Journal of Computer Science

Original Research Paper

A Hybrid Method of Long Short-Term Memory and Auto-Encoder Architectures for Sarcasm Detection

Mohammed M. AL-Ani, Nazlia Omar and Ahmed Adil Nafea

Center for Artificial Intelligence Technology (CAIT), Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Bangi, Selangor, Malaysia

Article history
Received: 29-03-2021
Revised: 09-07-2021
Accepted: 28-09-2021

Corresponding Author:
Mohammed M. AL-Ani
Center for Artificial Intelligence Technology (CAIT), Faculty of Information Science and Technology Universiti Kebangsaan Malaysia (UKM), Bangi, Selangor, Malaysia
Email: mohmed_alanni@yahoo.com

Abstract: Sarcasm detection is considered one of the most challenging tasks in sentiment analysis and opinion mining applications in the social media. Sarcasm identification is therefore essential for a good public opinion decision. There are some studies on sarcasm detection that apply standard word2vec model and have shown great performance with word-level analysis. However, once a sequence of terms is being tackled, the performance drops. This is because averaging the embedding of each term in a sentence to get the general embedding would discard the important embedding of some terms. LSTM showed significant improvement in terms of document embedding. However, within the classification LSTM requires adding additional information in order to precisely classify the document into sarcasm or not. This study aims to propose two technique based on LSTM and Auto-Encoder for improving the sarcasm detection. A benchmark dataset has been used in the experiments along with several pre-processing operations that have been applied. These include stop word removal, tokenization and special character removal with LSTM which can be represented by configuring the document embedding and using Auto-Encoder the classifier that was trained on the proposed LSTM. Results showed that the proposed LSTM with Auto-Encoder outperformed the baseline by achieving 84% of f-measure for the dataset. The main reason behind the superiority is that the proposed auto encoder is processing the document embedding as input and attempt to output the same embedding vector. This will enable the architecture to learn the interesting embedding that have significant impact on sarcasm polarity.

Keywords: Sarcasm Detection, Irony, LSTM, Auto-Encoder, Sentiment Analysis

Introduction

The rise of social networks such as Facebook, Twitter and YouTube have a significant impact on the emergence of new fields such as sentiment analysis (Korayem *et al*., 2012). Sentiment analysis is the task of determining a subjective polarity for a particular social network post (Agarwal *et al*., 2015; Al-Moslmi *et al*., 2017; Altawaier and Tiun, 2016). Assuming a content posted by a regular user to express bad experience regarding a particular product, sentiment analysis aims to analyze the words of such a post in order to classify it into negative polarity. Sentiment analysis has benefited many organizations whom were interested to collect feedback regarding their product or services.

Moreover, the emergence of social networks has changed this study perception of obtaining the news. The nature of social networks enables any regular user to post a wide range of contents that might be spread over the globe (Bharti and Korra, 2019). Numerous fake news has been depicted by either corporate or individual accounts over social networks. Therefore, a new task called Sarcasm Detection has been proposed to detect fake, unbelievable or incorrect information over social networks (Pandey *et al*., 2019). In addition, sarcasm detection in medical care may act as early detection of brain injuries. In research undertaken at London University College, they could show that people with brain injuries had impaired understanding of the sarcasm compared to the control group (Channon *et al*., 2005). Since all the sentiment sentences contain at least

one or two words, the state of art in sarcasm detection studies was mainly depending on document embedding (Cai *et al*., 2019; Misra and Arora 2019; Tay *et al*., 2018). In fact, the standard word2vec model has shown great performance with word-level analysis (Tiun *et al*., 2020), but once a sequence of terms is being tackled, the performance is getting low. This is because averaging the embedding of each term in a sentence to get the general embedding would discard the important embedding of some terms. Therefore, researchers tend to utilize the Long Short-Term Memory (LSTM) where the embedding of each term is being fed to such architecture for obtaining a document embedding. LSTM showed significant improvement in terms of document embedding. However, within the classification of such document embedding into sarcasm or not, LSTM would process the embedding as input and try to output their polarity whether '1' (i.e., sarcasm) or '0' (i.e., not sarcasm). Requires adding additional information in order to precisely classify the document into sarcasm or not. According to (Goodfellow *et al*., 2016), unlike LSTM and CNN where the learning is based on dense, some architectures such as Auto-Encoder would have the ability to learn the deep and hidden features through a recursive process of encoding and decoding the feature space. Such process of encoding and decoding would lead to reconstruct the feature space which may help the architecture to identify significant relationship among the features. Therefore, this study aims to utilize the Auto-Encoder for the process of classification sarcasm detection through tweets reviews. In addition, it aims to improve the accuracy of sarcasm detection. As such, this study may contribute to the development of better sarcasm detector for possibility users or enterprises.

Related Work

Sarcasm is a kind of implicit emotions where the speaker talks in a positive but implies pessimistic or vice versa. Sarcasm is a sarcastic, or a taunting joke made to the listener (Poria *et al*., 2016). It is a kind of speech action in which speakers express their views in an inexplicit way. The complexity of sarcasm makes it difficult for people to identify it (Nafis and Khanna 2015). Sarcasm has been described by researchers as a verbal irony intended to express disdain or ridicule (Joshi *et al*., 2016). In comparison to a negative situation, sarcasm is often taken as positive (Riloff *et al*., 2013. Sarcasm exists in a variety of dimensions, namely missed expectations, cynical insincerity, negative stress and victim presence (Campbell and Katz, 2012). Sarcasm is closely connected with irony - in truth, it is a form of irony. Sarcasm is commonly confused with verbal irony or used interchangeably. Verbal irony is the presentation of one's intention by the use of language that usually signs the opposite, often humorous or emphatic etc.; especially in a manner, style, or attitude suggestive of the use of such language. Sarcasm is an unavoidable part of our lives.

The literature showed great interest for the task of sarcasm detection in which some studies have named such task as irony tweet detection. Tay *et al*. (2018) proposed a sarcasm detection technique using a method called Long-Short Term Memory (LSTM). Such method was intended to generate embedding for each word and then classify the document whether it contains sarcasm or not. This research proposes a neural model focused on attention that looks in between instead than across, allowing it to convey contrast and incongruity. They conducted extensive experiments on six benchmark datasets from Twitter, Reddit and the Internet Argument Corpus. They performed detailed experiments on six Twitter, Reddit and Internet Claim Corpus benchmarking datasets.

Wu *et al*. (2018) proposed another sarcasm detection method using LSTM. The authors have attempted to capture the semantic and syntactic features within the word embedding. Similarly, based on a densely linked LSTM network with multi-task learning strategy in this study proposed work. this research dense LSTM model, where each layer will take every outputs of previous layers as input. They will be used in three classification tasks. Where the last LSTM layer will output the hidden representations of texts.

Baziotis *et al*. (2018) proposed a bidirectional LSTM that was intended to generate both word and character embedding. The authors have utilized the proposed method for detecting sarcasm similarly. They collection 550 million tweets pretrained of English language. During capture all the information of syntactic and semantic. They are augmented model with mechanism attention to determine more informative words.

Cai *et al*. (2019) have proposed an LSTM method for sarcasm detection. The authors have collected Twitter data in order to test the proposed method where the word embedding has been generated similarly. They used multimedia to detect sarcasm for tweets contain image and text in twitter. They deal with feature of text or image attribute they propose fusion model to detect sarcasm on dataset tweets.

Misra and Arora (2019) have proposed a combination of LSTM and Convolutional Neural Network (CNN) for generating word embedding in the task of sarcasm detection. The authors have collected comments from newspapers for the dataset. Similarly, they propose a new dataset which includes news from headlines from websites that contains real and sarcastic ones. This

provides insights about what actually makes sentences sarcastic by using a hybrid Neural Network architecture with attention mechanism.

Methods

This section intends to explain the framework of the proposed embedding by tackling the components of such a framework consists of the following stages. The preprocessing phase on the dataset and its tasks include stop word removal, tokenization and special character removal. Both the term-to-term and document-to-term one-hot encoding will be processed via the LSTM architecture in order to produce the document embedding. LSTM showed significant improvement in terms of document embedding where the generated embedding can accurately simulate the context of significant terms then applied the classifier auto encoder. The following sections state these stages in further details. Figure 1 describes the research framework and along with its stages.

The first stage aims to assemble the dataset for the analysis and execute pre-processing tasks as splitting sentences, tokenization and removing stop words.

The second stage is intended to initiate the document embedding model i.e., Doc2vec. To this end, the data collection will be used to embed the documents. The one-hot encoding for both the words and their documents will be launched.

The obtained vectors will be an input to LSTM architecture to form the final embedding for each document.

Finally, the classification stage discusses the classification method that has been applied to the baseline and the proposed classifier to conduct sarcasm detection and to evaluate the testing data that has been classified using the proposed classifier. The evaluation metrics used in this phase are precision, recall and f-measure.

Dataset

In this study, a recent dataset from SemEval-2018 Task 3 "Irony detection in English tweets" (Van Hee *et al*., 2018) is used. The SemEval-2018 task 3 aims at recognizing tweet irony without clear hashtags of irony. The objective is to determine whether a tweet is ironic or non-ironic (with the binary label value given 0 or 1). The dataset used is irony and there are some study that focuses primarily on sarcastic irony (also known as sarcasm), a popular type of irony most frequently used in interpersonal communication situations (Hancock, 2004). There are two types of "SemEval-2018 task 3" dataset; type1 and type 2. The type1 dataset has two labels; ironic or non-ironic. The type2 dataset is the same one, but it is with different labels. The type2 has four labels; V-irony, O-irony, S-irony and non-ironic. This study uses the type1 of "SemEval-2018 task 3 "Dataset where the labels of the class are: Ironic or non-ironic. The dataset consists of 4618 tweet data written in English. The dataset was split into a distribution of 90% training and 10% test set. The training set consists of 3834 documents (1911 of sarcasm and 1923 of non-sarcasm) and 784 documents for the test set (311 of sarcasm and 473 of non-sarcasm). Table 1 shows contents of the information in the dataset. The detailed statistics of the dataset in this task are shown in Table 2 (Wu *et al*., 2018).

Preprocessing

After clearly defining the dataset, the preprocessing tasks are performed on these data. First, sentence splitting is used to break the text into a sequence of sentences by defining the borders of the sentences. Besides, tokenization has been used to convert sentences into sequences of words. Lastly, the stop words will be removed because these words are considered a language common words that do not hold any effect on the meaning.

One-Hot Encoding

This is considered one of the basic steps to get any word embedded (Xiao and Cho, 2016). The following is an example of two documents from dataset for simplicity and Table 3 illustrates these documents.

Each term is taken from the two documents in Table 3 and will be considered without the stop words as in Table 4. In addition to taking from vector of terms by using one hot encoding representing of term to term, there is also a need for vector of term with its document by using one hot encoding to represent term to document, as depicted in Table 5.

In order to produce document embedding, both of the term-to-term encodings as in Table 4 and document-to-term encoding as in Table 5 will be processed using LSTM.

Long-Short Term Memory

LSTM is like any simple neural network. It has the same layers but LSTM has an additional layer called the memory layer which basically extends their memory. So, it is suitable and easier to remember past data in memory Alameri and Mohd (2021). The popular application of LSTM is in various tasks involving Natural Language Processing (NLP) (Zaremba *et al*., 2014). This layer has the benefit of storing input information, which can be used for the next input.

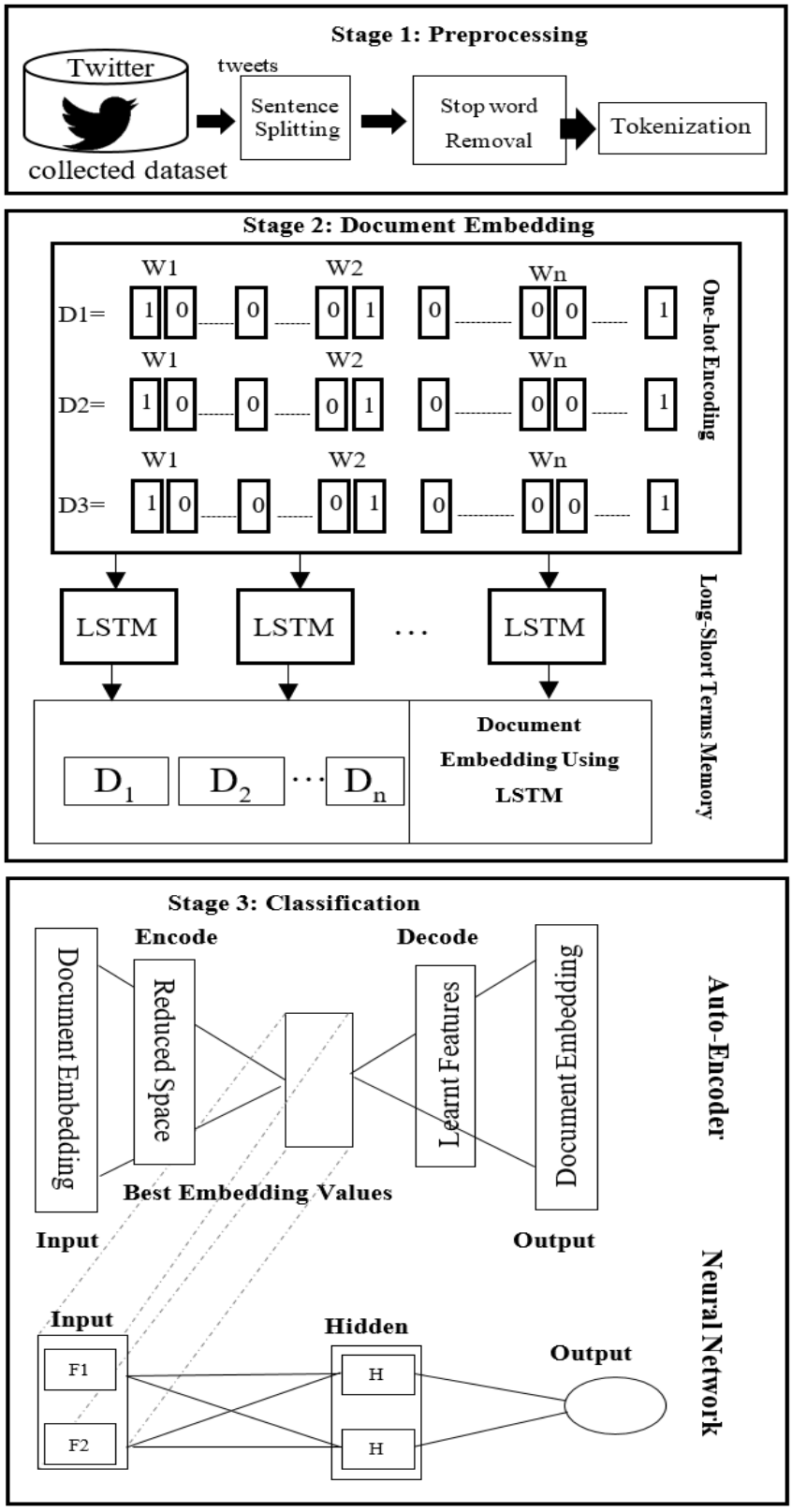

Fig. 1: Stage of the research framework of the proposed method

Table 1: The data consists of the dataset

Data	Description
Tweet index	Unique index for a tweet user.
Label	A label that marks the tweets as potentially sarcasm 1: For sarcasm; and 0: For non-sarcasm
Tweet text	The text of the tweets.

Table 2: The detailed Statistics of the dataset

Label	Irony	Non-ironic
Train	1911	1923
Test	311	473

Table 3: Sample of tweet documents

D_1	"Love working and be exhausted all the time"
D_2	"Love this weather"

Table 4: One-hot encoding for the terms

	Love	Working	Exhausted	Time	Weather
love	1	0	0	0	0
working	0	1	0	0	0
exhausted	0	0	1	0	0
time	0	0	0	1	0
weather	0	0	0	0	1

Table 5: One-hot encoding for the documents

	Love	Working	Exhausted	Time	Weather
D_1	1	1	1	1	0
D_2	1	0	0	0	1

Table 6: Experimental settings

Experiment	Description
Feature	Document Embedding (LSTM)
Classifiers	Auto-Encoder (AE)
Dataset	SemEval-2018 Task 3 Subtask A "Irony detection in English tweets (Van Hee *et al*., 2018)
Training and testing	90% for training (3834) and 10% for testing (784)

Table 7: Number of epochs for each neural network architecture

Architecture	Number of epochs
AE	Experimenting (100 - 800)
NN	1

Table 8: AE architecture details

AE Layer	Size	Details
Input Layer	4, 5, 10, 20 and 30	Hidden layer size of AE
Hidden layer	Input + output/2	The mean average between input size and output size
Output layer	1	Number of class labels

Table 9: Results of Auto-Encoder

	Recall	Precision	F-measure
Baseline1 (Wu *et al*., 2018) (LSTM)	79.92	78.65	70.54
Baseline2 (Baziotis *et al*., 2018) (LSTM)	80.06	63.04	78.56
Proposed Method (AE)	83.17	86.78	84.94

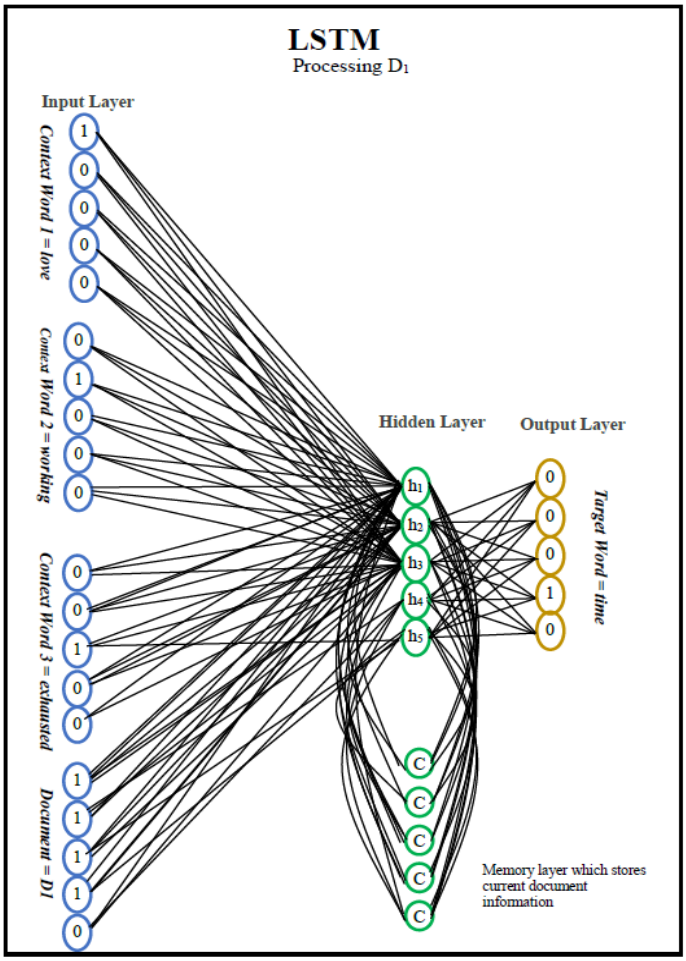

Fig. 2: Document Embedding Generation for D1 Using LSTM

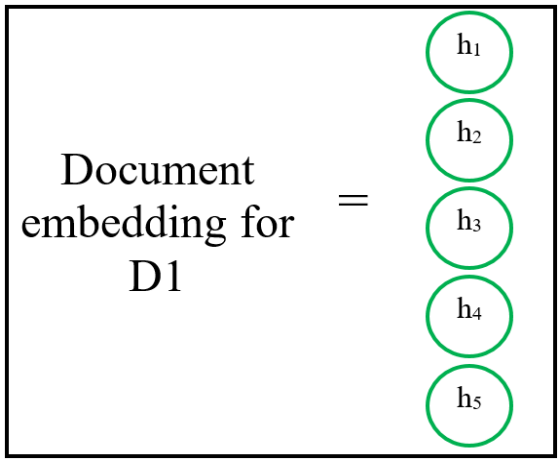

Fig. 3: Representation of document embedding

DOI: 10.3844/jcssp.2021.1093.1098

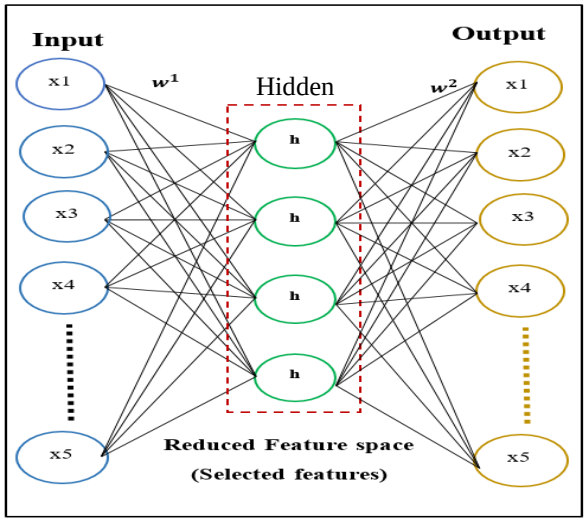

Fig. 4: Feature selection of auto-encoder

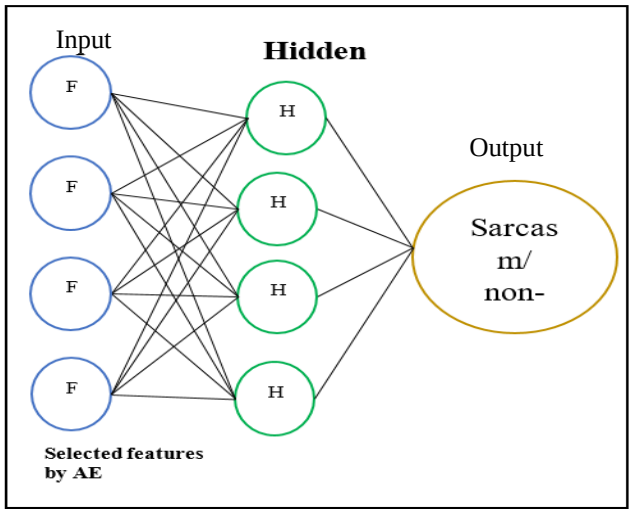

Fig. 5: Classification using NN

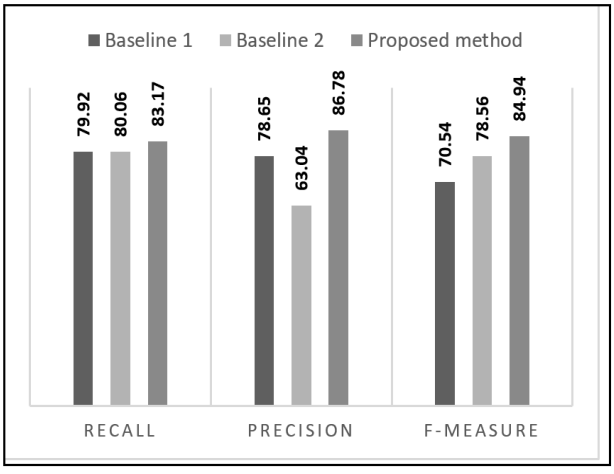

Fig. 6: Comparison between the proposed work and baselines

After getting the vector for each word and document, this study uses LSTM to get document embedding for each word which is "love" " working" and "exhausted " as a vector for D1 from Table 4 with its document as a vector for D1 from Table 5 and 6 output the last word which is "time" as a vector for D1 from Table 4. LSTM is needed to predict the next term in order to get the document embedding as shown in Fig. 2.

The LSTM training will be provided in which the weights are randomly assigned values and multiplied by the neurons to match with the output vectors. It is obvious that the initial epochs would reflect differences between the target output and the expected output. Backpropagation will be used to lower the error rate until the predicted output matches the actual output. The document embedding will be represented in the hidden layer and this document will be processed through the auto encoder as the input as shown in Fig. 3.

After getting the document embedding from the tweeted document generated by the LSTM document embedding method, the Auto-Encoder will be used to classify the documents into sarcasm or non-sarcasm.

Auto-Encoder

Auto-Encoder (AE) is one of Neural Network architectures that has unique and customized layers. In fact, the main aim behind AE is to learn a compressed and distributed representation of a given data (Mighan and Kahani, 2018). In other words, AE aims to process data as input and output the same data itself.

As shown in Fig. 4, the input of AE is the features of a document embedding from Fig. 3, while the output is the same values of the features. The first layer is also known as encoding within the AE where the data is being encoded and decoded. The input weights will be initiated with random values. Then the hidden layer weights will be computed. After that, the hidden weights will be initiated with random values in order to compute the output. After considering an activation function, the predicted output will be compared against the actual output to calculate the error. If there is an error, backpropagation will be used to lower the error rate until the predicted output matches the actual output. Once the error is being minimized to zero where precited output is identical to the actual output, the hidden neuron values will be considered as the selected and reduced feature space as shown in Fig. 4.

Neural Network Classification

After acquiring the selected features (hidden layer) by the proposed AE, simple neural network will be used to classify the documents embedding into sarcasm and non-sarcasm. As shown in Fig. 5, the input of this neural network is the set of selected features produced by the AE.

To evaluate the performance of the classification, there are three important variables should be declared which are True Positive (TP) is the amount of words that have been classified correctly, False Positive (FP) is the amount of words that have been classified incorrectly and True Negative (TN) is the amount of words that have not been classified (Alhutaish and Omar, 2015). Precision is the total of retrieval information, can be calculated as follow:

$$Precision = TP / TP + FP \tag{1}$$

Recall is the total of the correct items among those that retrieved and calculated as follows:

$$Recall = TP / Tp + TN \tag{2}$$

where F-measure is the most popular measures for evaluation of classification systems, where it combines precision and recall. It is a combination of precision and recall which can be calculated as follows.

$$F - measure = 2 \times Precision \times Recall / Precision + Recall \tag{3}$$

This study is using the same parameters as in Baseline 1 (Wu *et al*., 2018) for document embedding, but with AE, it is trained for 800 epochs with the learning rate set to 0.001. After determining the length of each layer within AE and NN, it is important to mention the activation functions used by both architectures. The activation function used by AE was the Rectified Linear Units (Relu) which can be calculated as follow:

$$Relu(x) = \max(0, x)\begin{cases} if\ \ x < 00 \\ if\ \ x \geq 0x \end{cases}, \tag{4}$$

According to Krizhevsky *et al*. (2012), Relu has remarkable performance with deep learning architecture. Therefore, it has been used with the proposed AE. On the other hand, the NN classification has utilized the Logistic Sigmoid, which can be calculated as follows:

$$Sigmoid(x) = \frac{1}{1 + e^{-x}}, \tag{5}$$

The last parameter related to the neural network architecture is the number of epochs. In fact, epoch is the iteration required to accommodate error-tuning by changing the weights' values in order to reduce the error rate. Table 7 represents number of epochs.

In fact, determining the number of epochs is a challenging issue. It is a common agreement that selecting a greater number of epochs would lead to better result of accuracy. This is because the greater number of epochs would contribute toward minimizing the error rate which directly improves the accuracy. However, in order to get an efficient performance, it is better to get a high accuracy as much as possible using the minimal number of epochs.

In this regard, this study has utilized different values for the epoch number for the proposed AE (i.e., from 100 iterations to 800). Yet, for the NN classification, only one epoch has been selected. The reason behind this is that AE would have been extensively trained on the data in order to produce the most accurate sub-set of features. Hence, there is no need for exhausted training conducted on the NN classification. Before applying the proposed AE, it is necessary to initiate a strategy for selecting the required number of features (i.e., hidden size of AE) (Xu *et al*., 2016). In this regard, many experiments have been performed on a general investigation where five number of features are being used to highlight the performances results of accuracy. For this purpose, the number of features used ranges from 30, 20, 10, 5 and 4. This study finds the number of features or hidden size of AE. Through our experiments, it was found that the hidden layer at 4 achieved the best result of f-measure The result increased as the number of epochs increased until it reaches the maximum f-measure of 84.94%. When the number of epochs was 800. This shows that four features are the most accurate reduction of the features produced by AE. Table 8 represents the size of the three layers used by NN.

Results and Discussion

The results of applying Auto-Encoder for sarcasm detection are depicted in this section. As similar to related work, the evaluation of the experiments has been performed using precision, recall and f-score. Table 9 shows these results.

The proposed method has gained better F-score as compared to the baseline by achieving 84.94% as compared to 70% in baseline 1 (Wu *et al*., 2018) and 78% in baseline 2 (Baziotis *et al*., 2018). In fact, the reason behind the superiority of the proposed method over the baseline studies lies on the mechanism of learning within the neural network architecture. While the baseline studies were intended to process the document embedding as input and attempted to output the polarity of sarcasm, the proposed auto encoder is processing the document embedding as input and attempt to output the same embedding vector. This will enable the architecture to learn the interesting embedding that has significant impact on sarcasm polarity. In general, the performance of the proposed AE is superior to that of the baseline ones in terms of detecting sarcasm. Figure 6 shows comparison of results between proposed and baseline approaches.

Regardless of the baseline, this study should be compared with the state of art methods such as Potamias *et al*. (2020). They have used transfer learning system (RCNN-RoBERTa) and achieved 80% of f-measure. Naseem *et al*. (2020) have used transformer based Deep Intelligent Contextual Embedding (T-DICE) with attention-based BiLSTM and achieved 82% of f-measure. The reason behind the outperformance of the proposed method over the state of art studies lies on the use of the auto-encoder. Auto-Encoder has the ability to learn the deep and hidden features through a recursive process of encoding and decoding the feature space. Such process of encoding and decoding would lead to reconstruct the feature space which may help the architecture to identify significant relationship among the features within the neural network architecture. Meanwhile the state of art methods requires adding additional information in order to precisely classify the document into sarcasm or not.

However Naseem *et al*. (2020) achieved f-measure of 93 and 90% respectively with Sarcastic Rillof's dataset. Kumar *et al*. (2019) have used sAtt-BLSTM convNet for the same task and acquired an f-measure of 93% with Sem Eval 2015 task 11 datasets. In this case, this comparison is not possible due to the difference in the datasets used even though the proposed method is still competitive and complex.

Conclusion

This study proposed a hybrid the LSTM of document embedding model and AE for sarcasm detection task. The proposed hybrid model has outperformed the baseline results. This study has proposed Auto encoder to improve the classification of sarcasm detection through tweets reviews. As such, this study may contribute to the development of better sarcasm detector for possibility users or enterprises.

The main limitation of this study was that no real-time dataset was used where a new tweet can be considered as sarcastic especially with the circumstances that we are living nowadays under covid-19 pandemic. Exploring real-time sarcasm tweets would contribute toward discovering new sarcasm for example in situation like the pandemic where this should be taken seriously even if it does not affect other people. In

addition, further examination on a new model architecture of auto encoder which combines it with LSTM could have promising results in sarcasm detection task.

Acknowledgement

The authors would like to thank all friends and colleagues at the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), for their support and assistance. The author would also like to thank the UKM for funding the research.

Funding Information

This project is funded by UKM under the research code GP-2020-K007009.

Author's Contributions

Mohammed M. AL-Ani: Executed methods and designed the research framework for sarcasm detection in this research and he applied the suggested work and conducted the experiments.

Ahmed Adil Nafea: Helped to bring the ideas and contributed in writing the paper.

Nazlia Omar: Advised on the writing and realization of this manuscript.

Ethics

This article is original and contains previously unpublished content. The corresponding author certifies that all other writers have read and accepted the manuscript and that there are no ethical concerns.

References

Agarwal, B., Mittal, N., Bansal, P., & Garg, S. (2015). Sentiment analysis using common-sense and context information. Computational intelligence and neuroscience, 2015. doi.org/10.1155/2015/715730

Alameri, S. A., & Mohd, M. (2021). Comparison of Fake News Detection using Machine Learning and Deep Learning Techniques. 2021 3rd Int. Cyber Resil. Conf., pp. 1–6. IEEE

Alhutaish, R., Omar, N. (2015). Arabic text classification using k-nearest neighbour algorithm. Int. Arab J. Inf. Technol. 12(2), 190-95. https://www.ccis2k.org/iajit/PDF/vol.12,no.2/5359.pdf

Al-Moslmi, T., Omar, N., Abdullah, S., & Albared, M. (2017). Approaches to cross-domain sentiment analysis: A systematic literature review. Ieee access, 5, 16173-16192. doi.org/10.1109/ACCESS.2017.2690342

Altawaier, M. M., & Tiun, S. (2016). Comparison of machine learning approaches on arabic twitter sentiment analysis. International Journal on Advanced Science, Engineering and Information Technology, 6(6), 1067-1073. https://core.ac.uk/download/pdf/296922515.pdf

Baziotis, C., Athanasiou, N., Papalampidi, P., Kolovou, A., Paraskevopoulos, G., Ellinas, N., & Potamianos, A. (2018). Ntua-slp at semeval-2018 task 3: Tracking ironic tweets using ensembles of word and character level attentive rnns. arXiv preprint arXiv:1804.06659. https://arxiv.org/abs/1804.06659

Bharti, S. K., & Korra, S. B. (2019). Sarcasm detection in twitter data: a supervised approach. In Semantic Web Science and Real-World applications (pp. 246-272). IGI Global. doi.org/10.4018/978-1-5225-7186-5.ch010

Cai, Y., Cai, H., & Wan, X. (2019, July). Multi-modal sarcasm detection in twitter with hierarchical fusion model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 2506-2515). https://aclanthology.org/P19-1239.pdf

Campbell, J. D., & Katz, A. N. (2012). Are there necessary conditions for inducing a sense of sarcastic irony?. Discourse Processes, 49(6), 459-480. doi.org/10.1080/0163853X.2012.687863

Channon, S., Pellijeff, A., & Rule, A. (2005). Social cognition after head injury: Sarcasm and theory of mind. Brain and Language, 93(2), 123-134. doi.org/10.1016/j.bandl.2004.09.002

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

Hancock, J. T. (2004). Verbal irony use in face-to-face and computer-mediated conversations. Journal of Language and Social Psychology, 23(4), 447-463. doi.org/10.1177/0261927X04269587

Joshi, A., Tripathi, V., Patel, K., Bhattacharyya, P., & Carman, M. (2016). Are word embedding-based features useful for sarcasm detection?. arXiv preprint arXiv:1610.00883. https://arxiv.org/abs/1610.00883

Korayem, M., Crandall, D., Abdul-Mageed, M. (2012). Subjectivity and sentiment analysis of arabic: A survey. Int. Conf. Adv. Mach. Learn. Technol. Appl., pp. 128-39. Springer

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Kumar, A., Sangwan, S. R., Arora, A., Nayyar, A., & Abdel-Basset, M. (2019). Sarcasm detection using soft attention-based bidirectional long short-term memory model with convolution network. IEEE access, 7, 23319-23328. doi.org/10.1109/ACCESS.2019.2899260

Mighan, S. N., & Kahani, M. (2018, May). Deep learning based latent feature extraction for intrusion detection. In Electrical Engineering (ICEE), Iranian Conference on (pp. 1511-1516). IEEE. doi.org/10.1109/ICEE.2018.8472418

Misra, R., & Arora, P. (2019). Sarcasm detection using hybrid neural network. arXiv Prepr. arXiv1908.07414

Nafis, S. T. O. P. T., & Khanna, S. (2015). An improved method for detection of satire from user-generated content. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.736.6121&rep=rep1&type=pdf

Naseem, U., Razzak, I., Musial, K., Imran, M. (2020). Transformer based deep intelligent contextual embedding for twitter sentiment analysis. Futur. Gener. Comput. Syst. 113:58–69. doi.org/10.1016/j.future.2020.06.050

Pandey, A. C., Seth, S. R., & Varshney, M. (2019). Sarcasm detection of amazon alexa sample set. In Advances in Signal Processing and Communication (pp. 559-564). Springer, Singapore. doi.org/10.1007/978-981-13-2553-3_54

Poria, S., Cambria, E., Hazarika, D., & Vij, P. (2016). A deeper look into sarcastic tweets using deep convolutional neural networks. arXiv preprint arXiv:1610.08815. https://arxiv.org/abs/1610.08815

Potamias, R. A., Siolas, G., & Stafylopatis, A. G. (2020). A transformer-based approach to irony and sarcasm detection. Neural Computing and Applications, 32(23), 17309-17320.Potamias, R. A., Siolas, G., & Stafylopatis, A. G. (2020). A transformer-based approach to irony and sarcasm detection. Neural Computing and Applications, 32(23), 17309-17320. doi.org/10.1007/s00521-020-05102-3

Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., & Huang, R. (2013, October). Sarcasm as contrast between a positive sentiment and negative situation. In Proceedings of the 2013 conference on empirical methods in natural language processing (pp. 704-714). https://aclanthology.org/D13-1066.pdf

Tay, Y., Tuan, L. A., Hui, S. C., & Su, J. (2018). Reasoning with sarcasm by reading in-between. arXiv preprint arXiv:1805.02856. https://arxiv.org/abs/1805.02856

Tiun, S., Mokhtar, U. A., Bakar, S. H., & Saad, S. (2020, April). Classification of functional and non-functional requirement in software requirement using Word2vec and fast Text. In Journal of Physics: Conference Series (Vol. 1529, No. 4, p. 042077). IOP Publishing. https://iopscience.iop.org/article/10.1088/1742-6596/1529/4/042077/meta

Van Hee, C., Lefever, E., & Hoste, V. (2018, June). Semeval-2018 task 3: Irony detection in english tweets. In Proceedings of The 12th International Workshop on Semantic Evaluation (pp. 39-50). https://aclanthology.org/S18-1005.pdf

Wu, C., Wu, F., Wu, S., Liu, J., Yuan, Z., & Huang, Y. (2018, June). Thu_ngn at semeval-2018 task 3: Tweet irony detection with densely connected lstm and multi-task learning. In Proceedings of The 12th International Workshop on Semantic Evaluation (pp. 51-56). https://aclanthology.org/S18-1006.pdf

Xiao, Y., & Cho, K. (2016). Efficient character-level document classification by combining convolution and recurrent layers. arXiv preprint arXiv:1602.00367. https://arxiv.org/abs/1602.00367

Xu, Q., Zhang, C., Zhang, L., & Song, Y. (2016, August). The learning effect of different hidden layers stacked autoencoder. In 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC) (Vol. 2, pp. 148-151). IEEE. doi.org/10.1109/IHMSC.2016.280

Zaremba, W., Sutskever, I., & Vinyals, O. (2014). Recurrent neural network regularization. arXiv preprint arXiv:1409.2329. https://arxiv.org/abs/1409.2329