Received March 27, 2021, accepted April 15, 2021, date of publication April 30, 2021, date of current version May 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3076789

Sarcasm Detection Using Deep Learning With Contextual Features

MD SAIFULLAH RAZALI[1,2], **ALFIAN ABDUL HALIN**[1], **LEI YE**[2], **SHYAMALA DORAISAMY**[1], **(Member, IEEE), AND NORIS MOHD NOROWI**[1]

[1]Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Seri Kembangan 43400, Malaysia
[2]Faculty of Engineering and Information Sciences, University of Wollongong, Wollongong, NSW 2522, Australia

Corresponding author: Alfian Abdul Halin (alfian@upm.edu.my)

This work was supported by Universiti Putra Malaysia through the Geran Putra Inisiatif Siswazah (Research Title: Sarcasm Detection Using Machine Learning to Enhance Sentiment Analysis).

ABSTRACT Our work focuses on detecting sarcasm in tweets using deep learning extracted features combined with contextual handcrafted features. A feature set is extracted from a Convolutional Neural Network (CNN) architecture before it is combined with carefully handcrafted feature sets. These handcrafted feature sets are created based on their respective contextual explanations. Each feature sets are specifically designed for the sole task of sarcasm detection. The objective is to find the most optimal features. Some sets are good to go even when it is used in independence. Other sets are not really significant without any combination. The results of the experiments are positive in terms of Accuracy, Precision, Recall and F1-measure. The combination of features are classified using a few machine learning techniques for comparison purposes. Logistic Regression is found to be the best classification algorithm for this task. Furthermore, result comparison to recent works and the performance of each feature set are also shown as additional information.

INDEX TERMS Sarcasm detection, natural language processing, deep learning.

I. INTRODUCTION

The magnitude of data generated through social media today is colossal. They are good for data analysis since they are very personal [1]. For years, companies have been analyzing this type of data to leverage their position in the market of their choice [2]. This field is called sentiment analysis [3].

On the other hand, sarcasm is defined as a positive utterance or sentence with underlying negative intention [4]. It is regarded as one of the most challenging issues in the Natural Language Processing (NLP) field [5]. Spotting and handling them correctly is crucial in an automated NLP systems, mainly since sarcasm can flip the polarity of a sentence [5], [6]. Traditional studies as from Davidov *et al.* [7] and Riloff *et al.* [4] used rule-based techniques to tackle sarcasm detection. However, more recent studies [8], [9] have shifted towards deep learning to automatically detect the discriminatory features.

In this work, the features extracted by a deep learning architecture is combined with the ones that are manually created through specific contextual understanding and processes. Tweets are used as the main source of input. Unlike in writing, different tones and gestures can be utilized to portray sarcasm in the real world [7]. As a counter-measure for this short-coming, writers of tweets tend to leave contextual clues for sarcasm in creative ways such as hashtags and hyperboles [4], [7]. This kind of clues is what this work is trying to find and exploit.

II. MOTIVATION

Several NLP studies have tried to come up with automatic detection models for sarcasm. Features are either discovered through deep learning or manual handcrafting (feature engineering) methods [10], never both. There is too much reliance on a deep learning architecture for some researchers [8], [9], and vice-versa for manual handcrafting [4], [11], [12]. This leave some room for experiments.

III. RELATED WORK

In sarcasm detection datasets, sarcastic tweets commonly contain hashtag keywords such as #sarcasm, #sarcastic, #not [4], [7], [12]. This group of studies believe that hashtags are the best indicators to initially detect sarcasm.

Recently, deep learning are used for sarcasm detection [8], [9]. This is following CNN's good track record in

The associate editor coordinating the review of this manuscript and approving it for publication was Shadi Alawneh.

solving NLP problems. For example, Poria *et al.* [8] used four datasets to extract four feature sets using CNN. Then these feature sets are combined and classified by an SVM classifier. Another work from Ilić *et al.* [13] used a deep learning model based on character-level word representations derived from the Embeddings from Language Models (ELMo). ELMo is a representation technique which use vectors derived from a bidirectional Long Short Term Memory (LSTM) [14]. Both of the aforementioned works used a dataset that is created by using hashtags [15]. This is also the dataset that is used in our work.

Besides sole reliance on hashtag keywords, some studies also add rule-sets. For example, Barbieri *et al.* [16] use frequency and rarity of words as their main features. This technique is also used by Bouazizi and Ohtsuki [17] with additional rules on extracting sarcastic word patterns. The rules include counting the number of positive/negative words in the tweet and counting the number of highly emotional positive/highly emotional negative words. A more recent study by Shmueli *et al.* [18] used a seed phrase ''being sarcastic'' as in ''I was being sarcastic'' or ''She was being sarcastic''. The seed is then used to collect sarcastic instances in Twitter. Ultimately, a new dataset is created to help solve the issue of dataset scarcity for sarcasm detection. These are not the only researches that use rule-based techniques for sarcasm detection. Riloff *et al.* [4] had already used classifiers that looks for positive verbs found with negative situations in a sentence in 2013. They created a lexicon for the verbs and situational words. Then they used this lexicon to differentiate between the sarcastic and normal sentences in their test. This research has inspired other researches that use the same methodology of using hashtags assisted detection augmented by rule-sets [8], [12], [19].

Apart from hashtag and rules related detection methods, other studies depended on user historical tweets to create their features. This idea was first experimented by [12] with thorough analysis on what are the most relevant features in a thread of tweets from the same user. The features is as shown in Figure 1. It is obvious that the highest accuracy can only be achieved when all of the relevant features are combined. All these relevant features has their own specific rules. For example, audience feature use historical communication between author of the tweet and the person that tweet is intended to using the @ function in Twitter. This idea is inspired by the work by Kreuz and Caucci [20] which states sarcasm is likely to happen between the people that knows each other. Then the rank, the frequency of messages and if there have been at least one mutual @-message between the author and this other user are also added as part of the audience feature. This technique is followed by a few other studies [21]–[23]. For example, Rajadesingan *et al.* [23] created a framework called Sarcasm Classification Using a Behavioral modeling Approach (SCUBA) where they classify user's behaviour using a similar approach as done by Bamman and Smith [12] (using historical tweets) but added aspects like the difference in the length of the words between the user's current tweet and their past tweets. They have also experimented on whether there is a direct correlation between the availability of historical information with sarcasm detection performance, as shown in Figure 2.

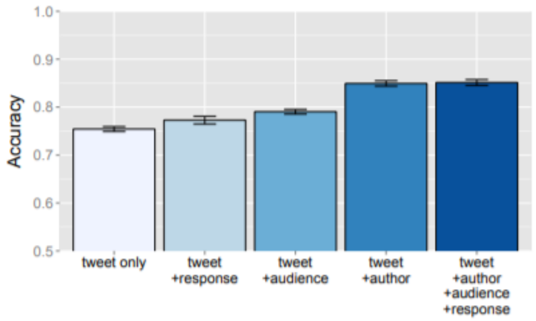

FIGURE 1. Most relevant features for sarcasm detection according to Bamman & Smith [12].

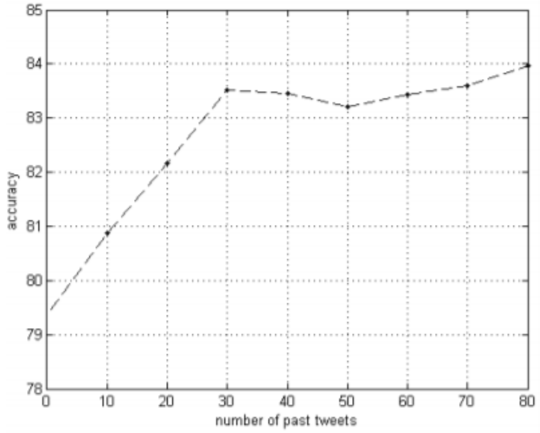

FIGURE 2. Direct correlation between availability of historical information and sarcasm detection performance according to Rajadesingan *et al.* [23].

A recent study [24] used a deep learning Bidirectional Encoder Representations from Transformers (BERT) architecture to experiment on the idea of using historical information to detect sarcasm. They used historical conversational features such as response, last utterance, last 2 utterance and last 3 utterance.

Another sarcasm detection study focused on the contextual difference in specific tweets [25]. This study yielded a better result than the studies using historical information mentioned above. Additionally, all the other context-based sarcasm detection studies [4], [8], [12] somehow incorporated historical information in their experiments. In our study, only information given in the dataset is used.

Linguistic markers such as exaggeration or hyperboles [26], [27], interjections such as ''gee'' or ''gosh'' and punctuation symbols such as ''?'' [20] could also be good features for sarcasm detection. Tsur *et al.* [28] even claims that modified words like ''yay!'' or ''nay!'' as a recurring aspect of sarcastic patterns in Amazon product reviews. These has all been done in rule-based systems.

The work by Liebrecht *et al.* [29] used "#not" to label their tweets as sarcastic or not. For example the tweet "Donald Trump is the best president ever #not" would be decided as sarcastic. They also used bigrams and trigrams to determine the rank of the features. For example, in their dataset the term "Nineteen Eighty" is found 836 times while the term "One hundred" is found 636 times which makes it lower in ranking for bigrams. "This gave another indication that N-Grams can be expanded into many types of grams to achieve an objective.

Another prominent way to detect sarcasm is the use of lexicons [4], [20]. Many studies have used lexicons to assist their sarcasm detection methods. For example, Riloff *et al.* [4] created their own lexicons of positive verbs and negative situations using a boot-strapping technique. Then they use back these lexicons for the main sarcasm detection task. Existing lexicons such as Wordnet [30] has also been used to assist in different sarcasm detection tasks, mainly in the process of word counting [6], [23].

Apart from short-texts such as tweets, existing researchers also work on long-texts such as product reviews and online discussions [31], [32]. Interestingly, features that excelled in these studies are in the form of N-Grams and Part-of-speech N-Grams. This gives a strong support to the researchers working with short texts. N-Grams based techniques could yield a good result if used correctly.

Sarcasm could also be temporal [33] in the sense that a sarcastic sentence could be regarded as ill-intentioned in a year but then good-intentioned in another year. However, many researchers believed that sarcasm are first created to be ill-intentioned unlikeness [4], [34]. If it uses any temporal words, it just means that the user is hoping that the situation changes in the future [33].

IV. PROPOSED METHOD

The overall framework of this study starts with data acquisition and ends with evaluation. The bird's eye view of the whole process is shown in Figure 3.

The feature extraction methods mentioned in the related works section is not enough to detect all the sarcastic tweets that might be present. In the real world, sarcastic utterances tends to utilize abnormal tones [35], [36] or exaggerations [26]. In a written statement, these are translated into certain use of words or symbols, or the way they are written. This work is focusing on the detection of such features.

A. DATA ACQUISITION

The dataset used in this work is created and shared publicly [15]. It consists of real twitter posts which used "#sarcasm" as the indicator to collect sarcastic instances. Normal instances are more likely to happen in the tweets as well as in the real world in comparison to sarcastic ones [15]. Hence, this dataset is an imbalanced distribution of 780,000 English tweets (130,000 sarcastic and 650,000 non-sarcastic). In our work, the datasets were split into 80 percent training and 20 percent testing.

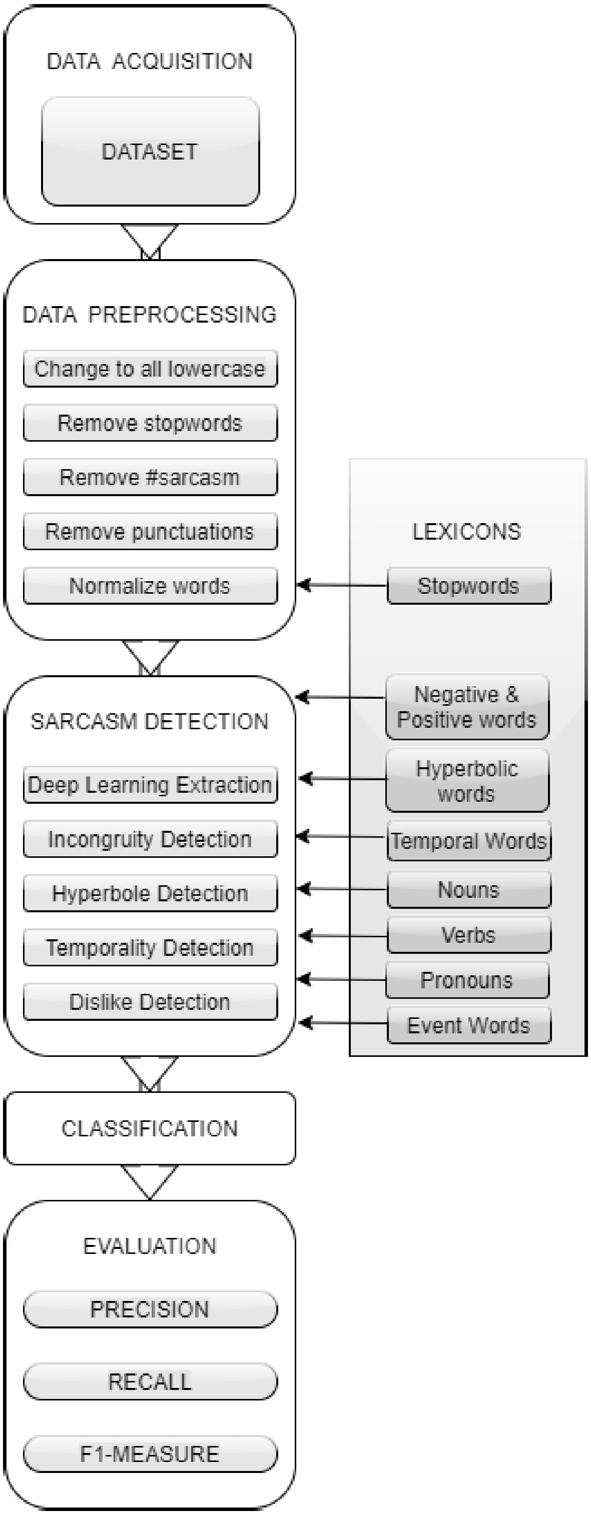

FIGURE 3. Overall framework of the study.

B. DATA PREPROCESSING

Preprocessing is an integral part of any NLP study [22]. It is done so that the preprocessed part would not give any weight and biasness to the experiments. For this purpose, five types of preprocessing techniques which were used.

First all the text in the dataset in converted to lowercase. Then all the stopwords are removed. Then any occurrence of "#sarcasm" in the document is removed. This is followed by the removal of any punctuation signs. Finally, all the words are changed to their root form. The 'lemma' style is chosen for this work as it is more general in contrast to the 'stem' style.

C. SARCASM DETECTION

1) DEEP LEARNING EXTRACTION

Many recent researches on sarcasm detection are moving towards deep learning [8], [9], [22], [37] given its high reputation in NLP. The biggest advantage of using deep learning is its ability to automatically gather optimum features for a given task [8], [9].

In this work, a vanilla Convolutional Neural Network architecture is proposed to extract ten deep features to match the 10 other features described in the next subsections. The features are also balanced so there is no bias when we do the comparison. The overall architecture of the Sarcasm Detector is shown in Figure 4.

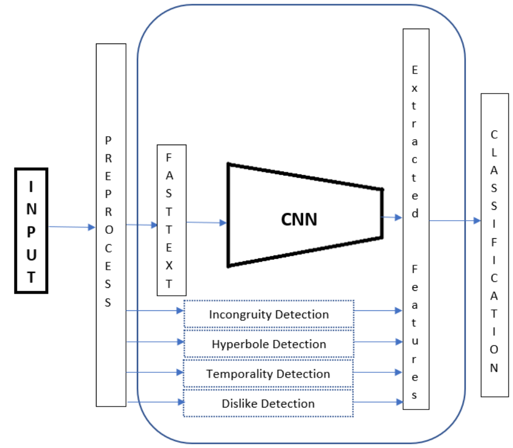

FIGURE 4. Overall architecture of the Sarcasm Detector.

As shown in Figure 4, the deep features extractor (CNN) is a large part of the overall architecture of the Sarcasm Detector. It is also assisted by a word-embedding technique as a way to convert tweet sentences into feature vectors as input. The details are described below.

2) FASTTEXT

We have decided to use FastText [38] as our word embedding technique instead of the commonly used Word2Vec [39]. This is because FastText breaks every word into N-grams instead of using individual words as with Word2Vec. Then the words are fed it into a neural network.

FastText feeds the words into a neural network in the form of unigram, bigram and trigram. For example, the word "human" is broken into "hum", "uma" and "man" as a trigram for the word "human". The vectors for "human" will be the total of all the broken N-grams. Artificial Neural Network (ANN) is used as the training method.

The output is a word-embedding vector for all the broken N-grams in the training dataset. Hence, FastText gives a better representation even for the rare and misspelled words. This makes it very effective for social networks analysis.

3) CONVOLUTIONAL NEURAL NETWORK (CNN)

For the purpose of extracting the deep features, a vanilla CNN architecture as shown in Figure 5 is used. This is the detail of the CNN part from the overall architecture of the Sarcasm Detector shown in Figure 4.

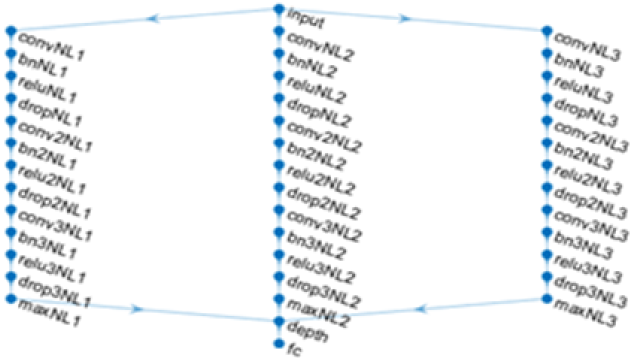

FIGURE 5. CNN architecture used in the Sarcasm Detector.

The CNN architecture in Figure 5 is shown in a top-down manner starting from the start (top) to the finish (bottom) node. "NL" stands for N-gram Length. The breakdown is:

1) An input layer of size 1 × 100 × N where N is the number of instances from the dataset. Vectors of embedded-words are used as the initial input.
2) Then the layers between the input and the concatenation is introduced:
    - One convolutional layer with 200 neurons to receive and filter size 1 × 100 × N where N is the number of instances from the dataset. The stride is [1 1].
    - Two convolutional layer with 200 neurons to receive and filter size 1 × 100 × 200. The stride is [1 1].
    - Three batch normalization with 200 channels.
    - Three ReLU activation layers.
    - Three dropout layers with 20 percent dropout.
    - A max pooling layer with stride [1 1].
3) A depth concatenation layer to concatenate all the last max pooling layers.
4) A fully connected layer with ten neurons.

In this architecture, the focus is on the convolutional layers which are used to produce the feature maps. This is followed by the batch normalization layers to improve the speed, performance, and stability by re-centring and re-scaling the input data. The activation functions are used for the scanning of input data. Finally, the minimum 0.2 dropout layers are used to avoid overfitting and increase the validation accuracy, as well as to increase the generalizing power. Then the max pooling layer is used to do the final vote. The initial inputs are the word vectors created using the Fasttext. The vector size is set to [1 100]. These vectors are then split into three groups- first group (N-Gram Length-1 or normally known as unigram), second group (N-Gram Length-2 or normally known as bigram), and third group (N-Gram Length-3 or normally known as trigram).

After the word vectors has been split into their respective N-Gram Length groups, they are fed into three graph architecture which begins after the input layer. The three graph architecture is running concurrently before combined at the concatenation layer. Then the network goes to a fully connected layer with 10 neurons, which are extracted to be our deep features. This feature set is then combined with the manual features before the classification.

D. INCONGRUITY DETECTION

Incongruity is disagreement of a sentence with the context. According to [40], the time it takes for the understanding of a sarcastic sentence is related with the degree of incongruity between the sentence and the context. Campbell and Katz [41] echoed this by stating that it is compulsory for sarcasm to use context incongruity. Ramteke *et al.* [42] has also came up with the same idea in their paper, even though they call it thwarting instead of incongruity. Another study [4] call this idea ‘‘a contrast’’ and came up with their own system to detect it. Hence, subsets of ideas from both studies by Ramteke *et al.* [42]and Riloff *et al.* [4] is used in the attempt to extract some incongruity features.

E. HYPERBOLE DETECTION

Hyperbole has always been a good linguistic marker for sarcasm detection [26]. The two figurative languages sarcasm and hyperbole are related. Bharti *et al.* [27] stated that hyperbole can facilitate sarcasm detection. This is agreed by other studies [26], [43] with the argument that sarcasm-related utterances often used exaggerations or also known as hyperbole. For example, the sentence ‘‘Great! I love Mondays!’’ could be both a hyperbole because it is an exaggerated sentence and also sarcastic, since naturally humans hate Mondays. Hence, some hyperbolic features are extracted to investigate its helpfulness in sarcasm detection.

F. TEMPORALITY DETECTION

Another quality of sarcasm is temporality [33]. For example, the sentence ‘‘You think just like Donald Trump now’’ in 2016 after he won the US election would have a different connotation then in 2020 when he lost the US election.

G. DISLIKE DETECTION

Sarcasm is also usually used to portray a sort of dislike [34] of a person to another person or an event. To model dislike, some features are extracted based on one of the most prominent characteristic of sarcasm: it is seldom happening between strangers [44]. These features are further differentiated into specific groups: i. A person to a familiar person in the same tweet ii. A person to a familiar event in the same tweet.

On-line engagement is also one of the indications of whether a negative feeling is likely to be felt in the real world. Chen and Boves [45] explained this in two simple definitions, which are used as two more feature groups: i. The more there are tweets about something, the more we can reliably lean on it being the public’s negative feeling. ii. Highly active on-line talking points are usually more inclined to be a negative feeling.

H. FEATURE ENGINEERING FOR MANUAL FEATURES

For context incongruity feature set, the frequency of negative word happening after a positive word and vice versa is counted. Then the total number of positive words and negative words is counted as another feature. This is done to every instance of the tweets. For this process, positive and negative word lexicons described in the next subsection is used.

For each tweet, the total number of hyperbolic words is counted. We use hyperbolic words from a lexicon which is described in the next subsection. This lexicon is also used to see if there is any occurrence of two or more hyperbolic words in a tweet. If there is, it is counted that as another feature. This second hyperbolic feature is in Boolean form.

Temporal words in each tweets is counted as another set of features. Furthermore, another feature set is added by counting the cases where the word that comes in the vicinity of three words before or after the temporal word is a noun. This feature is also in Boolean form. For this purpose, we have used the lexicons of temporal words and nouns explained in the subsection below.

Then two dislike features are added. The first one is the presence of a self-pronoun word with another self-pronoun word in the same tweet. The second is the presence of a self-pronoun with an event word such as ‘‘affair’’, ‘‘incident’’ and ‘‘episode’’ in the same tweet. Both of these features are in Boolean form. Then two dislike with engagement features are also added. Every verb that happens more than once in the whole dataset is counted and used as a feature for every tweet that uses the word. Verbs are direct indicators of how someone is doing [4]. The same is also done for event words. For the purpose of extracting these dislike related features, three lexicons which are explain in the subsection below are used.

In total, 10 manual features is extracted. All these features are related to the context explained in the subsection above.

I. LEXICONS

In order to extract all the manual features, some lexicons are used. These lexicons are directly correlated with the manual feature engineering in the subsection above based on its order.

1) STOPWORDS

There are 225 stopwords used in this work. This lexicon is used in the stopwords removal process in this work.

2) POSITIVE AND NEGATIVE WORDS

The positive and negative words lexicon is downloaded from an existing resource [46] that is studying sentiment analysis. It consists of 6800 words.

3) HYPERBOLIC WORDS

This dataset is a subset from that of another study [47] that works on hyperbole. It consists of 710 instances. This lexicon is used in the hyperbole-related features extraction process.

4) TEMPORAL WORDS

Temporal or transition words are words that deals with time. We downloaded a list of these words from a prominent website providing it [48]. It consists of 52 instances. This lexicon is used in the process of extracting the temporal-related features.

5) NOUNS

This lexicon is downloaded from a website that has the most comprehensive list of nouns [49]. This list contains 1500 instances. This lexicon is used in the process of extracting the features for temporal-related features.

6) VERBS

This lexicon is downloaded from a website with the most comprehensive list of verbs [50]. This list contains about 600 instances. Then another list is downloaded from another website [51] containing the most comprehensive list of irregular verbs. This list contains about 300 instances. These lexicon are used in the process of extracting the features for dislike-related features, same as the last two lexicons below.

7) PRONOUNS

There are many types of pronouns. That includes personal, objective, subjective, possessive, demonstrative and many more. We have collected all the instances for the purpose of this study. They are downloaded from a prominent website [52]. It has about 300 instances.

8) EVENT WORDS

A list of synonyms for the word "event" is downloaded from an online dictionary for the purpose of having an event words lexicon. It consists of the words: "affair", "circumstance", "episode", "hap", "happening", "incident", "occasion", "occurrence" and "thing".

J. MACHINE LEARNING CLASSIFIERS

For the purpose of comparison, five classification algorithms are used. This include: i. Support Vector Machine ii. K-Nearest Neighbor iii. Logistic Regression iv. Decision Tree v. Discriminant Analysis. All of these are well-known methods in statistics, pattern recognition and machine learning.

1) SUPPORT VECTOR MACHINE

The support vector machine (SVM) binary classification is a searching algorithm that searches for an optimal hyperplane that can partition a set of data into two classes, negative and positive [53]. In this work, we have decided to use the Radial Basis Function (RBF) as the kernel. An RBF is a function that depends only on the distance between the input and another fixed point. This has yielded the best result amongst all the other kernels for binary classification. This CNN-SVM scheme has also been proven to be quite useful for text classification with deep learning from a previous study [8].

2) K-NEAREST NEIGHBOR

K-nearest neighbor (KNN) is a classification algorithm that puts the outputs into classes by looking at the neighboring nodes [54]. This algorithm depends on a threshold "k". The default value is used for the threshold in this work; "k = 10".

3) LOGISTIC REGRESSION

Same as most keyword-based models, logistic regression (LR) has two varieties: regression and classification [55]. In a binary classification model, logistic regression simply models probability of output in respond to the input given. As a binary classifier, a cutoff value is chosen and the classification is based on whether the probability of the inputs are greater than the cutoff which is going to be put in one class or below the cutoff which is going to be put in the other class For this study, the default cutoff value of 10 is used.

4) DECISION TREE

A decision tree (DT) is an algorithm that uses nodes to represent tests which are ran on an attribute. It also uses branches to represent the result of the tests and leaf nodes to represent labels or conclusions made after calculating all attributes in the experiments. Rules of the experiments are represented by the paths from the root of the tree to the leaf nodes [56].

5) DISCRIMINANT ANALYSIS

Discriminant analysis (DISCR) or also known as Linear Discriminant Analysis (LDA) models the differences between the classes of data given. It can only work when the measurements on the variables used are continuous. It can also be used when groups are known in advance. Each case must have a score on one or more quantitative measures and a score on group measures [57]. Basically, DISCR is the algorithm to group or classify instances of the same type into one group or class.

K. EVALUATION

Once the features are extracted, we proceed to our experiments. The metrics used to evaluate the approach are F1-measure, precision, recall and accuracy. The formulas used for F1-measure, precision, recall and accuracy are given in the equations (1), (2), (3) and (4) below respectively.

F1-measure:

$$F1 = 2.\frac{precision\ .\ recall}{precision + recall} \tag{1}$$

Precision:

$$precision = \frac{TP}{TP + FN} \tag{2}$$

Recall:

$$recall = \frac{TP}{TP + FN} \tag{3}$$

Accuracy:

$$accuracy = \frac{Correct\ predictions}{Total\ predictions} \tag{4}$$

A classification algorithm's effectiveness is usually measured in accuracy, that is denoted in equation (4) [58]. However, the accuracy could be very high while still not bringing any significant value to a detection algorithm. For example, a president of a nation could write one normal sentence and it could be predicted as sarcastic. This mistake might lead to unwanted or serious ramifications even though its only one sentence. The accuracy of the system could still be very high even though the significance or implication of the mistake would be very serious. There is no clear indication of whether the mistake is caused by the algorithm or not.

This is the reason why precision, recall and F1-measure is used in this work. Precision is denoted in equation (2). It is basically the count of instances that are true positive from the total of true positive and false positive instances. This way, the percentage of how many of the real sarcastic instances over everything that is predicted as sarcastic by the system would be known. If the precision is low, the meaning is that the system is not doing a good job in predicting real sarcastic sentences.

Recall is denoted in equation (3). It is basically the count of the instances which are true positive over the total of the instances that are true positive and false negative. In this case, the percentage of the real sarcastic instances over everything that is actually sarcastic would be known. If the recall is low, the meaning is some sarcastic sentences are labeled as normal by the system.

The final equation F1-measure is denoted in equation (1). It is used to put balance between precision and recall, especially when the dataset is imbalanced. Naturally, a sarcasm dataset would have more normal instances than sarcastic ones. This is a direct relation to the real world where human seldom use sarcasm in their conversations in comparison to the normal sentences.

The publicly shared dataset [15] used for the experiments in this work is initially split into two sets, Train and Test in the ratio of 80:20. Then the Train set is split again into Train and Validation in the ratio of 80:20. In short, the whole dataset is split into Train, Test and Validation sets in the ratio of 64:16:20. This dataset has 780,000 English tweets with 83 percent normal instances and the rest are sarcastic instances. Hence, the number of instances in the training set would be 499,200 (414,336 normal and 84,864 sarcastic), the validation set would be 124,800 (103,504 normal and 21,296 sarcastic) and the testing set would be 150,000 (124,500 normal and 25,500 sarcastic).

V. SETUP

The environment used to carry out the experiments is a computer running on 64-bit Windows 10 with an Intel(R) Core(TM) i7 8th Gen with NVIDIA(R) GeForce(R) GTX. The software used is Mathworks Matlab 2019a.

VI. RESULTS AND DISCUSSION

In this section, the results produced by all the experiments is given. This is followed by the explanation of how each component in the experiments are used to fulfill the objective of the study.

A. CLASSIFICATION RESULTS

This subsection consist of performance results for every classification algorithm in the experiment using all the feature sets combined. It is shown in terms of F1-measure, Precision, Recall and Accuracy.

TABLE 1. Performance comparison for classification algorithms using all the feature sets combined.

Classification Algorithm	F1	Precision	Recall	Accuracy
SVM	0.92	0.92	0.92	92%
KNN	0.91	0.91	0.91	91%
LR	**0.94**	**0.95**	**0.94**	**94%**
DT	0.92	0.92	0.92	92%
DISCR	0.92	0.92	0.91	91%

SVM and Decision Tree are both good classifiers for this task, with high accuracies and F1-scores. However, the performance of Logistic Regression are the highest. For the rest of the analysis in this paper, the results used are those from the classifier Logistic Regression.

B. PERFORMANCE COMPARISON WITH EXISTING WORKS

For comparison, two recent works that used the same dataset as in our experiment are chosen. Both of the chosen works are in the domain of sarcasm detection. The results are compiled and shown in Table 2.

TABLE 2. Performance comparison with existing works.

Method	F1	Precision	Recall	Accuracy
Ilić et al. [13]	0.87	0.87	0.87	88%
Shmueli et al. [18]	0.86	0.87	0.91	87%
Proposed Method	**0.94**	**0.95**	**0.94**	**94%**

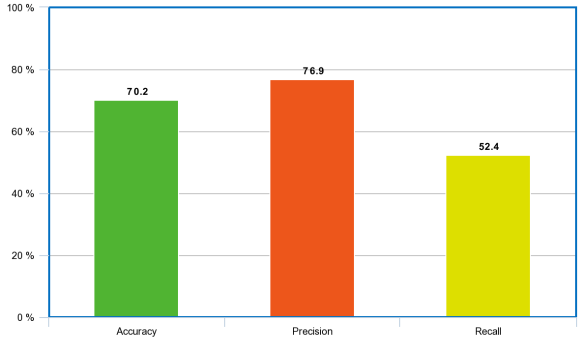

FIGURE 6. Performance of deep feature set.

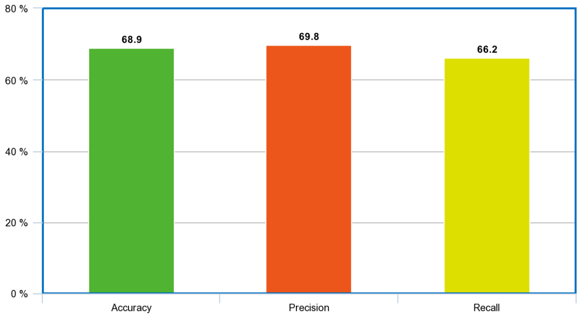

FIGURE 7. Performance of incongruity feature set.

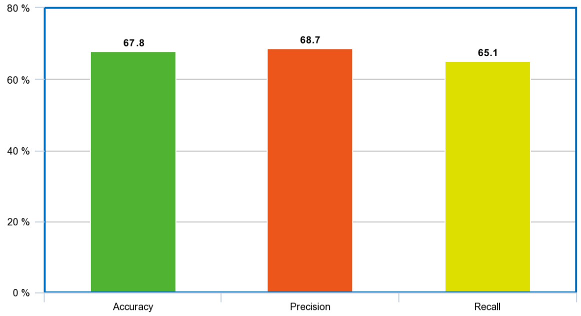

FIGURE 8. Performance of hyperbolic feature set.

The proposed method showed significantly better performance in comparison to the others across all metrics used. This supports our claim that manually extracted contextual feature sets are useful for this task. The performance for each of the feature sets are also experimented.

C. PERFORMANCE COMPARISON AMONG FEATURE SETS

This subsection consist of performance results for each of the feature set. It is shown in terms of Accuracy, Precision and Recall.

Figures 6-10 show the performance of each of the feature sets used in the study. The performance of the deep feature set is obviously the highest in comparison to the others shown below. The performance for the incongruity and hyperbolic feature sets are comparable to the highest.

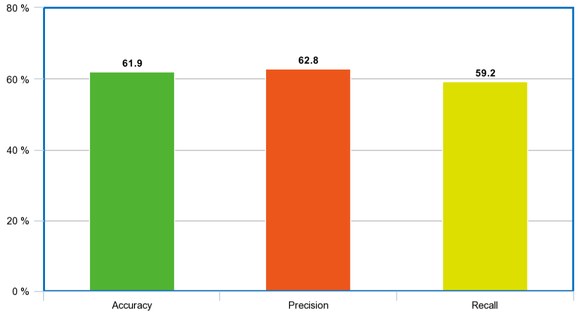

FIGURE 9. Performance of temporality feature set.

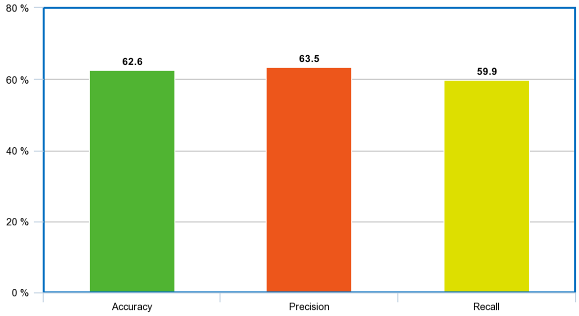

FIGURE 10. Performance of dislike feature set.

Compared to the other three, temporality and dislike feature sets show low performances. According to observation, the first reason for this to happen is that these features has low presence in the data set. Secondly, due to the informal language used in Twitter, temporal words, event words, nouns, verbs, and pronouns are becoming hard to detect. However, the precision given by these feature sets which is more than 60 percent shows the importance of such features for the task of sarcasm detection. Although the performance is not good as stand-alone features, they might have higher added value when correlated with other features.

VII. CONCLUSION AND FUTURE WORK

The results of our experiments provides some valuable insights into the usages of different features of tweets. The features are then exploited to build a framework that's proven useful to detect sarcasm. The method also significantly improves the F1-measure from the existing study using the same dataset. This work also demonstrates the generality of a deep learning architecture. For future work, a few datasets will be used to further generalize the comparisons. The process of extracting and gathering meaningful features could be expanded further.

REFERENCES

[1] A. Ghosh and T. Veale, "Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 482–491.

[2] J. Aquino, "Transforming social media data into predictive analytics," *CRM Mag.*, vol. 16, no. 11, pp. 38–42, 2012.

[3] B. Liu, "Sentiment analysis and opinion mining," *Synth. Lect. Hum. Lang. Technol.*, vol. 5, no. 1, pp. 1–167, 2012.

[4] E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang, "Sarcasm as contrast between a positive sentiment and negative situation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 704–714.

[5] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining Text Data*. Boston, MA, USA: Springer, 2012, pp. 415–463.

[6] A. Joshi, S. Agrawal, P. Bhattacharyya, and M. J. Carman, "Expect the unexpected: Harnessing sentence completion for sarcasm detection," in *Proc. Int. Conf. Pacific Assoc. Comput. Linguistics*. Singapore: Springer, 2017, pp. 275–287.

[7] D. Davidov, O. Tsur, and A. Rappoport, "Semi-supervised recognition of sarcasm in twitter and amazon," in *Proc. 14th Conf. Comput. Natural Lang. Learn.*, 2010, pp. 107–116.

[8] S. Poria, E. Cambria, D. Hazarika, and P. Vij, "A deeper look into sarcastic tweets using deep convolutional neural networks," 2016, *arXiv:1610.08815*. [Online]. Available: http://arxiv.org/abs/1610.08815

[9] N. Majumder, S. Poria, H. Peng, N. Chhaya, E. Cambria, and A. Gelbukh, "Sentiment and sarcasm classification with multitask learning," *IEEE Intell. Syst.*, vol. 34, no. 3, pp. 38–43, May/Jun. 2019.

[10] M. Abulaish, A. Kamal, and M. J. Zaki, "A survey of figurative language and its computational detection in online social networks," *ACM Trans. Web*, vol. 14, no. 1, pp. 1–52, Feb. 2020.

[11] M. Bouazizi and T. Ohtsuki, "Sarcasm detection in Twitter: 'All your products are incredibly amazing!!!'—Are they really??" in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2015, pp. 1–6.

[12] D. Bamman and N. A. Smith, "Contextualized sarcasm detection on twitter," in *Proc. Int. AAAI Conf. Web Social Media*, 2015, pp. 1–4.

[13] S. Ilić, E. Marrese-Taylor, J. A. Balazs, and Y. Matsuo, "Deep contextualized word representations for detecting sarcasm and irony," 2018, *arXiv:1809.09795*. [Online]. Available: http://arxiv.org/abs/1809.09795

[14] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018, *arXiv:1802.05365*. [Online]. Available: http://arxiv.org/abs/1802.05365

[15] T. Ptáček, I. Habernal, and J. Hong, "Sarcasm detection on czech and English Twitter," in *Proc. 25th Int. Conf. Comput. Linguistics, Tech. Papers (COLING)*, 2014, pp. 213–223.

[16] F. Barbieri, H. Saggion, and F. Ronzano, "Modelling sarcasm in twitter, a novel approach," in *Proc. 5th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.*, 2014, pp. 50–58.

[17] M. Bouazizi and T. Otsuki Ohtsuki, "A pattern-based approach for sarcasm detection on Twitter," *IEEE Access*, vol. 4, pp. 5477–5488, 2016.

[18] B. Shmueli, L.-W. Ku, and S. Ray, "Reactive supervision: A new method for collecting sarcasm data," 2020, *arXiv:2009.13080*. [Online]. Available: http://arxiv.org/abs/2009.13080

[19] A. Joshi, P. Bhattacharyya, and M. J. Carman, "Automatic sarcasm detection: A survey," *ACM Comput. Surv.*, vol. 50, no. 5, pp. 1–22, 2017.

[20] R. Kreuz and G. Caucci, "Lexical influences on the perception of sarcasm," in *Proc. Workshop Comput. Approaches Figurative Lang.*, 2007, pp. 1–4.

[21] S. Amir, B. C. Wallace, H. Lyu, and P. C. M. J. Silva, "Modelling context with user embeddings for sarcasm detection in social media," 2016, *arXiv:1607.00976*. [Online]. Available: http://arxiv.org/abs/1607.00976

[22] A. Ghosh and T. Veale, "Fracking sarcasm using neural network," in *Proc. 7th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.*, 2016, pp. 161–169.

[23] A. Rajadesingan, R. Zafarani, and H. Liu, "Sarcasm detection on Twitter: A behavioral modeling approach," in *Proc. 8th ACM Int. Conf. Web Search Data Mining*, Feb. 2015, pp. 97–106.

[24] A. Baruah, K. Das, F. Barbhuiya, and K. Dey, "Context-aware sarcasm detection using bert," in *Proc. 2nd Workshop Figurative Lang. Process.*, 2020, pp. 83–87.

[25] A. Joshi, V. Sharma, and P. Bhattacharyya, "Harnessing context incongruity for sarcasm detection," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process.*, vol. 2, 2015, pp. 757–762.

[26] F. Kunneman, C. Liebrecht, M. van Mulken, and A. van den Bosch, "Signaling sarcasm: From hyperbole to hashtag," *Inf. Process. Manage.*, vol. 51, no. 4, pp. 500–509, Jul. 2015.

[27] S. K. Bharti, K. S. Babu, and S. K. Jena, "Parsing-based sarcasm sentiment recognition in Twitter data," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Aug. 2015, pp. 1373–1380.

[28] O. Tsur, D. Davidov, and A. Rappoport, "Icwsm-a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews," in *4th Int. AAAI Conf. Weblogs Social Media*, 2010, pp. 1–8.

[29] C. Liebrecht, F. Kunneman, and A. van Den Bosch, "The perfect solution for detecting sarcasm in tweets# not," in *Proc. 4th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.* Atlanta, GA, USA: Association for Computational Linguistics, Jun. 2013, pp. 29–37.

[30] C. Fellbaum, "Wordnet," in *Theory and Applications of Ontology: Computer Applications*. Dordrecht, The Netherlands: Springer, 2010, pp. 231–243.

[31] A. Reyes, P. Rosso, and D. Buscaldi, "From humor recognition to irony detection: The figurative language of social media," *Data Knowl. Eng.*, vol. 74, pp. 1–12, Apr. 2012.

[32] D. Hazarika, S. Poria, S. Gorantla, E. Cambria, R. Zimmermann, and R. Mihalcea, "CASCADE: Contextual sarcasm detection in online discussion forums," 2018, *arXiv:1805.06413*. [Online]. Available: http://arxiv.org/abs/1805.06413

[33] M. Ebrahimi, A. H. Yazdavar, and A. Sheth, "Challenges of sentiment analysis for dynamic events," *IEEE Intell. Syst.*, vol. 32, no. 5, pp. 70–75, Sep./Oct. 2017.

[34] L. Peled and R. Reichart, "Sarcasm SIGN: Interpreting sarcasm with sentiment based monolingual machine translation," 2017, *arXiv:1704.06836*. [Online]. Available: http://arxiv.org/abs/1704.06836

[35] S. Attardo, J. Eisterhold, J. Hay, and I. Poggi, "Multimodal markers of irony and sarcasm," *Humor*, vol. 16, no. 2, pp. 243–260, 2003.

[36] P. Rockwell, "Vocal features of conversational sarcasm: A comparison of methods," *J. Psycholinguistic Res.*, vol. 36, no. 5, pp. 361–369, Jul. 2007.

[37] M. Zhang, Y. Zhang, and G. Fu, "Tweet sarcasm detection using deep neural network," in *Proc. 26th Int. Conf. Comput. Linguistics, Tech. Papers (COLING)*, 2016, pp. 2449–2460.

[38] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.

[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781

[40] S. L. Ivanko and P. M. Pexman, "Context incongruity and irony processing," *Discourse Process.*, vol. 35, no. 3, pp. 241–279, 2003.

[41] J. D. Campbell and A. N. Katz, "Are there necessary conditions for inducing a sense of sarcastic irony?" *Discourse Process.*, vol. 49, no. 6, pp. 459–480, Aug. 2012.

[42] A. Ramteke, A. Malu, P. Bhattacharyya, and J. S. Nath, "Detecting turnarounds in sentiment analysis: Thwarting," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics, Short Papers*, vol. 2, 2013, pp. 860–865.

[43] S. K. Bharti, B. Vachha, R. K. Pradhan, K. S. Babu, and S. K. Jena, "Sarcastic sentiment detection in tweets streamed in real time: A big data approach," *Digit. Commun. Netw.*, vol. 2, no. 3, pp. 108–121, Aug. 2016.

[44] P. Rockwell, "Empathy and the expression and recognition of sarcasm by close relations or strangers," *Perceptual Motor Skills*, vol. 97, no. 1, pp. 251–256, Aug. 2003.

[45] A. Chen and L. Boves, "What's in a word: Sounding sarcastic in British English," *J. Int. Phonetic Assoc.*, vol. 48, no. 1, pp. 57–76, Apr. 2018.

[46] B. Liu, M. Hu, and J. Cheng, "Opinion observer: Analyzing and comparing opinions on the Web," in *Proc. 14th Int. Conf. World Wide Web (WWW)*, 2005, pp. 342–351.

[47] E. Troiano, C. Strapparava, G. Özbal, and S. S. Tekiroğlu, "A computational exploration of exaggeration," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3296–3304.

[48] *Temporal Words*. Accessed: Feb. 21, 2021. [Online]. Available: https://grammar.yourdictionary.com/style-and-usage/list-transition-words.html

[49] *Noun*. Accessed: Feb. 21, 2021. [Online]. Available: https://www.talkenglish.com/vocabulary/top-1500-nouns.aspx

[50] *Verb*. Accessed: Feb. 21, 2021. [Online]. Available: https://www.englishclub.com/vocabulary/regular-verbs-list.htm

[51] *Iverb*. Accessed: Feb. 21, 2021. [Online]. Available: https://www.englishclub.com/vocabulary/irregular-verbs-list.htm

[52] *Pronoun*. Accessed: Feb. 21, 2021. [Online]. Available: https://www.really-learn-english.com/list-of-pronouns.html

[53] V. Vapnik and R. Izmailov, "Knowledge transfer in SVM and neural networks," *Ann. Math. Artif. Intell.*, vol. 81, nos. 1–2, pp. 3–19, Oct. 2017.

[54] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[55] A. Christmann, "Least median of weighted squares in logistic regression with large strata," *Biometrika*, vol. 81, no. 2, pp. 413–417, 1994.

[56] B. Kamiński, M. Jakubczyk, and P. Szufel, "A framework for sensitivity analysis of decision trees," *Central Eur. J. Oper. Res.*, vol. 26, no. 1, pp. 135–159, Mar. 2018.

[57] G. D. Garson, "Discriminant function analysis. statnotes: Topics in multivariate analysis," *Retrieved March*, vol. 29, p. 2010, Mar. 2008.

[58] J. Huang and C. X. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 3, pp. 299–310, Mar. 2005.

MD SAIFULLAH RAZALI is currently pursuing the dual Ph.D. degree with the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Malaysia, and the Faculty of Informatics, University of Wollongong. His research interests include figurative language, machine learning, and big data.

ALFIAN ABDUL HALIN received the Bachelor of Science degree (Hons.) in information technology from the Universiti Technology MARA, Malaysia, the Master of Multimedia Computing degree from Monash University, Australia, and the Ph.D. degree in computer science from the Universiti Sains Malaysia. He is currently a Senior Lecturer with the Multimedia Department, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia. He belongs to the Digital Information Computation and Retrieval Group, Universiti Putra Malaysia. His current research interests include computer vision and machine learning applications, and digital image/video processing applications.

LEI YE received the B.Eng. and Ph.D. degrees from Xidian University, China, in 1982 and 1989, respectively. After worked for several years in the industry with Motorola Australian Research Centre, he joined the University of Wollongong, Australia, in 2004. His recent research outcomes in image retrieval have led to the creation of a spin off company of the University of Wollongong. His current research interests include image processing, retrieval and annotation, multimedia communication and computing, multimedia content management, rights management, and security.

SHYAMALA DORAISAMY (Member, IEEE) received the Ph.D. degree from Imperial College London, in 2004, with a focus on music information retrieval. She is currently an Associate Professor with the Department of Multimedia, Faculty of Computer Science and Information Technology, University Putra Malaysia (UPM). She also heads the Digital Information Computation and Retrieval Research Group, UPM. Her research interest includes multimedia information processing, focusing in particular on audio content analysis and applications for music and health informatics.

NORIS MOHD NOROWI received the degree in computer science with a focus on multimedia and the Master of Science degree in multimedia systems from the Universiti Putra Malaysia, and the Ph.D. degree in computer music from the University of Plymouth, U.K. She is currently a Senior Lecturer with the Multimedia Department, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia. She belongs to the Human–Computer Interaction Research Group, where her researches include artificial intelligence in music, immersive technologies, sound cognition, and sound synthesis.