving the context which is discussed in the following sections.

i) Frequency-based word embedding: As the Bag of Words (BoW), term frequency, TF-IDF all represents the words in the vector space model for the learning algorithm and takes less time for computation but they also have some drawbacks and those are mentioned in this section. This research is inspired by the work done in [5]. A supervised classification method for news articles that analyzes the titles and constructs multiple types of inputs including single words and n-grams of variable sizes. The tokens and their scores are stored in a support structure subsequently used to classify the unlabeled articles and provide 95% accuracy. The feature extraction methods used in [5] does not preserve the context and the relationships of the words in the documents. This research utilizes methods mentioned in [5] as one of the possible ways to extract features from the dataset and also looks into the semantic side to obtain the best possible features that act as input to ML classifiers.

Several studies have been established on comparative analysis of word embedding techniques in various domains i.e. biomedical, twitter elections [8]. There are two particular approaches for prediction-based word embedding that are known for their efficiency, accuracy, and for capturing semantics are Word2Vec and FastText.

ii) Prediction based word embedding: In Mikolov et. al [8], a model was proposed for Word2Vec which includes two architectures to learn and represent the words in the vector space. The first architecture is the continuous bag-of-words (CBOW) and second the skip-gram architecture. They performed word analogy to assess their model. The word analogy was based on semantic and syntactic questions that were produced by the authors. The authors observed superior accuracy for both semantics and syntactic questions when both dimensional size and the number of training vocabulary were increased. But Word2vec only considers the context locally in a document without making the advantages of the occurrence in various documents.

To overcome the previously mentioned problem, Stein, R.et al. [14] proposed FastText Model. FastText (which is essentially an extension of Word2vec model), treats each word as composed of character n-gram. So the vector is a combination of multiple n-grams. It similarly calculates embeddings as the Continuous bag of words (CBOW) model does [9], but with the label as the middle word and a bag of n-grams rather than a bag of words. The experiments were conducted in [14] using these FastText word-embedding models on the IMDB dataset to achieve 89.30% precision.

2.2 Classification

The second task is to classify the news articles basis the feature extracted using vectors. The machine learning models are specifically designed for classification. The approaches that design statistical models typically identify possible strong relationships between the words of a title and the provided categories.

Al-Tahrawi et al. 2015 [7] proposed the use of logistic regression (LR) classifier with a news category dataset. It has recorded a precision of 96.5% on one category and above 90% for three out of the five categories. Another research [2] used Naïve Bayes (NBC) on 250 news articles divided into five categories as learning documents. The trial results of [2] showed that the system can generate such accuracy in delivering news articles classification with the average recall value of 92.87% and precision value of 91.16%. Another research mentioned in Kabir, F et.al 2015 [12] uses Stochastic Gradient Descent (SGD) to classify Bangla news titles and proposes that SGD outperform compared to NBC and LR.

The hybrid model achieved the highest outcome with 75-25 data sampling divisions for training and testing. Pambudi et al. [10] classified the Indonesian news into a multi-class classification using Pseudo Nearest Neighbour (PNNR). Several proximity functions were used, and Cosine proximity similarity measure produced the highest results as compared to Manhattan and Euclidean. The research includes some of the methods such as traditional frequency-based embedding with NBC, LR, and SGD for the categorization of news. Also in the later part of the research, the Word2Vec and FastText embeddings are used for adding semantic-based features.

3. METHODOLOGY

The data source with the name as 'News Aggregator' is a UCI machine learning dataset [11] that contains almost 422,937 news titles from 10-March-2014 to 10-August-2014. News in this dataset belongs to four different topics (labels) namely *technology*, *business*, *health*, and *entertainment*. Each news record consists of several attributes from which we are using only 'Category','Title', and 'Publisher' in this analysis. Also, it is a combination of data attributes 'Title' and 'Publisher' into the single attribute 'Text' as the input data for classification. The distribution of the data in each of the above-mentioned categories is as shown in Fig 1. As we can see the data across categories are evenly undistributed.

Firstly, concerning the unbalanced data set, a problem often found in the real-world application can cause serious negative effects on the classification performance of machine learning algorithms.

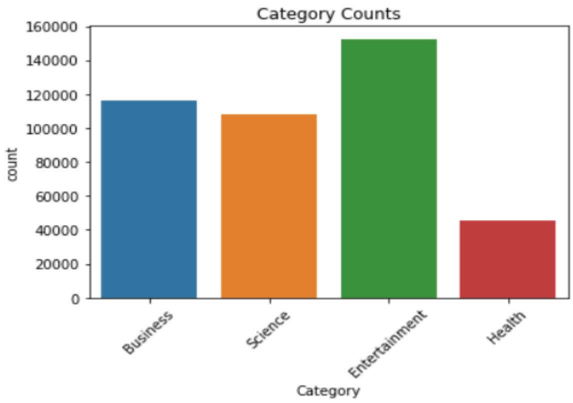


Fig 1. Unbalanced category-wise distribution of the news in the dataset

Hence to overcome this issue the dataset is balanced first across all the categories and then used for evaluation. The method used to balanced the dataset is as follows:

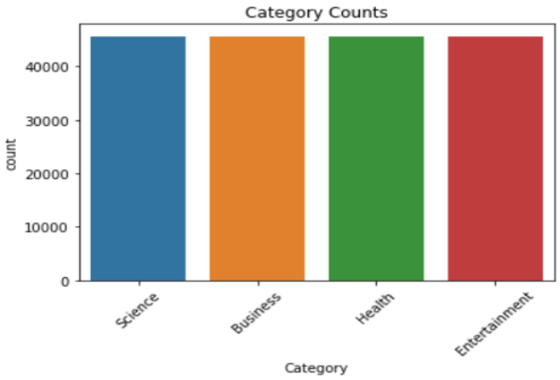


Fig 2. Balanced category-wise distribution of the news in the dataset

As referred to in Fig.1 the health category only contains 44,000 records hence this research takes random samples of 44,000 each from the remaining three categories and does the analysis on 176,000 rows of the dataset. The balanced dataset with an equal number is shown in Fig.2

3.1 Steps for implementing word embeddings

The approach in this research adopted is similar to the baseline model discussed in the literature, but there are several additional changes implemented that provided better results.

a. Combination of ‘Title’ column with ‘Publisher’ in a new column ‘Text.’
b. ‘Text’ acts as an input data cleaning block mentioned in 4.1. The blank values and duplicates are removed in this step.
c. A data pre-processing step removes punctuation, stopwords, lowercasing, and generates tokens. This research includes the use of lemmatization (mentioned in 4.2.3) in the pre-processing stage.
d. The length of tokens taken under consideration is from length 2 to length 14 and unique tokens in a sentence are considered for analysis.
e. The tokenized text is given to the embeddings mentioned in 4.4. with n-gram range for converting text to vectors so that ML models can classify it.
f. Finally, the vectorized data is given to machine learning classifiers. In this research, three machine learning classifiers are used namely naive bayes [2], logistic regression [7], SGD [12] for classification. The results are discussed in the later section of this research.

In this research, the classification is performed on the entire dataset. The dataset is divided into 70-30 with random samples. The models are trained on a training dataset (70% of the dataset) and the performance is tested on (30 % of the dataset). The evaluation is carried out by analyzing the precision for all models and then some additional metrics are also considered for performance evaluation.

4. EXPERIMENTAL SETUP

This research uses some of the states of art methods discussed in the literature and proposes design as indicated below for news categorization.

The setup of the system is as follows.

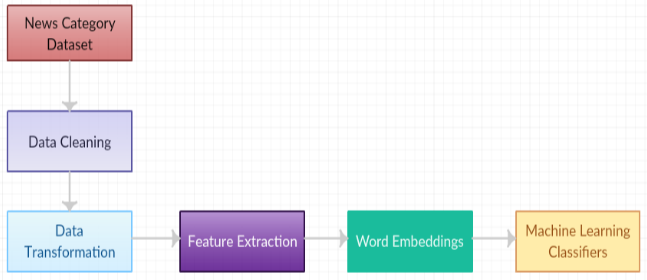


Fig 3. An overview of the proposed setup for news category prediction.

4.1. Data Cleaning

This block is a Python script. The script is responsible for doing the below tasks. The block ensures that the data is cleaned so that the result won’t be distorted at the end.

- Convert Date Time in the correct format. i.e *dd-mm-yyyy*;
- Removal of duplicate entries;
- Removal of blank/missing entries and punctuations.

4.2. Data transformation

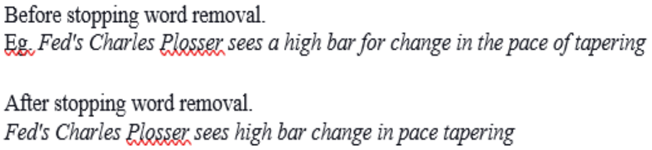
As cleaned data is passed to the adjacent block ‘Data Transformation.’ It ensures data is in the correct format and features can be extracted. It involves the checks described below.

4.2.1 Tokenization

It is a process that splits the sentences in several words referred to as tokens. These tokens are the fragments of a larger sentence represented as an array of words. The tokenization can be done manually or using predefined tokenizers. The final output is then served as input for further processing of text mining.

4.2.2 Stop Word removal

Stop words represent the English words that do not add much meaning to a sentence. They can safely be overlooked without sacrificing the meaning of the sentence. Words like *the*, *he*, *have*, for example.



Before stopping word removal.

Eg. Fed's Charles Plosser sees a high bar for change in the pace of tapering

After stopping word removal.

Fed's Charles Plosser sees high bar change in pace tapering

Fig 4. Stop word removal

4.2.3 Lemmatization

Lemmatization, unlike stemming, reduces the inflected words properly ensuring the root word belongs to the language. In lemmatization, the root word is called a lemma. A lemma (plural lemmas or lemmata) is the canonical form, dictionary form, or citation form of a set of words.

4.2.4 Label Encoding

Machine learning models require numeric features and labels to provide a prediction. For this reason, we must create a dictionary to map each label to a numerical ID. I have created this mapping scheme:

Category Name	**Category Code**
Entertainment	0
Business	1
Technology	2
Health	3

Table 1: Label encoding

4.3 Feature Extraction

First, using the pre-processed news title unique words are counted in a corpus. The total number of unique words is around 60,000 then, I extracted the preceding word features for the classification task.

4.3.1 Number of words in a single news title

The number of words in each sentence is considered excluding the white spaces. The average word count of each category is 150. The rows word count above 150 is ignored before giving it to the classifier.

4.3.2 Dataset column merged

Initially, the *Title* column is considered to predict the news category but at a later point in the experiment. The *Publisher* column is merged with the *Title* to observe if that makes any improvements in classification.

4.4 Word embedding

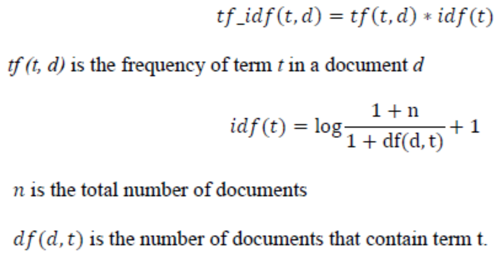
Word embedding is the collective name for a set of language modeling and feature learning techniques in natural language processing where words or phrases from the vocabulary are mapped to vectors of real numbers [16]. Words with similar meanings typically yield numerical word-vectors with a cosine similarity closer to one than 0.

4.4.1 Frequency-based embedding

Count vectorizer - These are the similar vectorizer models that learn vocabulary from all of the documents, then model each document by counting the number of times each word appears. For example, consider we have T titles and W is the number of different words in our vocabulary then the size of the count vector-matrix will be given by T*W. Besides, it includes the functionality of using the n-gram range. For instance, San Francisco (is a 2-gram). The Three Musketeers (is a 3-gram), She stood up slowly (is a 4-gram) [18]. With larger n, a model can store more context with a well-understood space-time tradeoff, enabling small experiments to scale up efficiently.

For example, the word company may be correlated with many categories, like Economics, Politics, or even Society. For this reason, single words are not adequate to identify the desired relationships, and more complex representations are required. To overcome this issue, apart from single words, research also utilizes n-grams, which, in contrast to single title words, have a smaller probability of being ambiguous.

TF-IDF vectorizer - To re-weight the count features into floating-point values suitable for usage by a classifier, it is very common to use the TF-IDF transform. This method takes into account, not just the occurrence of a word in a single document but in the entire corpus.



$$tf\_idf(t,d) = tf(t,d) * idf(t)$$

tf (t, d) is the frequency of term *t* in a document *d*

$$idf(t) = \log\frac{1+\mathrm{n}}{1+\mathrm{df(d,t)}} + 1$$

n is the total number of documents

$df(d,t)$ is the number of documents that contain term t.

Fig 5. TF-IDF equation

Both the frequency-based embeddings are implemented on the existing dataset. The performance also tested on unseen news articles randomly picked from the web. One of the problems found with randomly picked news titles is that the above-mentioned setup failed to obtain the context behind the news title hence wrong classification.

eg. *Samsung to pay Apple $539 million in iPhone patent case*

This title is tagged as ‘Technology’ by the classifier. The true label was ‘Business.’ The news seems to be technology news but in this context it is business. Hence, to overcome this issue, the research also introduces prediction based embeddings.

4.4.2 Prediction based embedding

Word2Vec - Word2vec is a tool based on deep learning and released by Google in 2013 [1]. It provides an efficient implementation of the CBOW and Skip-gram architectures for computing vector representations of words and these representations can then be used in many natural language processing applications and for further research.

FastText - Stein, R.et al. [14] proposed the FastText Model.

The word vector in the FastText model is represented by the n-gram range rather than just a bag of words. It generates better word embeddings for rare words (even if words are rare, their character n-grams are still shared with other words - hence the embeddings can still be good. Out of vocabulary words can be handled by FastText to construct the vector for a word from its character n-grams even if the word doesn't appear in the training corpus.
There are two methods to apply prediction based embeddings.

a) Custom Vectors - This type of method involves the training own word vectors from the corpus but it takes lots of CPU power and time [23]
b) Pre trained vectors - An alternative to custom vectors is to use an existing pre-trained word embedding. Word2Vec and FastText have existing pre-trained vectors trained on huge corpus.

This research uses both the methods to get the best possible results.

4.5 Machine learning classifiers

Classification is the process of predicting the class of given data points [15]. The predesigned algorithms implemented for classification are machine learning classifiers. The performance of the classifiers is investigated on different hyperparameters using grid search and the best parameters are chosen for analysis. This classification task starts with a training set T= (t1, t2, …, tn) of documents that are already labelled with classes C1, C2,.. (e.g., Entertainment, Health). Then, a classification model which can assign the correct class 'Cj' to a new document 'ti'. Text classification is of two types: single label and multi-label. A single label document belongs to only one class and a multi-label document may belong to more than one class. In this project, only consider a single label document classification as each news title is only having a single label.

4.5.1 Algorithms

The three machine learning classifiers are used in this research including some of the traditional ML models and deep learning models.

Naive Bayes:
The Naive Bayes classifier is the base model in this research. As per Asy'arie et al. [2] it produced a state of the art result in classification for Indonesian news articles. Hence the same model is used in this research to implement English news title classification. In this case, I have not tuned any hyperparameter.
Logistic regression:
Logistic regression is also the base model in this research. As per Al-Tahrawi et al. [7], it produced a state of the art result in classification for Arabic text categorization. Hence the same model is used in this research to implement English news titles. The list of hyperparameters has tuned are:

Hyperparameter	**Brief Description**
C	The inverse of regularization strength. Smaller values specify stronger regularization
multi_class	We'll choose multinomials because this is a multi-class problem.
solver	Algorithm to use in the optimization problem. For multiclass problems, only newton-cg, sag, saga, and lbfgs handle a multinomial loss.
class_weight	Weights associated with classes.
penalty	Used to specify the norm used in the penalization. The newton-cg, sag and lbfgs solvers support only l2 penalties

Table 2. LR Hyperparameters

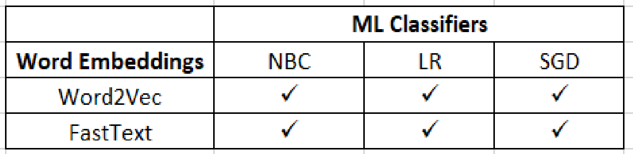
Stochastic Gradient Descent (SGD):
Linear SGD is the most naive algorithm for classification. In this algorithm, the gradient descent approach of gradually increasing or decreasing parameters to achieve the goal [12]. With the combination of linear regression, randomly initialize the parameters and compute accuracy through error function. The list of hyperparameters I have tuned are:

Hyperparameter	**Brief Description**
n_jobs	The number of CPUs to use to do the OVA (One Versus All, for multi-class problems) computation. -1 means using all processors.
max_iter	The maximum number of passes over the training data (aka epochs). It only impacts the behavior in the fit method and not the partial_fit method.
alpha	Constant that multiplies the regularization term. The higher the value, the stronger the regularization.

Table 3 . SGD Hyperparameters

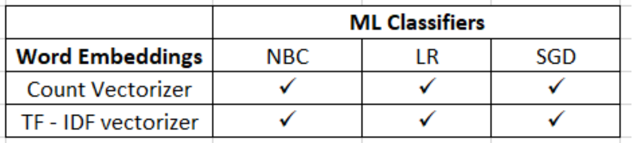
Long Short Term Memory (LSTM) -
LSTM to solve the classification problem of large-scale news text. To better extract the characteristics of the text. In this paper, the Bi-LSTM model is used to obtain the representation of two directions, and then the two directions representations are combined into a new expression through the convolutional neural network.

The combination of classifiers used with both frequency-based embeddings and prediction based embeddings are as follows:



	ML Classifiers		
Word Embeddings	NBC	LR	SGD
Word2Vec	✓	✓	✓
FastText	✓	✓	✓

Table 4. Prediction based word embeddings with classifiers



	ML Classifiers		
Word Embeddings	NBC	LR	SGD
Count Vectorizer	✓	✓	✓
TF - IDF vectorizer	✓	✓	✓

Table 5. Frequency-based word embeddings with classifiers

5. EVALUATION

5.1 Results

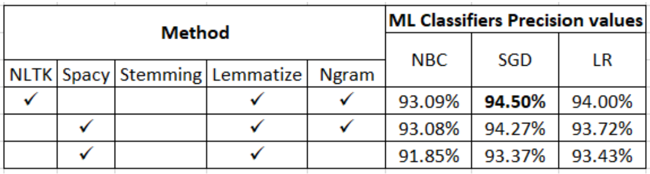
When performing classification problems, some metrics can be used to gain insights to check how the model is performing. Some of them used in this research are:

Accuracy: The accuracy metric measures the ratio of correct predictions over the total number of instances evaluated [22].
Precision: Precision is used to measure the positive patterns that are correctly predicted from the total predicted patterns in a positive class [22].
Recall: Recall is used to measure the fraction of positive patterns that are correctly classified [22].
F1-Score: This metric represents the harmonic mean between recall and precision values [22].
Cohen’s kappa: This function computes Cohen’s kappa [20], a score that expresses the level of agreement between two annotators on a classification problem.
Balanced Accuracy: The balanced accuracy [20] in binary and multiclass classification problems to deal with imbalanced datasets. It is defined as the average recall obtained in each class.
Matthews correlation coefficient (MCC): The Matthews correlation coefficient [21] is used in machine learning as a measure of the quality of binary and multiclass classifications [21]. It takes into account true and false positives and negatives and is generally regarded as a balanced measure that can be used even if the classes are of very different sizes.

I have considered precision when comparing models with each other for performance. Once I got the best pair of ML classifiers to the word embeddings (a reference to Table 4. & Table 5.) Then I calculated Cohen's Kappa score, Balanced Accuracy, and MCC scores for each of the combinations. I also obtained the confusion matrix and classification report (which computes precision, recall, and F1-score for all the classes) for the top scorer pair of ML classifiers and word embedding.

5.1.1 Results for Countvectorizer

Countvectorizer is the basic word embedding used in this research. The performance of the count vectorizer was tested with three ML classifiers namely NBC, SGD, LR, However, before classification, the count vectorizer performance was tested with different pairs of methods mentioned in Table 5. The results showed the best pair of count vectorizer was when the SGD classifier provided the lemmatizing. n-gram methods are implemented along with the natural language toolkit [3](NLTK). The highest precision 94.50% was observed and that is highest compared with other results.

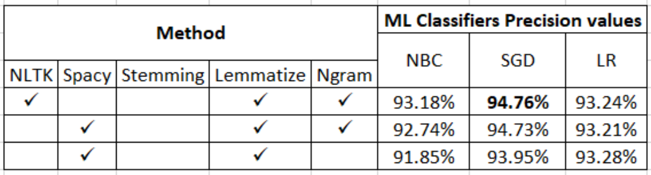


Method					ML Classifiers Precision values		
NLTK	Spacy	Stemming	Lemmatize	Ngram	NBC	SGD	LR
✓			✓	✓	93.09%	**94.50%**	94.00%
	✓		✓	✓	93.08%	94.27%	93.72%
	✓		✓		91.85%	93.37%	93.43%

Table 6. Results of Countvectorizer

5.1.2 Results for TF-IDF vectorizer

The following frequency-based word embedding tested with classifier is TF-IDF in this research. As mentioned in section 5.1.2.



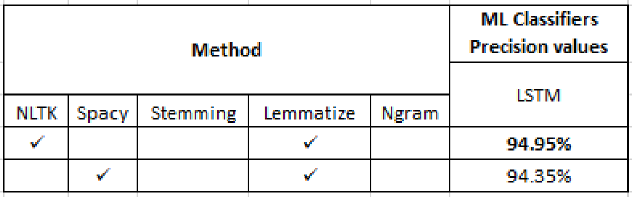
Method					ML Classifiers Precision values		
NLTK	Spacy	Stemming	Lemmatize	Ngram	NBC	SGD	LR
✓			✓	✓	93.18%	**94.76%**	93.24%
	✓		✓	✓	92.74%	94.73%	93.21%
	✓		✓		91.85%	93.95%	93.28%

Table 7. Results for TF-IDF

The TF-IDF performs a scoring system and gives more scores to the words that are less frequent by considering the occurrence in the overall corpus. It is used with spaCy[4] to perform sentence segmentation in a dataset. The results of spaCy along with SGD generated good results with precision up to 94.76%.

5.1.3 Results for LSTM

The bi-directional LSTM neural network model handles a substantial amount of data efficiently [13]. The LSTM is employed along with preprocessing techniques like NLTK and lemmatize to get precision up to 94.95%.



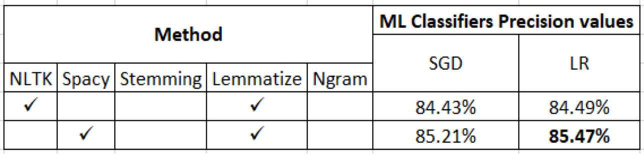
Method					ML Classifiers Precision values
NLTK	Spacy	Stemming	Lemmatize	Ngram	LSTM
✓			✓		**94.95%**
	✓		✓		94.35%

Table 8. Results for LSTM

5.1.4 Results for Word2Vec embedding

a) Results of pre-trained word embeddings

To implement contextual information Word2Vec embedding is used. Word2Vec provided less precision compared to methods like count vectorizer, TF-IDF, and LSTM but was successful in capturing the semantics with precise categorization. Meanwhile, it provided the highest precision with a combination of spaCy, a lemmatizer, n-gram, and LR classifier.



Method					ML Classifiers Precision values	
NLTK	Spacy	Stemming	Lemmatize	Ngram	SGD	LR
✓			✓		84.43%	84.49%
	✓		✓		85.21%	**85.47%**

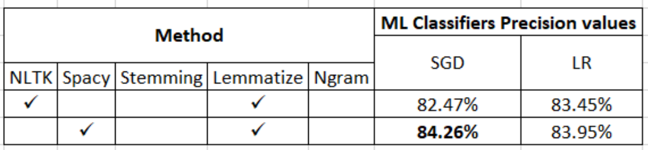
Table 9. Results for pre-trained Word2Vec

b) Results of custom word embeddings

The research additionally uses custom word embeddings. Rather than using already developed word vectors on a different dataset, this research produces its own word vectors from its corpus and makes a comparison with pre-trained word embeddings. Custom word embeddings produce good results when paired with a lemmatizer, n-gram, and SGD classifier.

[3] https://www.nltk.org/

[4] https://spacy.io/



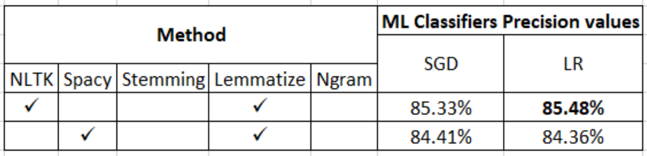
Method					ML Classifiers Precision values	
NLTK	Spacy	Stemming	Lemmatize	Ngram	SGD	LR
✓			✓		82.47%	83.45%
	✓		✓		**84.26%**	83.95%

Table 10. Results for custom Word2Vec

5.1.5 Results for FastText embedding

a) Results of pre-trained word embeddings

FastText performs similarly compared to Word2Vec. However, FastText takes more time for computation compared to Word2Vec but works well without vocabulary words. FastText provided better results when paired with NLTK, a lemmatizer, and LR as ML classifiers.

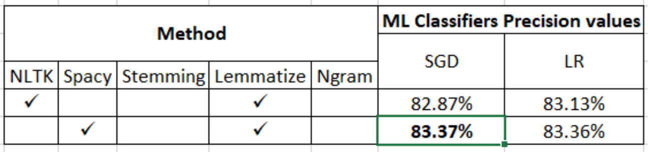


Method					ML Classifiers Precision values	
NLTK	Spacy	Stemming	Lemmatize	Ngram	SGD	LR
✓			✓		85.33%	**85.48%**
	✓		✓		84.41%	84.36%

Table 11. Results for pre-trained FastText

b) Results of custom word embeddings

The custom word embeddings are also developed to work with FastText similar to developed for Word2Vec and produced good results with spaCy, a lemmatizer, n-gram, and SGD classifier.



Method					ML Classifiers Precision values	
NLTK	Spacy	Stemming	Lemmatize	Ngram	SGD	LR
✓			✓		82.87%	83.13%
	✓		✓		**83.37%**	83.36%

Table 12. Results for custom FastText

The method obtained the features by summing all the vector representations of the tokens that exist in the title and dividing it by the total number of vectors (tokens). By that, we would have the average vector served as a feature vector. The length of this feature vector would be the same with the length of word vector from Word2vec & FastText, if we choose a length value of 300 for word vector, then we would also have a 300-dimensional feature vector.

5.2 Discussion

This section compares the performance of the experiments of this research and also manages to compare it with similar results experiments performed in the literature. Fig 6. Show the best pair of word embeddings with the classifiers. Fig 6. Also, show that the pre-trained word embeddings achieved more precision score when compared with custom word embeddings and this is due to small corpus size. Word2Vec and FastText perform well with a huge vocabulary in place. The SGD performs well with TF-IDF, Countvectorizer, custom word embeddings and LR with pre-trained Word2Vec and FastText for category prediction in the news aggregator dataset. So as mentioned in 5.1 additional performance metrics are implemented on the best performing ML classifier and word embedding. In this case, it is TF-IDF and SGD. The results displayed in Table 13. Shows that the balanced accuracy is 94.79 for TF-IDF and SGD so that means the average precision calculated in section 5.1.2 is validated and we can say the combination of TF-IDF and SGD outperformed other pairs. Also, the MCC of TF-IDF and SGD that represent the quality of prediction is still second best compared to other methods. Hence, the confusion metric and classification report is generated for the best pair.

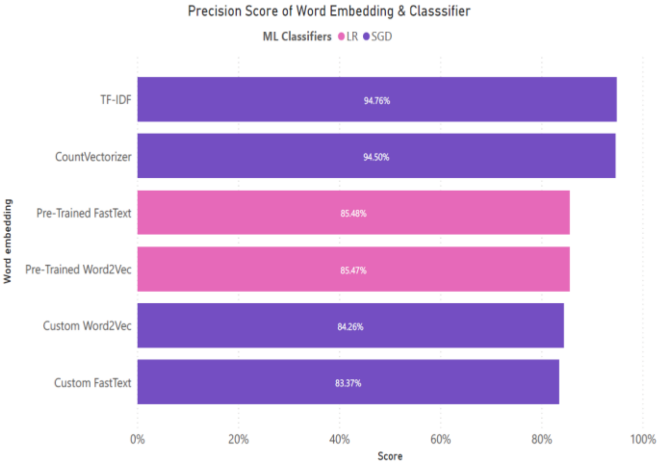
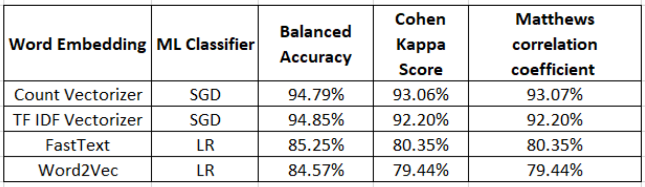


Fig 6. The best combination of word embedding and classifiers



Word Embedding	ML Classifier	Balanced Accuracy	Cohen Kappa Score	Matthews correlation coefficient
Count Vectorizer	SGD	94.79%	93.06%	93.07%
TF IDF Vectorizer	SGD	94.85%	92.20%	92.20%
FastText	LR	85.25%	80.35%	80.35%
Word2Vec	LR	84.57%	79.44%	79.44%

Table 13. Performance check with additional metrics

Below Fig 7. & Fig 8. are the confusion matrix and classification report of TF-IDF and SGD it clearly shows that the Entertainment category was easy to predict. The titles that have true and predicted labels as Entertainment are highest i.e. 13,434 and remaining 420 are misclassified.

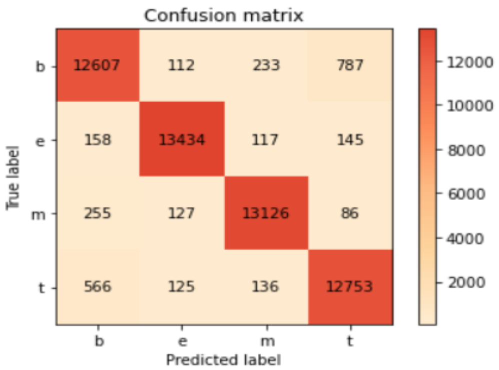
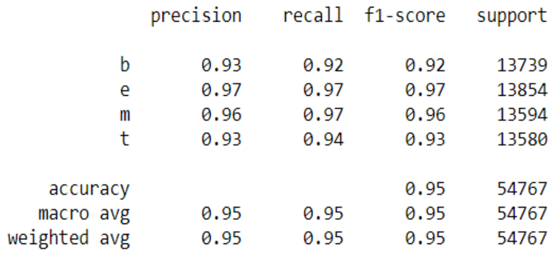


Fig 7. Confusion matrix

However, the classification report clearly shows business and technology categories having fewer precision scores when compared with the other two categories. It is believed that this is because the genre of the news titles most similar to the vector of the target title obtained in the calculation process was a technology article. In this method, it is only the calculation process to consider the genre in the label assignment. In the process of producing a vector, only the features of the text of the article are acquired. Therefore, depending on the target article, there is a possibility that labels assigned by calculation may contain features of articles of other genres.

The experiments performed in this paper are also compared with similar experiments in the literature. Some of the cases are discussed in the below section:

[12] Used TF-IDF along with SGD to classify Bangla news articles by testing on 9,127 news articles. [12] Additionally managed to gain a maximum of 93.86% precision.



```
              precision    recall  f1-score   support

           b       0.93      0.92      0.92     13739
           e       0.97      0.97      0.97     13854
           m       0.96      0.97      0.96     13594
           t       0.93      0.94      0.93     13580

    accuracy                           0.95     54767
   macro avg       0.95      0.95      0.95     54767
weighted avg       0.95      0.95      0.95     54767
```

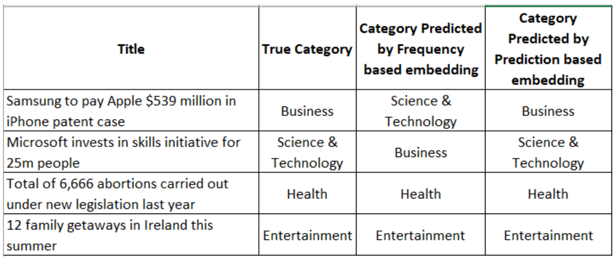
Fig 8. Classification report

In contrast, my work demonstrates techniques mentioned in [12] but performs experiments on the English news title dataset by implementing TF-IDF and SGD in addition to NLTK, a lemmatizer, and n-gram to score 94.76% precision displayed in Table 7.

The inspiration for this research, [5], achieved 95% accuracy with their supervised news classifier (SNC) method. However, in this current research, I first balance the data amongst four categories before using the n-gram method proposed in [5] along with TF-IDF, NLTK, and SGD classifiers to reach 94.76%, only a marginal drop in precision compared to [5].

5.3 Error analysis

To indicate the degree to which each approach can work in practice, I additionally evaluated Word2Vec and FastText in terms of how well they classify unseen news titles randomly picked from the web. Due to Word2Vec and FastText having the property to capture semantics, it predicted true tags for news titles.



Title	True Category	Category Predicted by Frequency based embedding	Category Predicted by Prediction based embedding
Samsung to pay Apple $539 million in iPhone patent case	Business	Science & Technology	Business
Microsoft invests in skills initiative for 25m people	Science & Technology	Business	Science & Technology
Total of 6,666 abortions carried out under new legislation last year	Health	Health	Health
12 family getaways in Ireland this summer	Entertainment	Entertainment	Entertainment

Table 14. Results for unseen news title

6. CONCLUSION & FUTURE WORK

Both frequency-based embeddings and prediction based embeddings have been applied in this experiment to solve single-label classification problems for news aggregator dataset. The result shows that the frequency-based embeddings approach provided better results than prediction based embeddings. The primary reason behind the poor performance of these predictions based embeddings is due to the absence of keywords in pre-trained word vectors. The news aggregator dataset might be compiled manually with each news category labelled against the new title in the dataset. Most of them consist of the words that are either slang, proper nouns, past tense or plural words that are not present in Word2Vec, and FastText pre-trained embedding vector vocabulary. The secondary reason is related to the size of the dataset. Word2Vec & FastText failed to capture word relationships in the embedding space with limited information.

This research provides answers to the first research question *How efficiently trained ML models can predict the news categories by analysing news titles?* The ML trained models performed well for the news aggregator dataset when combined with frequency-based embedding. The SGD produced 94.50% precision and 94.79% balanced accuracy with a lemmatizer, n-gram, count vectorizer. And 94.76% precision 94.85% balanced accuracy with a lemmatizer n-gram and TF-IDF.

The second research question of this paper is *Can prediction-based word embeddings improvise the result over traditional frequency-based embeddings?* As per the experiments performed in this research, I can say that frequency-based embeddings are still better than prediction-based word embeddings. They provided good results when compared to prediction-based embeddings (a reference to section 5.1) on the news category dataset.

The model will be improved by mending the corpus to make it more ideal, which is more representative. It can be performed by including more data, including multilabel data to increase label correlation. The solution needs to be found for unseen news articles that don't belong to any of the categories present in the dataset. The model shouldn't categorize unseen news titles in one of the categories of the dataset. Although this model still needs some improvements, I believe my model can aid readers to easily filter the news titles.

7. APPENDIX

7.1 Python Scripts

Scripts used in this research to perform analysis

Python Scripts: News Categorization

REFERENCES

[1] Mikolov, T., Chen, K., Corrado, G., and Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[2] Asy'arie, A.D. and Pribadi, A.W., 2009, December. Automatic news articles classification in the Indonesian language by using naive bayes classifier method. In Proceedings of the 11th International Conference on Information Integration and Web-based Applications & Services (pp. 658-662).

[3] Rahmawati, D. and Khodra, M.L., 2015, August. Automatic multi-label classification for Indonesian news articles. In the 2015

2nd International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA) (pp. 1-6). IEEE.

[4] Rahmawati, D. and Khodra, M.L., 2016, August. Word2vec semantic representation in multilabel classification for Indonesian news articles. In 2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA) (pp. 1-6). IEEE.

[5] Akritidis, L., Fevgas, A., Bozanis, P. and Alamaniotis, M., 2019, July. A Self-Pruning Classification Model for News. In 2019 10th International Conference on Information, Intelligence, Systems, and Applications (IISA) (pp. 1-6). IEEE.

[6] Bogery, R., Al Babtain, N., Aslam, N., Alkabour, N., Al Hashim, Y. and Khan, I.U., Automatic Semantic Categorization of News Headlines using Ensemble Machine Learning: A Comparative Study.

[7] Al-Tahrawi, M.M., 2015. Arabic text categorization using logistic regression. IJ Intelligent System and Applications, 6, pp.71-78.

[8] X. Yang, C. Macdonald, and I. Ounis, “Using word embeddings in Twitter election classification,” Inf. Retr. J.vol. 21, no. 2–3, pp. 183–207, 2018.

[9] T. Mikolov, G. Corrado, K. Chen, and J. Dean, “Vector Space,” pp. 1–12.

[10] R. A. Pambudi, Adiwijaya, and M. S. Mubarok, “Multi-label classification of Indonesian news topics using Pseudo Nearest Neighbor Rule,” J. Phys. Conf. Ser., vol.1192, no. 1, 2019.

[11] UCI News Aggregator UCI Machine Learning Repository: News Aggregator Data Set

[12] Kabir, F., Siddique, S., Kotwal, M.R.A. and Huda, M.N., 2015, March. Bangla text document categorization using stochastic gradient descent (sgd) classifier. In 2015 International Conference on Cognitive Computing and Information Processing (CCIP) (pp. 1-4).

[13] Li, C., Zhan, G. and Li, Z., 2018, October. News text classification based on improved Bi-LSTM-CNN. In 2018 9th International Conference on Information Technology in Medicine and Education (ITME) (pp. 890-893). IEEE.

[14] Stein, R.A., Jaques, P.A. and Valiati, J.F., 2019. An analysis of hierarchical text classification using word embeddings. Information Sciences, 471, pp.216-232.

[15] Bojanowski, Piotr and Grave, Edouard and Joulin, Armand and Mikolov, Tomas. Enriching Word Vectors with Subword Information. arXiv preprint arXiv:1607.04606

[16] Wikipedia. (2020). Word embedding. [online] Available at: https://en.wikipedia.org/wiki/Word_embedding.

[17] Bogery, R., Al Babtain, N., Aslam, N., Alkabour, N., Al Hashim, Y. and Khan, I.U., Automatic Semantic Categorization of News Headlines using Ensemble Machine Learning: A Comparative Study.

[18] Mohammad Alshaer. An Efficient Framework for Processing and Analyzing Unstructured Text to Discover Delivery Delay and Optimization of Route Planning in Realtime. Data Structures and Algorithms [cs.DS]. Université de Lyon; Université Libanaise, école doctorale des sciences et technologies, 2019. English. ffNNT : 2019LYSE1105ff. fftel-02310852f

[19] Ng, Andrew. “Machine Learning and AI via Brain simulations”. Stanford University. https://ai.stanford.edu/~ang/slides/DeepLearning-Mar2013.pptx

[20] J. Cohen (1960). “A coefficient of agreement for nominal scales”. Educational and Psychological Measurement20(1):37-46. doi:10.1177/001316446002000104.

[21] Jurman, G., Riccadonna, S. and Furlanello, C., 2012. A comparison of MCC and CEN error measures in multi-class prediction. PloS one, 7(8), p.e41882.

[22] Sokolova, M., Japkowicz, N. and Szpakowicz, S., 2006, December. Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In Australasian joint conference on artificial intelligence (pp. 1015-1021). Springer, Berlin, Heidelberg.

[23] PhD, J. B. (2019, August 7). How to Develop Word Embeddings in Python with Gensim. Retrieved from machinelearningmastery: https://machinelearningmastery.com/develop-word-embeddings-python-gensim/