Using LSTM for Context Based Approach of Sarcasm Detection in Twitter

Siti Khotijah[†]
School of Computing
Telkom University
Bandung, Indonesia
sitikhotijah@student.telkomuniversity.ac.id

Jimmy Tirtawangsa
School of Computing
Telkom University
Bandung, Indonesia
jimmytirtawangsa@telkomuniversity.ac.id

Arie A Suryani
School of Computing
Telkom University
Bandung, Indonesia
ardiyanti@telkomuniversity.ac.id

ABSTRACT

In this research, we propose a sarcasm detection by taking into consideration its many varying contexts, related to the word or phrase in a tweet. To get the related context, we extract the information with paragraph2vec to simplify the process of finding the contextual meaning. The result paragraph2vec will provide the features to help classification in Long Short Term Memory (LSTM). Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. We applied a sarcasm detection method to identify sarcasm in two different languages: English and Indonesian and classification with balanced and imbalanced data. It aims to measure the reliability of the proposed approach and how effective the method is in detecting sarcasm. The result of the experiment shows that in Indonesian, balanced data has a good accuracy of 88.33 % and imbalanced data of 76.66 %, whereas in English the balanced data has an accuracy of 79% and imbalanced data of 54.5%.

CCS CONCEPTS

• Computing methodologies • Artificial intelligence • Natural language processing

KEYWORDS

Sarcasm detection, lstm, paragraph2vec, context, deep learning

ACM Reference format:
Siti Khotijah, Jimmy Tirtawangsa and Arie A Suryani. 2020. Using LSTM for Context Based Approach of Sarcasm Detection in Twitter. In Proceedings of International Conference on Advances in Information Technology (IAIT2020), July 1-3, 2020, Bangkok, Thailand. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3406601.3406624

IAIT2020, July 1–3, 2020, Bangkok, Thailand
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7759-1/20/07…$15.00
https://doi.org/10.1145/3406601.3406624

1 Introduction

Social media is a place to find new friendships and to openly express opinions. One of the most used social media today is Twitter. Twitter allows users to write and read messages commonly called tweets. Millions of tweets are written by more than 288 million active users every day [1]. Twitter is used by several political figures to campaign and increase its following by 61% across Social Media platforms such as Twitter popularity ahead of general elections.

In addition, it is common to find people using sarcasm in their opinions. Sarcasm is a word that has the opposite meaning of what is said that is used to mock or show resentment [2]. The difficulty is to analyze sarcasm automatically[3].

Technically, sarcasm according to D. Wilson [4] is a situational disparity between the text and its context. Sarcasm requires 6-tuples consisting of a speaker, listener, context, and speech, literal propositions and intended propositions[5]. Sarcasm is used as a form of negation in which the negation marker is less explicit, implying that the sarcasm functions as polarity-shifters[6]. Sarcasm is divided into 3 categories: sarcasm as wit, sarcasm as whimper and sarcasm as evasion. Sarcasm as wit is used with the purpose of being funny. Sarcasm as a whimper is sarcasm used to show how annoyed or angry the person is. As for sarcasm as evasion, it refers to situations when people want to avoid giving clear answers[1].

Sarcasm detection is often challenging for chatbots and even more so for the average person. In order for these such technologies to identify sarcasm, they need to understand contextual clues. On top of that, they must do so with very limited nonverbal indications (e.g like vocal tonality and body language play crucial roles) in determining sarcasm [20].

The definition in this research inspired by [2] Sarcasm is a word that has the opposite meaning of what is said that is used to mock or show resentment. The opposite meaning in this research is when positive sentiments are followed by negative sentiments.

For example, consider the tweets below:

a. I love being ignore #sarcasm

b. Prestasi Anies Menginternasional. pengen tau kan? Neh-- Banjir Jakarta jadi sorotan dunia. KwAk kwaK KwAk -- #4niesBebalSokHebat

   "*Anies International achievements. want to know? This - Jakarta floods have become the world's spotlight. KwAk kwaK KwAk #StupidButActAsAGreatMan*"

From the examples above, the positive sentiment is underlined, and the negative sentiment is highlighted in pink. The sarcasm in this research arises when we take into consideration the juxtaposition of positive words with negative words, which should have opposite meaning.

In this paper, we propose an efficient way to detect sarcasm on twitter with the use of Long Short Term Memory (LSTM). The main contributions of this paper are as follows:

- We propose a method to solve sequence to sequence text classification problems for which sarcasm is detected.
- We propose an effective method to detect sarcasm with context representation using word sequences.
- We propose the method to recognize sarcasm in two different languages: English and Indonesian.

2 Related Work

Sarcasm is a sophisticated form of irony [1]. In the fields of text mining specifically: sentiment analysis, detecting sarcasm automatically is still considered a difficult problem since lexical features do not provide enough information to detect sarcasm [7]. Several approaches from previous research on sarcasm, conducted by Lunando and Purwarianti [2] proposed two features namely the negativity information and the number of interjection words. The use of negativity information represents the percentage of the negative sentiment in the topic of the text message. In order to obtain this feature, the topic of the text message should be extracted first. They did the topic extraction manually, while the number of interjection words from the text message fed them to an SVM for the final classification. The experiment resulted in a 54.1 % accuracy. Bouazizi and Ohtsuki [1] proposed the four-feature extraction: sentiment, punctuation, semantic, and pattern related features. Extraction of the features is used to build a classification model using SVM, Random Forest, KNN, and Maximum Entropy. The experiment resulted in a Random Forest with a higher accuracy compared to others at 83.1%. In their research, Sounjaya poria, [8] used Word2vec as embeddings which was trained on 100 billion words from Google News. The vectors are dimension 300, trained using the continuous bag-of-words architecture. Two kinds of experiments were conducted during the research: first, they used CNN for the classification; second, they extracted features from the fully-connected layer of CNN and fed them to an SVM for the final classification. The experiment resulted in a 90.70% F1-score. Devamanyu Hazarika [10] proposed a hybrid network, named CASCADE, which utilizes both content and contextual information. CASCADE first learns user embeddings and discourse features of all users and discussion forums; respectively following this phase CASCADE then retrieves the learnt user embedding $\bar{u}_i$ of user u_i and discourse feature vector $\bar{t}_j$ of forum t_i. Finally, all three vectors $\bar{C}_{i,j}$, $\bar{u}_i$, and $\bar{t}_j$ are concatenated and used for the classification CNN. The experiment resulted in a 77 % F1-score of balanced data.

3 Proposed Research

Given a set of tweets, we aim to classify whether or not the tweets contain sarcasm. We expect to solve this problem with Paragraph2vec as a context feature and Long Short Term Memory as task classification. We applied several preprocesses, which are normalization, lower case, stopwords removal, a reverse word in each sentence order, and padding & truncating sequence. Based on the preprocesses above, we are then able to extract the information context with paragraph2vec and focus on the contextual meaning. The result paragraph2vec are used as features to help classification in Long Short Term Memory (LSTM). The Diagram process is shown in Figure 1.

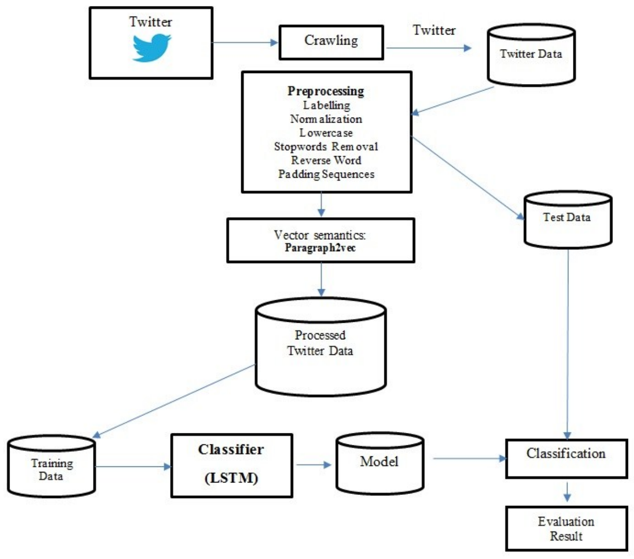


Figure 1: Diagram Process

Explanation of the above Diagram Process is as follows:

3.1 Data

For the Indonesian dataset, we have collected data using Twitter API within a period starting from March 2013 to February 2020. In total we have collected more than 1000000 tweets with various hash tags (``#sarcasm, #banjir *"#flood"*, #bodoh *"#stupid*" ,

#anies, etc.). After we cleaned up the collection by removing noisy and irrelevant tweets, we have relevant 17838 tweets. Among these tweets, there are 4350 sarcastic tweets. The data was verified by an expert. For the second dataset, the English dataset was collected by Erol Özkan [11] After performing similar noisy and irrelevant tweets removal process, we have 4554 tweets. From these tweets we identified 1000 sarcastic tweets.

3.2 Prepocessing

This section consists of:

3.2.1 Labelling . In order to build classifiers, we need labeled data, which consists of documents and the corresponding categories or tags or labels (e.g sarcasm or not). Manual annotation is conducted in the consideration of the rules of positive sentiments, followed by negative sentiments. The process was performed by five people without any background on the tweets or the users, who posted them, and ambiguous data labeling was resolved by a language expert. The data validation is performed by five people, and we took majority voting when there are disagreement among them.

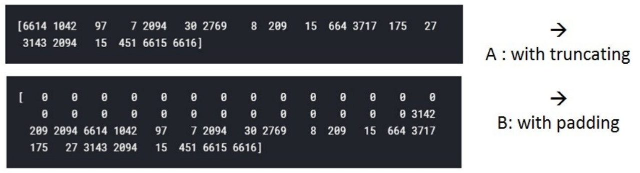
3.2.2 Normalization . Normalization is a process that converts a list of words to a more uniform sequence by transforming the words into a standard format (e.g loe, kmu normalise kamu *"you"*). In the process, we used two corpora, in which the first contains a normal word, and the second some slang words. Every word in the slang corpus will be replaced with the normal corpus. This process helps reduce words with the same meaning.

3.2.3 Lowercase . This is the simplest preprocessing technique which serves to convert all uppercase characters to lowercase forms. (e.g prestasi anies menginternasional. pengen tau kan? neh-- banjir jakarta jadi sorotan dunia. kwak kwak kwak -- #4niesbebalsokhebat "*Anies International achievements. want to know? This - Jakarta floods have become the world's spotlight. KwAk kwaK KwAk* #StupidButActAsAGreatMan*")*. We applied a rule in which every input process must do lower case first. Despite its desirable property of reducing sparsity and vocabulary size, lowercasing may negatively impact a system's performance by increasing ambiguity[16].

3.2.4 Stop Words Removal . Stop words are words which are filtered out before or after processing of natural language data (text) [16]. Stop word removal is applied by using two corpora, dataset corpus, and stopwords corpus. Every sentence in the dataset corpus which has a stop word will be removed based on stopword corpus and replacement by (e.g Cerdaaas... Beruntunglah bagi mereka yg mengenal tanda²...Tanda ketidakbecusan @aniesbaswedan Memimpin jakarta. #AzabGabenerBodong *"Smart ... Luckily for those who know the signs ... Signs of incompetence @aniesbaswedan Lead Jakarta. #Governor'sDoom"* stop word "*Cerdaaas… Beruntunglah mereka mengenal tanda²...Tanda ketidakbecusan @aniesbaswedan Memimpin jakarta. #AzabGabenerBodong*" . Examples of Stop Words that must be removed are "*itu, juga, bagi, yang, pada, selalu*". Stop word is applied because most common words and do not have contextual meaning which affects the process of the readers' understanding.

3.2.5 Reverse Word in Each Sentence Order . We assume that truncation is done in pre-sequence because we use max length to denote the maximum length of the sentence in the training set; we cut extra words at the end of these sentences to reach max length. The input sequence is fed to the decoder in reverse (e.g welcome to Indonesia virus saja disambut apalagi dajjal *"welcome to Indonesia, not only Dajjal, even virus is welcomed"* reverse dajjal apalagi disambut saja virus Indonesia to welcome "*welcomed is virus even Dajjal, only not Indonesia, to welcome"*. The reverse process was done by applying the reverse transformation rules for every word in a sentence. Based on the preliminary study of training, the use of reverse word has a better end result than if the sequences were in normal order. If we reverse the input sequence, the average distance between the input/ output words will not change, but the first input words will be very similar to the first output words.

3.2.6 Padding and Truncating Sequences . Padding and Paragraph Vector Initialization First, we use maxlen to denote the maximum length of the sentence in the training set. As the LSTM layer in our model requires fixed max sequence length input, we pad each sentence that has a length less than maxlen with 0 at the beginning (pre-sequence) that indicates the unknown words. For a sentence in the test dataset, we pad sentences that are shorter than maxlen in the same way, but for sentences that have a length longer than maxlen, we simply cut extra words in the beginning of these sentences to reach maxlen. The Padding & Truncating Sequences is shown in figure. 2.



```
[6614 1042   97    7 2094   30 2769    8  209   15  664 3717  175   27
 3143 2094   15  451 6615 6616]
```

```
[   0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0 3142
  209 2094 6614 1042   97    7 2094   30 2769    8  209   15  664 3717
  175   27 3143 2094   15  451 6615 6616]
```

Figure 2: Padding and Truncating Sequences

In the figure. 2. above we show an example of both padding & truncating sequences. In illustration A, it shows that a sequence with truncatin, set at maxlen: 20, determines that sentences with words exceeding maxlen need to be omitted. While B presents an example with padding, set at maxlen: 50, shows that if the input is less than maxlen, we pad with 0.

3.3 Vector Semantic : Paragraph2vec or Doc2vec

Vector semantics is a technique used for a model that deals with the different aspects of word meaning (word senses, word similarity, and relatedness, lexical fields and frames, connotation). Vector semantics represent words in a multidimensional vector space. The vector models are called embedding [12]. We use Paragraph2vec or Doc2vec as the embedding used for extracting information context. The Paragraph Vector is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts such as sentences, paragraphs, and documents [13]. Paragraph2vec has two models that are a distributed memory model (PV-DM) and a distributed bag of words (PV-DBOW).

In this research, we use the model PV-DM to recognize the context in tweets. The context in this research is related to the word/phrase in the tweets. Vector semantics is used to detect whether or not the tweets contain sarcasm. The vector semantics is shown in figure. 3.

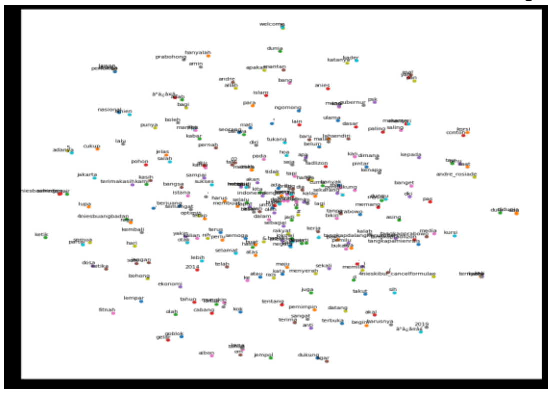


Figure 3: Vector Semantics

In figure.3 . above Paragraph2vec is based on the distributional hypothesis that the meaning of a word can be gauged by its context. In this research, every paragraph and every word in the paragraph are mapped to unique vectors. The paragraph vector and word vectors are averaged, and we concatenate both vectors. The result of the embedding used as features to create classification models using Long Short Term Memory (LSTM).

3.4 Long Short Term Memory

Long Short Term Memory (LSTM) is a special kind of Recurrent Neural Network (RNN), capable of learning long-term dependencies. These long-term dependencies have a great influence on the meaning and overall polarity of a document. Long short-term memory networks (LSTM) address this long-term dependency problem by introducing a memory into the network. It was first introduced by Hochreiter & Schmidhuber [14]. The LSTM architecture has a range of repeated modules for each time step as in a standard RNN. At each time step, the output of the module is controlled by a set of gates, as a function of the old hidden state h_{t-1} and the input at the current time step x_t : the forget gate f_t, the input gate i_t , and the output gate O_t . These gates collectively decide how to update the current memory cell C_t and the current hidden state h_t . The LSTM transition functions are defined as follows:

$$
\begin{aligned}
i_t &= (W_i[h_{t-1}, x_t] + b_i) \\
\acute{C}_t &= tanh(W_c[h_{t-1}, x_t] + b_C) \qquad (1) \\
f_t &= \left(W_f[h_{t-1}, x_t] + b_f\right) \\
O_t &= (W_o[h_{t-1}, x_t] + b_o) \\
C_t &= f_t * C_{t-1} + i_t * \acute{C}_t
\end{aligned}
$$

Here σ is logistic sigmoid function that has an output in $n\ [0,1]$,tanh denotes the hyperbolic tangent function that has an output $h\ in\ [-1,1]$, and $*$ denotes the pointwise multiplication, as proved by [14].

In this research we are able to establish the contexts in tweets through the input gate, forget gate and output gate. Where the input gate decides what information is relevant to add from the current step, the forget gate decides what is relevant to keep from prior steps, and the output gate determines what the next hidden state should be.

Therefore, we decided to use Long Short Term Memory (LSTM) for task classification, considering that sarcasm detection is a sequence text problem. To understand whether the tweet is sarcastic or not we have to pay attention to the related context. In the classic approach for sarcasm detection, a common approach for document classification might include take a label set of text a corpus, tokenize take to create a bag of the word, convert the bag of the word into TF-IDF, train classification on matrix, such as Naive Bayes, SVM, Random Forest, etc. It produced good results, but its context is lost. The approach is also very sensitive to misspellings. The context is quite important because generally a language is often ambiguous and the exact meaning of each word has to be determined by the context where it is used. The context of a word is defined as vector of words often used before and after that word. To get such a vector we extract information using paragraph2vec. Then it will be used as features to help classification process in the LSTM.

3.4.1 LSTM Architecture . The architecture of the LSTM model is shown in figure. 4. The model consists of two LSTM layers.

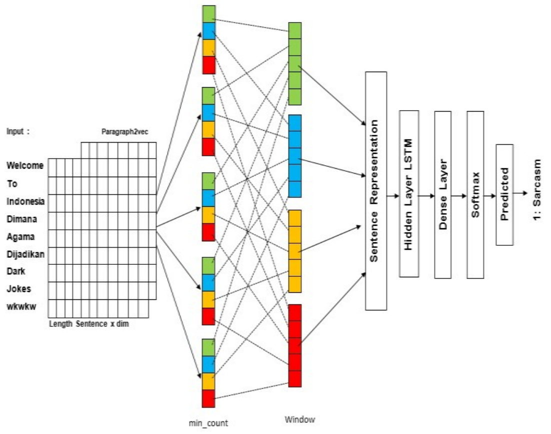


Figure 4: LSTM Architecture

The illustration above represents the process of how to feed input sentences to the decoder sentences in Paragraph2vec. In the first step, as the time when the input sequence is fed to the decoder, we set parameters in Paragraph2vec such as min_count, size of embedding, and window size. After the training process in Paragraph2vec, we obtain sentence representation such as word vector and information context, then we initialize the number of hidden LSTM layers and we initialize dense layers tailored with classes that are in the dataset then apply activation function softmax. We are using softmax because we want to convert out last layer output values into probability values as sarcasm and non-sarcasm. Finally, the result is a predicted class (e.g sarcasm).

3.5 Experimental Setting

In this research, we set several parameters for the optimization method. The parameters are defined as follows:

1. Dataset
   In Indonesian, We prepared 3 data sets for our work as follows:
   Set 1: this set contains 8700 tweets, half of which are sarcastic, and the other half not. It is used for the implementation of balanced data.
   Set 2: this set contains 17.718 data, in which 4.350 are sarcastic and the others not. It is
   used for the implementation of imbalanced data.
   Set 3: this contains 120 tweets, in which 56 tweets are sarcastic and the rest not. The Set 3 is used for new testing data which served as an outside model. This data is fresh data and is not related to either data set 1 or data set 2. We tokenized the words, reversed the word order, and predicted theirs classes.

   In English, We prepared 3 data sets for our work as follows:
   Set 1: this set contains 1800 tweets, half of which are sarcastic, and the other half not. It is used for the implementation of balanced data. While
   Set 2: this set contains 4.554 data, in which 900 are sarcastic and the remaining not. It is used for the implementation of imbalanced data.
   Set 3: the last set contains 200 tweets, in which 100 are sarcastic and the other are not. It is used for testing.

   To reduce overfitting effect, we split 90:10 both Indonesian and English data set (set 1 and set 2). As the original data is imbalance between the number of sarcastic tweets and non-sarcastic tweets, we also tried to train the model using a more balance data set by taking a data subset containing a more balance number of sarcastic and non-sarcastic tweets.
2. Maxlen: 30, 36, 50
   Maxlen is the maximum length of all sequences. In the preliminary study, the optimum maxlen is in the range of 30,36,50.
3. Embedding dim: 20
   We have a 17.718 dataset with the number of vocabulary around 15.000; we represent each one using 20 embeddings dimension in a vector. We assume with 20 embedding dimensions the result will be optimal because we want to distinguish the local context and not the global context within Paragraph2vec.
4. Min_count:1
   Min_count is the word's frequency count in the corpus/ dataset. We set min_count: 1 because we used model DM=1 in Paragraph2vec, therefore preserving word order.
5. Window: 8
   The window is the size of the sampling window; we set the window size at 8. The aim is to find a local context related to the word in the sentence context. Technically the window is computed:
   $$w - i \quad will\ be \quad [i - windowsize + windowsize + 1]$$
6. Hidden Layer LSTM : 50
   In the preliminary study, we experiment to use several sizes of hidden layers LSTM, but we have optimum results when we apply 50 hidden layers. Thus, the reason why we use it.
7. Batch size : 32 (default)
   Batch size is the number of a sentence per gradient update; we set it with a default value of 32.
8. Number of epochs : 20 – 1000 epochs
   In the preliminary study training with 20 epochs with three maxlens, the result is stable enough, therefore we decided to experiment with an epoch that had a greater epoch with the aims of achieving convergence.

The parameter chosen is small since big parameters are not optimal and the system will fail to attain context sarcasm detection.

4 Experimental Result

When the classification process is completed. We evaluate the results using Confusion Matrix. A confusion matrix is a summary of prediction results on a classification problem [19]. It aims to measure the reliability of the proposed approach and how effective the proposed method is to detect sarcasm. The formula is defined as follows:

Accuracy: $\frac{TP}{TP+TN+FP+FN}$

Precision: $\frac{TP}{TP+FP}$

Recall: $\frac{TP}{TP+FN}$

F1. Score: $2\frac{Precision\ x\ Recall}{Precision+Recall}$

TP (True Positive) , FN (False Negative) , TN (True Negative) , FP (False Positive). We run the classification using LSTM with the result as shown in table 1 and table 2 below:

4.1 Experiments on Indonesian Sarcasm Dataset

Below is the result from the first experiment in which we evaluated sarcastic tweets in Indonesian:

Table 1. Experimental results of Sarcasm Indonesian Data

Dataset	Training data (90%)	Validation data (10%)				New and different test dataset			
		Accuracy	Precision	Recall	F1. Score	Accuracy	Precision	Recall	F1. Score
Balanced	100%	95.40%	95.63%	95.19%	95.40%	88.33%	83.92 %	90.38 %	87.03 %
Imbalanced	100%	96.78%	98.52%	97.31%	97.64%	76.66%	53.57 %	93.75%	68.18%

We are able to evaluate the result of balanced and imbalanced data (New and different test dataset) having significant differences. In this experiment, there were better results for the balanced data rather than imbalanced data. The reason is because imbalance due to rare instances is representative of domains where the minority class examples are very limited,i.e. where the target concept is rare.

In this situation, the lack of representative data will make learning difficult regardless of the between-class imbalance [18].

In both experiments in balanced and imbalanced data, we have a high recall and low precision: This indicates that most of the sarcasm is correctly recognized (low FN).

4.2 Experiments on English Sarcasm Dataset

The second experiment we evaluated sarcasm tweets in English; the result is shown in the table below:

Table 2. Experimental results of Sarcasm English Data

Dataset	Training data (90%)	Validation data (10%)				New and different test dataset			
		Accuracy	Precision	Recall	F1. Score	Accuracy	Precision	Recall	F1. Score
Balanced	99.81%	98.48%	98.07%	99.02%	98.04%	79%	53.93 %	97.95 %	69.56 %
Imbalanced	99.85%	94.33%	96.50%	93.05%	94.74%	54.5%	40.48 %	96.73%	46.43 %

We are able to evaluate the result of balanced and imbalanced data (New and different test dataset) having significant differences. In this experiment, there were better results for the balanced data rather than imbalanced data. The reason is because imbalance due to rare instances is representative of domains where the minority class examples are very limited,i.e. where the target concept is rare. In this situation, the lack of representative data will make learning difficult regardless of the between-class imbalance [18].

In both experiments in balanced and imbalanced data, we have a high recall and low precision: This indicates that most of the sarcasm is correctly recognized (low FN).

4.3 Experiments During Cross Validation

The third experiment we evaluated sarcasm tweets in Indonesian and English; the result is shown in the table below:

Table 3. Experimental results of cross-validation

Dataset		Accuracy Mean cv=10	New and different test dataset			
			Accuracy	Precision	Recall	F1. Score
Balanced	Indonesian	94.38%	78.33%	75%	77.77%	76.36%
	English	97.62%	66%	33%	97.05%	49.25%
Imbalanced	Indonesian	96.31%	70%	42.85%	85.71%	57.14%
	English	93.89%	63%	27%	96.42%	42.18%

In the third experiment, we tried to cross-validate with cv=10. From the experiment result, we calculated the mean of accuracy and we used model cross-validation for classification test data which served outside the model. However, we can notice that the cross-validation it changes the value of accuracy, precision, recall, and F1. Score. This can be shown in the table above the value turns down.

4.4 The Effect of Padding

We experiment with three maxlen as shown in figure .5. The pad_sequence function is used to pad sequences to a preferred length that may be longer than any observed sequence. This is accomplished by specifying the "maxlen" argument to the desired length. Padding will then be performed on all sequences to achieve the desired length

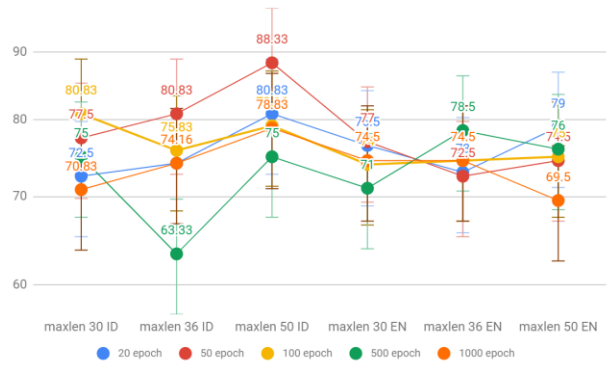


Figure 5: Comparison of experimental results using different maxlen

In figure. 5. above, we compare the accuracy-test which served outside the model of each maxlen between Indonesian (ID) and English (EN) datasets which train in balanced data. The outcome is that maxlen increases the level of accuracy towards the test result. Based on both experiments, the dataset was optimal when using maxlen: 50. Regarding linguistics differences between Indonesia and English, Indonesian sarcasm seems mostly satire while English sarcasm is more to the point type of sarcasm, and this might give difference experiment result.

We compared a more realistic test accuracy of the model using new and different data set aside from the data to build the model. The test from 90:10 split data should give as only the quality of the model.

5 Conclusion

In this research, we solve sarcasm detection with Paragraph2vec as a context feature and LSTM as task classification. The approach accuracy is at least 88.33% for the Indonesian Dataset and 79% for the English Dataset. We have shown that the effect of additional padding sequences for detecting sarcasm can increase the accuracy average of at least 7 %. But the system is insufficient for recognizing English tweets because the sarcasm is shorter than in Indonesian. In the end, there is still much room for improvement, particularly when considering the connection between context and lexicon. Context helps to find contextual meaning while the lexicon shows emotional sentiment. Both factors are of great interest for future study in sarcasm detection.

ACKNOWLEDGMENTS

We thank all respondents who have helped label and verify data in this research. We would also like to express gratitude to all reviewers for their detailed and sharp comments and suggestions to improve the quality of the paper.

REFERENCES

[1] Mondher Bouazizi and Tomoaki Otsuki, "A Pattern-Based Approach for Sarcasm Detection on Twitter," IEEE Access Volume 4, 2016. pp. 5477-5488.

[2] Lunando, E. and Purwarianti, A. Indonesian social media sentiment analysis with sarcasm detection. In 2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS), 2013.

[3] D. Maynard and M. A. Greenwood, ``Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis," in Proc. 9th Int. Conf. Lang. Resour. Eval., May 2014, pp. 4238_4243 .

[4] D. Wilson, ``The pragmatics of verbal irony: Echo or pretence?" *Lingua*,vol. 116, no. 10, pp. 1722_1743, Oct. 2006.

[5] S. L. Ivanko and P. M. Pexman, ``Context incongruity and irony process-ing," *Discourse Process.*, vol. 35, no. 3, pp. 241_279, 2003.

[6] R. Giora, ``On irony and negation," *Discourse Process.*, vol. 19, no. 2,pp. 239_264, 1995.

[7] Gibbs, R. W. and Colston, H. L. 2007. Irony in Language and Thought. Routledge (Taylor and Francis), New York.

[8] S Poria, E Cambria, D Hazarika, P Vij. "*A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks*". 2017.

[9] Filatova, Elena. "Sarcasm Detection Using Sentiment Flow Shifts." Copyright c 2017 Association for the Advancement of Artificial Intelligence.

[10] D Hazarika, S Poria, S Gorontla, E Cambria, R Zimmermann, R Mihalcea. "CASCADE: *Contextual Sarcasm Detection in Online Discussion Forums*" source: arXiv:1805.06413v1 [cs.CL] 16 May 2018.

[11] Erol Özkan. SARCASM DETECTION IN TWITTER. Yaaay! it's a holiday weekend and I have a sarcasm detection project to complete! couldn't be more thrilled!.Department of Computer Engineering, Hacettepe University.https://github.com/ErolOZKAN-/NaturalLanguageProcessing-SarcasmDetection

[12] Jurafsky, Daniel, & Martin , James. (2015). Vector Semantics. In Daniel Jurafsky & James Martin (Eds.), Speech and Language Processing (3rd edition draft).

[13] Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on Machine Learning.

[14] Hochreiter, S., & Schmidhuber, J. (1997). *Long short-term memory* (Neural Computation 9(8):1735{1780, 1997). Fakultät für Informatik, Technische Universität München, Licence details :https://www.bioinf.jku.at/publications/older/2604.pdf

[15] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In Proceedings of EMNLP, pages 1532–1543.

[16] Rajaraman, A.; Ullman, J. D. (2011). "Data Mining". *Mining of Massive Datasets* (PDF). pp. 1–17. doi:10.1017/CBO9781139058452.002. ISBN 9781139058452.

[17] He, H. and Garcia, E.A. (2009) Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering, 21, 1263-1284. http://dx.doi.org/10.1109/TKDE.2008.239

[18] G.M. Weiss, "Mining with Rarity: A Unifying Framework," ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, pp. 7-19, 2004.

[19]Confusion Matrix in Machine Learning. https://www.geeksforgeeks.org/confusion-matrix-machine-learning, accessed march 15 2020.

[20] Hunter Heidenreich, CS Undergrad at Drexel University, interested in AI and ethics http://hunterheidenreich.com/blog/nlp-and-sarcasm/ , Acces 23 March 2020.