A Mini-project Report

on

Scalable Semantic Sentiment Analysis on Large Review Data

Carried out as part of the course Automata and Compiler Design (IT303)

Submitted by

Bhuvan MS (12IT16)
Siddharth Jain (12IT78)
Vinay Rao D (12IT94)
V Sem B.Tech (IT)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

Department of Information Technology

National Institute of Technology Karnataka, Surathkal
November 2014

CERTIFICATE

This is to certify that the project entitled **"Scalable Semantic Sentiment Analysis on Large Review Data"** is a bonafide work carried out as part of the course **Automata and Compiler Design (IT303)**, under my guidance by Bhuvan MS (12IT16), Siddharth Jain (12IT78), Vinay Rao D (12IT94), students of V Sem B.Tech (IT) at the Department of Information Technology, National Institute of Technology Karnataka, Surathkal, during the academic semester V, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, at NITK Surathkal.

Place: NITK Surathkal

Date: Signature of the Instructor

DECLARATION

I hereby declare that the project entitled "**Scalable Semantic Sentiment Analysis on Large Review Data**" submitted as part of the partial course requirements for the course Automata and Compiler Design (IT303) for the award of the degree of Bachelor of Technology in Information Technology at NITK Surathkal during the Jul - Nov 2014 semester has been carried out by us. I declare that the project has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles elsewhere.

Further, I declare that I will not share, re-submit or publish the code, idea, framework and/or any publication that may arise out of this work for academic or profit purposes without obtaining the prior written consent of the course Faculty Mentor and Course Instructor.

Signature of the Students:

Name: Bhuvan M S

Name: Siddharth Jain

Name: Vinay Rao D

Place: NITK Surthkal

Date: 17 Nov 2014

ABSTRACT

The increasing numbers of e-commerce sites, social networking sites are producing large amount data pertaining to reviews of a product, review, restaurant etc. These calls for scalable sentiment analysis on such reviews from social sites, which would help to gather people's sentiment on the product in turn it would help in improving the product in future. Such results can be helpful in decision making with respect to production of future products as well. A keen observation reveals that the text data gathered from any social review site are specific to the context. Hence context specific grammar could be defined as semantics for a particular data. Our research aims to develop a scalable model using Apache Spark where semantic pattern matched features are used to predict the sentiment polarity of reviews and also provide a sentiment score for reviews. The proposed model is intended to be flexible so that it could be applied to any domain by redefining the semantics specific to that domain.

Keywords: Large review data; Scalable model; Context-Specific-Grammar; Semantic pattern matching; Apache Spark; Flexible model; Sentiment Polarity; Sentiment Score.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Internet today contains a huge quantity of textual data, which is growing every day. The text is prevalent data format on the web, since it is easy to generate and publish. In recent years, we became witnesses of a large number of websites that enable users to contribute, modify, and grade the content. Users have an opportunity to express their personal opinion about specific topics. The examples of such web sites include blogs, forums, product review sites, and social networks.

Online reviews have become extremely famous. The times when people ask their neighbors' suggestion before buying a product or watching a movie is obsolete. Most of the people nowadays choose to read reviews before buying any product. All the e-commerce websites like Amazon, Flipkart have review section for every product. They read movie reviews and decide whether to watch it or not.

Sentiment Analysis is the task of identifying whether the opinion expressed in text is positive or negative in general. Sentiment analysis aims to uncover the attitude of the author on a particular topic from the written text. Other terms used to denote this research area include "opinion mining" and "subjectivity detection". It uses natural language processing and machine learning techniques to find statistical and/or linguistic patterns in the text that reveal attitudes. It has gained popularity in recent years due to its immediate applicability in business environment, such as summarizing feedback from the product reviews, discovering collaborative recommendations, or assisting in election campaigns. Hence it is very important to classify the data.

The amount of reviews available online is huge. Yelp is one of the top ratings and review sites with more than 78 million unique visitors a month. There are over 30 million reviews on Yelp alone [1]. Movie review sites like Internet Movie Data Base (IMDB) and Rotten Tomatoes also have millions of reviews. E-commerce sites such as Amazon also have millions of reviews. The following figure shows review data is distributed over different domains covering about 30 million reviews from yelp site.

Fig.1.1: Different domains of Reviewed Business in Every Category on Yelp

We can see that reviews are found in different domains. A domain specific sentiment analysis could be developed instead of general sentiment analysis. These calls for the need of a *flexible* sentiment analysis model which can be applied to any domain by just defining domain specific data. Also considering the size of data as of 2011, i.e. 30 million, we can easily expect it to grow over billions by 2015. Same is the case for other review sites. Hence we need to ensure that the model is *scalable* to support any amount of data.

The focus of our project is to develop a model that is both scalable and flexible. For the model to be flexible, context specific grammar could be defined as semantics for a particular topic. Scalability is achieved through the use of Apache Spark. Apache Spark is a fast and general-purpose cluster computing system. Machine Learning Library (mllib) is available in Apache Spark. With the help of this, semantic pattern matched features are used to predict the sentiment polarity of reviews and also provide a sentiment score for reviews.

2. LITERATURE SURVEY

The internet is a valuable resource for information and reviews on products and service. This information can be used by various service providers to their benefit to improve in areas they lack. But the major problem being faced is that the information is so vast that classifying it is a very hard task.

Lot of efforts has been put in classifying the information using different methods. Machine Learning algorithms are quite popular for this kind of problem. Support Vector Machine has been used to classify each blog post as positive, negative or neutral. The basic lexical model involves the same steps, i.e. preprocessing, tokenizing and giving score to the post. There is a threshold value decided and if score is more than that, then post is considered as positive. If the score is less than threshold, then the post is negative and if equal to threshold value, then it is neutral post. Many variants of this basic model have been used.

For support vector machine, a popular word set was composed for comparison. This set contained most used adjectives and adverbs. Based on this word set and model mentioned above, feature vector was developed. This can be varied in different way. For instance, a lot of feature vectors can be developed considering most frequently occurring words with weight assigned to each word. Instead of giving weights, we can use binary representation also signifying presence or absence of the word.

The results showed that alone stemming is not much helpful. The accuracy was between 50-60 % depending on the approach (variants of basic lexical model).But with application of machine learning, a drastic improvement was seen. Depending on number of features, accuracy varied from 66 to 77% (50-2000 features) [2].

Sentiment analysis has also been considered as text categorization problem where the topic of a given textual document is guessed from a set of the entire possible sentiment label. Using Natural Language Processing (NLP) algorithms, various features from text were extracted and were used with classifier. 3 different classifiers were tested for the best accuracy (Naïve Bayes, kNN, SVM) on IMDB and Twitter data. Naïve Bayes misses some dependencies as it considers all features as independent. kNN was the least efficient of the

three. Naïve Bayes can give results as good as SVM if the considered features are actually almost independent, which was the case here. So while considering sentiment analysis as classification problem for 2 different datasets (IMDB and Twitter), it was found that it's harder to classify twitter data than IMDB data just using word based features and accuracy of 0.8385 can be achieved using SVM-POLY version [3].

The above two ideas were further generalized by using "bag of words" with a machine learning algorithm for classifying text messages based on their semantic meaning. This method was applied to a large movie review dataset, restricting to adjectives and adverbs, taking care of negations occurring and using a threshold as a limit. This skeleton was used with 4 different machine learning methods to see which combo works the best. Four methods were Decision Tree, Naïve Bayes, Maximum Entropy and K means clustering method. Feature extraction and selection is similar to the basic lexical mode with additional model, bag of words. This model considers individual words in sentence as features assuming their conditional independence. Effectively a unigram model, each feature in the vector stands for presence of the word and all the features in the vector constitute a dictionary. This makes the whole process very generalized. This model does not capture relations among the words. As mentioned above, it assumes there conditional independence. To handle this, WordNet [16], which is a lexical database was used to map the relations among different words. Handling negation is a daunting task. One way to do this is by including another feature as" NOT", 0 indicating absence of not and 1 otherwise. This is done at feature level. This is more efficient and less costly than handling this at extraction level. If supervised data was available, then supervised learning algorithms were applied for classifying like Naïve Bayes, Decision Tree or Maximum Entropy. But getting large trained data set is rare, so algorithms like K Means clustering for unsupervised learning can also be used. Comparison was done between the 4 different methods used. Various variants of bag of words like one using frequencies from movie reviews or one using only adjectives and adverbs were also put in use to see which one performs better . Performance of Maximum Entropy was quite low (around 33% when classification by polarity). Naïve Bayes was seen to be most effective (supporting the bag of words conditional independence assumption). In some cases depending on data, maximum entropy and K means were able to perform as well as naïve bayes and decision tree. The analysis was done using 2 movies. For one of the movies, supervised algorithms gave results as good as 70-80% accuracy. Maximum entropy when used with variant of bag of words seemed to perform equally good as naïve bayes. If

classification is done by strict polarity, considering negation as a separate feature vector improvises result but does the opposite when classification is done based on subjectivity. Bag of words improved the results but still faced some problems.

The methods discussed above are either not performing well or are too generalized with no consideration among relations between words. To bring relation into picture, it took introducing another level (WordNet) which can be costly and complex. There is a need of something more context specific and simple without affecting the speed or accuracy of the result.

3. METHODOLOGY

Users all over the world review the products or services they have used on a social review website. A social review site may contain reviews of various products and services. All those reviews belonging to a particular product or type of service can be considered as one domain which may be owned by company. Our approach concentrates on analyzing reviews pertaining to a single domain i.e. Movie Reviews. This is a multi-stage approach to obtain sentiment polarity and a sentiment score for each of the reviews in the domain. For training the model we take labeled data available so that we check the accuracy after predicting using supervised learning algorithms. The architecture of the approach has been depicted as a block diagram in fig.3.1.

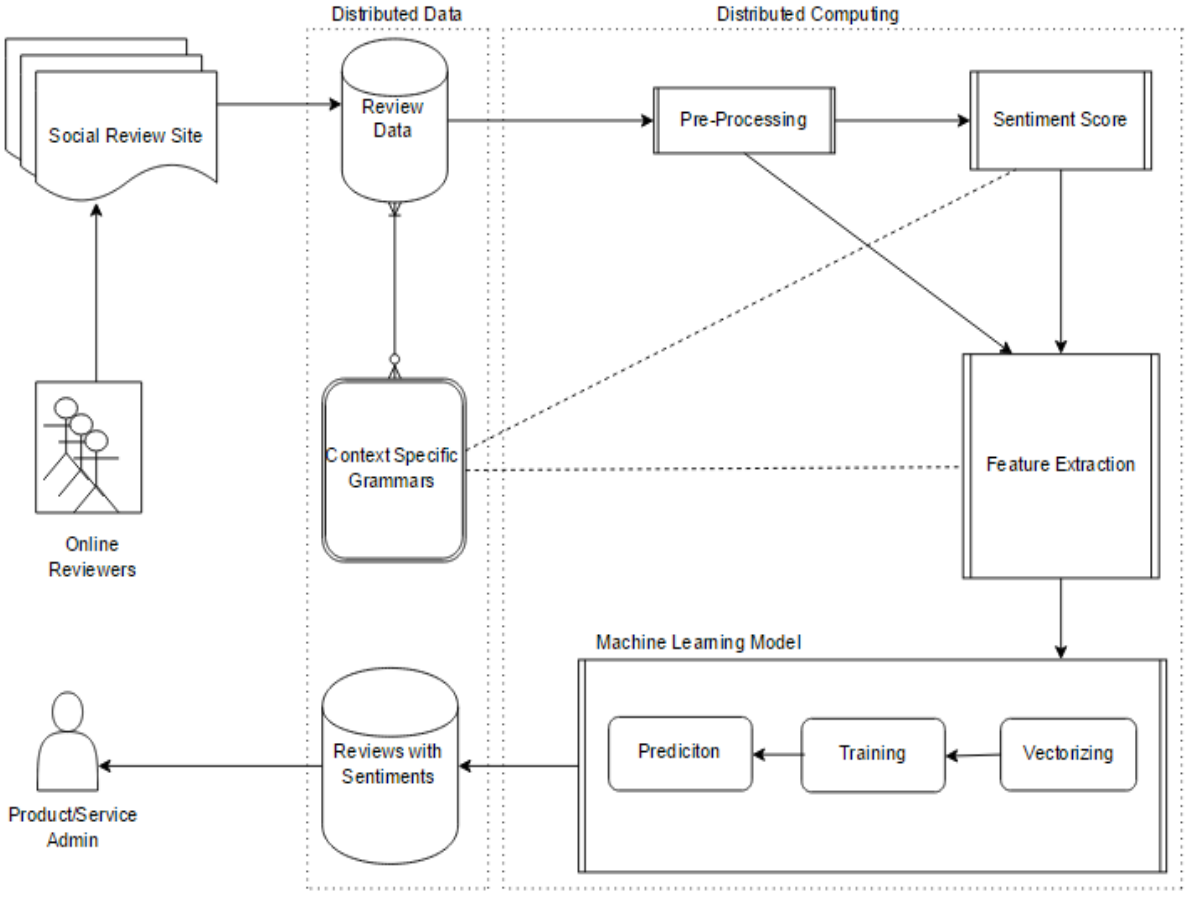

Fig.3.1. Block Diagram of Scalable Semantic Sentiment Analysis model

1. Data aggregation of reviews of specific domain

Movie domain was chosen, and Large Movie Review Dataset v.10 [5] was obtained. This dataset contains 50000 movie reviews which is uniformly distributed to contain 25000 positive and 25000 negative reviews about movies which was collected from IMDB [Internet Movie Data Base]. It is aggregated to a single file with each line containing one review along with its label. To randomize the data the lines are shuffled. Since the data can be very huge, in order to make the model scalable the randomized data is loaded onto distributed memory. The data is loaded using Apache Spark [6] which is built on Hadoop Distributed File System (HDFS) to maintain distributed data which can be accessed through Resilient Distributed Dataset (RDD) through a spark context. PySpark a python API for Spark has been used for the implementation.

2. Defining Context Specific Grammars

Since the movie domain has been chosen, a set of positive and negative grammars need to be defined specific to context of movies. This can be done by identifying the general structure of positive and negative reviews respectively from the labeled review data. These grammars should be independent of one another; otherwise the results may get skewed with overlapping matches against grammars. This can also include emoticons to catch the sentiment polarity associated with emoticons used in the reviews. This can be implementing by defining Regular Expressions for grammars. Some examples have been shown in fig.3.2. This approach is better than simply defining a bag of words because the grammar defined the semantic structure of the reviews of the selected context, hence would give better accuracy. While defining the regular expressions for the grammar it should be noted that the word stems have to be described instead of the whole word as it should match all morphological forms of the word. These are the generic grammar based on which the machine will learn how many matches of which grammar would imply a polarity of the sentiment associated with review. The model is flexible because it can be used for data pertaining to any domain, by just defining new grammars specific to that domain.

Positive	1. (I | is | was | it)?(.*) (ador|admir|breathtak|color|convinc|effortless|enjoy|exceed|excel|extraordin|fascin|flawless|best|impress|enthrall|lovab) 2. (good| great| best| fun| weird)(.*)(movie | film | actor | actr| direct| ride| music | song) 3. (:\)|:p|:P|:d|:D|;\)|B\)|B||O:\)|:*|<3|^_^|\(Y\)|\(y\)) 4. it was?(.*) (good|awesome|splendid|spellbinding|mindblowing|fantastic|best|fabulous|breathtaking|captivat|comsistent|cool|catchy|deserv|effec|elegant|enchant|entertain|entic|motivat|praiseworthy|rejoic|mesmeriz|tantaliz|vibrant|ecstat|fascinat|impress|genuinity|satisfy|spectacular|wow|worth|wonderful)
Negative	1. I (.*) (hate|dislike|bumm|depress|fool|angry|belliger|restless|idiot|violat|awful| avers|blame|bor|bonkers|hurt|disgust|dismay|displeas|disquiet|dissapoint|offen|punish|destroy|ruin|sad|tortur|traumatiz|vomit|filth|fault|frustrat|hopeless|mad) 2. (Cast(ing?)|actors?|acting) (.*) (suck|bad|disappoint|off putting|banal|depress|lunatic|awful| mist|naive|obnoxious|overkill| piti|horrendous|shame|stupid|tortur| cliche|degrade|far-fetched|fault|irksome|mess| mis|blank|obscure|blunder|peculiar|screw up|senseless) 3. it was? not|doesn't|lacked (.*) (good|awesome|splendid|spellbinding|mindblowing|fantastic|best|fabulous|breathtaking|captivat|comsistent|cool|catchy|deserv|effec|elegant|enchant|entertain|entic|motivat|praiseworthy|rejoic|mesmeriz|tantaliz|vibrant|ecstat|fascinat|impress|genuinity|satisfy|spectacular|wow|worth|wonderful) 4. (:\(|:'\(|;\(|:O|:o|>:\(|:/|o.O|>:O|>:o)

Fig.3.2. Examples of Context Specific Grammars for movie review data

3. *Preprocessing*

Each review item has to be broken into sentences before trying to match against any grammar. This is crucial because a grammar should not be matched across sentences as they define the semantic structure of a single sentence and not a paragraph. Stop words should not be removed as certain verbs, auxiliary verbs and articles can be an integral part of the grammar. Stemming of words is not necessary as the words in the grammar contain only word stems, so all morphological forms of words in the review get matched along with the grammar.

4. *Getting Sentiment Score*

Each preprocessed review is given a sentiment score. It is calculated as:

Sentiment Score = num_of_positive_matches / num_of_negative_matches

Where num_of_positive_matches is the number of matches of review against positive grammars, and num_of_negative_matches is the number of matches of review against

negative grammars. Here to avoid divide by zero exception, the number of negative matches is set to 1 in case if it is 0. Hence this score denotes the positivity and negativity of a review; if it is 1, then it is neutral as there are equal numbers of positive and negative matches, likewise if it is lesser than 1 it denotes negativity and greater than 1 denotes positivity.

A naïve approach could be used where reviews are classified based the sentiment score itself, but it is found to be very ineffective compared to machine learning approach.

5. Feature Extraction and Selection

In order to make the machine learn the characteristics of a review so that it can classify it accurately as positive or negative, a robust feature extraction and selection mechanism is essential. Each one of the grammars promote for one feature. The number of matches of the review against a grammar is taken as a feature value represented by that grammar. Hence number of features is equal to the number of the positive and negative grammars defined. Additionally, Sentiment Score that was calculated in the previous step is used as a complex feature. Sentiment Score gives the notion of the sentiment scale considering those positive and negative grammars, but the other features are just number of matches against a grammar, the machine cannot distinguish them as positive or negative grammar. These grammars are described to be positive or negative by the person who defined the grammar. A negative grammar defined in fact may be a characteristic of a positive review. This can happen due to the use of various figures of speech, sarcasm and ironic phrases in a review. These cases are also taken care of because, the machine doesn't distinguish a grammar as positive or negative, but classifies the review as positive or negative looking at its labels and feature values which is the number of matches with positive and negative grammar. There is a possibility that majority of the positive labeled reviews may get a good match for a negative grammar, so after observing this scenario, the system will classify a review as positive even if it gets a match for that negative grammar. The Sentiment Score will compensate this effect as it provides the float value as a ratio of number of matches considering the positive and negative grammars separately. The above hypothesis would be tested for accuracy after evaluating the predictions by the machine learning algorithms which take the features extracted by this method as input.

As number of features i.e. number of grammars, its effect on the accuracy of results should be tested. It is found that as number of features increases, the accuracy gradually increases. This is also supported by the result of the research by Michelle Annett and Grzegorz Kondrak [2].

The feature extraction process is computationally expensive, as it has to accommodate large data. Therefore, feature values for each review have to be extracted in parallel. Apache Spark framework is a good choice where map-reduce programming paradigm is used to apply functions on each data item. Extracting number of matches of the review against all grammar should be defined as a mapper function and has to be called on the RDD of randomized review data which is loaded into the Spark Context. Spark distributes the work, such that each slave machine works on extracting the feature values for a particular review in parallel. If there are as many slave nodes as number of reviews then features for all reviews can be extracted in constant time as each slave would only work on a single review. Spark which has Hadoop underneath it is a robust framework. It takes care of synchronization and resource management. The model is highly scalable, as computation speed increases with increase in cluster size. For testing the approach, standalone implementation of spark is used on Intel i7 processor, which utilizes multiple cores of the processor for its parallel execution.

6. Machine Learning Model

Supervised Machine Learning algorithms enable the machine to learn the characteristics of the review based on the training data and then predict the label of a new unlabeled review. The proposed Scalable Machine learning model is composed of three stages.

a. Vectorizing

The extracted features values needed to be in vectorized form as an RDD of Labeled Points in order to feed it to Spark. A labeled point is an ordered pair or a list of feature values mapped to a class label. This vectorizing function can be applied as a mapper function where feature values of each of the reviews are converted to vectorized form by different slave nodes in parallel.

b. Training

Spark provides a machine learning library (mllib) which parallel implementations of various machine learning algorithms. Since this is targeted at Big Data, Stochastic Gradient Descent (SGD) [7] is used in Spark instead of Gradient Descent (GD) [8] to reduce the model training time. Accuracy with SGD needs to be compared with results of accuracy with GD by testing with WEKA [12] tool which uses GD in its machine learning algorithms. We need to ensure that we don't lose accuracy by using SGD.

The vectorized data need to be split into training data and testing data. The optimal split ratio has to be decided by experimenting with various ratios. This can be done easily using the WEKA tool, and the split ratio which results in good accuracy should be used. We found that 90% training data 10% test data is reasonable with good accuracy in general, and hence it is suggested to use this split ratio.

This is a binary classification problem; there are various machine learning algorithms for binary classification, but we need to use the optimal algorithms to get best accuracy. WEKA supports various classifiers, which can be tested on our data, and the one which provides best accuracy should be used. Our experiments provided good accuracies with Logistic Regression [10] and Support Vector Machines (SVM) [11]. The Apache Spark version 1.1.0 supports Logistic Regression, Linear SVM with SGD, hence the these machine learning models are trained on the vectored data.

c. Prediction

The trained models are used to predict the class labels i.e. Sentiment Polarity of the reviews in the testing data set. Predicted values can be compared with the original class labels to determine the accuracy of the results. Accuracy maybe calculated with Mean Absolute Error and Root Mean Squared Error [9]. This can be implemented in Spark in the same way as previously described i.e. by using mapper function, for parallel execution. The algorithm which provides best accuracy should be used for real applications. Our experiments suggest that Logistic Regression is best suited giving higher accuracies compared to other algorithms.

7. Semi Supervised Approach

In many scenarios, it is difficult to obtain the training data for reviews. Hence a semi-supervised approach can be used. This uses the naïve approach described in the section 3.D where sentiment score is used to determine the sentiment polarity. Henceforth this can be used as training data, to feed into machine learning model.

The complete model can be implemented on distributed framework like Spark, making this model highly scalable to support large review datasets. The model is flexible enough to accommodate data from different domains, with a small overhead of redefining context specific grammars specific to that domain.

4. RESULTS AND DISCUSSION

Evaluation of Sentiment Analysis models as per Wikipedia [13], is in principle how well it agrees with human judgments. A 70% accurate program is considered to be same as human, because sentiment perceptions of humans vary from one person to another. Even if the program is 100% right humans would still disagree with it about 20%. It is legitimate to provide a sentiment scale rather than a sentiment polarity to describe the sentiments of a text. Our model provides a sentiment score along with the sentiment polarity to describe the sentiments in a review. The proposed model was implemented on Apache Spark version 1.1.0, hence making the model scalable. WEKA version 3.7.11 was used for determining the best suited machine learning algorithm suited for the model. Henceforth, the selected machine learning algorithms with optimal parameters was implemented on Spark.

1. *Determining Best Suited Machine Learning Algorithm*

Getting the sentiment polarity just based on the sentiment score without learning model resulted in accuracy of around 55%. In order to achieve higher accuracies we experiment with various machine learning techniques. The vectorized data was fed into various binary classifying machine learning algorithms (Naïve Bayes [14], SVM, Logistic Regression, SMO [15]). This experimentation was carried out in WEKA. In each case Accuracy (Mean Absolute Error) and Root Mean Squared Error [RMSE] was calculated as explained in the previous section.

	Accuracy	RMSE
Naïve Bayes	60.44	0.4961
SVM	63.58	0.6035
Logistic	63.38	0.4708
SMO	62.54	0.612

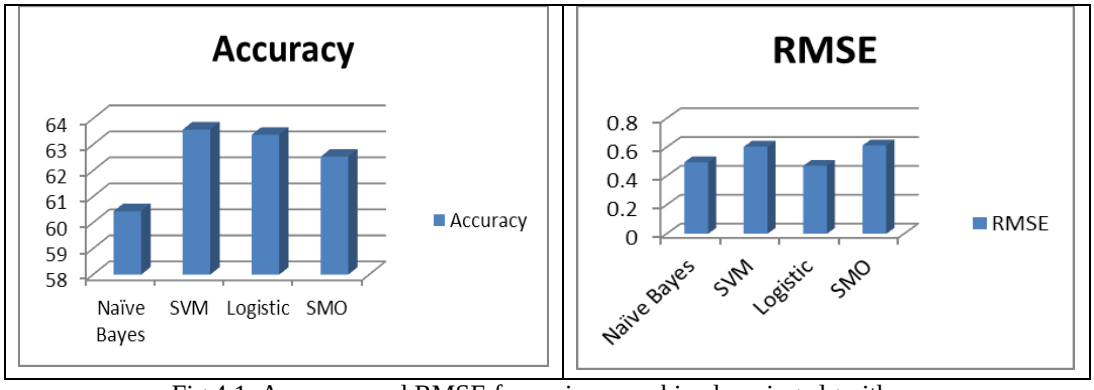

Fig.4.1. Accuracy and RMSE for various machine learning algorithms

From the above results we can observe that Logistic Regression and SVM have good accuracies. But we can see that Logistic Regression has the least RMSE value. Hence we can consider SVM and Logistic regression to be well suited.

2. *Determining Best Train/Test Split Ratio of Dataset*

The randomized review dataset has to be split into training set and testing set for feeding into machine learning model. This experiment aims to find the best split ratio.

Logistic	50	70	90
Accuracy	63.468	63.34	63.38
RMSE	0.4727	0.4727	0.4708

This is the variation of accuracy and RMSE for different split ratios used against Logistic Regression in WEKA. As we can observe, there is very less variations in the accuracy and RMSE with different split ratios. But 90% split ratio with 90% training data and 10% testing data can be considered optimal, as it has least RMSE and a good accuracy as well.

3. *Determining the variation of Accuracy with SGD and GD*

From the above results we can conclude that SVM and Logistic are Optimal. The variation of accuracy and RMSE when GD is implementation in WEKA versus SGD implementation in Apache Spark need is tabulated below. SGD is faster but we need to check if it gives accuracies in par with GD.

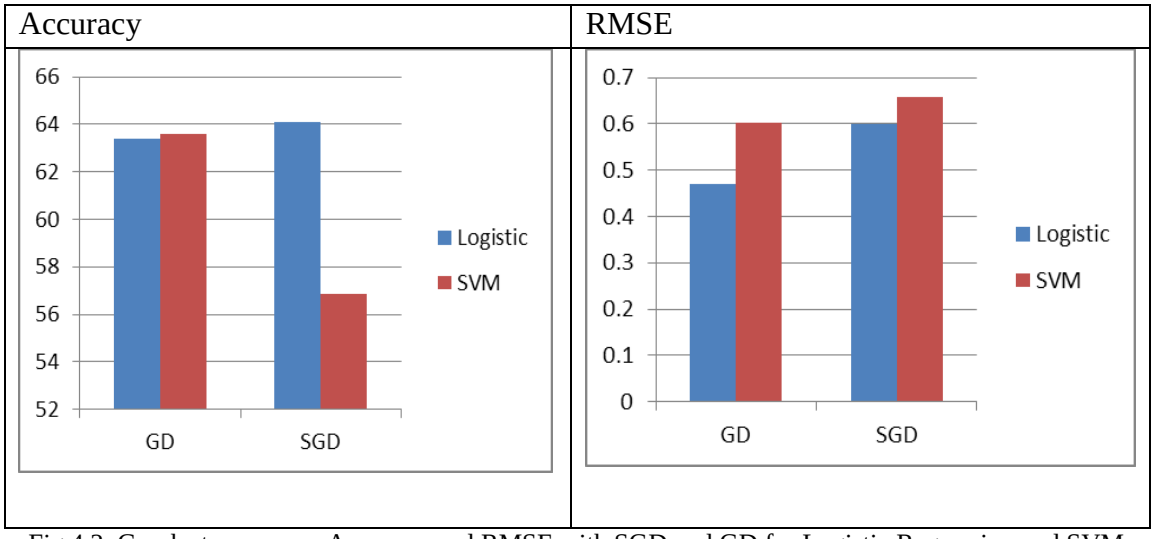

Fig.4.2. Graphs to compare Accuracy and RMSE with SGD and GD for Logistic Regression and SVM

		GD	SGD
Accuracy	Logistic	63.38	64.12
	SVM	63.58	56.84
RMSE	Logistic	0.4708	0.5989
	SVM	0.6035	0.6569

With these results we can observe that SGD is giving good in fact better accuracies compared to GD in case in Logistic but is not doing that good with SVM. There is a slight rise the RMSE which is in allowable range. As Spark implementation is essential for the scalability of the model, Logistic Regression is very well suited with high accuracies with SGD as well, and with an allowable RMSE.

4. Spark Results

Logistic Regression and Linear SVM with SGD was carried out considering 90% training set and 10% testing set. Accuracies, RMSE, Training Time and Confusion Matrix have been tabulated below.

ML Algorithm	Accuracy	RMSE	Training Time
Logistic Regression	64.12%	0.5989	12.47 sec
Linear SVM	56.84%	0.6569	7.94 sec

Logistic Regression Confusion Matrix	Linear SVM Confusion Matrix
64.12% | pos | neg --- pos | 1868 | 703 neg | 1091 | 1338	56.84% | pos | neg --- pos | 2454 | 117 neg | 2041 | 388

The above results clearly suggest that Logistic regression is the best algorithm for the model with best accuracy even with SGD when implemented on Spark with 90% training set and 10% testing set. Being accurate it also promises high scalability to accommodate Big Data.

5. CONCLUSION AND FUTURE WORK

Large review data available at social web sites needs a Scalable Sentiment Analysis Model which is flexible enough to be applied to reviews pertaining to various domains. The popular method of using bag of words would not be sufficient as it can be specific to a domain. On the other hand, defining Context Specific Grammars for a domain would be a novel idea as it can address the domain by describing more specific structure of reviews. This also makes the model scalable as the same model can be used for different domain by just defining new Context Specific Grammars for that Domain. To make the model more scalable the proposed model is implemented on Apache Spark Framework enabling parallel execution on a reliable cluster.

Extracting and Selecting Features from the context specific grammar and preprocessed reviews is an important step. By conducting various experiments with parameters for the machine learning model, we can conclude that Logistic Regression with SGD implemented on Spark with 90% training set and 10% testing set would be optimal with accuracy in the acceptable range. This provides Sentiment Polarity along a Sentiment Scale for a more clear representation of sentiment in a review. Even if Training data is not available, the suggested semi supervised approach could be followed. The model being Scalable and Flexible makes it practically very applicable for Sentiment Analysis of Large review datasets available on the web.

Future work could be done to experiment with different number of grammar and variations of feature extraction and selection model. New experiment could be done on an unsupervised approach which can be carried out by using K-Means Clustering algorithm to find the break point in the sentiment score at which a review gets classified as positive or negative assuming uniform distributed dataset. The proposed model implicitly assumes equal importance to each grammar. Another idea would be use different weights associated with each of the grammar which denotes its role in describing the level of positivity or negativity of a review which gets matched against the grammar. To find optimal weights Neural Networks algorithms could be used. By conducting above suggested future research, more robust sentiment analysis models can be evolved.

REFERENCES

[1] Introduction to ratings and review sites, available at: http://www.socialquickstarter.com/content/78-introduction_to_ratings_and_review_sites, visited on: November 16, 2014

[2] Annett, Michelle, and Grzegorz Kondrak. "A comparison of sentiment analysis techniques: Polarizing movie blogs." *Advances in artificial intelligence*. Springer Berlin Heidelberg, 2008. 25-35.

[3] Sentiment Analysis of Movie Reviews and Twitter Statuses, Machine Learning Final Project Kfir Bar, available at: http://www.cs.tau.ac.il/~kfirbar/mlproject/project.html, Visited on: December 7, 2014

[4] Yessenov, Kuat, and Saša Misailovic. "Sentiment analysis of movie review comments." *Methodology* (2009): 1-17.

[5]. Maas, Andrew L., et al. "Learning word vectors for sentiment analysis."Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

Large Movie Review Dataset, available at http://ai.stanford.edu/~amaas/data/sentiment/, visited on December 7, 2014.

[6] Apache Spark – Lightning Fast Cluster Computing, available at https://spark.apache.org/, visited on December 7, 2014.

[7] Wikipedia Stochastic Gradient Descent, Available at: http://en.wikipedia.org/wiki/Stochastic_gradient_descent, Visited on: December 7, 2014

[8] Wikipedia Gradient Descent, Available at: http://en.wikipedia.org/wiki/Gradient_descent, Visited on: December 7, 2014

[9] Wikipedia Root-Mean-Squared-Deviation, available at: http://en.wikipedia.org/wiki/Root-mean-square_deviation, visited on December 7, 2014.

[10] Wikipedia Logistic Regression, Available at: http://en.wikipedia.org/wiki/Logistic_regression, Visited on: December 7, 2014

[11] Wikipedia Support Vector Machines, Available at: http://en.wikipedia.org/wiki/Support_vector_machine, Visited on: December 7, 2014

[12] WEKA, Available at: http://www.cs.waikato.ac.nz/ml/weka/, Visited on: December 7, 2014

[13] Wikipedia, Sentiment Analysis, Available at: http://en.wikipedia.org/wiki/Sentiment_analysis, Visited on: December 7, 2014

[14] Wikipedia Naïve Bayes, Available at: http://en.wikipedia.org/wiki/Naive_Bayes_classifier, Visited on: December 8, 2014

[15] Wikipedia Sequential Minimal Optimization (SMO), Available at: http://en.wikipedia.org/wiki/Sequential_minimal_optimization, Visited on: December 8, 2014

[16] WordNet, Available at: http://wordnet.princeton.edu/, Visited on: December 8, 2014