Interpretability of Computational Models for Sentiment Analysis

Han Liu, Mihaela Cocea and Alexander Gegov

School of Computing, University of Portsmouth, Buckingham Building, Lion Terrace, Portsmouth, PO1 3HE, United Kingdom
{Han.Liu, Mihaela.Cocea and Alexander.Gegov}@port.ac.uk

Abstract Sentiment analysis, which is also known as opinion mining, has been an increasingly popular research area focusing on sentiment classification/regression. In many studies, computational models have been considered as effective and efficient tools for sentiment analysis. Computational models could be built by using expert knowledge or learning from data. From this viewpoint, the design of computational models could be categorized into expert based design and data based design. Due to the vast and rapid increase in data, the latter approach of design has become increasingly more popular for building computational models. A data based design typically follows machine learning approaches, each of which involves a particular strategy of learning. Therefore, the resulting computational models are usually represented in different forms. For example, neural network learning results in models in the form of multi-layer perceptron network whereas decision tree learning results in a rule set in the form of decision tree. On the basis of above description, interpretability has become a main problem that arises with computational models. This chapter explores the significance of interpretability for computational models as well as analyzes the factors that impact on interpretability. This chapter also introduces several ways to evaluate and improve the interpretability for computational models which are used as sentiment analysis systems. In particular, rule based systems, a special type of computational models, are used as an example for illustration with respects to evaluation and improvements through the use of computational intelligence methodologies.

Keywords: Sentiment Prediction, Computational Intelligence, Machine Learning, Rule Based Systems, Interpretability Analysis, Interpretability Evaluation, Fuzzy Computational Models, Rule Based Networks

1 Introduction

Sentiment analysis, which is also known as opinion mining, generally aims to identify the attitude of a speaker/writer using natural language processing, text analysis and computational linguistics. In recent years, research has been increasingly focusing on sentiment classification/regression. In other words, sentiment analysis can be regarded as a classification task if the sentiment is analyzed in qualitative terms, e.g. positive/negative and rating, and as a regression task if the sentiment is analyzed in quantitative terms, e.g. numerical values.

Many studies have shown that computational models are effective and efficient tools for sentiment classification/regression. Such models can be designed through adopting computational intelligence approaches. In particular, these can be done by using expert knowledge or learning from real data. From this point of view, the design of computational models can be divided into expert based design and data based design. The former follows traditional system engineering approaches [1, 2] whereas the latter typically follows machine learning approaches [3].

The expert based approach is in general domain dependent. It is necessary to have knowledge or requirements acquired from experts at first and then to identify the relationships between inputs and outputs. Modelling, which is the most important step, is to be executed in order to build a system. Once the modelling is complete, then simulation is started to check the model towards fulfillment of systematic complexity such as predictive accuracy and computational efficiency. Finally, statistical analysis is undertaken in order to validate whether the computational model is reliable and efficient in practical application.

On the other hand, the data based approach is in general domain independent. As mentioned earlier in this section, it typically follows machine learning approaches. Machine learning is a branch of artificial intelligence and involves two stages: training and testing [3]. Training aims to learn something from known properties by using learning algorithms and testing aims to make predictions on unknown properties by using the knowledge learned in the training stage. From this point of view, training and testing are also known as learning and prediction respectively. In practice, a machine learning task aims to build a model that is further used to make predictions through the use of learning algorithms. Therefore, this task is usually referred to as predictive modelling. In the context of sentiment analysis, supervised modelling approaches, which can be involved in classification or regression tasks, are popularly used for solving particular issues.

As justified in [4], the data based approach is more effective and efficient than expert based approach in modern development. The main reason is explained with the following arguments: expert knowledge may be incomplete or inaccurate; some of experts' viewpoints may be biased; engineers may misunderstand requirements or have technical designs with defects. In other words, with regards to solving problems with high complexity, it is difficult for both domain experts and engineers to

have all possible cases considered or to have perfect technical designs. Once a failure arises, domain experts and engineers may have to find the problem and fix it through reanalysis or redesign. However, the real world has had big data available. Some previously unknown information or knowledge may be discovered from data. Data may potentially be used as supporting evidence to reflect some useful and important pattern by using modeling techniques. More importantly, the model could be modified automatically along with the update of database in real time if the data based modeling technique is used. On the basis of above description, the data based approach is recommended instead of the expert based approach for design of computational models.

In machine learning research, computational models can be evaluated in three main dimensionalities, namely accuracy, efficiency and interpretability. This chapter focuses on the analysis of interpretability for computational models since interpretable models have been increasingly required. In particular, computational models can be used not only to make predictions but also to extract knowledge which needs to be communicated to people. For example, a computational model is built as a sentiment analysis system in order to provide people with the judged rating score of a review. People may not trust the judgment made by the sentiment analysis system unless they can fully understand the reasons of the judgment making. Consequently, interpretability of computational models has become a significant aspect which needs further research.

As different machine learning algorithms involve different learning strategies, the computational models built by using these algorithms are usually represented in different forms and thus are characterized by different levels of interpretability. Some machine learning algorithms lead to computational models that operate in a black box manner. In other words, the models can make highly accurate predictions as outputs but it is difficult to interpret the reason why the outputs are derived from the models. On this basis, the interpretability of a computational model has become increasingly significant for knowledge usage.

As introduced in [4], most computational models can work based on different types of logic such as deterministic logic, probabilistic logic and fuzzy logic. Ross stated in [5] that logic is a small part of human capability for reasoning, which is used to assist people in making decisions or judgments. As mentioned in [6], in the context of Boolean logic, each variable is only assigned a binary truth value: 0 (false) or 1 (true). It indicates that reasoning and judgment are made under certainty resulting in a deterministic outcome. From this point of view, this type of logic is also referred to as deterministic logic. However, in reality, people usually can only reason and make decisions and judgments under uncertainty. Therefore, the other two types of logic, namely probabilistic logic and fuzzy logic, are used more widely, both of which can be seen as an extension of deterministic logic. The main difference is that the truth value is not binary but continuous between 0 and 1. The truth value implies a probability of truth between true and false in probabilistic logic and a degree of that in fuzzy logic, both of which need to be interpretable through the use of computational modes in order to indicate explicitly the probability or degree.

The rest of this paper is organized as follows: Section 2 explores the significance of interpretability. Section 3 identifies a list of impact factors that may affect the interpretability of a computational model. Section 4 introduces some criteria for evaluation against interpretability. Some recommendations are given with respect to improvements on interpretability in Section 5. Section 6 summarizes the completed work and suggests some further directions on the development of computational modelling with respect to interpretability.

2 Significance of Interpretability

As mentioned in Section 1, interpretability of computational models has been increasingly significant in practice for knowledge usage. This section justifies why interpretability of a computational model is so significant.

In practice, machine learning methods can be used for two main purposes. One is to build a predictive model that is used to make predictions. The other one is to discover some meaningful and useful knowledge from data [7]. For the latter purpose, the knowledge discovered is used to provide insights for a knowledge domain. From this point of view, it is required to have a computational model which works in a white box manner. This is in order to make the model transparent so that people can understand the reasons why the output is derived from the model.

However, as mentioned in Section 1, different learning algorithms involve different strategies of learning that usually result in different ways of representing knowledge. In terms of knowledge representation, Higgins justified in [8] that an interpretable model needs to be able to provide the explanation with regard to the reason of an output and that a rule based knowledge representation makes computational models more interpretable with the following arguments:

- A network was conceived in [9], which needs a number of nodes exponential in the number of attributes in order to restore the information on conditional probabilities of any combination of inputs. It is argued in [8] that the network restores a large amount of information that is mostly less valuable.
- Another type of networks known as Bayesian Networks introduced in [10] needs a number of nodes which is the same as the number of attributes. However, the network only restores the information on joint probabilities based on the assumption that each of the input attributes is totally independent of the others. Therefore, it is argued in [8] that this network is unlikely to predict more complex relationships between attributes due to the lack of information on correlational probabilities between attributes.
- There are some other methods that fill the gaps that exist in Bayesian Networks by deciding to only choose some higher-order conjunctive probabilities, such as the first neural networks [11] and a method based on correlation/dependency measure [12]. However, it is argued in [8] that these

methods still need to be based on the assumption that all attributes are independent of each other.

On the basis of above arguments, Higgins motivated the use of rule based knowledge representation due mainly to the advantage that rules specify relationships between attributes, which can provide explanations with regard to a decision from a computational model [8]. Therefore, Higgins argues the significance of interpretability, i.e. the need to explain the output of a computational model based on the reasoning of that model.

In the machine learning context, due to the presence of massively large data, it is very likely to have a complex model built, which makes the knowledge extracted from such a model cumbersome and less readable for people. In this case, it is necessary to represent the model in a way that has a high level of interpretability.

On the other hand, different people would usually have different level of cognitive capability. In other words, the same message may take different meanings to different people due to their different level of capability of reading and understanding. In addition, different people would also have different levels of expertise and different preferences with regard to the way of receiving information. All these issues raised above make it necessary that knowledge extracted from a computational model needs to be represented in a way that suits people to read and understand. In other words, high interpretability of a computational model is required in order to make the knowledge extracted from the model more readable and understandable to different groups of people.

In sentiment analysis, a great number of studies focus on sentiment classification/regression. In particular, machine learning methods, such as Naïve Bayes and Support Vector Machine, are popularly used to improve the accuracy of sentiment prediction [13, 14]. However, as mentioned in Section 1, sentiment analysis is also known as opinion mining, which means to discover opinions from texts. In other words, sentiment analysis is considered as a data mining task and the opinions discovered from texts need to be communicated to people, which again indicates the significance of interpretability of computational models. Those popular machine learning methods used in sentiment analysis such as Naïve Bayes and Support Vector Machine are useful to build accurate sentiment prediction systems but are difficult to provide interpretable models due to the nature of their learning strategies.

In terms of Naïve Bayes, the interpretability problem is similar to that of Bayesian Networks. As mentioned earlier in this section, this type of learning algorithms need to work based on the assumption that all input attributes are totally independent of each other. This assumption makes generated models unable to represent the correlations among input attributes and thus results in insufficient interpretability.

In terms of Support Vector Machine, the interpretability problem is present due to the limitations in transparency and depth of learning. With regard to transparency, support vector machine is not working in a black box manner but models built by using this method are still less transparent to a general audience. This is because the model built by using this algorithm appears to be function lines as decision boundaries in geometric form or a piecewise function in algebraic form. As mentioned

earlier, due to different levels of cognitive capability from different people, this type of model representation is usually less interpretable to a non-technical audience who does not know mathematics well. With regard to the depth of learning, this method involves a sound theory in learning strategy but does not aim to learn in depth. In particular, this method does not go through all data instances but just takes a look at a few instances that are selected as support vectors, in order to make predictions. In this case, there could be a great amount of useful information failed to be discovered from data.

On the basis of the above discussion, interpretability is considered as a significant issue in sentiment analysis, and thus it is required to be improved through advancing computational intelligence approaches. However, interpretability is in general a domain-independent problem. Therefore, the rest of this chapter provides a theoretical analysis of issues relating to interpretability as well as an empirical investigation on ways of improving interpretability of computational models in a domain-independent manner.

3 Impact Factors for Interpretability

As described in Section 2, the interpretability of computational models is significant due to the presence of some problems from machine learning methods, size of data, model representation and human expertise and preferences. This section discusses how each of these factors has an influence on interpretability.

3.1 Learning Strategy

As mentioned in Section 2, different machine learning algorithms usually involve different strategies of learning. This would usually result in differences in two aspects namely, transparency and model complexity.

In terms of transparency, a typical example would be the natural difference between neural networks and rule based learning methods in terms of learning strategy. Neural network learning aims to construct a network topology that consists of a number of layers and that has a number of nodes, each of which represents a perceptron. As a neural network is working in a black box manner with regard to providing an output, the transparency is poor, i.e. people cannot see in an explicit way the reason why the output is given. On the basis of the above description, neural networks have been judged poorly interpretable in [15]. On the other hand, a rule based method aims to generate a rule set typically in the form of either a decision tree or if-then rules. As also mentioned in Section 2, rule based knowledge representation is able to explain the reason explicitly with regard to providing an output

and, thus, it is well transparent. In addition, another popular machine learning algorithm, which is known as k nearest neighbor, involves lazy learning. In other words, the learning algorithm does not aim to learn in depth to gain some pattern from data but just to make as many correct predictions as possible. In the training stage, there is no actual learning but just some data loaded into computer memory. In this sense, there is no model built in the training stage so there is nothing to be visible for people to gain some useful patterns. On the basis of the above description relating to transparency, the strategies of learning involved in learning algorithms are an impact factor that affects interpretability.

In terms of model complexity, the learning strategy is a significant impact factor. For example, as mentioned earlier, k nearest neighbor does not build a model in the training stage whereas a rule based method would build a rule set. For the former algorithm, the model complexity is 0 as there is no model built. For the latter algorithm, the model complexity would be determined by the overall number of rule terms that is dependent on the number of rules and the average number of terms per rule, i.e. a large number of complex rules indicates the rule set has a high complexity whereas a small number of general rules indicates the rule set has a low complexity. For rule based methods, the strategy of rule induction would usually significantly affect the model complexity. As mentioned in [16, 17], the induction of classification rules could be divided into two categories: ‘Divide and Conquer’ [18] and ‘Separate and Conquer’ [19]. The two approaches are also referred to as Top-Down Induction of Decision Trees (TDIDT) and covering approach respectively. As introduced in [20, 21], Prism, which is a rule induction method that follows the ‘Separate and Conquer’ approach, is likely to generate fewer and more general rules than decision tree, which is another rule induction method that follows the ‘Divide and Conquer’ approach. The above phenomenon is due mainly to the strategy of rule learning. As mentioned in [20], the rule set generated by TDIDT needs to have at least one common attribute to be in the form of decision trees. The same also applies to each of the subtrees of a decision tree, which requires to have at least one common attribute represented as the root of the subtree. Due to this requirement, TDIDT is likely to generate a large number of complex rules with many redundant terms such as the replicated subtree problem [20] illustrated in Fig.1 and thus results in a model of high complexity.

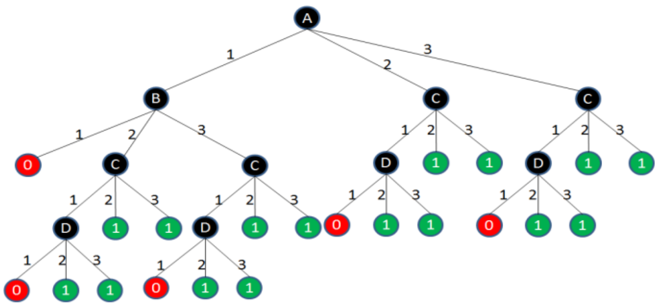

Fig. 1. Cendrowska’s replicated subtree example [22]

The above description relating to model complexity also indicates that learning strategies involved in learning algorithms can be an impact factor that may affect the interpretability.

3.2 Data Size

As mentioned in Section 3.1, different learning algorithms involve different strategies of learning and thus generate models with different level of complexity. In this sense, when the same data set is used, different learning algorithms would usually lead to different model complexities. However, for the same algorithm, data of different size would also usually result in the generation of models with different levels of complexity. The rest of this subsection justifies the potential correlation between data size and model complexity using rule based methods and Naïve Bayes as examples.

As mentioned earlier, rule based methods involve the generation of rule sets. The complexity of a rule set is determined by the total number of rule terms, which is dependent upon the number of rules and the average number of terms per rule. However, the total number of rule terms is also affected by the data size in terms of both dimensionality (number of attributes) and sample size (number of instances). For example, a data set has n attributes, each of which has t values, and its sample contains m instances and covers all possible values for each of the attributes. In this example, the model complexity would be equal to $\sum t^i$, while $i=0, 1, 2 \ldots n$, but no greater than $m \times n$ in the worst case. This indicates that a rule set consists of a default rule, which is also referred to as 'else' rule, and t^i rules, each of which has i terms, for $i=0, 1, 2 \ldots n$ respectively. However, each rule usually covers more than one instance and the rule set is expected to cover all instances. Therefore, the number of rules from a rule set is usually less than the number of instances from a data set. As also justified above, each rule would have up to n (the number of attributes) terms due to the requirement that each attribute can only appear once comprising one of its possible values in any of the rules. On the basis of above description, the complexity of a rule set is up to the product of dimensionality and sample size of a data set.

On the other hand, Naïve Bayes, which is another machine learning algorithm, aims to identify a list of probabilistic correlations, such as $P(class=1|x_1=1)= 0.5$, between an input attribute and the class attribute [23]. The model complexity would be dependent on the number of correlation pairs. For example, a data set contains n input attributes plus one class attribute and each of the attributes (including the class attribute) has t values included in the data set. In this example, the complexity of the model would be equal to $n \times t^2$. This is because Naïve Bayes aims to identify the probabilistic correlation of each attribute-value pair to each class label. In this sense,

each attribute has t values and the class attribute has t labels so the number of correlation pairs is t^2 for the attribute and thus the overall number is the same as n times the above number (t^2).

The above description relating to Naïve Bayes also indicates that a large number of attributes with many possible values is likely to make a data set in a large size, and thus data size is also one of the impact factors that may affect the interpretability.

3.3 Model Representation

As mentioned in Section 2, different types of machine learning algorithms may generate models represented in different forms. For example, the 'divide and conquer' approach generates a rule set in the form of a decision tree as illustrated in Fig.1 whereas the 'separate and conquer' approach would generate if-then rules represented in a linear list. In addition, neural network learning algorithm would generate a multi-layer network with a number of interconnected nodes, each of which represents a perceptron. As also described in Section 2, models generated by rule based learning methods are in white box and thus well transparent whereas models constructed by neural network learning methods are in black box and thus poorly transparent. As justified in Section 3.1, the level of transparency can affect the level of interpretability. However, models that demonstrate the same level of transparency may also have different levels of interpretability due to their differences in terms of representation such as rule based models. The rest of this subsection justifies why and how the nature of model representation can affect the level of interpretability using rule based models as a special example.

As mentioned earlier in this subsection, a rule set can be represented in two different forms namely, decision tree and linear list.

Decision tree representation is criticized by Cendrowska and identified as a major cause of overfitting in [20] due to the replicated sub-tree problem as illustrated in Fig.1. It can be seen from the Fig.1 that the four sub-trees which have node C as their roots are identical. This is due to the requirement that all rules must have at least one common attribute involved as mentioned in Section 3.1. This kind of problems mentioned above could be referred to as redundancy, which would lower the interpretability as argued in [19] that decision trees are often quite complex and difficult to understand and thus inscrutable to provide insight into a domain for knowledge usage [18, 19].

In comparison with decision trees, linear lists do not have the constraint that all rules must have common attributes and thus reduces the presence of redundant terms in a rule set. However, redundancy may still arise with this representation. This is because the same attribute may repetitively appear in different rules as illustrated by the example below:

Rule 1: If x=0 and y=0 Then class= 0;
Rule 2: If x=0 and y=1 Then class= 1;
Rule 3: If x=1 and y=0 Then class= 1;
Rule 2: If x=1 and y=1 Then class= 0;

When a training set is large, there would be a large number of complex rules generated. In this case, the presence of redundancy would make the rule set (represented in a linear list) very cumbersome and difficult to interpret for knowledge usage. In other words, a large number of complex rules represented in this way is quite like a large number of long paragraphs in an article that would be very difficult for people to read and understand. Instead, people prefer to look at diagrams to gain information. In this sense, a graphical representation of rules would be expected to improve the interpretability of a model. More details about ways to improve interpretability will be presented in Section 5.

3.4 Human Characteristics

As mentioned in Section 2, different people may have different levels of expertise and preferences and thus different levels of cognitive capability to understand the knowledge extracted from a particular computational model. The rest of this subsection justifies why and how human expertise and personality may affect the interpretability of computational models.

In terms of expertise, due to the fact that an expert system is used to act as a domain expert to extract knowledge or make predictions, people need to have the relevant expertise in order to be able to understand the context. From this point of view, the exactly same model may demonstrate different levels of interpretability for different people due to their different levels of expertise in this domain.

In terms of preferences, due to the fact that different people may have different preferences with respect to the way of reading, the exactly same model may also demonstrate different level of interpretability for different people due to their different preferences. From this point of view, human characteristics are also an impact factor that may affect the interpretability of a model.

As mentioned in Section 3.3, model representation can affect the interpretability with respect to the level of redundancy. In other words, the same model can have different levels of interpretability depending on its representation. However, due to the difference in expertise and preferences, a particular representation may be understandable to some people but not to others. For example, some people in nature sciences/engineering would prefer to read diagrams/ mathematical formulas whereas others may dislike them. From this point of view, model representation, human expertise and characteristics may jointly determine the cognitive capability for people to read and understand the knowledge extracted from a model.

4 Criteria for Evaluation of Interpretability

On the basis of description in Section 3, the list of identified impact factors would have causal relationships to the interpretability of a model as illustrated in Fig.2.

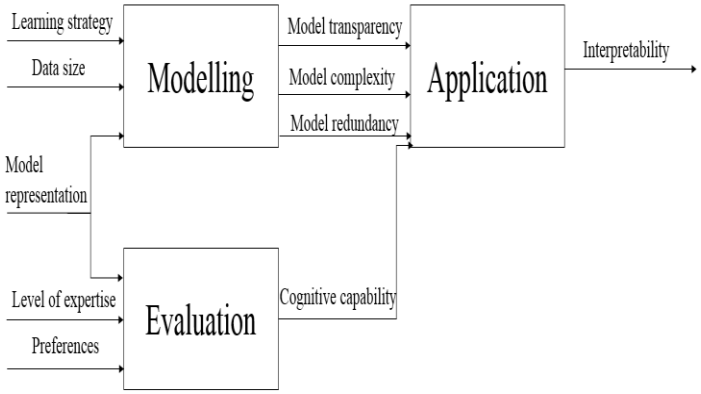

Fig. 2. Causal relationship between impact factors and interpretability [24]

Fig.2 indicates that the evaluation of a model's interpretability could be based on several criteria, namely model transparency, model complexity, model redundancy and cognitive capability, due to their direct relationships to interpretability.

In terms of model transparency, as mentioned in Section 3.1, the evaluation is based on information visualization. In other words, to what extent the information is visible or hidden to people. For example, a neural network learning method generates a model in black box, which means that the information in the model is mostly hidden to people and thus poorly transparent. In contrast, a rule based learning method generates a model in white box, which means the information in the model is totally visible to people and thus well transparent.

In terms of model complexity, the evaluation is subject to the type of learning algorithms to some extent. For example, with regard to rule based methods, the model complexity could be measured by checking the total number of rule terms in a rule set, which is referred to as rule set complexity. For the rule set given below, the complexity would be 8.

Rule 1: If x=0 and y=0 Then class= 1;
Rule 2: If x=0 and y=1 Then class= 0;
Rule 3: If x=1 and y=0 Then class= 0;
Rule 4: If x=1 and y=1 Then class= 1;

In addition, with regard to Naïve Bayes method, the complexity could be measured by checking the number of probabilistic correlation pairs such as

P(class=1|x1=1) = 0.5. For the example shown in Table 1 and in the list of probabilistic correlations, the complexity would be 8. Besides, the model complexity also partially depends on data size as mentioned in Section 3.2.

Table 1. Probabilistic correlations

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	1

Pair 1: $P(y=0|x_1=0) = 0.5$
Pair 2: $P(y=1|x_1=0) = 0.5$
Pair 3: $P(y=0|x_1=1) = 0.5$
Pair 4: $P(y=1|x_1=1) = 0.5$
Pair 5: $P(y=0|x_2=0) = 0.5$
Pair 6: $P(y=1|x_2=0) = 0.5$
Pair 7: $P(y=0|x_2=1) = 0.5$
Pair 8: $P(y=1|x_2=1) = 0.5$

In terms of model redundancy, the evaluation could be based on the extent of information duplication. For example, a rule set may be represented in different forms namely, decision tree, linear list and rule based network. As mentioned in Section 3.3, both of the first two representations may include duplicated information. For decision tree, the replicated subtree problem is a typical example to indicate that redundancy is a principal problem that arises with this representation. As can be seen from Fig.1, there are four identical subtrees. For a linear list, as can be seen from the rule set given earlier in this section, all of the four rules have two common attributes, namely 'x' and 'y', which are repeated. The authors have recently developed two types of network topologies [22] in order to reduce redundancy. One is attribute-value-oriented and the other one is attribute oriented. More details on the improvements of interpretability by using network topologies will be described in Section 5.

In terms of cognitive capability, the evaluation would be based on empirical analysis following machine learning approaches. This is in order to analyze to what extent the model representation is understandable to particular people. In detail, this could be designed as a classification task to predict the cognitive capability in qualitative aspects or as a regression task to predict in quantitative aspects. Briefly speaking, the analysis could be done by collecting the data records on expertise and preferences from previous people who have high similarities to the current people and then taking the majority voting for a classification task or averaging for a regression task. With respect to the cognitive capability. The above task can be done effectively using k nearest neighbor, which is a popular machine learning method.

5 Improvement of Interpretability

Section 3 listed some impact factors for interpretability and Section 4 introduced some criteria with regard to evaluation of interpretability. In particular, transparency is dependent on learning algorithms. Model complexity is subject to both learning algorithms and data size. Model redundancy is subject to the form of model representation. Cognitive capability is determined jointly by model representation, human expertise and characteristics. This section demonstrates in detail some ways recommended in [24] towards improvements of interpretability through advancing and using computational intelligence approaches.

5.1 Scaling Up Algorithms

As mentioned in Section 3, the strategy of a learning algorithm may affect the model transparency. In this case, the transparency could be improved by scaling up algorithms in terms of depth of learning. For example, rule based methods usually generate models with good transparency because this type of learning is in great depth and on an inductive basis. As also mentioned in Section 2, models represented by rules can provide straightforward explanations with regard to the outputs of a computational model.

On the other hand, the performance of a learning algorithm would also affect the model complexity as mentioned in Section 3. In this case, the model complexity could be reduced by scaling up algorithms. In the context of rule based models, complexity could be reduced through proper selection of rule generation approaches. As mentioned in Section 3, there are typically two generation approaches namely, 'divide and conquer' and 'separate and conquer'. As mentioned in [19], the latter is usually likely to generate less complex rule sets than the former. The following example is given for illustration:

Table 2. Data example from [21]

Object	Height	Hair	Eyes	Class
O1	short	blond	blue	C1
O2	short	blond	brown	C2
O3	tall	red	blue	C1
O4	tall	dark	blue	C2
O5	tall	dark	blue	C2
O6	tall	blond	blue	C1
O7	tall	dark	brown	C2
O8	short	blond	brown	C2

According to [21], both ID3 and Prism generate four rules as follows:

Rule set by ID3:

- if Hair= red then Class= C1;
- if Hair= blond and Eyes= blue then Class= C1;
- if Hair= blond and Eyes= brown then Class= C2;
- if Hair= dark then Class= C2;

Rule set by Prism:

- if Hair= red then Class= C1;
- if Hair= blond and Eyes= blue then Class= C1;
- if Eyes= brown then Class= C2;
- if Hair= dark then Class= C2;

As can be seen from the above example, ID3, which follows the divide and conquer rule learning, generates a redundant term (Hair= blond) in the third rule whereas Prism, which follows the separate and conquer rule learning, successfully manages to remove the redundancy. The above example also indicates Prism manages to reduce the complexity of rule set through the removal of the redundant term from the third rule. In addition, Cendrowska compared ID3 with Prism in [20] on the training set, which has 7 attributes and 647 instances and is provided by the King-Knight-King-Rook chess end game [25]. As reported in [20], the rule set generated by ID3 contains 52 rules and 337 terms whereas that generated by Prism only has 15 rules and 48 terms.

On the other hand, the following empirical investigation using the data sets retrieved from UCI repository [26], which is illustrated in Table 3, also indicates that separate and conquer rule learning algorithms are likely to generate fewer and more general rules than algorithms that follow divide and conquer rule learning even when small training data is used.

Table 3. Number of rules and average number of rule terms

Data set	C4.5		Prism	
	Count(rules)	Avg(Terms)	Count(rules)	Avg(Terms)
anneal	35	3.02	**18**	**2.28**
breast-w	14	4.33	**12**	**1.42**
cmc	157	6.23	**36**	**4.67**
credit-a	30	3.9	**12**	**1.0**
credit-g	103	7.3	**20**	**1.8**

In addition, it is also helpful to employ pruning algorithms, such as reduced error pruning [27] and Jmid-pruning [16], to simplify rule sets [18, 19, 20]. In this way, some redundant or irrelevant information is removed and thus the interpretability is improved. Empirical results, which are reported in [24] and illustrated in Table 4, indicate that applying Jmid-pruning to Prism usually reduces the number of rules and the average number of terms per rule and thus helps improve the interpretability of a rule set.

Table 4. Number of rules and average number of rule terms by Prism [24]

Dataset	Prism without pruning		Prism with Jmid-pruning	
	Count(rules)	Avg(terms)	Count(rules)	Avg(terms)
cmc	36	4.67	**25**	**4.48**
vote	25	6.28	**15**	**5.13**
kr-vs-kp	63	5.84	**21**	**5.52**
ecoli	24	**1.88**	**17**	1.94
anneal.ORIG	16	**1.56**	**12**	3.67
audiology	48	3.60	**38**	**2.79**
car	**2**	**1.0**	3	2.0
optdigits	431	7.46	**197**	**6.53**
glass	26	**2.85**	**24**	3.29
lymph	**10**	1.3	**10**	**1.11**
yeast	37	1.68	**20**	**1.5**
shuttle	30	3.87	**12**	**1.0**
analcatdataasbestos	**5**	1.6	**5**	**1.4**
irish	**10**	1.5	11	**1.27**
breast-cancer	**11**	1.09	**11**	**1.0**

In sentiment analysis, Support Vector Machines and Naïve Bayes are two popular methods used for prediction of sentiments. However, as mentioned in Section 2, computational models generated by using these two methods are generally less transparent, which again indicates the necessity of improvement of model transparency through scaling up algorithms.

Reduction of model complexity is also necessary in sentiment analysis. This is because sentiment data is typically in the form of text which is unstructured unlike the data sets retrieved from machine learning repositories such as UCI [26]. As introduced in [28, 29], features in sentiment data usually include unigrams, bigrams and trigrams or various combinations between them, which indicates that such data is generally of high complexity. This again shows that it is highly necessary to involve effective management of model complexity through scaling up algorithms. For example, if rule learning methods are used to build models for sentiment analysis, proper selection of rule generation approaches and employing pruning algorithms are generally helpful for reducing model complexity.

As mentioned in [28], general preprocessing techniques partially aim to remove redundancy that exists in data. However, it is currently still difficult to guarantee that such redundancy can be effectively eliminated from data. When the preprocessing techniques fail to detect and remove some redundancy from data, it would be very likely to result in the occurrence of model redundancy. On the basis of the above description, it is worth to manage the effective reduction of model redundancy on algorithms side through scaling up algorithms rather than to rely only on data preprocessing techniques. The proper selection of rule generation approaches and employing pruning algorithms mentioned above are also helpful here.

5.2 Scaling Down Data

As mentioned in Section 3, the size of data may also affect the model complexity. In other words, if a data set has a large number of attributes with various values and instances, the generated model is very likely to be complex.

The dimensionality issue can be resolved by using feature selection techniques such as entropy [30] and information gain [3], both of which are based on information theory pre-measuring uncertainty present in the data. In other words, the aim is to remove those irrelevant attributes and thus make a model simpler. In addition, the issue can also be resolved through feature extraction methods, such as Principal Component Analysis (PCA) [31] and Linear Discriminant Analysis (LDA) [32].

When a dataset contains a large number of instances, it is usually required to take advantage of sampling methods to choose the most representative instances. Some popular methods comprise simple random sampling [33], probabilistic sampling [34] and cluster sampling [35].

Besides, it is also necessary to remove attribute values due to the presence of irrelevant attribute values. For example, in a rule based method, an attribute-value pair may be never involved in any rules as a rule term. In this case, the value of this attribute can be judged irrelevant and thus removed. In some cases, it is also necessary to merge some values for an attribute in order to reduce the attribute complexity especially when the attribute is continuous with a large interval. There are some ways to deal with continuous attributes such as ChiMerge [36] and use of fuzzy linguistic terms [5].

In the context of rule based systems, as analyzed in Section 3.2, dimensionality reduction can effectively reduce the average number of rule terms per rule. This is because each single rule can have only up to n rule terms, where n is the data dimensionality. As also analyzed in Section 3.2, reduction of the number of values for each input attribute can effectively reduce the number of rules. For example, three attribute a, b, c have 2, 3 and 4 values respectively. In this case, the number of first order rules (with one rule term) is 2+3+4=9; the number of second order rules (with two rule terms) is 2×3+2×4+3×4=26; the number of third order rules (with three rule terms) is 2×3×4=24. In addition, effective sampling of data can reduce the number of instances and may also reduce the number of values for some or all input attributes. Therefore, dimensionality reduction, data sampling and reduction of the number of attribute values are all generally effective towards reduction of model complexity.

On the basis of above description, the complexity could be reduced through dimensionality reduction, data sampling and removal or merging of attribute values. In sentiment analysis, such scaling down data is even more necessary due to the variety of features in sentiment data as mentioned in Section 5.1. In addition, as reported in [37], rule learning methods have been popularly used as sentiment classification techniques [38, 39, 40], which indicates to a larger extent the necessity of scaling down data through the ways introduced above.

5.3 Selection of Model Representation

As mentioned in Section 3, a change of model representation would usually result in the change of model interpretability. As also introduced, rule based models could be represented in different forms such as decision tree and linear list. These two representations usually have redundancy issues. For example, a decision tree may have the replicated subtree problem and a linear list may have the attribute appear in different rules on a repetitive basis. This kind of problem may be resolved by converting to a rule based network representation as illustrated in Fig.3.

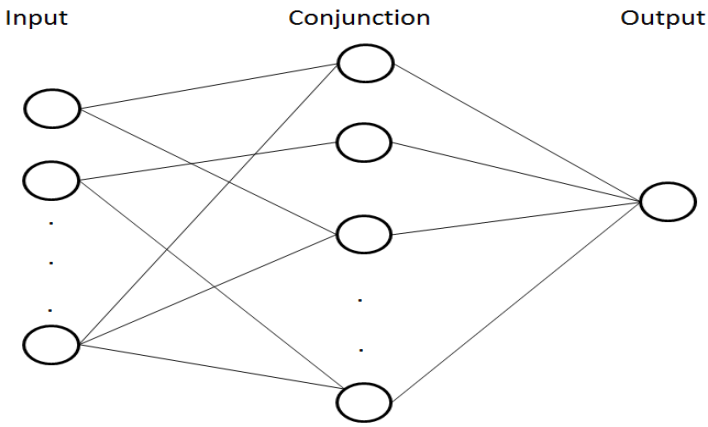

Fig. 3. Deterministic rule based network [22]

In general, this is a three layer network. In the first layer, each node represents an input attribute and this layer is referred to as input layer. In the middle layer, each node represents a rule to make the conjunction among inputs and provide outputs for the node in the last layer and thus the middle layer is referred to as conjunction layer. The only node in the last layer represents the class output and thus this layer is referred to as output layer. In addition, the nodes in the input layer usually have connections to other nodes in the conjunction layer. Each of the connections represents a condition judgment which is explained further using specific examples. However, a node in the input layer may not necessarily have connections to other nodes in the conjunction layer. This is due to a special case that an attribute may be totally irrelevant to making a classification. In other words, this attribute is not involved in any rules in the form of rule terms.

In terms of model redundancy, this network based representation solves this problem. As illustrated in Fig.3, each of the attributes is located in the input layer and only appears once with some branches connected to the nodes, each of which represents a rule, in the conjunction layer. This totally removes the repetitive ap-

pearance of the same attribute in different rules, which is likely to arise with decision tree and linear list representations. In the context of programming, a selection structure could be represented in the form of 'if-else' statement or 'switch-case'. The network based representation is philosophically similar to the 'switch-case' whereas the linear list is similar to the 'if-else' statement. Due to the requirement that the program needs to have a good readability, 'switch-case' is used instead of 'if-else' statement in many cases. In 'switch-case', a variable only appears once as an input parameter of a switch function and each of the possible values of the variable is involved in a particular case respectively. This removes the repetitive judgment with regard to the value of the same variable in comparison with the use of the 'if-else' statement.

In addition, the network based representation has another advantage that the network could be represented in colors to highlight important information. For example, there is a rule set given below:

Rule 1: If x_1=0 and x_2=0 Then class= 0;
Rule 2: If x_1=0 and x_2=1 Then class= 0;
Rule 3: If x_1=1 and x_2=0 Then class= 0;
Rule 2: If x_1=1 and x_2=1 Then class= 1;

The corresponding representation in network form is illustrated in Fig.4. In this diagram, the rules that are firing and the conditions that are satisfied for each of the rules can be explicitly highlighted by using the two colors namely, green and red. The former color generally means the condition is met and it is permitted to pass through whereas the latter color means the opposite case.

On the other hand, the network based representation can also be generalized to different logics such as deterministic, probabilistic and fuzzy logic, see Fig.5.

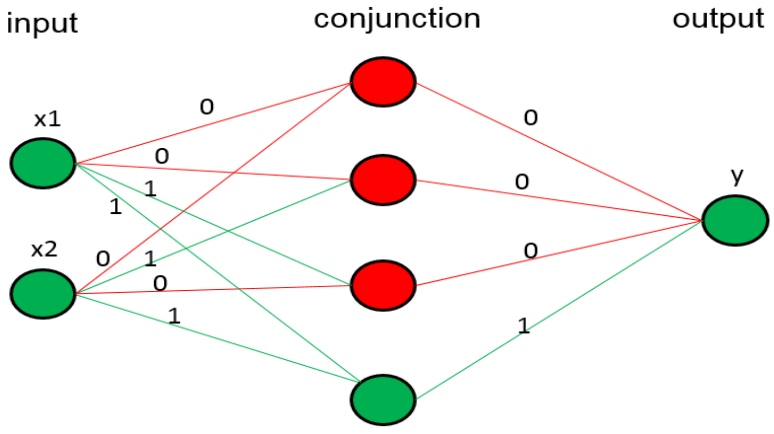

Fig. 4. Deterministic rule based network example [22]

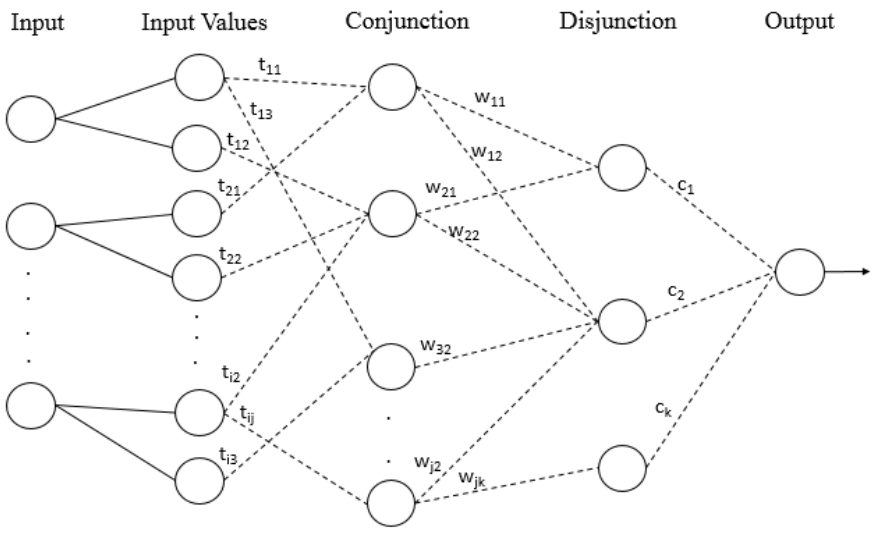

Fig. 5. Unified Rule Based Network

In this network topology, the modifications are made to the one illustrated in Fig.3 by adding two new layers called input values and disjunction respectively, and assigning a weight to each of the connections between nodes. In the input values layer, each node represents one of the values of an input attribute. For those values of the same input attribute, they cannot commonly appear in the same rule as rule terms. Therefore, for these nodes in the input values layer, if they are connected to the same node in the input layer, then they cannot be commonly connected to the same node in the conjunction layer. In the disjunction layer, each node represents a class label. The topology allows representing inconsistent rules, which means that the same rule antecedent could be mapped to different classes (consequents). For example, the first node in the conjunction layer is mapped to both the first and the second node in the disjunction layer as illustrated in Fig.5. With regard to the weights assigned to the connections between nodes, they would represent the truth values if the computation is based on deterministic or fuzzy logic. The truth value would be crisp (0 or 1) for deterministic logic whereas it would be continuous (between 0 and 1) for probabilistic and fuzzy logic.

For fuzzy computational models, it is required not only to interpret the way that the inputs determine the outputs but also the degree of likelihood. The nature of the generalized network representation would fulfil the above requirement by simply assigning a weight to each of the connections between nodes as illustrated in Fig.5. This is because each of the connections is only involved in one rule in this representation. In contrast, in a decision tree representation, a connection may be involved in more rules with the need that different weights are assigned for the involvements in different rules. In addition, for a linear list representation, if each of the rule terms is assigned a weight, it is likely to make the rules less readable.

On the other hand, the network topology illustrated in Fig.5 can interpret well the fuzzy truth values derived in different stages of fuzzy simulation namely, fuzzification, application, implication, aggregation and defuzzification.

In particular, each of the weights assigned to each of the connections between the second and third layers indicates the fuzzy membership degree of each input value derived in the fuzzification stage.

In the application stage, the firing strength of a fuzzy rule is computed through conjunction of all the fuzzy membership degrees derived in the first stage. In other words, the minimum among these fuzzy membership degrees is taken as the firing strength of the fuzzy rule.

Each of the weights assigned to each of the connections between the third and fourth layers indicates the fuzzy membership degrees of given rule consequents, each of which is derived in the implication stage through the conjunction of the rule firing strength and the fuzzy membership degree of the corresponding output value.

Each of the weights assigned to each of the connections between the last two layers indicates the overall fuzzy membership degree of each output value, which is derived through the disjunction of the fuzzy membership degrees for each output value from a particular fuzzy rule.

The final output illustrated in Fig.5 is derived in the defuzzification stage through weighted majority voting for fuzzy classification or weighted averaging for fuzzy regression.

On the basis of above description in this section, the model interpretability can be improved by selecting a model representation with high transparency and low redundancy. In sentiment analysis, both probabilistic and fuzzy logic have become increasingly popular for uncertainty handling in prediction of sentiments [41, 42, 43]. From this point of view, advancing interpretability of such computational models has become highly significant, which again shows the necessity of proper selection of a model representation.

5.4 Assessment of Cognitive Capability

As mentioned in Section 3, due to the difference in level of expertise and personal preferences, the same model representation may show different levels of comprehensibility for different people. For example, people who do not have a good background in mathematics may not like to read information in mathematical notations. In addition, people in social sciences may not understand technical diagrams used in engineering fields. On the basis of above description, cognitive capability needs to be assessed to make the knowledge extracted from computational models more interpretable to people in different domains, due to the fact that sentiment analysis is actually involved in different domains. This can be resolved by using expert knowledge in cognitive psychology and human-machine engineering, or following machine learning approaches to predict the capability as mentioned in Section 4.

6 Conclusion

As mentioned in Section 2, many studies focused on the improvement of accuracy of sentiment predictions using computational intelligence approaches. However, there are very few studies focused on addressing the issues of interpretability of sentiment analysis systems. This paper justifies the significance of interpretability of computational models, which are used as sentiment analysis systems, and identifies some significant impact factors for their interpretability. According to each of the impact factors, some criteria are proposed for the evaluation of interpretability using computational intelligence methodologies. This paper also provides recommendations towards improvements in terms of interpretability through the use of computational intelligence approaches. In particular, scaling up algorithms and scaling down data, which can be effectively used for reduction of model overfitting, are considered also capable of improvement of interpretability through reduction of model complexity. In addition, a generalized rule based network topology is proposed for the improvement of interpretability through reduction of model redundancy. These proposed solutions have also been investigated theoretically and empirically with expected outcomes. The aspects relating to human characteristics introduced in the paper will be further investigated in the future research. In addition, in sentiment analysis, data preprocessing is an important stage in order to have the data structured and provided with relevant features [13, 28]. Therefore, in what way to provide the structural data with an acceptable number of relevant features would be a significant impact factor for interpretability of sentiment analysis systems. In other words, it is significant to determine what elements need to be kept or removed as well as ways to use n-grams [28] through the selection of features such as unigrams, bigrams and trigrams or combinations between them. The above description indicates the needs of further research in computational intelligence towards improvement of model interpretability without loss of accuracy for sentiment analysis.

References

[1] J. Schlager, "Systems engineering: key to modern development," *IRE Transactions EM,* vol. 3, no. 3, pp. 64-66, 1956.

[2] A. D. Hall, A Methodology for Systems Engineering, Van Nostrand Reinhold, 1962.

[3] T. M. Mitchell, Machine Learning, McGraw Hill, 1997.

[4] H. Liu, A. Gegov and F. Stahl, "Categorization and Construction of Rule Based Systems," in *15th Inter-national Conference on Engineering Applications of Neural Networks*, Sofia, Bulgaria, 2014.

[5] T. J. Ross, Fuzzy Logic with Engineering Applications, West Sussex: John Wiley & Sons Ltd, 2004.

[6] S. G. Simpson, Mathematical Logic, PA, 2013.

[7] P.-N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, New Jersey: Pearson Education, 2006.

[8] C. M. Higgins, "Classification and Approximation with Rule Based Networks," Pasadena, California, 1993.

[9] A. M. Uttley, "The Design of Conditional Probability Computers," *Information and control,* vol. 2, pp. 1-24, 1959.

[10] I. Kononenko, "Bayesain Neual Networks," *Biologi-cal Cybernetics,* vol. 61, pp. 361-370, 1989.

[11] F. Rosenblatt, Principles of Neurodynamics: Perceptron and the Theory of Brain Mechanisms, Washington, DC: Spartan Books, 1962.

[12] O. Ekeberg and A. Lansner, "Automatic generation of internal representations in a probabilistic artificial neural network," in *Proceedings of the First European Conference on Neural Networks*, 1988.

[13] N. Altrabsheh, M. Cocea and S. Fallahkhair, "Sentiment analysis: towards a tool for analysing real-time students feedback," in *2014 IEEE 26th international conference on tools with artificial intelligence*, Limassol, Cyprus, 2014.

[14] B. Liu, Sentiment Analysis and Opinion Mining, Morgan & Claypool Publishers, 2012.

[15] F. Stahl and I. Jordanov, "An overview of use of neural networks for data mining tasks," *WIREs: Data Mining and Knowledge Discovery,* pp. 193-208, 2012.

[16] H. Liu, A. Gegov and F. Stahl, "J-measure Based Hybrid Pruning for Complexity Reduction in Classification Rules," *WSEAS Transactions on Systems,* vol. 12, no. 9, pp. 433-446, 2013.

[17] H. Liu, A. Gegov and F. Stahl, "Unified Framework for Construction of Rule Based Classification Systems," in *Inforamtion Granularity, Big Data and Computational Intelligence*, vol. 8, W. Pedrycz and S. Chen, Eds., Springer, 2015, pp. 209-230.

[18] R. Quinlan, C4.5: programs for machine learning, Morgan Kaufman, 1993.

[19] J. Fürnkranz, "Separate-and-Conquer Rule Learning," *Artificial Intelligence Review,* vol. 13, pp. 3-54, 1999.

[20] J. Cendrowska, "PRISM: an algorithm for inducing modular rules," *International Journal of Man-Machine Studies,* vol. 27, p. 349–370, 1987.

[21] X. Deng, "A Covering-based Algorithm for Classification: PRISM," Regina, 2012.

[22] H. Liu, A. Gegov and M. Cocea, "Network Based Rule Representation for Knowledge Discovery and Predictive Modelling," in *IEEE International Conference on Fuzzy Systems*, Istanbul, 2015.

[23] I. Rish, "An Empirical Study of the Naïve Bayes Classifier," *IJCAI 2001 workshop on empirical methods in artificial intelligence,* vol. 3, no. 22, pp. 41-46, 2001.

[24] H. Liu, A. Gegov and M. Cocea, Rule Based Systems for Big Data: A Machine Learning Approach, vol. 13, Berlin: Springer, 2016.

[25] R. Quinlan, "Discovering rules from large collections of examples: a case study," in *Expert Systems in the Micro-Electronic Age*, D. Michie, Ed., Edinburgh, Edinburgh, 1979, pp. 168-201.

[26] M. Lichman, "Machine Learning Repository," University of California, School of Information and Computer Science, 2013. [Online]. Available: http://archive.ics.uci.edu/ml.

[27] T. Elomaa and M. Kääriäinen, "An analysis of reduced error pruning," *Journal of Artificial Intelligence Research ,* vol. 15, no. 1, pp. 163-187, 2001 .

[28] N. Altrabsheh, M. Cocea and S. Fallahkhair, "Learning sentiment from students' feedback for real-time interventions in classrooms," in *Adaptive and intelligent systems:Third International Conference, ICAIS 2014*, Bournemouth, 2014.

[29] O. Kummer and J. Savoy, "Feature Selection in Sentiment Analysis," in *CORIA 2012*, Bordeaux, 2012.

[30] C. E. Shanno, "A mathematical theory of communication," *Bell System Technical Journal,* vol. 27, no. 3, pp. 379-423, 1948.

[31] I. T. Jolliffe, Principal Component Analysis, New York: Springer, 2002.

[32] H. Yu and J. Yang, "A direct LDA algorithm for high-dimensional data — with application to face recognition," *Pattern Recognition,* vol. 34, no. 10, p. 2067–2069, 2001.

[33] D. S. Yates, D. S. Moore and D. S. Starnes, The Practice of Statistics, 3rd ed., Freeman, 2008.

[34] W. E. Deming, "On probability as a basis for action," *The American Statistician,* vol. 29, no. 4, pp. 146-152 , 1975.

[35] S. M. Kerry and J. M. Bland, "The intracluster correlation coefficient in cluster randomisation," *British Medical Journal,* vol. 316, no. 7142, p. 1455–1460, 1998.

[36] R. Kerber, "ChiMerge: Discretization of Numeric Attributes," in *Proceedings of the 10th National Conference on Artificial Intelligence*, California, 1992.

[37] W. Medhat, A. Hassan and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal,* vol. 5, no. 4, p. 1093–1113, 2014.

[38] H. Yi and W. Li, "Document sentiment classification by exploring description model of topical terms," *Computer Speech Language,* vol. 50 , p. 386–403, 2011.

[39] D. D. Lewis and M. Ringuette, "A comparison of two learning algorithms for text categorization," in *Third Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, 1994.

[40] S. Chakrabarti, S. Roy and M. V. Soundalgekar, "Fast and accurate text classification via multiple linear discriminant projections," *The International Journal on Very Large Data Bases,* vol. 12, no. 2, p. 170–185, 2003.

[41] C. Zhao, S. Wang and D. Li, "Fuzzy Sentiment Membership Determining for Sentiment Classification," in *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, Shenzhen, 2014.

[42] S. Nadali, M. Murad and R. Kadir, "Sentiment classification of customer reviews based on fuzzy logic," in *Information Technology (ITSim), 2010 International Symposium in* , Kuala Lumpur, 2010.

[43] F. Colace, M. D. Santo and L. Greco, "A probabilistic approach to Tweets' Sentiment Classification," in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, Geneva, 2013.