Modified Convolutional Neural Network Filter Gate for Social Media Text Classification

Nur Suhailayani Suhaimi[1], Zalinda Othman[2*], Mohd Ridzwan Yaakub[3]

Universiti Teknologi MARA, Universiti Kebangsaan Malaysia, Universiti Kebangsaaan Malaysia

Summary
The capacity of the Convolution Neural Network (ConvNet) to handle unpredictable and continuous stream input has piqued researchers' attention in a variety of fields. One of ConvNet's features involves filtering during input reception. However, sampling filters alone lead to low pre-processing accuracy and precision problems. This work offers an upgraded Filter Gate with Text Pre-processing (FGTP) to address this pre-processing problem during input reception. We filtered the input using three algorithms: Fourier Transform (FT), Porter's Algorithm (PA), and Correction Filter (CF) for the continuous and uncertain size of text input data placed into the ConvNet algorithm for classification. We use the FT method to delete redundant or similar text to deal with duplication and repetitive text. We use the PA approach to stem and reduce out-of-vocabulary words in continuous sentences. After that, the CF algorithm deals with misspellings and typos. The filtered results in this paper are compared to random sampling filtering and word detection accuracy with and without FGTP. Finally, we compare the classification accuracy of ConvNet with FGTP to ConvNet with random sampling. This proposed method significantly contributes to proving the influence of multiple filtering of text pre-processing in text classification with a gap of over 27 per cent better than the conventional method. The proposed method yielded 83.4% accuracy, while conventional filtering provides a 65.33% accuracy value.

Keywords:
Text pre-processing; text input filtering; convolutional neural network; multi-gate text filtering.

1. Introduction

Stream data handling for deep learning analysis or as an algorithm efficiency testing data has grown increasingly important in recent years. With the massive increase in data rates and dimensions in many data types, processing and analysing this continuous unbounded data stream sequence has grown more complex [16],[33]. The size of a data stream is potentially limitless. The data amount is significant and irregular [16]. As a result of the rise of stream-based applications, there is a greater demand for massively high-speed and efficient solutions than traditional Data Stream Management Systems can handle (DSMSs). DSMSs are often append-only data systems, with new data streams arriving in the system in an unordered way [16],[30],[33]. Stream arrival rates can vary dramatically, resulting in erratic and burst flow.

Manuscript received May 5, 2022
Manuscript revised May 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.5.86

In recent years, a plethora of data stream research has emerged with the expansion of data stream dimensions. Convolutional Neural Network (ConvNet) is a deep neural network approach that is space and shift-invariant. ConvNet is derived from a Multilayer Perceptron with a high volume of connectivity; it is also known as whole connectedness.

Overfitting is a problem that affects the entire connectivity network [3],[18]. As a result, ConvNet takes advantage of its hierarchical patterns to conduct a smaller and simpler network pattern. ConvNet is used because it can learn significant features from continuous data at different levels, similar to how a human brain does. This ability to learn is known as feature learning, and it is something that a traditional Neural Network (NN) cannot perform. Weight sharing is another advantage of ConvNet [3],[32],[33]. ConvNet features filters that use weight sharing to simplify networks. The overfitting problem of the entire connectivity network was decreased due to this weight sharing. As a result, ConvNet is more memory and complexity efficient. Compared to traditional NN with billions of neurons, ConvNet is less complex and consumes less memory [3],[16]. ConvNet outperforms NN on image recognition tasks and many other tasks in terms of performance and is also a suitable feature extractor for a completely new task or problem. ConvNet extracts only good attributes from an already-trained network with its taught weights. The feature extraction is by feeding stream data at each level and fine-tuning it for the task. It was, for example, inserting a classifier after the last layer with task-specific labels. Pre-training refers to the process of selecting features, and ConvNets outperform NNs in this regard. Another benefit of this pre-training is that it avoids ConvNet training, saving memory and time. The classifier at the end layer, which has class labels, is the only item to be trained.

Existing ConvNet-based frameworks, on the other hand, have several flaws, including the traditional pooling procedure losing essential feature information and being unlearnable [30]. Another disadvantage is that classic convolution optimises slowly and does not fully exploit hierarchical characteristics from multiple layers [3],[18],[33]. Filtering stream data input using ConvNet takes up much memory and takes a long time during the initial data extraction [4],[7],[32]. Because ConvNet does not encode the object's location or orientation in its

convolutional layer, it cannot recognise the relationships between distinct objects [31]. Furthermore, low-quality data from the first data extraction of filtering input may result in slower performance in ConvNet's max pool operation [22],[23],[28]. [10] employed a filtering strategy to improve the quality of stream data by deleting redundant input data before performing further analysis. However, the problem of recognising data kinds and distinguishing noise in-stream data input has yet to be investigated [5],[6],[22]. The process of processing text stream data with variation feature representation usually starts with random initialisation (within the filtering layer), which has an impact on the outcome [19],[26]. All filter coefficients are usually learned when applying ConvNet to image data and filtering by random sampling. The lack of subjectivity in non-probabilistic data is exacerbated by random sampling [22].

Extracting textual data from an online application is a complex problem for many applications. Unlike character recognition in physical documents, text identification in unconstrained media social applications is complicated by various properties, data formats, repeating content, and lexical conditions. In the field of text classification, text preparation is critical. Text processing activities are usually challenging in reality since textual data is informative and generally unstructured. Text processing research such as web scraping, text clustering, and sentiment analysis focuses on handling text data fast and effectively to extract relevant information from a big pool of online text. Research by [25] also showed the importance of text pre-processing in increasing the learning ability of Natural Language Processing (NLP). Furthermore, specific text data from web applications is extensive and comes in a variety of text formats, making it challenging for a machine to comprehend the semantics of the text. As a result, text pre-processing is essential in the filtering [6],[10] of input features in this study to prevent the loss of critical feature information during sampling filtering and improve classification accuracy.

Generally, the ConvNet algorithm is used in image processing to extract meaningful features. Due to its ability to pool samples of several frames randomly and identify essential features intelligently, we use this feature to extract text from image data. However, text data in social media images consists of inconsistencies, noisy data, missing values and redundant text (repeating text). These issues of text data will eventually affect the data quality to be classified for sentiment analysis. Without text filtering gates to pre-process the extracted data, the accuracy of the classifier tends to be lower due to the existence of free-text data from media social. Previous research in text mining generally performed the task of data pre-processing separately from the classifier algorithm. This separation of tasks requires repetitive processes and inconsistency in compiling the continuous stream data. Therefore, this paper aimed to enhance ConvNet by embedding text filtering gates after the feature extraction before the classification phase. This paper aims to increase the accuracy of the ConvNet classifier towards continuous stream data, reduce error rates in classifying unclean text data, and increase the learning rate on the ConvNet by remembering the previous updated weight of the classifier. Our central hypothesis is that the classifier's performance will be increased when the unnecessary text data is successfully filtered within the algorithm feature.

2. Related Works

2.1 Social Media Text Data Challenges

Social media is one of the powerful communication platforms in our era of knowledge discovery. Many online discussions on the internet are meant for users to express, discuss, and exchange their views and opinions on various topics. Online Social networks (OSN) have been widely studied in recent years as ways to communicate and share information [21]. For example, news portals, blogs, and social media channels like YouTube and Twitter. One of the platforms that are majorly used is the Twitter application. A large amount of user-generated data from Twitter is an informative data source, even though it is not easy to extract the implicit knowledge from this vast and noisy data warehouse.

According to [20], the main challenge in text classification is to analyse social media text, especially in Twitter applications. Since tweets are short text structures that look like colloquial text compared to written documents, the classification process became challenging. Social media text also consists of informal language, slang and missing words which makes text classification very difficult to distinguish the text categories correctly. A tremendous amount of research has been done to classify social media text data. Since classification is an essential part of text mining, many techniques have been proposed to classify this kind of information. Based on user-generated social media posts on Twitter, [24] has developed a tweet classification system using deep learning-based sentiment analysis techniques to classify tweets as extremist or non-extremist. Text classification by popular machine learning algorithms such as Naive Bayes, Support Vector Machines, Decision Trees, Random Forest, AdaBoost, and Gradient Boosting for analysing online reviews is also widely used as a baseline model to be compared. The explosive growth of online social media content has received substantial attention in recent years to address the problem of automatic sentiment analysis during input extraction into the

classification network [2]. Thus, text filtering is required as a phase of preprocessing before classification occurs to prevent the loss of critical feature information during sampling filtering and improve classification accuracy.

2.2 Deep Neural Networks for Text Filtering

Filtering is an essential text feature for text mining and information retrieval. Deep learning (DL) has made considerable achievements in text filtering. DL methods are representation learning methods with multiple levels of representation obtained by composing nonlinear but straightforward modules that transform the representation at one level (raw data) into a higher representation. DL is slightly more abstract, with the composition of such transformations and very complex functions. DL approaches based on Convolutional Neural Networks (ConvNet) have shown success in many areas of machine learning and pattern recognition research due to their filtering feature during input reception. However, mapping the ConvNet filtering usually used in image data into filtering text data requires extensive exploration in understanding text structure and algorithm features. [9] successfully applied a novel approach to pruning ConvNet's convolution filters using a data-driven utility score for text classification. Filtering by pruning achieved both better task performance and increased computational efficiency.

DL approaches based on Neural Networks are also used for detecting inappropriate content in the text as one of the filtering methods [12]. This method is alternately used as a selection process or eliminating process. Filtering text is also one crucial text processing for feature extraction and selection. Selection of text feature content is an essential process for text mining and knowledge discovery. Text feature extraction that filters text information is a filtering process to select and represent a text message; this is the basis of many text processing tasks. Filtering a set of features from some datasets is an effective way to reduce feature space dimensions. [11] claimed the purpose of this process is feature extraction in the text filtering phase. During feature extraction, uncorrelated or superfluous features will be deleted. As a method of data pre-processing of learning algorithms, filtering text during the feature extraction process can improve the accuracy of learning algorithms and shorten the time taken to classify the dataset.

The critical aspect of DL, especially in neural networks, is that human engineers do not design these layers of features; they are learned from data using a general-purpose learning procedure [11]. DL requires a small amount of manual engineering; thus, it can easily take advantage of the available computation and data increase. It provides the advantage of processing a model of unstructured data. Everybody is familiar with the media, such as images, sound, video, and text, belonging to such data. DL has proven to contribute promising results for various tasks in sentiment analysis or also known as opinion mining. Its profound architecture nature is an open path for deep learning and the possibility of solving much more complicated and narrowed AI tasks. [1] ensemble learning method in DL to classify poverty and produce multidimensional poverty indicators with higher accuracy than the linear method. In order to determine only the best inputs used in a DL, [27] hybridized Singular Spectrum Analysis and Deep Neural Network to control the input and eventually succeed in producing higher accuracy in forecasting. DL is then further used for optimization [13], where text classification performance is improved by injecting with an optimizer to increase the model learning ability. Deep Learning is also used for prediction as [14] utilizes DL to predict movie revenue using user reviews.

2.3 Text Classification Accuracy Issues

Quality data is challenging to obtain due to the uncertainty of user input, especially when involved with media social as its source. The accuracy of a classification model is very much dependent on the quality of data input. A text classifier is not functioning without accurate training data. The suitability of social media text in text mining to train the model classifier is always in an argument due to its characteristic of the unstructured, abundance of noisy data, not sufficient facts due to its free-writing style. However, as for sentiment analysis and the natural language processing domain, the source of knowledge can be retrieved from social media somehow very implicit, trivial and exciting. It is a proven challenge to construct a reliable algorithm with significant accuracy when involving a large volume of data from various sources and languages in free form.

Therefore, pre-processing is an essential part when dealing with online stream data. Pre-processing tasks organize data, sort it and produce ready-to-mine data to ensure the desired knowledge can be obtained. Even with a powerful and efficient algorithm to mine the data without pre-processing tasks, the final accuracy result will be affected. A study by [8] successfully improved prediction accuracy when pre-processing feature selection was performed before the prediction process. [29] performed text clustering as its pre-processing task to reduce the semantic representation of the text data to ensure only valuable data is used to discover knowledge. These proved the essential pre-processing needs before proceeding to the knowledge discovery, prediction or classification modelling

process. Raw data can have missing or inconsistent values and redundancies, especially from online sources. This proposed method justified the need for pre-processing filtering within the classifier.

3. Model Description

In this section, we describe our approach for filtering text regarding the problem of the conventional sampling method in ConvNet filtering together with selected datasets.

3.1 Data Description

The filtering model was trained using six datasets from Twitter. The following is a list of data acquired from the Twitter application, as shown in Table 1.

Table 1. Experimental Dataset

Dataset	Details	Class Label
A	Iphone SE reviews	1 – not recommended ; 2 – slightly good 3- good 4- very good 5- strongly recommended
B	Branded items reviews	-1 : not recommended; 0 : neutral; +1 :recommended
C	e-clothing reviews	*1 – not recommended ;* *2 – slightly good* *3- good* *4- very good* *5- strongly recommended*
D	Movie Review From HBO	-1 : negative; 0 : neutral; +1 :positive
E	Movie Recommendation For Netflix Viewer	-1 : not recommended; 0 : neutral; +1 :recommended
F	Movie Recommendation For DisneyPlus Viewer	-1 : not recommended; 0 : neutral; +1 :recommended

Dataset A: The dataset recorded from this Twitter data was pulled from the Indian e-commerce website via Flipkart. Ratings from users are specifically collected and classified into five (5) labels, namely 1 represents ‘not recommended’, 2 represents ‘slightly good’, 3 represents ‘good’, 4 represents ‘very good’, and 5 represents ‘strongly recommended’.

Dataset B: The data set is in a CSV record, where each line is a snippet of text. Each included conversation has at least one request from the user and at least one response from the company. This set consists of 20 major brands for items in this data set record. The collected data is regarding the sample of customer support data on Twitter.

Dataset C: In an e-commerce-driven business world where customers are unable to experience physically recognizing a product before purchasing, many consumers turn to online product reviews. For any company that exists in the digital space, online reviews are crucial to winning business strategies and maintaining a positive reputation. This data set was collected from the Twitter app to the Kaggle website https://www.kaggle.com/saadbinmanjuradit/ecommerce-reviews-for-women-clothings. Reviews from customers on clothing, in particular, are used to raise the rating of an item.

Dataset D: This set of movie review data from the HBO channel was collected based on ratings given by Reelgood and IMDb. Reelgood is the most extensive streaming guide in the United States and the United Kingdom, with every movie online. Like Reelgood, IMDb (Internet Movie Database) is also a popular source for movie content, ratings and reviews.

Dataset E: Following the growing popularity of the Netflix App, various community reactions emerged and were shared on the Twitter space. The variety of shows featured on Netflix gives viewers space to choose the right movie to watch. This data set has three (3) class labels for recommending suitable shows to watch, namely -1 represents "not recommended", 0 represents "neutral", and +1 represents "recommended". Figure 3.11 shows a sample of movie recommendation data on Netflix.

Dataset F: This data set contains text data regarding movie plots that can influence the percentage of viewers choosing to watch or not. Data were collected from IMDb (Internet Movie Database), specifically on the DisneyPlus channel.

3.2 The ConvNet Architecture

In the ConvNet architecture, all convolutional blocks contain 3 x 3 size kernels, 1 x 1 size pads, and 1 x 1 size strides, while all pooling (max) blocks have 2 x 2 size kernels. After each convolution, batch normalisation (BN) and rectified linear unit (ReLU) activation. The experiment uses 256 memory blocks and 37 output units (26 letters, 10 digits, and 1 set of symbols). We keep the architecture as similar as possible to the conventional convolution networks to measure the effectiveness of modified filtering gates.

ConvNet's layer parameters comprise a set of learnable filters known as Kernels. These kernels are small receptive inputs deep in a convolutional layer, resulting in a two-dimensional activation map. The dot product of text input and text features produced the 2-D activation map. The first layer sets up the sentences in its input neuron as an initialisation. The extraction characteristics with complicated text features make up the second layer. Then,

using text pre-processing algorithms, proceed to the third layer of the modified filter gates.

Initialization;

$$X = [X1, X2, \ldots\ldots, Xn] \quad (1)$$

Equation 1 shows text input initialization X is the text input and n represent the dataset limit.

$$XW \in RDW \; x \; n \quad (2)$$

Equation 2 represents weight and input summation where XW denotes the samples of the weight for the n^{th} view, and DW is the dimension of the n^{th} view's weight. As the size of twitter text has a maximum of 280 characters, the maximum n-value is set to 280 unicode glyphs.

Extraction Features;

$$\text{features, } F_0 = H_{conv1}(input) \quad (3)$$

Equation 3 represents the feature frame for one block of extracted input. H is a unit-impulse response towards the first convoluted frame which relates to the system input and output. $H_{conv1}(X)$ represents separable convolution operation. This general feature extraction formula is then derived as equation 4 to include the value of bias (b) for each batch after normalization and rectification.

$$\text{features, } F_1 = HBN\,(W \; x \; HRELU\,(input) + b), \quad (4)$$

where,

$HBN\,(X)$ = batch normalization

$HRELU\,(X)$ = Rectified Linear Units operation

3.3 The Proposed Model: ConvNet Filter Gate for Text Preprocessing (FGTP)

The ConvNet input neuron has overlapping visual and receptive fields, similar to the animal visual cortex. This overlapping area necessitated the use of filters. Thus we added three algorithms to the ConvNet filter gate to pre-process text. Figure 1 shows the FGTP with Fourier Transform (FT), Porter's Algorithm (PA), and Correction Filter (CF). The FT algorithm's first filter removes redundant data in a dataset by acting as an overlapping text remover. Out-of-vocabulary (OOV) terms were stemmed and removed using PA. Then, any error or misspelt word in text data was rectified using the CF algorithm and the SymSpell API as its library.

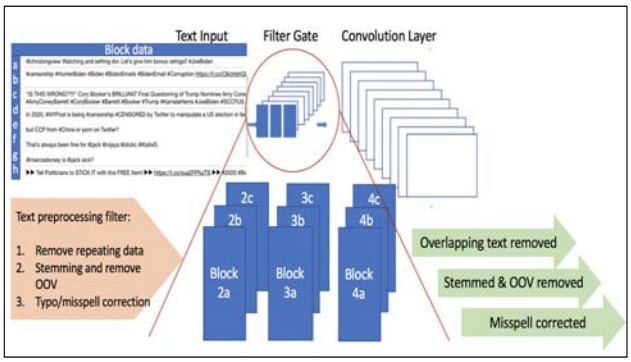

Fig. 1. The proposed FGTP model

3.3.1 Filter Gate I: Text Repetition Removal

The objective of this filter gate is to remove repetition text from input data. Repetition leads to redundancy issues—Fourier Transform method is set into this filtering gate to solve the redundancies problem in the text. The output from this gate is a set of cleaned data from repetitive words.

Gate I: Fourier Transform Algorithm

Step 1: Map string to an array of integers [block = array (2,5,7,7,5, 2, ……..);]

Step 2: Apply Fourier transform on array to get character frequency spectrum
[FT = fft (block);]

Fourier transform can be interpreted as the inner product of the signal x with a complex sinusoid at radian frequency ω. Fourier Transform equation as below:

$$F(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t}dt \quad (5)$$

Step 3: Detect similar text patterns in the character space

Step 4: Detect peaks of the function, FT. (measure relative heights of the peaks, compared to the noise in the background)

Step 5: Set a threshold for the peaks. Peak above the threshold return a similar pattern as shown in Figure 2.

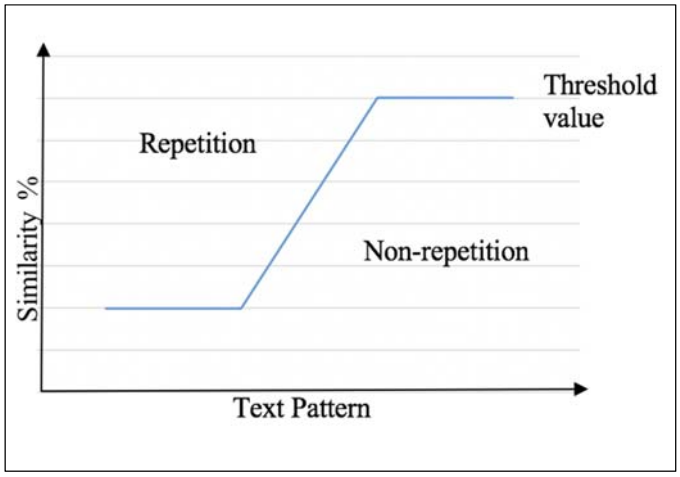

Fig. 2. Text repetition threshold graph

3.3.2 Filter Gate II: Stemming and Remove Out-Of-Vocabulary (OOV) words

The objective of this filtering gate is to remove OOV words from input data. Data from online sources usually have the long text and out-of-vocabulary words that may affect the classification result if not processed. Thus, this gate filters unnecessary text and produces cleaner text data.

Gate II: Porter's Algorithm
Step 1: Lowercasing all text data
Step 2: Perform Porter's algorithm to stem
- 2.1 Remove and recode plurals
- 2.2 Remove 'ed' and/or 'ing'
- 2.3 If vowel = 'yes'; recode 'y' to 'i'
- 2.4 Index penultimate letter to stem
- 2.5 If double suffix = 'yes'; map to single suffix
- 2.6 Index final letter to stem
- 2.7 If endings match stem; remove endings
- 2.8 Index penultimate letter to stem
- 2.9 If stem = <c>vcvc<v>; remove endings
- 2.10 If more than one consonant sequence is present; remove final 'e'
- 2.11 Output stemmed text

Step 3: Perform lemmatization to map a word to its form using WordNet
Step 4: Remove punctuation from each word
Step 5: Filter out non-alphabet text
Step 6: Check for OOV words from GloVe and remove OOV detected.
- 6.1 Input filtered text
- 6.2 Load GloVe vectors of predicted OOV
- 6.3 If OOV word detected;
  remove word;
  update GloVe vocabulary;
- 6.4 Output remaining text

3.3.3 Filter Gate III: Correction Filter

Since the online dataset, mainly from Twitter, consists of a free-writing style from users, the input data comes with misspelt words. Thus, the objective of this gate is to correct the misspelt word based on the built-in library API (SymSpell). The output of this gate is processed text data without spelling errors and is ready to undergo the classification process.

Gate III: Typo/misspell Correction Algorithm
Step 1: Build corpus from Gate 2 filtered text
Step 2: Detect the misspelt word using spell_check_dictionary.txt
Step 3: Correct the misspelt word using SymSpell API
Step 4: Replace the misspelt word with a corrected word in convolutional layer's text block.

3.3.4 Text Classification

The neuron transmits the text's value onto a hidden layer for calculation and classification after ConvNet FGTP filters the text data. All of the dataset's class labels are converted to binary classes of 0 and 1. All datasets were then tested against the traditional ConvNet using Sampling Filtering (SF) as a comparative measure of classification accuracy. For all datasets, Table 2 shows the transformed class label.

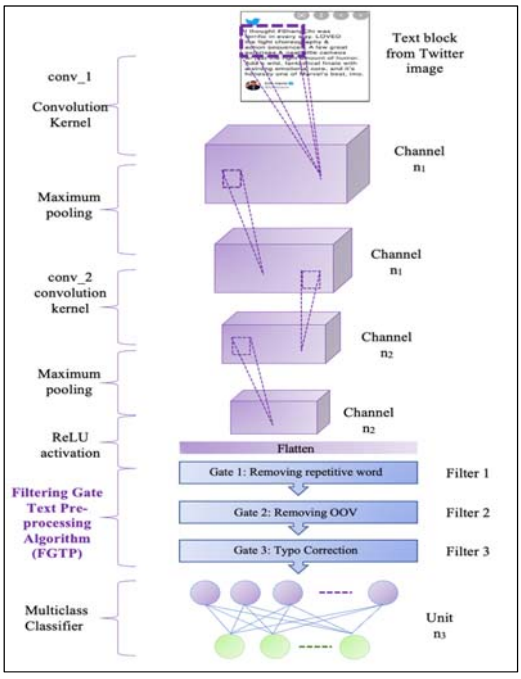

Fig. 3. Classifier with FGTP block diagram

Figure 3 shows a block diagram consisting of the flow of data filtering and data classification. Three filtering gates embedded in Convolution Neural Network to pre-process the extracted text from the data sources are Gate 1: removing the repetitive word, Gate 2: removing the out-of-vocabulary word and Gate 3: typo correction gate. Then the data is flattened and transferred into the input unit to undergo the classification process. We perform two classification methods which are binary classification and multi-class classification, by using the same datasets.

Table 2. Class label for training and testing dataset

Dataset	Class Label	Description
A	Speech class	0 – hate speech; 1– offensive language
B	Feedback type	0 – complaint; 1– support
C	Sentiment value	0 – negative; 1 – positive
D	Sentiment value	0 – negative; 1 – positive
E	Rating	0 – not recommended; 1 – recommended
F	Sentiment value	0 – negative; 1 – positive

To study the effectiveness of binary classification on text, we normalized data sets with label classes exceeding 2 labels. In general, the categories not recommended (not recommended) and slightly good (slightly good) can be combined into one label not recommended. While the sound, excellent and highly recommended categories are combined into the recommendation label. Neutral and recommended categories are also combined into one recommendation label. Table 2 above elaborates on the normalization of class labels for six data sets from multi-class to binary.

4. Experiments

To evaluate the effectiveness of our approach, we run both models of SF and FGTP on the same six datasets. Each dataset was divided into 80% training and 20% evaluation testing for the overall classification model. As for the filtering gate, we set a measurement of precision and recall text data being filtered by the algorithm filtering gate before undergoing the classification process. Results were obtained in each dataset's precision value filtered from input data for filtered text measures. In the iteration of epochs, precision on exact the recognised text is filtered. During the filtering procedure, an error in identifying text was also logged. The precision, recall and error detected during text processing from one gate by one gate as a single filter are demonstrated in section 4.1. Then we run the same experiment measure on a combination of all three gates and yield another table of results, as shown in section 4.2. Finally, we compare the combined FGTP with SF classification performance. We recorded all results and the classification accuracy after filtering, as shown in sections 4.3 and 4.4.

4.1 Single Gate FGTP

The value of precision, recall, and error in detecting the text was recorded for all six datasets. In the next section, the F-score by each gate is also computed as one of the comparison measures with combination gates of FGTP. Table 3 shows the result of text filtering precision and recall by using one by one pre-processing text gate as a single filter gate.

Dataset	Gate I		
	Precision	Recall	f-score
A	0.4113	**0.9100**	0.5665
B	0.6177	0.6406	**0.6289**
C	**0.6210**	0.4987	0.5532
D	0.3744	0.5031	0.4293
E	0.3212	0.7305	0.4462
F	0.5990	0.4063	0.4842

Dataset	Gate II		
	Precision	Recall	f-score
A	0.5843	0.6361	0.6091
B	0.6126	0.3750	0.4652
C	0.4940	0.5014	0.4977
D	**0.7148**	**0.8534**	**0.7780**
E	0.3392	0.7360	0.4644
F	0.5084	0.8413	0.6338

Dataset	Gate III		
	Precision	Recall	f-score
A	0.1032	0.4460	0.1676
B	0.2417	0.4132	0.3050
C	0.1401	0.5080	0.2196
D	0.1763	0.6193	0.2745
E	0.2358	0.6317	0.3434
F	**0.3507**	**0.7280**	**0.4734**

Table 3. Single gate FGTP

Based on text filtering results shown in Table 3, we managed to analyse the f-score value for all three text pre-processing tasks using three different methods, as mentioned earlier. For filtering repetitive words using Gate I, the highest precision of 0.6210 was achieved from dataset C, while the highest recall value was recorded from dataset A. As for the derived value off-score for Gate I, dataset B

yielded the highest value. In a separate experiment, we run the same dataset into Gate II filtering only, where its job is to correct typo errors in the detected text. As for Gate II filtering, the highest precision (0.7148), recall (0.8534) and f-score (0.7780) were recorded on dataset D. While for the subsequent text pre-processing task in Gate III of OOV word, the highest value of precision, recall and f-score recorded on dataset F with 0.3507, 0.7280 and 0.4734 respectively.

4.2. Combined Gate FGTP

The value of precision, recall, and error in detecting the text was also recorded for all six datasets. During the filtering process of all three gates, time is also logged as one of the comparison measures for the combined gate FGTP. Table 4 shows the result of text filtering precision and recall by using combined FGTP.

Table 4. Combined Gate FGTP

Data	Precision	Recall	f-score	Text Error (%)	Time (s)
A	0.731707	**1**	0.845070	24.3	**0.037**
B	0.656716	0.588233	0.620589	**5.11**	0.081
C	0.724138	0.736363	0.743362	14.67	0.62
D	0.826631	0.572529	0.676506	8	0.043
E	0.582433	0.793881	0.671914	36.9	0.711
F	**0.884725**	0.929553	**0.906585**	11	0.05

The precision, recall, and text error observed during data extraction from the convolution layer are shown in Table 4. Precision and recall results are used to calculate the f-score as the equation 6.

$$f - score = 2\,x\,\frac{precision\ x\ recall}{precision+recall} \quad (6)$$

Fig. 4 depicts the filtering precision graph for each of the six datasets. The f-score is the harmonic mean of precision and recall, where precision and recall are determined. There is always a trade-off between precision and recall in most situations.

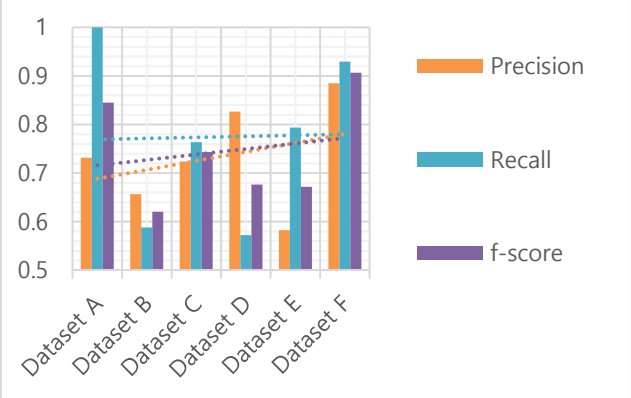

Fig. 4. f-score for combined gate FGTP

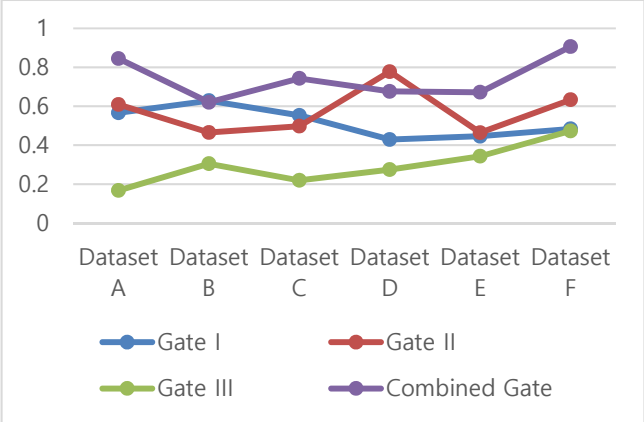

Fig. 5. f-score for single gate FGTP versus combined gate FGTP

Based on a graph shown in Fig.5, we analyse the influence of single gate text pre-processing and the combination of all three gates filtering towards the value of the f-score. For combined FGTP filtering, the f-score value is recorded as better performance from a single gate in four datasets which are set A (0.845070), C (0.743362), E (0.671914) and F (0.906585). Filtering by using Gate II alone for dataset D showed the highest recorded f-score. High typos may influence its typo correction usage in the dataset. Gate I recorded the highest f-score value for dataset A (hate speech and offensive language) consisting of highly repetitive words in the same sentence within their gathered text data. Thus, combined gate FGTP yielded significant text pre-processing results for text classification usage.

4.3. Text classification after FGTP filtering

Table 5 shows the text classification performance after using FGTP as a filter during the text pre-processing. After all six datasets underwent filtering from all the three FGTP gates, the results were recorded.

Table 5. Text Classification Results after FGTP Filtering

Dataset	f_{score}	Classification Accuracy	MSE (%)	Time (s)
A	0.845070	**89.80**	$1e^{-04}$	7.1532
B	0.620589	88.68	$1e^{-06}$	7.0981
C	0.743362	88.91	$1e^{-07}$	7.1095
D	0.676506	79.00	$1e^{-05}$	8.0065
E	0.671914	71.57	$1e^{-07}$	12.1215
F	**0.906585**	82.50	$1e^{-07}$	6.105

With 89.8% classification accuracy, filtered dataset A was the most accurate among all datasets. Meanwhile, the classification accuracy values for datasets B, C, and F are higher than 80%, at 88.68%, 88.91%, and 82.54%, respectively. Datasets D and E had less than 80%

classification accuracy, with 79% and 71.57%, respectively. The mean squared error measures the networkability in adjusting to an error within the classification process. With FGTP as input filtering, time was collected to test algorithm performance in text classification.

4.4. FGTP Filtering vs. Sampling Filtering

The classification accuracy is recorded for each of the six datasets based on their class label. We run the same six datasets through a traditional ConvNet algorithm with random Sampling Filtering (SF) as the filter gate as a comparison measure. Table 6 compares the results of FGTP with the SF filtering approach.

Table 6. Classification Accuracy of FGTP vs SF

Dataset	f-score		Accuracy		MSE (%)		Time (s)	
	FGTP	SF	FGTP	SF	FGTP	SF	FGTP	SF
A	**845070**	**0.77**	**89.80**	**88.65**	$1e^{-04}$	0.01	7.1532	6.0500
B	620589	0.48	88.68	63.50	$1e^{-06}$	$1e^{-04}$	7.0981	5.0015
C	743362	0.45	88.91	61.84	$1e^{-07}$	$1e^{-05}$	7.1095	6.3542
D	676506	0.41	79.00	58.77	$1e^{-05}$	0.1	8.0065	6.8085
E	**671914**	**0.44**	71.57	59.19	$1e^{-07}$	0.001	12.1215	7.4601
F	906585	0.53	82.50	60.03	$1e^{-07}$	$1e^{-04}$	6.1050	3.1068

FGTP produces a greater f-score and accuracy for filtered text than SF, as seen in Table 6. Dataset A produced the highest f-score value of FGTP relative to SF among all six datasets, whereas dataset A yielded the highest classification accuracy using FGTP. FGTP produced lower error rates during the classification process than SF, according to the MSE. FGTP, on the other hand, took longer to process due to its three-layer filtering. The amount of the data and the number of errors will affect the pre-processing time during input filtering.

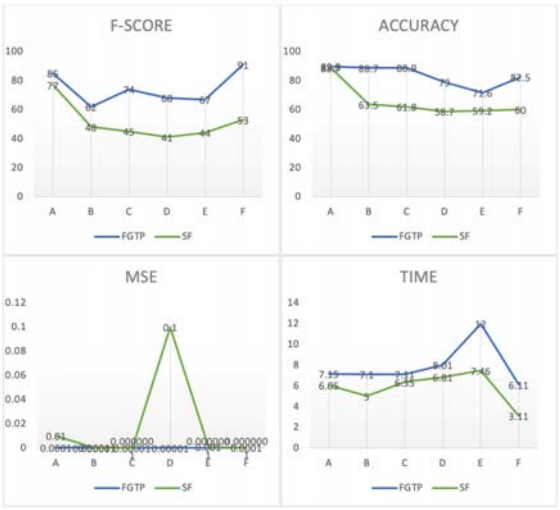

Fig. 6. FGTP performance versus SF

Text pre-processing is used to prepare filtered text data before the classification process to alleviate the problem of data loss in sampling and the difficulty of handling dynamic input. All datasets were successfully filtered using FGTP with various data types, including string, Boolean, ID, DateTime, integer, decimals, and other data types (i.e. symbols). The proposed FGTP precisely filtered text input from dataset A by 0.731707 with a recall value of 1, yielding an f-score of 0.845070. The number of errors in a dataset is represented by text error. For dataset A, the proportion of text error reported is 24.3 per cent, and the filtering time for this dataset is 0.037 seconds. Because the dataset contains two categories of data, string and integer, filtering takes less time than datasets B, C, D, E, and F. The f-score values for datasets B and C are 0.620589 and 0.743362, respectively. The f-score values for datasets D and E, on the other hand, are 0.676506 and 0.671914, respectively. The greatest f-score from dataset F indicates that this set contributed to the most precise outcome of the six datasets. The f-score value for Dataset B, which comprises specific recurring terms of complaint in text data, was the smallest of the six datasets used. Meanwhile, because dataset E had the most detailed text data in every row of records, it took the longest to filter. The filtering time depends on the data column size, but the findings show that FGTP can filter text errors in less than a second. Furthermore, the f-score number was higher than the allowed limit of 60%.

Table 7. Gap Result Between FGTP and SF

Dataset	f-score	Classification Accuracy (%)	Time (s)
A	0.075	1.15	1.1032
B	0.141	25.18	2.0966
C	0.293	27.07	0.7533
D	0.267	20.23	1.198
E	0.232	12.38	4.6614
F	0.377	22.47	2.9982

Table 7 shows the text classification accuracy gap for each dataset based on its class label. FGTP provided somewhat greater classification accuracy than SF in all datasets in this experiment. Dataset F has the most significant f-score value discrepancy between FGTP and SF, with a value of 0.377. In contrast, the difference between datasets A and B is merely 0.075. The f-score ranged from 0.1 to 0.2 on average for the remaining dataset. In terms of classification accuracy, it has been demonstrated that FGTP improved the filter gate within the ConvNet algorithm, resulting in higher accuracy when compared to SF. However, SF took less time to process, ranging from 0.7 to 4.6 seconds shorter than FGTP. Even though SF has a faster processing time than FGTP in this experiment, it has a disadvantage in terms of the high MSE value. A high MSE value will lower classification accuracy in the data processing. In this study experiment,

FGTP with three gates of filtering improved the filtering f-score value and classification accuracy.

5. Conclusion

Stream filtering is one of the most important and practical techniques for efficient stream evaluation because of the nature of data streams. It is done either implicitly by the system to ensure the reliability of stream processing during overload conditions or explicitly by the evaluating function. Six text datasets used three layers of filter to address the issue of random sampling, which results in a lower f-score value of filtered data and lower text classification accuracy. Text preparation characteristics help the convolution layer improve network performance by lowering input reception and classification mistakes. For all datasets used, the accuracy of text classification after additional filter gates were added to the algorithm averaged 83.41 per cent. As a result, the model is suitable for additional analysis, including deep learning research. Finally, the FGTP improved filtering model satisfactorily solves the problem of preparing text data during classification. Our proposed algorithm successfully reduced errors during classification due to filtering gates embedding removed noisy data and improved the accuracy of the classifier model. The performance of classifying text after FGTP is implemented has also proven to be significant compared to conventional Convolution Neural networks. This classifier will be used in future works of classifying multi-class text data for further analysis.

Acknowledgments

We are very grateful to Universiti Teknologi MARA for supporting this research and sincere thanks to our research lab, Data Mining and Optimization (DMO) of Fakulti Teknologi dan Sains Maklumat (FTSM), Universiti Kebangsaan Malaysia, for the expert's knowledge sharing. Massive appreciation to Universiti Kebangsaan Malaysia for providing grant code GGP-2020-032 for this research funding.

References

[1] A.A. Bakar, R. Hamdan, and N.S. Sani, Ensemble learning for multidimensional poverty classification, *Sains Malaysiana*, **49**, 2, (2020), 447-459.

[2] A. Bhardwaj, Sentiment Analysis and Text Classification for Social Media Contents Using Machine Learning Techniques, *2nd International Conference on IoT, Social, Mobile, analytics and Cloud in Computational Vision and bio-Engineering* (ISMAC – CVB 2020).

[3] B. Shi, X. Bai and C. Yao, An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition, *IEEE Transactions on Pattern Recognition*, **39**, 11, (2013).

[4] C. Shuai, J. Lei, Z. Gong and X. Ouyang, Suitability of a New Bloom Filter for Numerical Vectors with High Dimensions, *Journal of PLoS ONE*, **13**, 12, e0209159 DOI: 10.1371/journal.pone0209159, (2018).

[5] E. Mehmood and T. Anees, Challenges and Solutions for Processing Real-Time Big Data Streams: A Systematic Literature Review. *IEEE Access*: DOI: 11.1109/ACCESS.2020.3005268, (2020).

[6] F. Arolfo, K.C. Rodriguez and A. Vaisman, Analyzing the Quality of Twitter Data Streams, *Springer Nature 2020.* DOI: 10.1007/s10796-020-10072-x, (2020).

[7] H. El-Amir and M. Hamdy, Convolutional Neural Network: Deep Learning Pipeline, DOI: 10.1007/978-1-4842-5349-6_11, (2020).

[8] H.I.S. Abuhamad, A.A. Bakar, S. Zainudin, M. Sahani, and Z.M. Ali, Feature selection algorithms for Malaysian dengue outbreak detection model, *Sains Malaysiana*, **46**, 2, (2017), 255-265.

[9] H.J. Yoon, S. Robinson, J.B. Christian, J.X. Qiu and G.D. Tourassi, Filter Pruning of Convolutional Neural Networks for Text Classification: A Case Study of Cancer Pathology Report Comprehension, *IEEE Explore*. DOI: 10.1109/BHI.2018.8333439, (2018).

[10] H. Kamaludin, H. Mahdin and J.H. Abawajy, Filtering Redundant Data from RFID Data Streams, *Journal of Sensors 2016*, DOI: 10.1155/2016/7107914, (2016).

[11] H. Liang, X. Sun, Y. Sun and Y. Gao, Text Feature Extraction Based on Deep Learning: A Review, *EURASIP Journal on Wireless Communications and Networking*, 2017:211, DOI 10.1186/s13638-017-0993-1, (2017).

[12] H. Yenala, A. Jhanwar, M.K. Chinnakotla and J. Goyal, Deep Learning for Detecting Inappropriate Content in Text, *International Journal of Data Science and Analytics*, **6**, (2017), 273-286.

[13] Ihsan, Afdhalul, Rainarli, Ednawati, Optimization of k-Nearest Neighbour to categorize Indonesian's news articles, *Asia-Pacific Journal of Information Technology and Multimedia,* **10**, 1, (2021), 43-51.

[14] I.S. Ahmad, A.A. Bakar, M.R. Yaakub and S.H. Muhammad, A Survey on Machine Learning Techniques in Movie Revenue Prediction, *Journal of Springer Nature Computer Science*, **1**, 235, (2020).

[15] K. Gao, J.V. Hulse and T. Khoshgoftaar, An Evaluation of Sampling on Filter-Based Feature Selection Methods. *Proceedings of the Twenty-Third International Florida Artificial Intelligence Research Society Conference,* (2010).

[16] M. Delakis and C. Garcia, Text Detection with Convolutional Neural Networks, *International Journal on Document Analysis and Recognition (IJDAR),* (2018).

[17] M. Dorfler, T. Grill, R. Bemmer and A. Flexer, Basic Filters for Convolutional Neural Networks: Training or

Design? *Journal of Neural Computing and Applications*, **32**, 12, DOI:10.1007/s00521-01803704-x, (2017).
[18] M. Yousef, K.F. Hussain and U.S. Mohammed, Accurate, data-efficient, unconstrainted text recognition with Convolutional Neural Network, *Elsevier : Pattern Recognition,* **108**, (2020).
[19] N.I. Widiastuti, Convolution Neural Network for Text Mining and Natural Language Processing, *IOP Conference Series: Materials Science and Engineering* **662**, (2019).
[20] P. Jotikabukkana, V.Sornlertlamvanich, O. Manabu and C. Haruechaiyasak, Social Media Text Classification by Enhancing Well-Formed Text Trained Model, *Journal of ICT Research and Application* **10**, 2, (2016), 177-196.
[21] R.A. Rahman, K. Omar, S.A.M. Noah and M.S.N.M. Danuri, A Survey on Mental Health Detection in Online Social Network, *International Journal on Advanced Science Engineering Information Technology*, **8**, 4-2, (2018).
[22] R. Balasubramaniam and K. Nandhini, Algorithms Associated with Streaming Data Problems. *International Journal of Applied Engineering Research* ISSN 0973-4562, **14**, 9, (2019), 2238-2243.
[23] R. Cheng, B. Kao, S. Prabhakar, A. Kwan and Y. Tu, Filtering Data Streams for Entity-Based Continuous Queries, *IEEE Transactions on Knowledge and Data Engineering*, **22**, 2, (2010), 234-248.
[24] S. Ahmad, M.Z. Asghar, F.M. Alotaibi and I. Awan, Detection and Classification of Social Media-Based Extremist Affiliations using Sentiment Analysis Technique, *Human-Centric Computing and Information Sciences,* **9**, 1, 24, (2019).
[25] S. Gaber, M.Z.A. Nazri, N. Omar and S. Abdullah, Part-Of-Speech (POS) Tagger for Malay Language Using Naïve Bayes and K-Nearest Neighbor Model, *Journal of Critical Reviews*, **7**, 16, (2020).
[26] S. Ramesh, A guide to an efficient way to build Neural Network Architectures-PartII: Hyper-parameter Selection and Tuning for Convolutional Neural Network using Hyperas on Fashion-MNIST, *Towardsdatascience.com/a-guide-to-an-efficient-way-to-build-neural-network-architectures-part-i-hyper-parameter-8129009f131b* (2018).
[27] Suhartono, Ashari, D. Ewin, Prastyo, D. Dwi, Kuswanto, Heri and M.H. Lee, Deep Neural Network For Forecasting Inflow And Outflow In Indonesia. *Sains Malaysiana*, **48**, 8, (2019) 1787-1798.
[28] S. Wares, J. Isaacs and E. Elyan, Data Stream Mining: Methods and Challenges for Handling Concept Drift. *Springer Nature Journal* **1**:1412, (2019).
[29] T.N.T. Zakaria, M.J. A. Aziz, M.R. Mokhtar, and S. Darus, Text clustering for reducing semantic information in Malay semantic representation, *Asia-Pacific Journal of Information Technology and Multimedia*, **9**, 2, (2020),11-24.
[30] T. Wang, D.J. Wu, A. Coates and A.Y. Ng, End-to-End Text Recognition with Convolutional Neural Networks, *International Conference on Pattern Recognition,* (2012).
[31] X. Fang, J. Huang, R. Zhang and L. Xu, Convolutional Neural Network-Based Prediction of Protein Thermostability, *Journal of Chemical Information and Modeling*, **59**, 11, (2019).
[32] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu and S. Zhou, AON: Towards Arbitrarily-Oriented Text Recognition, *IEEE/CVF Conference on Computer Vision and Pattern Recognition,* DOI: 10.1109/CVPR40276.2018, (2018).
[33] Z. Zhang, Z. Tang, Z. Zhang, Y. Wang, J. Qin, and M. Wang, Fully-Convolutional Intensive Feature Flow Neural Network for Text Recognition, *24th European Conference on Artificial Intelligence*, (2019).

Nur Suhailayani Suhaimi a lecturer at Universiti Teknologi MARA, Malaysia. A PhD student at Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. One of research member in Data Mining and Optimization Lab of UKM. Area of interests are optimization, Sentiment Analysis and Data Mining.

Zalinda Othman an associate professor (PhD) at Universiti Kebangsaan Malaysia. An active senior lecturer in teaching and research for Faculty of Information Science and Technology. One of research member in Data Mining and Optimization Lab of UKM. Area of interests are optimization, IT in Manufacturing and Data Mining.

Mohd Ridzwan Yaakub an associate professor (PhD) at Universiti Kebangsaan Malaysia. An active senior lecturer in teaching and research for Faculty of Information Science and Technology. Leader of Sentiment Analysis Research Lab. Area of interests are Sentiment Analysis, Opinion Mining and Artificial Intelligence.