Sentiment Analysis of Cultural Product Reviews

Sara Amirsardari

Université du Québec à Montréal, 405 Rue Sainte-Catherine Est, Montréal

Abstract__To extract and sentiment analysis from a verbal description, text-based sentiment detection is employed. Text-based sentiment detection categorized into two main phases, including language representation and classification. Language representation proposes a robust technique to extract the contextual information from the text to increase the quality of feature extraction. Classification employed neural networks to increase classification performance. These techniques have been applied to extract sentiment from the "IMDB" movie review dataset. Three general approaches are represented to detect sentiment analysis, including Rule Construction, Machine Learning (ML), and Hybrid Approaches. The ML approach solves the SD problem by classifying texts into various sentiment categories through the implementation of ML algorithms. Sentiment Analyzer with a classical ML approach achieved fairly good results. The hybrid approach combines the rule-construction and ML approaches into a unified model. This approach has a higher probability of transcending the other two approaches individually. The hybrid approach offers clear comprehension of the extracted feature prior to classification.

Keywords—sentiment analysis; review polarity; language representation, text classification, neural networks, rule construction, machine learning, hybrid approach

I. INTRODUCTION

The study of consumer behavior begun with traditional research in the form of an information processing model [1]. This model defines a consumer as a logical thinker who makes rational choices to make purchasing decisions [1]. Nevertheless, consumer behavior has evolved from a traditional perspective to focus on important consumption phenomena which are a result of irrational buying needs. This viewpoint of consumer behavior has defined hedonic consumption [2].

The hedonic consumption perspective is called motivation research [2]. Unlike traditional consumer research that emphasis on cognitive information, motivation research seeking for sensory-emotional stimulation [2]. This research focused on neglected consumption phenomena that originated from customer behavior. The first consumption phenomenon is called multisensory that consists of tastes, sounds, scents, tactile impressions, and visual images [2]. Emotional arousal is another consumption phenomena related to hedonic consumption that generating psychological and physiological altered states in both the mind and body [1, 2]. Therefore, the hedonic consumption perspective has a huge effect on the emotional aspects of products. It represents products as subjective symbols rather than objective entities that can be key determinants for selecting some products such as esthetic objects [1, 2]. Nevertheless, verbal descriptions derived from consumption phenomena need to be supplemented with sensory impressions [2].

Thus, motivation research to extract and analyze emotions from verbal description employed text-based emotion detection, which is a branch of sentimental analysis [3]. It seeks to extract fine-grained emotions such as happy, sad, angry, and so on, from human languages rather than coarse-grained and general polarity assignments in SA [3].

II. RELATED WORK

The general approaches to detect emotions from texts can be addressed in three categories included: (A) rule construction, (B) machine learning (ML), and (C) hybrid approaches [3].

A. Rule construction method

The rule construction approach works on grammar and logical rules to detect emotions from documents. This approach consists of keyword recognition (KR) and lexical affinity methods [3]. The KR method deals with the construction of emotion dictionaries or lexicons. The LA is responsible for this second stage of assigning probabilistic affinities to the random emotion words [3].

Many works have focused on the rule-based approach [4, 5, 6, 7]. The limitation associated with this approach is the lack of representation of the various categories of emotions. Hence, it reduces them into two "positive" and "negative" extreme states [3]. In addition, this approach can lead to inaccuracies in the classification of emotions depending on the context of the assigned words [3]. These barriers often necessitate the need for using other approaches for detecting emotions in texts.

B. machine learning method

The ML approach solves the ED problem by classifying texts into various emotion categories through the implementation of ML algorithms. On the one hand, the supervised ML algorithms have been offered compares better detection rates than unsupervised ML techniques were implemented. On the other hand, deeper learning techniques are more robust and their deep layers can extract the intrinsic/hidden details of text [3].

Several works have been interested in ML approach using supervised and unsupervised techniques [8, 9, 10, 11, 12, 14, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26]. Different type of limitations associated with their work, including the disregard for semantic information in texts and the role of context in sentences [27, 10], lack of consideration the relationship between features [11], and restricted categories of emotions [17].

C. Hybrid approaches

The hybrid approach combines the rule-construction and ML approaches into a unified model. This approach has a higher probability of transcending the other two approaches individually. However, the hybrid approach needs to obtain the most effective deep learning technique to enhance performance since the approach relies heavily on the particular type of deep learning technique used [3].

Many works have focused on hybrid approach [28, 29, 30, 31, 33, 34, 32]. The limitations associated with these works mentioned that more ML technique could improve their performance [28, 31]. In addition, the ensemble of classifiers performed better than situations where single classifiers were used [29].

This research categorized text-based ED into two main phases, including language representation and classification [3]. During language representation, the extraction of contextual information has a main role in improving classification accuracies [3]. Hence, the major issue is to propose a robust technique for extracting this contextual information from the text. In order to increase the quality of contextual information extraction, Neuro-fuzzy networks would help the limiting effects and increasing classification performance [3]. The attention networks are expected to focus on the extraction of relevant features while the Neuro-fuzzy networks offer clear comprehensibility and classification of the extracted feature prior to classification [3].

This research addresses implementation of a hybrid system that applying machine learning algorithms and their combinations for the classification of movie reviews. Based on my research, there are several feature categories that have been employed in the sentiment analysis included syntactic, semantic, link-based, and stylistic features [35]. As part of my project, it is essential to investigate an efficient feature extraction method for better effectiveness results.

III. DATA PREPARATION AND FEATURE EXTRACTION

A. Data Pre-processing

Data processing is one of the most common tasks in many ML applications. In the field of natural language processing, these applications deal with a huge amount of text to perform classification or translation. Therefore, the goal is transforming text into something that an algorithm can understand. There are several major steps involved in text processing including Tokenization, Converting all characters to lowercase, Removing stop words, removing punctuation, Stemming, and Handling n-grams [40].

In these researches, some of these methods are employed and some of them like removing stop words were skipped because depending on the application, it may be sensible to ensure that certain words are (or aren't!) considered to stop words [40]. Furthermore, BeautifulSoup is a library for extracting data from HTML and XML documents is used as well.

B. Feature extraction methods

In text processing, words of the text represent discrete, categorical features. The question that can be raised is how to encode such data in a way that is ready to be used by the algorithms. To solve this issue, feature extraction which uses the mapping from textual data to real-valued vector is used. Bag of words is one of the simplest techniques to numerically represent text in a bag of words. In this technique, unique words in the text corpus called vocabulary are saved as a list. Hence, each sentence or document represent as a vector. Term Frequency-Inverse Document Frequency (TF-IDF) is another technique that can count the number of times each word appears in a document. Bag of Words just creates a set of vectors containing the count of word occurrences in the document (reviews), while the TF-IDF model contains information on the more important words and the less important ones as well.

One of the major disadvantages of using the BOW technique is discards word order thereby ignoring the context and in turn meaning of words in the document. On the other hand, for natural language processing (NLP) maintaining the context of the words is important. Hence, to solve this problem, the word embedding is represented.

IV. CHOOSING ML ALGORITHMS

For automation of sentiment analysis, different approaches have been applied to predict the sentiments of words, expressions, or documents. In this research classical ML approaches such as Naive Bayes (NB) and Support Vector Machines (SVM) are used. SVM has been used extensively for movie reviews [36, 37, 38], while NB has been applied to reviews and Web discourse [36, 37, 39].

A. Implementation

To implement these techniques, the IMDB movie review provided by (*Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011*) is used. This dataset consists of 50,000 reviews for binary sentiment classification that contains a set of 25,000 highly polar movie reviews for positive reviews, and 25,000 for negative reviews. This dataset was used for the very popular paper 'Learning Word Vectors for the very popular paper 'Learning Word Vectors for Sentiment Analysis'.

B. Testing and evaluation of algorithms

In this research, a sentimental analyzer over the IMDB movie review dataset is applied to classical ML approaches like "Naïve Bayes" and "Support Vector Machines". The data set split into training data (80%) and testing data (20%). In addition, TF-IDF feature extraction is used to convert the text corpus into the feature vectors. TF-IDF feature extraction restricts the maximum features to 10000.The accuracy for Naïve Bayes and SVM algorithms is shown in Table 1:

TABLE I. RESULTS OF ML ALGORITHMS

Naive Bayes Accuracy Score ->	86.11
SVM Accuracy Score ->	89.41

Based on the processing time to get the result, SVM is not suitable for large datasets because of its high training time and it also takes more time in training compared to the naïve Bayes algorithm.

V. DEEP LEARNING TECHNIQUES

Deep learning techniques compare with ML algorithms achieve better results for NLP problems such as sentiment analysis and language translation. Nevertheless, Deep learning models are very slow to train. Thus, for simple text classification problems, classical ML approaches give similar results with quicker training time.

This research focuses on two major tasks that included building deep neural networks for Sentimental Classification and using Word2Vec as a feature extraction method to produce distributed representations of words.

A. Architecture

Deep learning text classification model architectures as shown in "Fig.1" generally consist of the following components connected in sequence.

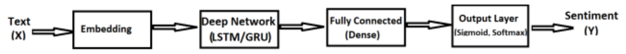


Fig. 1. Deep Learning Architecture [41]

1. Embedding Layer
Word Embedding is a representation of the text where words that have the same meaning have a similar representation. In other words, it represents words in a coordinate system where related words, based on a corpus of relationships, are placed closer together. In deep learning architecture, an embedding layer stores a lookup table to map the words represented by numeric indexes to their dense vector representations [41].
2. Deep Network
The deep network takes the sequence of embedding vectors as input and converts them to a compressed representation. The compressed representation effectively captures all the information in the sequence of words in the text. The deep network part is usually an RNN or some form of it, like LSTM/GRU [41].
3. Fully Connected Layer
This layer takes the deep representation from the RNN/LSTM/GRU and transforms it into the final output classes. This component is comprised of fully connected layers along with batch normalization and optionally dropout layers for regularization [41].
4. Output Layer
This layer can have either Sigmoid for binary classification or Softmax for both binary or multi- classification output.

In this research, four different types of deep neural networks are examined, which included Densely Connected Neural Network (basic neural network), Convolutional Neural Network (CNN). Long Short Term Memory Network (LSTM) and Gated Recurrent Units (GRUs) which are a variant of Recurrent Neural Networks.

B. Word Embedding Models

Word embedding encodes each word into a vector that captures some sort of relation and similarity between words within the text corpus. This means even the variations of words like a case, spelling, and punctuation, and so on will be automatically learned. Some of the well-known models of word embedding have been included Word2Vev and Glove.

Word2vec takes a large corpus of text as its input and produces a vector space with each unique word being assigned a corresponding vector in the space. GloVe is an extension to the word2vec method for efficiently learning word vectors. Glove constructs an explicit word-context or word co-occurrence matrix using statistics across the whole text corpus. GloVe and word2vec differ in their underlying methodology. In other words, word2vec uses predictive models, while GloVe is count based.

C. Train word2vec Embedding

The neural network uses the embedding layer as the first hidden layer. The Embedding layer is initialized with random weights and learns how embedding for all of the words in a training dataset while training a model. In this research, pre-trained word embedding through the word2vec method pass to the embedding layer. This approach uses the Genism implementation of Word2Vec. The first step is to prepare the text corpus for learning the embedding by creating word tokens, removing punctuation, removing stop words and etc. The word2vec algorithm processes documents sentence by sentence.

D. Using Pre-trained Embedding

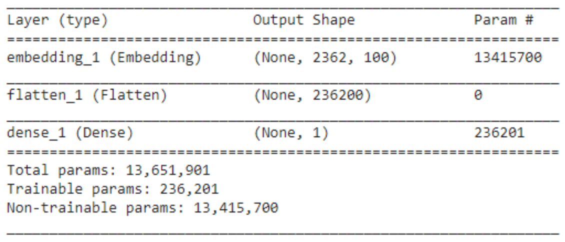
After training the Word2Vec model on the IMDB dataset, the next step is to load the word embedding as a directory of words to vectors. By extracting the word embedding, we are ready to convert it into a tokenized vector. To convert the word embedding into a tokenized vector, it is necessary to vectorize the text samples into two dimensions integer tensor, And then map embedding from the loaded word2vec model for each word to create a word-to-index vocabulary and create a matrix of vectors. In a word-to-index vocabulary, each word in the corpus is used as a key, while a corresponding unique index is used as the value for the key. This trained embedding vector is ready to be used directly in the embedding layer.

VI. TEXT CLASSIFICATION WITH DENSE NETWORKS

A baseline dense network model as the first neural network in this research is implemented. In this network, layers are fully connected by the neurons in a network layer. In other words, the dense layer is a fully connected layer, meaning all the neurons in a layer are connected to those in the next layer [40].

A. Dense Model

The embedding layer creates word vectors from a corpus of documents. In the following, the flatten () layer enables us to pass a many-dimensional output (in this research, a two-dimensional output from the embedding layer into a one-dimensional dense layer). A single one Dense () layer consists of sigmoid activations. Sigmoid activation was chosen because there are only two classes to classify. Hence, if one class has the probability p then the other class has the probability 1-p. Thus, only a single output neuron is required to present output probabilities between 0 and 1 [40]. The dense Model is shown in “Fig.2”.

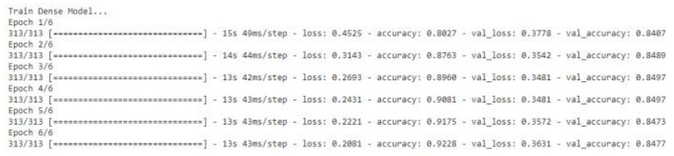


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 2362, 100)         13415700
_________________________________________________________________
flatten_1 (Flatten)          (None, 236200)            0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 236201
=================================================================
Total params: 13,651,901
Trainable params: 236,201
Non-trainable params: 13,415,700
_________________________________________________________________
```

Fig. 2. Dense Model

B. Testing and evaluation of algorithms

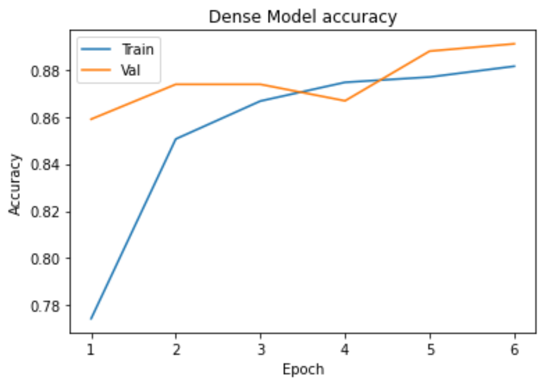
Training the model on the training set and cross-validation on the test set, as shown in “Fig.3”, represents the improvement of model accuracy after each epoch. After a few epochs, the model reaches a validation accuracy of around 92%.



```
Train Dense Model...
Epoch 1/6
313/313 [==============================] - 15s 49ms/step - loss: 0.4525 - accuracy: 0.8027 - val_loss: 0.3778 - val_accuracy: 0.8407
Epoch 2/6
313/313 [==============================] - 14s 44ms/step - loss: 0.3143 - accuracy: 0.8763 - val_loss: 0.3542 - val_accuracy: 0.8489
Epoch 3/6
313/313 [==============================] - 13s 42ms/step - loss: 0.2693 - accuracy: 0.8960 - val_loss: 0.3481 - val_accuracy: 0.8497
Epoch 4/6
313/313 [==============================] - 13s 43ms/step - loss: 0.2431 - accuracy: 0.9081 - val_loss: 0.3481 - val_accuracy: 0.8497
Epoch 5/6
313/313 [==============================] - 13s 43ms/step - loss: 0.2221 - accuracy: 0.9175 - val_loss: 0.3572 - val_accuracy: 0.8473
Epoch 6/6
313/313 [==============================] - 13s 43ms/step - loss: 0.2081 - accuracy: 0.9228 - val_loss: 0.3631 - val_accuracy: 0.8477
```

Fig. 3. Training Dense Model

To evaluate the performance of the model, the test set is passed through the evaluation method. The model reaches a test accuracy of 85%. The model is overwriting on the training set by comparing this result with training accuracy. The loss and accuracy differences between the training and testing set are shown in “Fig.4”.



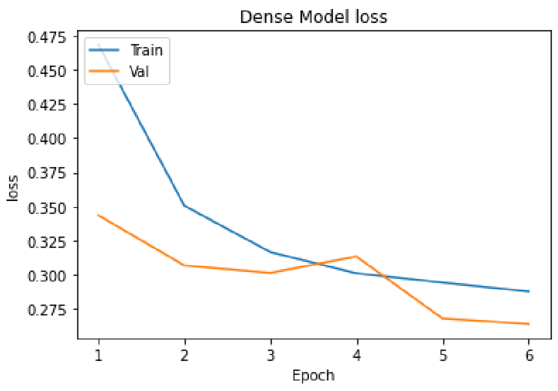


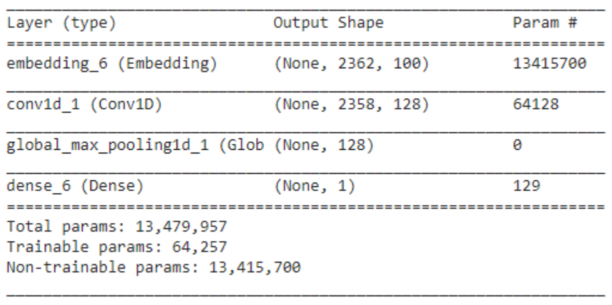
Fig. 4. Dense Model Accuracy and loss

VII. TEXT CLASSIFICATION WITH A CONVOLUTIONAL NEURAL NETWORKS

A convolutional neural network is a type of network that is primarily used for 2D data classification, such as images. A convolutional network tries to find specific features in an image in the first layer. In the next layers, the initially detected features are joined together to form bigger features. In this way, the whole image is detected. Convolutional neural networks have been found to work well with text data as well. Hence, 1D convolutional neural networks are employed to extract features from our data [40].

A. Build CNN Model

A simple convolutional neural network is created with one convolutional layer and one pooling layer. This step is similar to what had done earlier in the first model. The next step is creating a one-dimensional convolutional layer with 128 training examples used during each round of model training. The kernel size is 5 and the activation function is sigmoid. A global max-pooling layer is added to reduce feature size. Finally, a dense layer with sigmoid activation completes the model “Fig.5”.

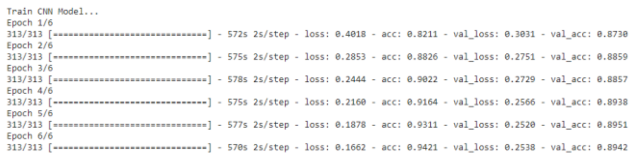


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_6 (Embedding)      (None, 2362, 100)         13415700
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 2358, 128)         64128
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 128)               0
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 129
=================================================================
Total params: 13,479,957
Trainable params: 64,257
Non-trainable params: 13,415,700
_________________________________________________________________
```

Fig. 5. CNN Model

B. Train and evaluation of model

Training the model on the training set and cross-validation on the test set, as shown in “Fig.6”, represents the training accuracy of around 94%, which is greater than the training accuracy of the simple neural network. The test accuracy is around 89%, which is also greater than the test accuracy of the simple neural network, which was around 84%.



```
Train CNN Model...
Epoch 1/6
313/313 [==============================] - 572s 2s/step - loss: 0.4018 - acc: 0.8211 - val_loss: 0.3031 - val_acc: 0.8730
Epoch 2/6
313/313 [==============================] - 575s 2s/step - loss: 0.2853 - acc: 0.8826 - val_loss: 0.2751 - val_acc: 0.8859
Epoch 3/6
313/313 [==============================] - 578s 2s/step - loss: 0.2444 - acc: 0.9022 - val_loss: 0.2729 - val_acc: 0.8857
Epoch 4/6
313/313 [==============================] - 575s 2s/step - loss: 0.2160 - acc: 0.9164 - val_loss: 0.2566 - val_acc: 0.8938
Epoch 5/6
313/313 [==============================] - 577s 2s/step - loss: 0.1878 - acc: 0.9311 - val_loss: 0.2520 - val_acc: 0.8951
Epoch 6/6
313/313 [==============================] - 570s 2s/step - loss: 0.1662 - acc: 0.9421 - val_loss: 0.2538 - val_acc: 0.8942
```

Fig. 6. Training CNN Model

However, the CNN model is still over fitting as there is a vast difference between the training and test accuracy. The loss and the accuracy difference between the training and testing set are shown in “Fig.7”.

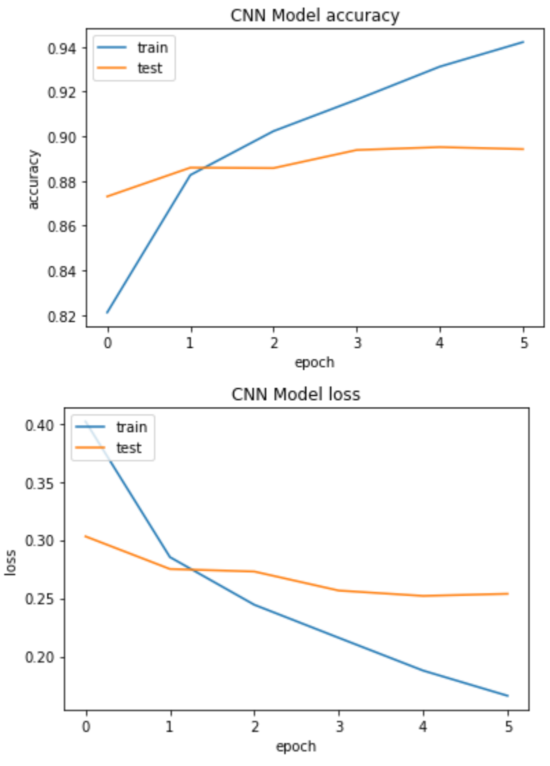


Fig. 7. CNN Model Accuracy and loss

VIII. RECURRENT NEURAL NETWORKS

The filter within the convolutional layers we built in the previous section tend to excel at learning short sentences like five words (kernel length =5). Considering a document of a natural language like a movie review might contain much longer sequences of words. Hence, to handle long sequences of data, a family of deep learning models exists called recurrent neural networks (RNNs). A recurrent neural network is a type of neural network that is proven to work well with sequence data [40]. Since the text is actually a sequence of words, a recurrent neural network is an automatic choice to solve text-related problems [40]. RNNs include specialized layer types including long short-term memory units (LSTMs) and gated recurrent units (GRUs) [40].

Consider the following sentences:” *Jon and Grant are writing a book together. They have really enjoyed writing it”.* With such a small window of text, that neural network had no capacity to assess what “they” or “it” might be referring to [40]. Our human brains can do it because our thoughts loop around each other, and we revisit earlier ideas in order to inform our understanding of the current context [40]. Nevertheless, the concept of recurrent neural networks, which have loops built into their structure that allow information to persist over time. As shown in “Fig.8”, a Schematic diagram of a recurrent neural network is displayed [40].

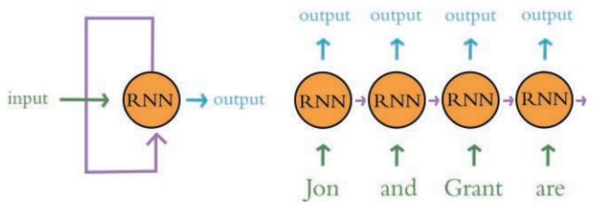


Fig. 8. Schematic diagram of a recurrent neural network [40]

IX. TEXT CLASSIFICATION WITH LSTMs

LSTMs and GRUs defined as the solution to short term memory. These models have internal mechanisms called gates that can regulate the flow of information. These gates can learn which data in a sequence are important to keep or throw away. Hence, the models learn to use relevant information to make predictions. A schematic diagram of an LSTM is shown in “Fig.9”.

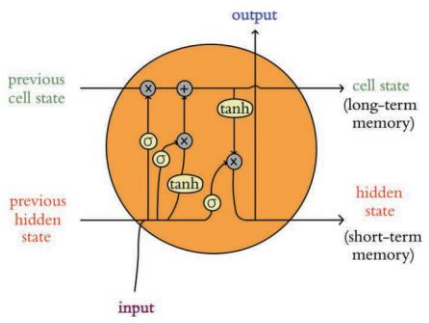
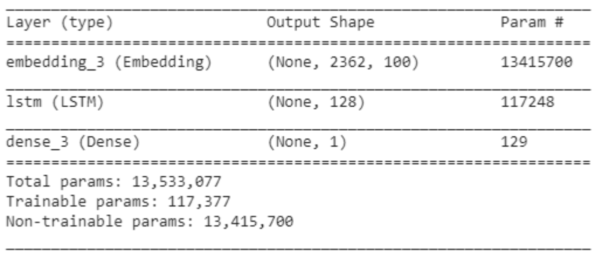


Fig. 9. Schematic diagram of an LSTM [40]

A. Build LSTMs Model

LSTMs model “Fig.10”, which is a variant of RNN is built to solve the sentiment classification problem [40]. In this model, initializing a sequential model followed by the creation of the embedding layer. In the following, a LSTMs layer with 128 training examples is created.

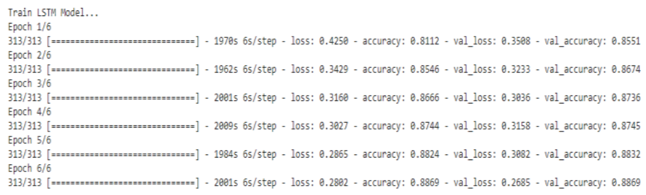


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 2362, 100)         13415700
_________________________________________________________________
lstm (LSTM)                  (None, 128)               117248
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 129
=================================================================
Total params: 13,533,077
Trainable params: 117,377
Non-trainable params: 13,415,700
_________________________________________________________________
```

Fig. 10. LSTMs Model

B. Train and evaluation of model

After building the model, Training the model on the training set and evaluate its performance on the test set, as shown in “Fig.11”, which represents the training accuracy of around 88% once the model is trained. The test accuracy is around 0.88 as well.



```
Train LSTM Model...
Epoch 1/6
313/313 [==============================] - 1970s 6s/step - loss: 0.4250 - accuracy: 0.8112 - val_loss: 0.3508 - val_accuracy: 0.8551
Epoch 2/6
313/313 [==============================] - 1962s 6s/step - loss: 0.3429 - accuracy: 0.8546 - val_loss: 0.3233 - val_accuracy: 0.8674
Epoch 3/6
313/313 [==============================] - 2001s 6s/step - loss: 0.3160 - accuracy: 0.8666 - val_loss: 0.3036 - val_accuracy: 0.8736
Epoch 4/6
313/313 [==============================] - 2009s 6s/step - loss: 0.3027 - accuracy: 0.8744 - val_loss: 0.3158 - val_accuracy: 0.8745
Epoch 5/6
313/313 [==============================] - 1984s 6s/step - loss: 0.2865 - accuracy: 0.8824 - val_loss: 0.3082 - val_accuracy: 0.8832
Epoch 6/6
313/313 [==============================] - 2001s 6s/step - loss: 0.2802 - accuracy: 0.8869 - val_loss: 0.2685 - val_accuracy: 0.8869
```

Fig. 11. Training LSTMs Model

In this model, the test accuracy is better than both the CNN and densely connected neural networks. Also, there is a very small difference between the training accuracy and test accuracy which means that the model is not overfitting.

The output shows that the difference between the accuracy values for training and test sets is much smaller compared to the simple neural network and CNN. Similarly, the difference between the loss values is also negligible, which shows the model is not overfitting. The loss and accuracy differences between the training and testing set are shown in “Fig.12”.

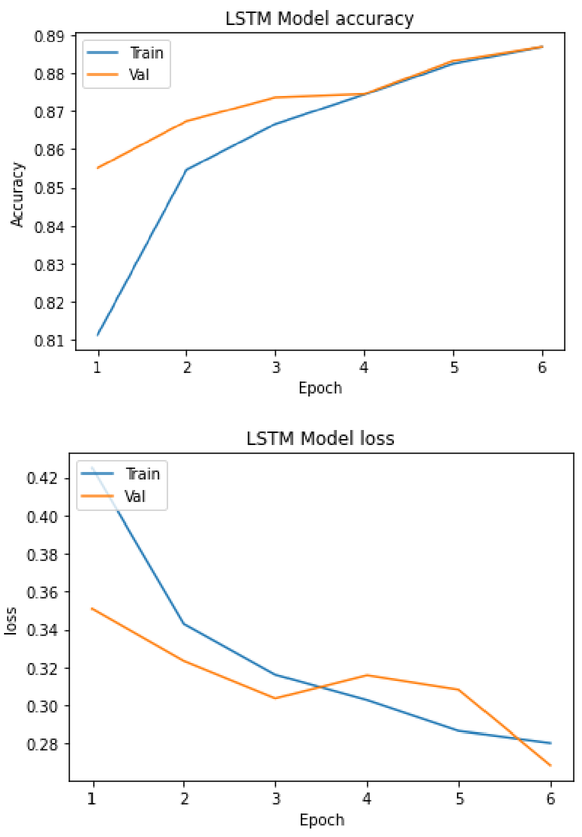


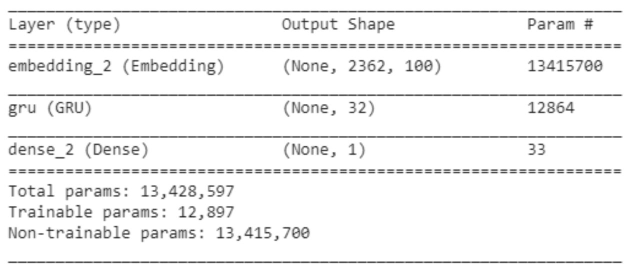
Fig. 12. LSTMs Model Accuracy and loss

X. TEXT CLASSIFICATION WITH GRUs

GRUs model trains faster and perform better than LSTMs model on less training data set. In addition, the GRUs model is simpler and thus easier to modify.

A. Build GRU Model

GRUs Model “Fig.13” is slightly less computationally intensive than the LSTMs model because it involves only three activation functions, and yet its performance often approaches the performance of LSTMs.

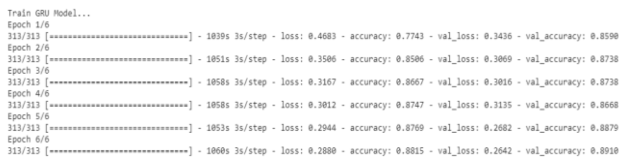


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 2362, 100)         13415700
_________________________________________________________________
gru (GRU)                    (None, 32)                12864
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 13,428,597
Trainable params: 12,897
Non-trainable params: 13,415,700
_________________________________________________________________
```

Fig. 13. GRUs Model

B. Train and evaluation of model

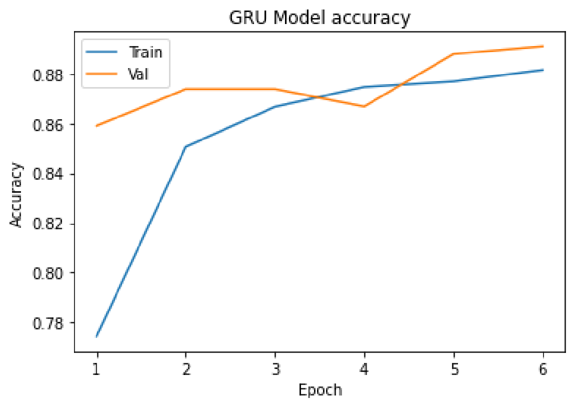
After building the model, Training the model on the training set and evaluate its performance on the test set, as shown in “Fig.14”, Represents the training accuracy of around 88% once the model is trained. The test accuracy is around 89%.



```
Train GRU Model...
Epoch 1/6
313/313 [==============================] - 1039s 3s/step - loss: 0.4683 - accuracy: 0.7743 - val_loss: 0.3436 - val_accuracy: 0.8590
Epoch 2/6
313/313 [==============================] - 1051s 3s/step - loss: 0.3506 - accuracy: 0.8506 - val_loss: 0.3069 - val_accuracy: 0.8738
Epoch 3/6
313/313 [==============================] - 1058s 3s/step - loss: 0.3167 - accuracy: 0.8667 - val_loss: 0.3016 - val_accuracy: 0.8738
Epoch 4/6
313/313 [==============================] - 1058s 3s/step - loss: 0.3012 - accuracy: 0.8747 - val_loss: 0.3135 - val_accuracy: 0.8668
Epoch 5/6
313/313 [==============================] - 1053s 3s/step - loss: 0.2944 - accuracy: 0.8769 - val_loss: 0.2682 - val_accuracy: 0.8879
Epoch 6/6
313/313 [==============================] - 1060s 3s/step - loss: 0.2880 - accuracy: 0.8815 - val_loss: 0.2642 - val_accuracy: 0.8910
```

Fig. 14. Training GRUs Model

In this model, the accuracy values for test sets are better than all the networks evaluated so far, including the CNN, densely connected neural network, and LSTMs. Moreover, the results show the test accuracy is higher than training accuracy which means that the model is not overfitting. Similarly, the difference between the loss values is also negligible, which shows the model is not over fitting. The loss and accuracy differences between the training and testing set are shown in “Fig.15”.



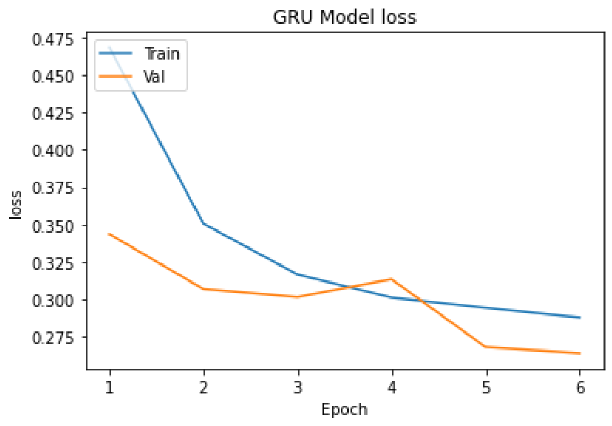


Fig. 15. GRUs Model Accuracy and loss

XI. COMPARISON OF THE PERFORMANCE OF SENTIMENT CLASSIFIER MODEL ARCHITECTURES

To evaluate the performance of the binary-classifying deep learning models, The ROC AUC metric is calculated for each model. As shown in Table 2, the GRUs model has the highest accuracy compared to the rest of the models. RNN is suitable for temporal data, also called sequential data. CNN is considered to be more powerful than RNN. RNN includes less feature compatibility when compared to CNN. RNN unlike feed-forward neural networks can use their internal memory to process arbitrary sequences of inputs.

TABLE II. MODEL ARCHITECTURES PERFORMANCE

Model	**Validation Accuracy**	**ROC AUC (%)**
Dense	84.77	92.19
Convolutional	89.42	96.24
LSTM	88.69	95.65
GRU	89.10	95.85

XII. CONCLUSION

We have described the text processing techniques used in NLP and different feature extraction methods, including Bag of words, TF-IDF, Word2Vec, and Glove. We also demonstrated the use of text processing and build a Sentiment Analyzer with classical ML approaches that achieved fairly good results. In the following, we described in detail the architecture of the Deep Learning model for sentiment classification. We also trained a word2vec model and used it as a pre-trained embedding for sentiment classification.

We applied this knowledge to experiment with deep learning NLP models to classify film reviews as positive or negative. Some of these models involved layer types (dense and convolutional layers), while later ones involved new layer types from the RNN family (LSTMs and GRUs). In a conclusion, deep learning models offer clear comprehensibility of the extracted feature prior to classification.

REFRENCES

[1] Holbrook, M. B., & Hirschman, E. C. (1982). The experiential aspects of consumption: Consumer fantasies, feelings, and fun. *Journal of consumer research*, *9*(2), 132-140.
[2] Hirschman, E. C., & Holbrook, M. B. (1982). Hedonic consumption: emerging concepts, methods and propositions. *Journal of marketing*, *46*(3), 92-101.
[3] Acheampong, F. A., Wenyu, C., & Nunoo-Mensah, H. (2020). Text-based emotion detection: Advances, challenges, and opportunities. *Engineering Reports*, e12189.
[4] Badugu, S., & Suhasini, M. (2017). Emotion detection on twitter data using knowledge base approach. *International Journal of Computer Applications*, *162*(10).
[5] Rabeya, T., Ferdous, S., Ali, H. S., & Chakraborty, N. R. (2017, December). A survey on emotion detection: A lexicon based backtracking approach for detecting emotion from Bengali text. In *2017 20th International Conference of Computer and Information Technology (ICCIT)* (pp. 1-7). IEEE.
[6] Kušen, E., Cascavilla, G., Figl, K., Conti, M., & Strembeck, M. (2017, August). Identifying emotions in social media: comparison of word-emotion lexicons. In *2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)* (pp. 132-137). IEEE.
[7] Seal, D., Roy, U. K., & Basak, R. (2020). Sentence-level emotion detection from text based on semantic rules. In *Information and Communication Technology for Sustainable Development* (pp. 423-430). Springer, Singapore.
[8] Lee, S. Y. M., & Wang, Z. (2015, October). Multi-view learning for emotion detection in code-switching texts. In *2015 International Conference on Asian Language Processing (IALP)* (pp. 90-93). IEEE.
[9] Wikarsa, L., & Thahir, S. N. (2015, November). A text mining application of emotion classifications of Twitter's users using Naive Bayes method. In *2015 1st International Conference on Wireless and Telematics (ICWT)* (pp. 1-6). IEEE.
[10] Allouch, M., Azaria, A., Azoulay, R., Ben-Izchak, E., Zwilling, M., & Zachor, D. A. (2018, December). Automatic detection of insulting sentences in conversation. In *2018 IEEE International Conference on the Science of Electrical Engineering in Israel (ICSEE)* (pp. 1-4). IEEE.
[11] Singh, L., Singh, S., & Aggarwal, N. (2019). Two-stage text feature selection method for human emotion recognition. In *Proceedings of 2nd International Conference on Communication, Computing and Networking* (pp. 531-538). Springer, Singapore.
[12] Chatterjee, A., Narahari, K. N., Joshi, M., & Agrawal, P. (2019, June). Semeval-2019 task 3: Emocontext contextual emotion detection in text. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 39-48).
[13] Tripto, N. I., & Ali, M. E. (2018, September). Detecting multilabel sentiment and emotions from bangla youtube comments. In *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)* (pp. 1-6). IEEE.
[14] Alotaibi, F. M. (2019). Classifying text-based emotions using logistic regression. *VAWKUM Transactions on Computer Sciences*, *16*(2), 31-37.
[15] Mashal, S. X., & Asnani, K. (2017, July). Emotion intensity detection for social media data. In *2017 International Conference on Computing Methodologies and Communication (ICCMC)* (pp. 155-158). IEEE.
[16] Abdullah, M., Hadzikadicy, M., & Shaikhz, S. (2018, December). SEDAT: sentiment and emotion detection in Arabic text using CNN-LSTM deep learning. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 835-840). IEEE.
[17] Ma, L., Zhang, L., Ye, W., & Hu, W. (2019, June). PKUSE at SemEval-2019 task 3: emotion detection with emotion-oriented neural attention network. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 287-291).
[18] Huang, J., Lin, Z., & Liu, X. (2019, April). Episodic Memory Network with Self-attention for Emotion Detection. In *International Conference on Database Systems for Advanced Applications* (pp. 220-224). Springer, Cham.
[19] Polignano, M., Basile, P., de Gemmis, M., & Semeraro, G. (2019, June). A comparison of word-embeddings in emotion detection from text using bilstm, cnn and self-attention. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization* (pp. 63-68).
[20] Ragheb, W., Azé, J., Bringay, S., & Servajean, M. (2019). Attention-based modeling for emotion detection and classification in textual conversations. *arXiv preprint arXiv:1906.07020*.
[21] Barbieri, F., Anke, L. E., Camacho-Collados, J., Schockaert, S., &

Saggion, H. (2018). Interpretable emoji prediction via label-wise attention LSTMs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (pp. 4766-4771).
[22] Malte, A., & Ratadiya, P. (2019, October). Multilingual cyber abuse detection using advanced transformer architecture. In *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)* (pp. 784-789). IEEE.
[23] Huang, C., Trabelsi, A., & Zaïane, O. R. (2019). ANA at SemEval-2019 Task 3: Contextual Emotion detection in Conversations through hierarchical LSTMs and BERT. *arXiv preprint arXiv:1904.00132*.
[24] Huang, Y. H., Lee, S. R., Ma, M. Y., Chen, Y. H., Yu, Y. W., & Chen, Y. S. (2019). EmotionX-IDEA: Emotion BERT--an Affectional Model for Conversation. *arXiv preprint arXiv:1908.06264*.
[25] Ahmad, Z., Jindal, R., Ekbal, A., & Bhattachharyya, P. (2020). Borrow from rich cousin: transfer learning for emotion detection using cross lingual embedding. *Expert Systems with Applications*, *139*, 112851.
[26] Suhasini, M., & Srinivasu, B. (2020). Emotion Detection Framework for Twitter Data Using Supervised Classifiers. In *Data Engineering and Communication Technology* (pp. 565-576). Springer, Singapore.
[27] Jayakrishnan, R., Gopal, G. N., & Santhikrishna, M. S. (2018, January). Multi-class emotion detection and annotation in malayalam novels. In *2018 International Conference on Computer Communication and Informatics (ICCCI)* (pp. 1-5). IEEE.
[28] Grover, S., & Verma, A. (2016, August). Design for emotion detection of punjabi text using hybrid approach. In *2016 International Conference on Inventive Computation Technologies (ICICT)* (Vol. 2, pp. 1-6). IEEE.
[29] Perikos, I., & Hatzilygeroudis, I. (2016). Recognizing emotions in text using ensemble of classifiers. *Engineering Applications of Artificial Intelligence*, *51*, 191-201.
[30] LeCompte, T., & Chen, J. (2017, December). Sentiment analysis of tweets including emoji data. In *2017 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 793-798). IEEE.
[31] Jain, V. K., Kumar, S., & Fernandes, S. L. (2017). Extraction of emotions from multilingual text using intelligent text processing and computational linguistics. *Journal of computational science*, *21*, 316-326.
[32] Tzacheva, A., Ranganathan, J., & Mylavarapu, S. Y. (2019, July). Actionable Pattern Discovery for Tweet Emotions. In *International Conference on Applied Human Factors and Ergonomics* (pp. 46-57). Springer, Cham.
[33] Ramalingam, V. V., Pandian, A., Jaiswal, A., & Bhatia, N. (2018, April). Emotion detection from text. In *Journal of Physics: Conference Series* (Vol. 1000, No. 1, p. 012027).
[34] Ghanbari-Adivi, F., & Mosleh, M. (2019). Text emotion detection in social networks using a novel ensemble classifier based on Parzen Tree Estimator (TPE). *Neural Computing and Applications*, *31*(12), 8971-8983.
[35] Abbasi, A., Chen, H., & Salem, A. (2008). Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. ACM Transactions on Information Systems (TOIS), 26(3), 1-34.[36] B. Pang, L. Lee and S. Vaithyanathan, “Thumbs up? Sentiment Classification using Machine Learning Techniques,” In Proceedings of CoRR 2002.
[37] B. Pang and L. Lee, “A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts,” Proceedings of the ACL, 2004.
[38] C. Whitelaw, N. Garg and Sh. Argamon, “Using Appraisal Groups for Sentiment Analysis,” In Proceedings of the 14th ACM Conference on Information and Knowledge Management, pp. 625-631. 2005
[39] M. Efron, “Cultural orientations: Classifying subjective documents by cocitation analysis,” In Proceedings of the AAAI Fall Symposium Series on Style and Meaning in Language, Art, Music, and Design, 2004, pp. 41–48.
[40] Krohn, J., Beyleveld, G., & Bassens, A. (2019).Deep Learning Illustrated: A Visual, Interactive Guide to Artificial Intelligence. Addison-Wesley Professional
[41]https://towardsdatascience.com/machine-learning-word-embedding-sentiment-classification-using-keras-b83c28087456