International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 04 | April - 2022 Impact Factor: 7.185 ISSN: 2582-3930

Attention Based Encoder Architecture for Automatic Text Classification: A Case Study on Text-news Classification

Joydeep Sinha Chowdhury[1], Tanmoy Roy[2]
[1]Birla Institute of Technology and Science, Pilani, Rajasthan, India,
[2]Independent Researcher, Kolkata, India

Abstract.
Text classification is a classical problem of natural language processing where given texts are classified into relevant classes. Text classification techniques are extensively used in finding news categories, search engine optimization, automated textual response and many more. With the exponential growth of digital data, it is inevitable that an automatic classification of documents is needed for efficient information retrieval and document archival. Over the years, text classification problems are approached by using classical machine learning techniques like *naive bayes*(NB), *support vector machine*(SVM), *logistic regression*(LR), *random forest*(RF) and others. In this article, an *Attention* based *deep learning* (DL) model is proposed and applied to the news corpus and then the results are compared with the results of classical machine learning models. A comparative analysis of classification accuracy of the DL model with prominent models like LR, RF and NB are presented in this work.

1. Introduction

The objective of the *text classification* (TC) methods is to solve the problem of assigning labels to different types of digitized textual data such as social media posts, digital news papers, review comments for products, movies and so on. Text classification has different use cases including sentiment analysis, spam filtering, news classification, question answering and many more. TC approaches can be grouped into two broad categories[1], such as: (i) rule-based models where texts are categorised based on some predefined rules, (ii) *machine learning* (ML) based methods where the underlying relations are discovered from existing labeled categories. As the rule based method is an extremely tedious and very time consuming process, the ML approach has become the most important method for the text classification task. This work is based on the ML approach.

ML approaches can be of three types: (i) *supervised learning* where the input texts are labeled and the goal is to classify unforeseen texts; (ii) *unsupervised learning* where input texts are not labeled and the goal is to find patterns in the text; (iii) *semi-supervised learning* which is a mix of both the supervised and unsupervised approaches. The purpose of this work is to apply prominent supervised learning approaches to classify text documents. We have chosen to find solution for news categorisation task. News classification, based on the text i.e its words,phrases and combination of words, is classified into few categories like sports, political, technology.

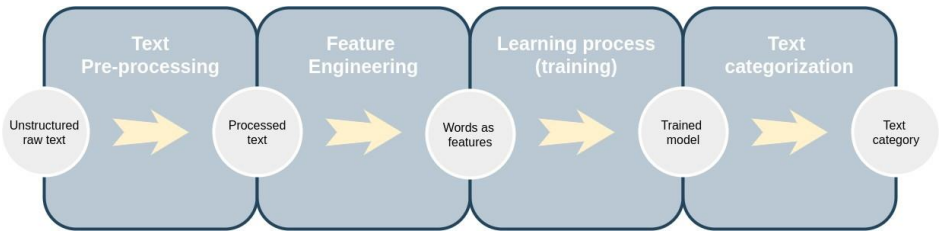

Figure 1. Text classification overview.

Automatic classification of news data can be very helpfull for the readers and the businesses. For example political and technological news data are rich information source for businesses to grow further.

1.1. Overview of text classification

Text classification or text categorization is a classical *natural language processing*(NLP) problem. With the burgeoning text data being generated, it is becoming very important to automatically classify those texts into predefined categories. The figure:1 shows an overview of the processes involved in a TC system using ML techniques. In the pre-processing stage, the text corpus is prepared for further processing by cleaning and trimming unnecessary words and characters. In the next stage relevant features are extracted or engineered using the selected words. The extracted features are used to learn the underlying relation among the features and corresponding labels. This learning phase is also called *training* in ML terminology and this training process produces the desired ML model which can later be used to categorize unseen text documents. TC can be used to take up various tasks like: (i) *sentiment analysis*, (ii) *news categorisation*, (iii) *topic modeling*, (iv) *question answering*, (v) *natural language inference*. For this work we have taken up *news classification* problem as case study.

1.2. News classification

News article classification task is a multi-class text classification problem where given a set of news corpus the trained ML model can predict the appropriate new category of the text. The complete process of classifying the document into predefined class is a two step process: (i) first, find the underlying relation among the input text and the class label called *training*; (ii) second, classify unseen news text into tentative news category called *prediction*.

1.3. Text Classification State of the Art

Machine learning based text classification has become an established method to rely upon with respect to improving classification accuracy. There are two types approaches are employed:
(i) *classical machine learning* approaches (ii) *deep learning* approaches

1.3.1. Classical Machine Learning Approach: In the classical ML approach, the news headlines or the text first goes through a cleaning process and after text cleaning, the text is broken into *tokens*(words) and each words are converted into vector using different strategies like *term- frequency inverse document frequency*(TF-IDF), *bag of words*(BOW), *bi-grams*, and *word2vec*. These vectors are representation of those sentences are passed into machine learning models like *naive bayes*(NB), *logistic regression*(LR), *support vector classifier*(SVC) *random forest*(RF), *gradient boosted decision tree* (see fig.1).

Mulahuwaish et al [2] in their article applied models like *support vector classifier*(SVM), K-Nearest Neighbors (kNN), Decision Tree (DT) and Long Short-Term Memory (LSTM) to perform classification of web based news into four classes i.e. business, technology & science, health and entertainment. SVM achieved highest accuracy of 95.04% and KNN performance was lowest among them and it is around 88.72%. Researchers [3] proposed TF-IDF term weighting scheme for optimization of classification techniques to get more optimized results and use two supervised learning approaches, i.e., SVM and kNN, and compared the results. Bijalwan [4] proposed a kNN based machine learning model for text and document mining and compared the results with *term graph* and *naive bayes*. kNN performed best, however term graph results are also very close. Mallick et al [5] used Bernoulli model for digital media news categorization. Their model produced accuracy of 98.4%, F1-score of 91.4%, and precision of 92.7%.

1.3.2. Deep Learning Approach: In the Deep learning approach, the news text headlines or news text data are converted into tokens which are nothing but numeric representation of the text and then this numeric vectors is passed through Neural Network based models. This approach is not heavily dependent on hand crafted feature extractions. Deep Learning text classification research can be divided into two categories. One group is more focused on the model based word embedding [6, 7]. A lot of studies shown that word embedding has a significant effect on the model performance [8, 9].

Kim [10] used *convolutional neural netowrk* (CNN) on top of static Word2Vector [6] on 100 billion words of Google News, and are publicly available. They [10] depicted that pre-trained vectors can be used as universal feature extractor and can be used for various classification tasks. Their simple CNN with one layer build on top of Word2Vec performed significantly well for TC tasks.

CNN models prioritise locality of information whereas RNNs capture the sequence but both of them cannot capture the positional importance and long distance semantics. Graph Based Neural Networks are designed to capture long distance relations in sequential data and also capable of providing positional [11, 12]. Recently a *graph based nueral network* (GCN) [11] is proposed. They compared the results with different base line models like TF-IDF-LR, LSTM, Bi-LSTM using different datasets like 20NG, R8, R52, Ohsumed, MR. GCN outperformed the base line models. Another interesting work [13] have shown that a shallow capsule network perform better than SVM, LSTM on the *blurb genre collection* (BGC) dataset.

Between 2013 and 2015, another area of NLP called neural machine translation become very popular. [14, 15, 16] proposed neural machine translation techniques. The concept of *jointly learning to align and translate* [17] or attention based models started becoming popular. In their paper,they proposed that the use of fixed-length vector is a bottleneck of the basic encoder- decoder architecture and they proposed an architecture that allows a model to automatically search for the source sentence that are relevant to predicting the target word. Deep learning based models including convolution neural networks(CNN) are used to learn text representations [10]. For further reference we can recommend the comprehensive review by Minaee et al [1] where they reviewed multiple DL based models.

1.4. Drawbacks of existing approaches

Existing text classification models based on manchine learning models have few drawbacks.

- Full meaning of the coherent documents is difficult to retain for machine learning algorithms [18]. Deep learning models proposes different way outs for this issue but for ML models it's difficult to deal with this.
- Text classification models are unable to learn long range dependencies in text sequence in input.

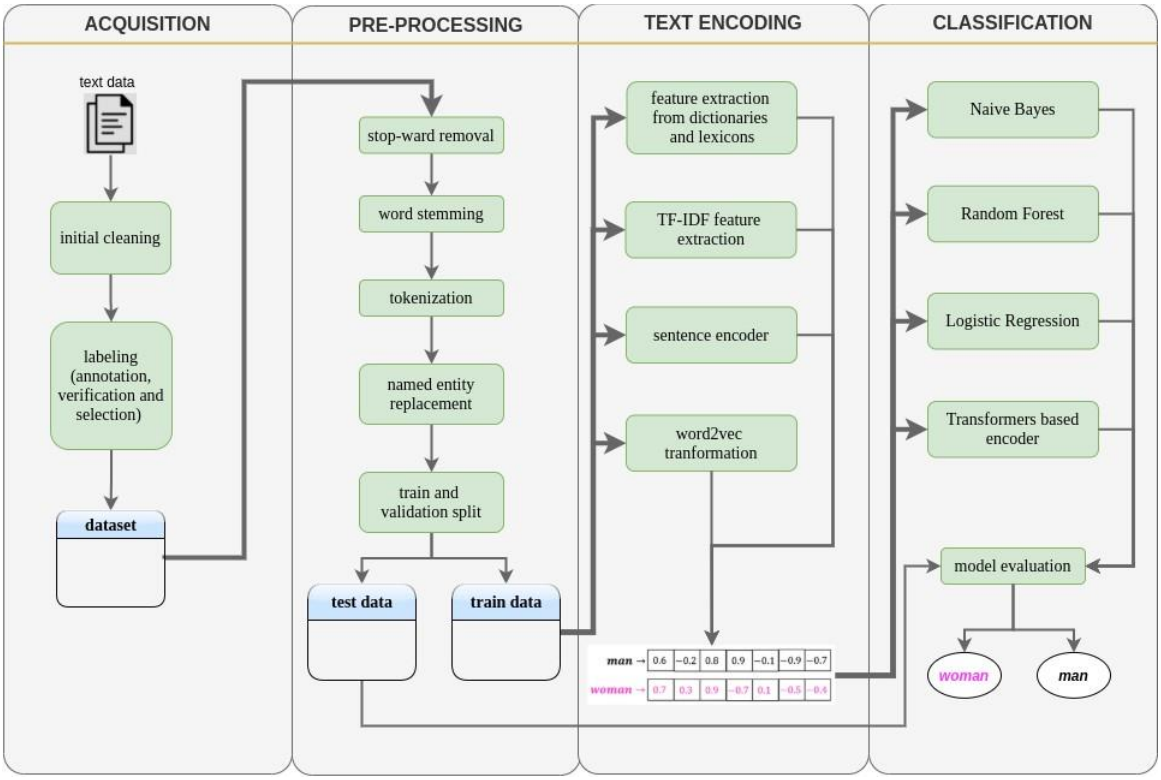

Figure 2. Different stages of text classification described in a flow diagram.

- There are scopes of improvement to find the optimal text representation method for training input.

2. Problem statement

Computers are not capable of handling texts in the way we humans do. Humans can interpret any texts of a known language by just looking at it. However, computers by default can display any text but they can't interpret the text. To make computers interpret texts is one of the primary goals of the NLP domain and text classification is a primary task of that domain. Classification is a techniques applied to classify data into some predefined classes, similarly here also the machine classifies texts into some specific classes. So, texts needs to be presented in some format from where the classification algorithms can start working upon the texts. Hence, the problem of text classification can be broken down into following steps:

- convert the text to machine readable format;
- transform the texts into some format on which classification algorithms work;
- define some classes as per the requirement;
- classify the transformed text into predefined classes using suitable classification algorithm.

input sources consists of some raw texts. Usually, text data sets contain document, text segment) with s number of sentences such that each sentence includes w_s words

sequences of text in documents as $A = \{a_1, a_2, \ldots, a_i\}$, where a_i refers to a data point (e.g. i^{th} data point) with l_w letters. Each data point is labeled with a class value from a set of k different discrete value indices [19, 18].

There are two primary problems in text classification: (i) feature extraction, (ii) classification model building Since, texts have no meaning for the computing system for learning any underlying patterns, the first problem is to transform the texts into a suitable representation. The representation will be the input for the classification model. So, a set of features can be extracted from A such that

$$X = \{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\} \quad (1)$$

where $X_i = (x_{i1}, x_{i2}, \ldots, x_{id}, y_i)$ where $i = 1, 2, \ldots, n$, is the feature vector and $x_{ij} \in R, j = 1, 2, \ldots, d$. The labels are $y_i \in Y$. Here n is the total number of text documents in the corpus. The feature set will be the input for classification model. Different classification models take different approaches to solve the TC problem. We have employed prominent algorithms like *naive bayes* (NB), *logistic regression* (LR), and latest *deep learning* (DL) based models to compare the results. Section.4 discusses the classification approaches in detail where the models will work on the input sequence defined by eq.1.

3. Dataset description

This work is based on the publicly available text corpus known as *BBC News dataset* [20]. BBC News Dataset is available for non- commercial research and a popular dataset among the NLP researchers. It contains 2225 documents collected from the BBC news website between year 2004 and 2005. There are news from 5 categories: *politics, sports, business, entertainment and tech.*

We found 99 duplicate documents which is removed so effectively 2126 documents are available for building the model.The distribution of documents among the 5 categories of news can be seen in fig.3 and according to the distribution, data is almost balanced that is classes are evenly distributed. There are many steps involved in the preprocessing stage which prepare the data for modeling. The steps are: (i) remove punctuations, stop words, words with numbers, (ii) remove html tags and urls, (iii) lemmatize the words to the root words, (iv) decontraction of words.

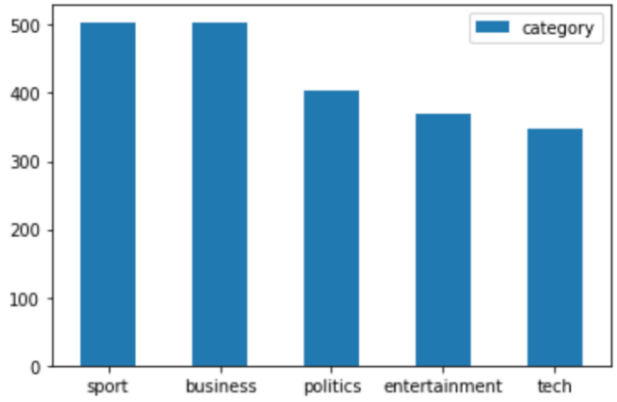

Figure 3. Document class distribution showing the number of documents in each class.

4. Proposed method

The text classification process involves different steps before actual classification task can be performed on the corpus. We have used two corpus for this work and the details of them are described in the section-3. In this section the proposed method is discussed in detail including the *preprocessing* methods (sec.4.1) and the classification models (sec.4.2 and sec.4.3).

4.1. Text Transformation Methods

This is a very important phase in text classification task where the input texts are transformed to some machine manageable format. Machine Learning and Deep Learning models cannot read texts directly as input. So, its mandatory to represent the texts in some digitized format that ML or DL models can read.

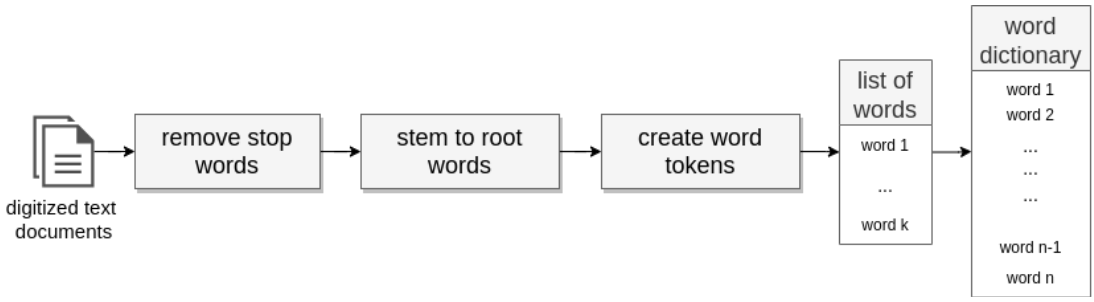

Figure 4. Text pre-processing steps.

The figure.4 shows an overview of the operations performed on the text. First, the stop words are removed from Texts the texts which have no relevance in classification. The there is a step called *stemming* where a word is trimmed to its root word and only the root words are kept as text. The third step is *tokenization* a fundamental step for ML based as well DL based NLP solutions. Texts are broken down into smaller units called tokens. Tokens can be words, sub-words or characters depending on the requirements of the NLP system.

Tokens are building blocks of NLP models because any NLP model need word tokens to build *vocabulary* (fig.4). Both DL and traditional NLP models need vocabulary for training. Because DL models use dictionary to build the input sequences while traditional ML models use vocabulary to prepare *TF-IDF* or *Count Vectorizer* feature set.

4.2. Machine Learning Classifiers

In this work few prominent classical ML classification techniques are applied on the news corpus to analyse and compare the results.

Logistic Regression Classifier: Logistic regression is a special type of model called the *generalized linear models* (GLMs). GLMs have the following characteristics:

(i) a probability distribution describing the outcome variable y

(ii) a linear model such that:
$\eta = w_0 + w_1 X_1 + \cdots + w_d X_d$ (2) where n is the number of features, $X_j, j \in 1, \ldots, d$ are the features or independent variables and $w_j, j = 0, \ldots, d$ are the parameters of the linear model.

(iii) a mapping or link function (f) that relates the linear model to the parameters of the outcome distribution, i.e.

where f is the link function.
$$f(p) = y \Rightarrow p = f^{-1}(y) \quad (3)$$

So, *logistic regression* (LR) analyzes the relationship between the dependent variable y and one or more independent variables $\boldsymbol{x}$ based on the probability by using a link function. The link function is a logistic (sigmoid) function. The optimization problem for LR can be established as follows:

$$\min_{\mathbf{w} \in R_d} \sum_i^n \log\left(\frac{}{1+e^{-y_i f(\mathbf{x}_i)}}\right) + \lambda \| \mathbf{w} \|^2 \quad (4)$$

The model estimates the coefficients $\boldsymbol{w}$ from training dataset to minimize the errors which are different between the real output and the predicted output. In eq.4 λ is the regularizer which

restricts the model to from overfitting. The extracted features from the text documents are X_i and the corresponding labels are y,based on these we can fit the LR model for classification.

Naive Bayes Classifier: The Naive Bayes classifier (NBC) is based on the principles of Bayes Theorem. This is one of the simplest classifiers but very effective one [21]. Bayes theorem states that: $P(A \mid B) = \frac{P(B|A) \times P(A)}{P(B)}$, where A and B are events and P(B) 0. Bayes rule can be interpreted from classification point of view as: $posterior = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$. The reason this algorithm is so powerful is that it assumes complete independence among the features in the data, even if dependence often naturally exists. This is primarily why it is known as 'Naive' Bayes. In terms of the data, it is defined [22] as follows: Given class label y and an independent

feature vector $X_i = x_1, \ldots, x_n$, where $i = 1, \ldots, d$ we have

$$P(y_i \mid X_i) = \frac{P(y_i) \times P(X_i \mid y_i)}{p(X_i)} \quad (5)$$

And based on the above assumptions the classifier can be sated as follows:

$$\hat{y} = \underset{c \in \{1,\ldots,C\}}{\operatorname{argmax}} P(y_c) \prod_{i=1}^{n} P(x_i \mid y_c) \quad (6)$$

$P(y_c)$ is the probability of getting class c. So, finally, we are left with the task of calculating $P(y_c)$ and $P(x_i \mid y_c)$ for classification and its assumed to be C here [23].

Random Forest Classifier: Random Forest (RF) is one of the most popular and effective ensemble methods. This is a *decision tree* (DT) based classifier which employs *bootstrap sampling* method. The sampled data-slices are used to build the separate DTs and later majority vote is computed to select the best possible outcome [24]. of preparing DT [25] is *entropy*. Lets consider the target variable y as defined as random variable which can have C number of classes $y = y_1, \ldots, y_C$. So *entropy*(H) can be The key concept

$$H(y) = -\sum_{c=1}^{C} p(y_c) \log_2(p(y_c)) \quad (7)$$

X_i, $i = 1, \ldots, d$ as: The concept of entropy in eq. 7 is further used for compute *information gain* (IG) for each of the features

$$IG(y, X_i) = H(y) - EH(X_i) \quad (8)$$

where, $EH(X_i)$ is the *expected entropy* such that

$$EH(X_i) = \sum_{p=1}^{P} \frac{D_p}{n} \times H^{D_p}(y)$$

where, D_p, $p = 1, \ldots, P$ is the p^{th} partition of the data and there are total P partitions based on the feature X_i. The feature with highest IG value is selected as the root for present tree level [18]. The DT is built level by level applying the same logic discussed above.

Random Forest (RF) creates t random decision trees in parallel [26] based on *bootstrap sampling* over the *rows* and *columns* of the dataset. A *voting* is performed after training all the decision trees [27] to predict the unseen data, in this case text documents.

Multi Layer Perceptron (MLP): Multi-layer perceptron, also called feedforward neural networks, are the prototypical deep learning models. The learning objective of an MLP is to approximate some function g based on the mapping $y = f(x; \theta)$ and networks, or θ that results in best possible g. When MLPs are composed deep feed forward as a chain different functions and represented as a directed acyclic graph, they are called *deep networks* and the length of the chain gives the depth of the model.

So, the training phase the MLP $f(x)$ is tuned to approximate $g(x)$ with the help of another non-linear function α called the *activation function* such that

$$y = \alpha(x;\theta)^T w \tag{9}$$

where θ is used to learn and w maps $\alpha(x)$ to desired output. Modern neural networks mostly use *maximum likelihood estimator* to estimate the parameters. The loss function (θ) here is the cross-entropy loss function.

$$L(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log p_{model}(y_n \mid \boldsymbol{x}_n, \boldsymbol{\theta}) \tag{10}$$

(i) for θ such that $y = \alpha(x)$ mapping is optimum we have to minimize (θ) with respect to θ. We can get output distribution parameterised by θ. To find a good parameter vector θ for the neural network. Where p_{model} is a particular During the backward pass, also called *back propagation*, the values for the weight vector w are make use of the loss function (θ) to find the optimal values for $\boldsymbol{w}$, updated using methods like *stochastic gradient descent* (SGD). Both forward and backward pass.

The architecture of the MLP designed for this work is shown in figure.7(b). Apart from dense layers we have used *dropout* layers to deal with over-fitting. Also there are special nodes specific for text classification problem such as:

(i) *GlobalAveragePooling1D* for generating one feature map for each corresponding category of the classification task, (ii) *Embedding* which turns positive integers (indexes) into dense vectors of fixed size, (iii) *TextVectorization* is a preprocessing layer which maps text features to integer sequences.

4.3. The Proposed Deep Learning Model

We have trained a custom variant of the *transformer* [28] model for news classification. Transformer is an *encoder-decoder* stack which was used for language translation and achieved new stat-of-the-art in the respective problem domain. However we require only the *encoder* section of the model since our task is to classify the text inputs and we do not need the decoder as required in case of language translation.

as in eq.1 $X = (x_1, x_2, \ldots, x_d)$ where d is the dimention of the input embeddings. This mod The input to the *encoder* is the sentence, or symbol representations called *embeddings* (fig.5) does not contain recurrent connections to capture the sequential order of the words, instead it uses a concept called *positional encoding* (*PE*) [28] to keep track of the sequential order of the words. The *PE* values are designed to be a function of *sine* and *cosine* values of their positions. This concept is explained in detail in the original article [28].

The proposed *encoder* model is composed of a stack of two identical layers. The model is auto-regressive [29] at each step and consumes the symbols generated by previous step to generate its own output. The figure.5 shows a block diagram of the model with two identical attention blocks are stacked over another. The blocks consists of a multi-head attention layer and a fully connected feed forward network. The attention block contains residual connections and layer normalization (fig.5).

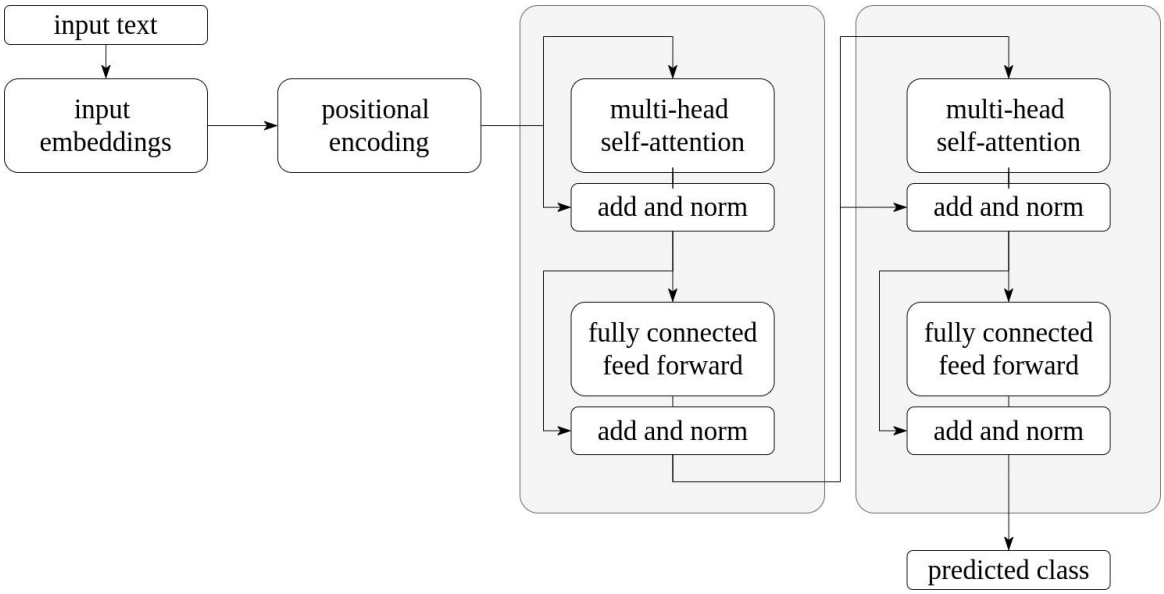

Figure 5. The transformer based encoder model.

One of the key concept of this model is the *attention* mechanism which can be seen as mapping of a query (Q) and a set of key(K)-value(V) pairs an output. This specific attention technique is name as *scaled dot product attention* [28] and it is computed using a *softmax* function as below:

$$attention(Q, K, V) = softmax(QK^T / \sqrt{d_k})V \quad (11)$$

is same as dot-product attention with the $1/\sqrt{d_k}$ scaling factor. This model [28] implemented h linear projections of the Qs, Ks and V s and performed attention function in parallel instead of performing one single attention. They called it *multi-head attention*. This multi-head attention generates d_v-dimensional output which is further projected to get the final values (fig.6). Multi-head attention is powerful in information retrieval because it can focus on different representations in parallel and finally collaborates to extract information. So, the attention model can be summarized as follows:

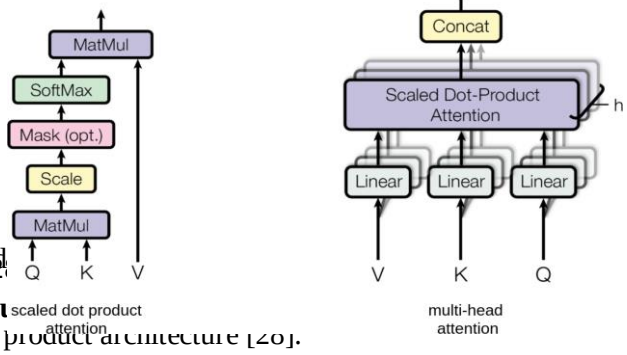

Figure 6. scaled dot product attention (left) and multi-head attention (right) which consists of several attention layers running in parallel [28].

$$multi\text{-}head(Q, K, V) = concat(head_1, \ldots, head_h)P^O \quad (12)$$

where, $head_i = attention(QP_i^Q, KP_i^K, VP_i^V)$ and the projections are parameter matrices $P_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $P_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $P_i^V \in \mathbb{R}^{d_{model} \times d_v}$ and $P^O \in \mathbb{R}^{hd_v \times d_{model}}$. For this work we used $h = 4$ parallel attention layers, or heads.

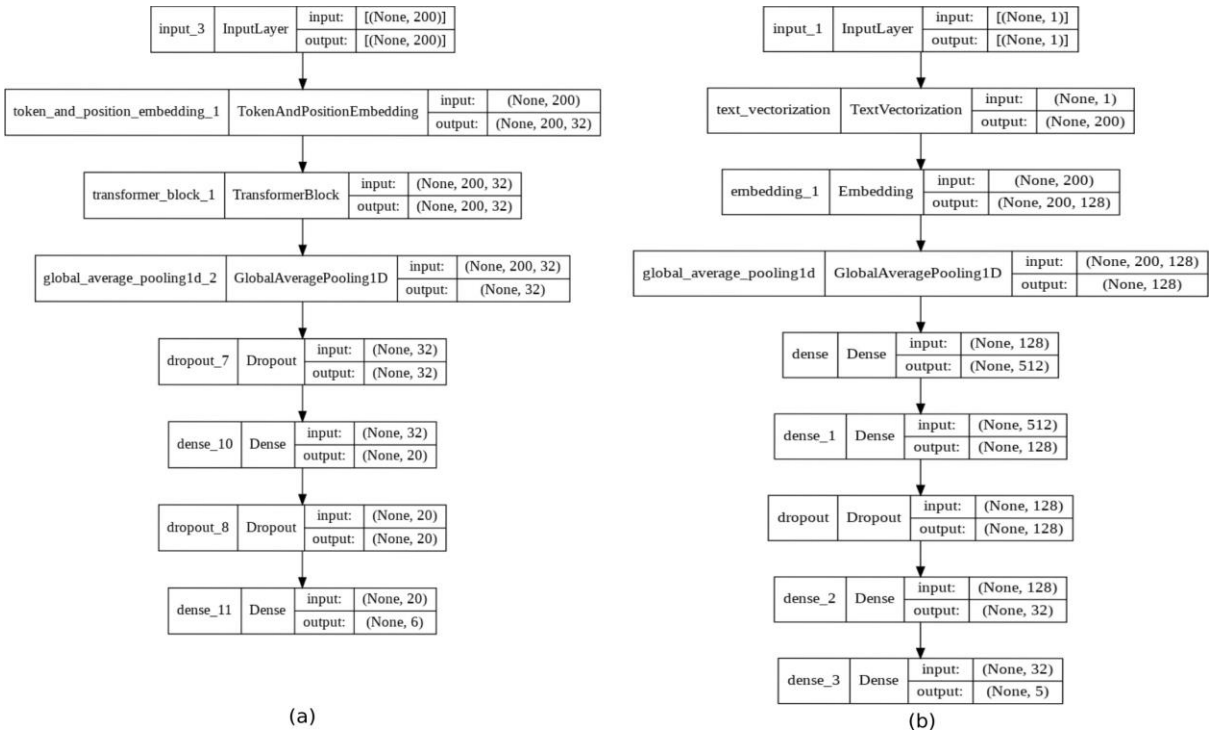

Figure 7. This figure is showing the structure of the two deep learning models Attention (a) and MLP (b).

The final model with the multi-head attention layers, the dense layers and dropout layers are shown in the figure.7(a). Figure.7 also shows the MLP model (b) so that two models can be seen side by side.

Advantages of the model: Some of the prominent advantages of this attention based model are mentioned below.

(i) better ability to learn long range dependencies in the input sequence [28].
(ii) computational complexity of the self-attention layers are lower than the recurrent layers.
(iii) This model uses dot-product attention which is faster than other attention models.
(iv) Its space efficient in practice.
(v) Highly optimized for matrix multiplication code.

5. Results and discussions

The results of the experiments conducted using the BBC dataset is discussed in this section. Figure.8 shows a comparative view of the performances of different the classifiers tested in this work and its apparent that the newly developed attention model out performs the other four classifiers with 96.41% test accuracy. MLP is the closest second with 94.62% test accuracy. Apart from test accuracy we computed three other prominent metrics such as precision, recall and f1-score. Attention model out performed other models in every metric.

Figure.9 shows the confusion matrices of the test results of the models. The training time loss and accuracy for the MLP and Attention model can be seen in the fig.10. Training and validation accuracy curves for both the MLP and Attention model remains close, which indicates that models are are not over-fitting. However, for both the DL models the loss increases, though

Result summary of the models					
	MLP	NB	LR	RF	Attention
accuracy	94.62%	95.52%	95.52%	92.83%	96.41%
precision	94.72%	95.62%	95.66%	93.01%	96.47%
recall	94.62%	95.52%	95.52%	92.83%	96.41%
f1-score	94.61%	95.51%	95.54%	92.86%	96.39%

Figure 8. The table summarised the results received during the experiment with the models *multi layer perceptron* (MLP), *naive bayes*(NB), *logistic regression* (LR), *random forest*(RF) and the *attention*.

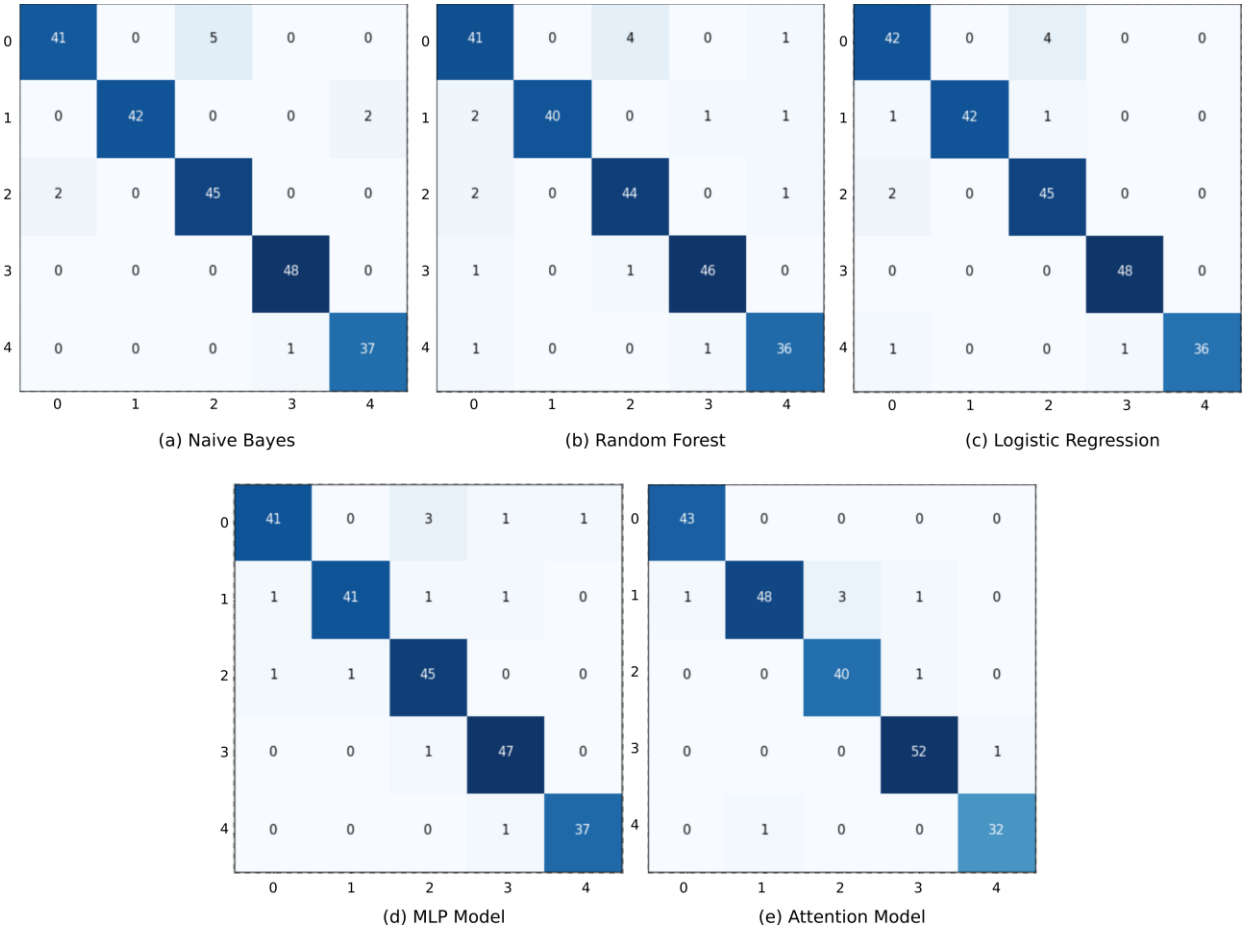

Figure 9. Confusion matrices for the models we have evaluated for this work. The axis shows the number of classes.

slightly, over time which is a common phenomenon in modern neural networks. The phenomenon is called the mis-calibration issue [30] where the model tend to beover-confident.

6. Conclusion

This work we have explored the text news classification problem of natural language processing domain. A transformer architecture based custom attention model is developed to solve this problem. Apart from that prominent ML classifiers like *Naive Bayes*, *logistic regression*, *random forest* and MLP is also evaluated on the same corpus. The comparative study between the results obtained, as discussed in section 5, shows that the attention based model performs better than

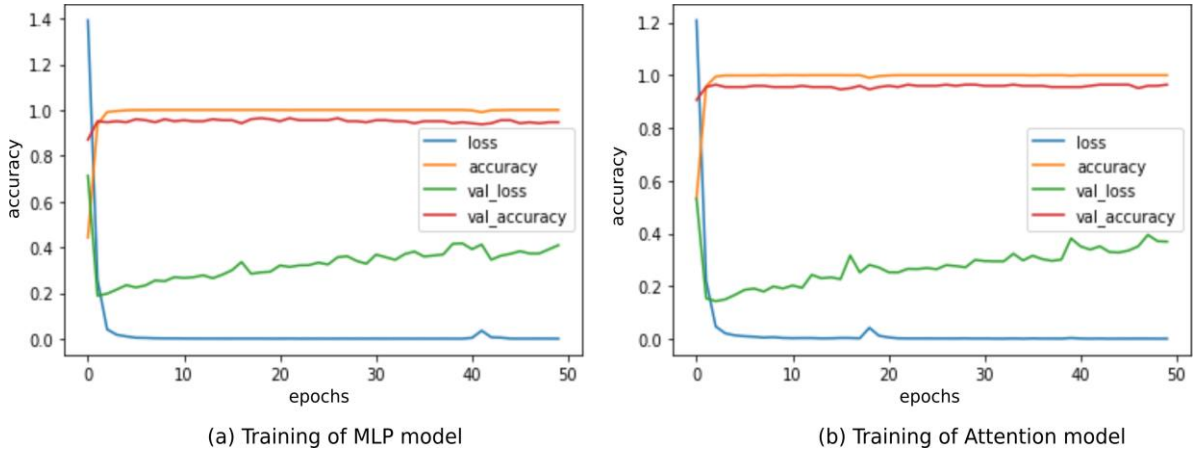

Figure 10. Graphs show how loss and accuracy change with each epoch for both the MLP and Attention model. Both the training and validation results are compared.

the other models.

The results are encouraging and we are looking forward to enhance the accuracy of the attention based model by employing *attention-based LSTM* [31]. There are possible candidates for the enhanced model to be deployed for solving other problems of NLP domain or of other domain such as gene expression base classification [32], where finding pattern in positional encoding is of prime importance. In this work one dataset is used so the baseline model ofthis work can be evaluated using other datasets as well.

References

[1] Minaee S, Kalchbrenner N, Cambria E, Nikzad N, Chenaghlu M and Gao J 2021 *ACM Com- puting Surveys* **54** 62:1–62:40 ISSN 0360-0300 URL https://doi.org/10.1145/3439726

[2] Mulahuwaish A, Gyorick K, Ghafoor K Z, Maghdid H S and Rawat D B 2020 *Computers & Security* **98** 102006 ISSN 0167-4048 URL https://www.sciencedirect.com/science/article/pii/S0167404820302790

[3] Kanika and Sangeeta 2019 *Smart Computational Strategies: Theoretical and Practical Aspects* ed Luhach A K, Hawari K B G, Mihai I C, Hsiung P A and Mishra R B (Singapore: Springer Singapore) pp 95–105 ISBN 978-981-13-6295-8

[4] Bijalwan V, Kumar V, Kumari P and Pascual J 2014 *International Journal of Database Theory and Application* **7** 61–70

[5] Mallick P K, Mishra S and Chae G S 2020 *Personal and Ubiquitous Computing* ISSN 1617-4917 URL https://doi.org/10.1007/s00779-020-01461-9

[6] Mikolov T, Sutskever I, Chen K, Corrado G and Dean J 2013 *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* NIPS'13 (Red Hook, NY, USA: Curran Associates Inc.) pp 3111–3119

[7] Pennington J, Socher R and Manning C 2014 *Proceedings of the 2014 Con- ference on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar: Association for Computational Linguistics) pp 1532–1543 URL https://www.aclweb.org/anthology/D14-1162

[8] Shen Y, Zhang Q, Zhang J, Huang J, Lu Y and Lei K 2018 *International Conference on Information Science and Applications* (Springer) pp 401–411

[9] Wang G, Li C, Wang W, Zhang Y, Shen D, Zhang X, Henao R and Carin L 2018 *arXiv preprint arXiv:1805.04174*

[10] Kim Y 2014 *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar: Association for Computational Linguistics) pp 1746– 1751 URL https://www.aclweb.org/anthology/D14-1181

[11] Yao L, Mao C and Luo Y 2019 *Proceedings of the AAAI Conference on Artificial*

Intelligence **33** 7370–7377 ISSN 2374-3468 number: 01 URL https://ojs.aaai.org/index.php/AAAI/article/view/4725

[12] Battaglia P W, Hamrick J B, Bapst V, Sanchez-Gonzalez A, Zambaldi V, Malinowski M, Tacchetti A, Raposo D, Santoro A, Faulkner R, Gulcehre C, Song F, Ballard A, Gilmer J, Dahl G, Vaswani A, Allen K, Nash C, Langston V, Dyer C, Heess N, Wierstra D, Kohli P, Botvinick M, Vinyals O, Li Y and Pascanu R 2018 *arXiv:1806.01261 [cs, stat]* URL http://arxiv.org/abs/1806.01261

[13] Aly R, Remus S and Biemann C 2019 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop* pp 323–330

[14] Kalchbrenner N and Blunsom P 2013 *Proceedings of the 2013 conference on empirical methods in natural language processing* pp 1700–1709

[15] Sutskever I, Vinyals O and Le Q V 2014 *Advances in neural information processing systems* pp 3104–3112

[16] Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H and Bengio Y 2014 *arXiv preprint arXiv:1406.1078*

[17] Bahdanau D, Cho K and Bengio Y 2014 *arXiv preprint arXiv:1409.0473*

[18] Kowsari K, Meimandi K J, Heidarysafa M, Mendu S, Barnes L E and Brown D E 2019 *Information* **10** 150 ISSN 2078-2489 URL http://arxiv.org/abs/1904.08067

[19] Aggarwal C C and Zhai C 2012 *Mining text data* (Springer) pp 163–222

[20] Greene D and Cunningham P 2006 *Proc. 23rd International Conference on Machine learning (ICML'06)* (ACM Press) pp 377–384

[21] Webb G I, Keogh E and Miikkulainen R 2010 *Encyclopedia of machine learning* **15** 713–714

[22] Rish I *et al.* 2001 *IJCAI 2001 workshop on empirical methods in artificial intelligence* vol 3 pp 41–46

[23] Murty M N and Devi V S 2011 *Pattern recognition: An algorithmic approach* (Springer Science & Business Media)

[24] Biau G and Scornet E 2015 *TEST* **25** 197–227

[25] Magerman D M 1995 *ArXiv* **cmp-lg/9504030**

[26] Ho T K 1995 *Proceedings of 3rd International Conference on Document Analysis and Recognition* vol 1 pp 278–282 vol.1

[27] Wu T, Lin C J and Weng R C H 2003 *J. Mach. Learn. Res.*

[28] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L and Polosukhin I 2017 *arXiv:1706.03762 [cs]* ArXiv: 1706.03762 URL http://arxiv.org/abs/1706.03762

[29] Graves A 2014 *arXiv:1308.0850 [cs]* ArXiv: 1308.0850 URL http://arxiv.org/abs/1308.0850

[30] Guo C, Pleiss G, Sun Y and Weinberger K Q 2017 *ArXiv* **abs/1706.04599**

[31] Zeng J, Ma X and Zhou K 2019 *IEEE Access* **7** 20462–20471

[32] Khan A and Lee B 2021 *ArXiv* **abs/2108.11833**