Big Data Sentiment Analysis based on PLSA and its Application

Ebenezer Komla Gavua, Seth Okyere-Dankwa, Gignarta Kasaye Soka

Abstract: ***Posting reviews online has become an increasingly popular way for people to express opinions and sentiments toward products bought or services received. Analyzing the large volume of online reviews available would produce useful actionable knowledge that could be of economic value to vendors and other interested parties. This study conducted a case study in the movie domain, and tackles the problem of mining reviews for predicting product sales performance. Based on an analysis of the complex nature of sentiments, this paper studies Sentiment PLSA (S-PLSA), in which a blog entry is viewed as a document generated by a number of hidden sentiment factors. Training an S-PLSA model on the blog data enables us to obtain a succinct summary of the sentiment information embedded in the blogs. The study then presents SAAR, a sentiment-aware autoregressive model, to utilize the sentiment information captured by S-PLSA for predicting product sales performance. Extensive experiments were conducted on a movie data set. In this study SAAR is compared with alternative models that do not take sentiment information into account; as well as a model with different feature selection methods. Experiments confirm the effectiveness and superiority of the approach studied.***

Keywords: Blog mining, Hadoop, MapReduce.

I. INTRODUCTION

Due to its high popularity, Weblogs (or blogs in short) present a wealth of information that can be very helpful in assessing the general public's sentiments and opinions. This paper focuses on studying the problem of mining sentiment information from blogs and investigates ways to use such information for predicting product sales performance. It has become a common practice for e-commerce websites to provide the venues and facilities for people to publish their reviews, with a prominent example being Amazon www.amazon.com.

This paper investigates on how to predict box office revenues using the sentiment information obtained from blog mentions. The choice of using movies rather than other products in our study is mainly due to data availability, in that the daily box office revenue data are all published on the Web and readily available, unlike other product sales data which are often private to their respective companies due to obvious reasons. Analyzing movie reviews is one of the most challenging tasks in sentiment mining (Liu et al. 2005). Analyzing movie reviews is one of the most challenging tasks in sentiment mining. Models and algorithms developed for box office prediction are expected to be easily adapted to handle other types of products that are subject to online discussions, such as books, Music CDs and electronics. Prior studies on the predictive power of blogs have used the volume of blogs or link structures to predict the trend of product sales, failing to consider the effect of the sentiments present in the blogs (Gruhl et al. 2005). It has been reported that although there seems to exist strong correlation between the blog mentions and sales spikes, using the volume or the link structures alone do not provide satisfactory prediction performance (Gruhl et al. 2005). Indeed, as the study will illustrate with an example, the sentiments expressed in the blogs are more predictive than volumes.

Mining opinions and sentiments from blogs, which is necessary for predicting future product sales, presents unique challenges that can't be easily addressed by conventional text mining methods. Therefore, simply classifying blog reviews as positive and negative, as most current sentiment mining approaches are designed for, does not provide a comprehensive understanding of the sentiments reflected in the blog reviews. In order to model the multifaceted nature of sentiments, this study view the sentiments embedded in the blogs as an outcome of the joint contribution of a number of hidden factors, and studied a novel approach to sentiment mining based on Probabilistic Latent Semantic Analysis (PLSA), which we call Sentiment PLSA (S-PLSA).

Different from the traditional PLSA, S-PLSA focuses on sentiments rather than topics. Therefore, instead of taking a vanilla "bag of words" approach and considering all the words (modulo stop words) present in the blogs, we focus primarily on the words that are sentiment-related (Hofmann, 1999). Studies adopt the appraisal words extracted from the lexicon constructed (Whitelaw et al, 2005). Despite the seemingly lower word coverage (compared to using "bag of words"), decent performance has been reported when using appraisal words in sentiment classification (Whitelaw et al, 2005). In S-PLSA, appraisal words are exploited to compose the feature vectors for blogs, which are then used to infer the hidden sentiment factors. Aside from the S-PLSA model which extracts the sentiments from blogs for predicting future product sales, also the past sale performance of the same product is considered as another important factor in predicting the product's future sales performance. This effect is captured through the use of an autoregressive (AR) model, which has been widely used in many time series analysis problems, including stock price prediction (Enders, 2004). Combining this AR model with sentiment information mined from the blogs, this study proposes a new model for product sales prediction called the Sentiment Aware Autoregressive (SAAR) model. Extensive experiments on the movie dataset has shown that the SAAR model.

Revised Version Manuscript Received on August 16, 2017

Ebenezer Komla Gavua, Koforidua Technical University, Ghana Email: mgavua@yahoo.com

Seth Okyere-Dankwa, Koforidua Technical University, Ghana E-mail: Sokyeredankwa@yahoo.com

Gignarta Kasaye Soka, Omaha, Nebraska, USA, E-mail: kasayegignarta@gmail.com

provides superior predication performance compared to using the AR model alone, confirming our expectation that sentiments play an important role in predicting future sales performance.

II. SENTIMENT MINING

Sentiment mining or (opinion mining) is a type of natural language processing for tracking the mood of the public about a particular product. Opinion mining, which is also called sentiment analysis, involves building a system to collect and examine opinions about the product made in blog posts, comments, reviews or tweets. Automated opinion mining often uses machine learning, a component of artificial intelligence.

Opinion mining can be useful in several ways. If you are in marketing, for example, it can help you judge the success of an ad campaign or new product launch, determine which versions of a product or service are popular and even identify which demographics like or dislike particular features. For example, a review might be broadly positive about a digital camera, but be specifically negative about how heavy it is. Being able to identify this kind of information in a systematic way gives the vendor a much clearer picture of public opinion than surveys or focus groups, because the data is created by the customer.

Most existing work on sentiment mining focuses on determining the semantic orientations of documents. Among them, some of the studies attempt to learn a positive/negative classifier at the document level employ three machine learning approaches (Naive Bayes, Maximum Entropy, and Support Vector Machine) to label the polarity of IMDB movie reviews. In a follow up work, they propose to firstly extract the subjective portion of text with a graph min-cut algorithm, and then feed them into the sentiment classifier (Pang et al, 2002). Instead of applying the straightforward frequency-based bag-of-words feature selection methods (Whitelaw et al, 2005). This defined the concept of "adjectival appraisal groups" headed by an appraising adjective and optionally modified by words like "not" or "very". Each appraisal group is further assigned four types of features: attitude, orientation, graduation, and polarity. They report good classification accuracy using appraisal groups. They also show that when combined with standard "bag-of words" features, the classification accuracy can be further boosted. There are also studies that work at a finer level and use words as the classification subject. They classify words into two groups, "good" and "bad", and then use certain functions to estimate the overall "goodness" or "badness" score for the documents. (Kamps et al, 2002) propose to evaluate the semantic distance from a word to good/bad with WordNet. Turney (2001) measures the strength of sentiment by the difference of the mutual information (PMI) between the given phrase and "excellent" and the PMI between the given phrase and "poor". Pushing further from the explicit two-class classification problem, Pang et al (2005) and Zhang (2005) attempt to determine the author's opinion with different rating scales (i.e., the number of stars). Liu et al (2005) build a framework to compare consumer opinions of competing products using multiple feature dimensions. After deducting supervised rules from product reviews, the strength and weakness of the product are visualized with an "Opinion Observer". This method departs from classic sentiment classification in that we assume that sentiment consists of multiple hidden aspects, and use a probability model to quantitatively measure the relationship between sentiment aspects and blogs, as well as sentiment aspects and words.

Figure 1 Work flow of Sentiment analysis model. This figure shows how sentiment classifier model process data/

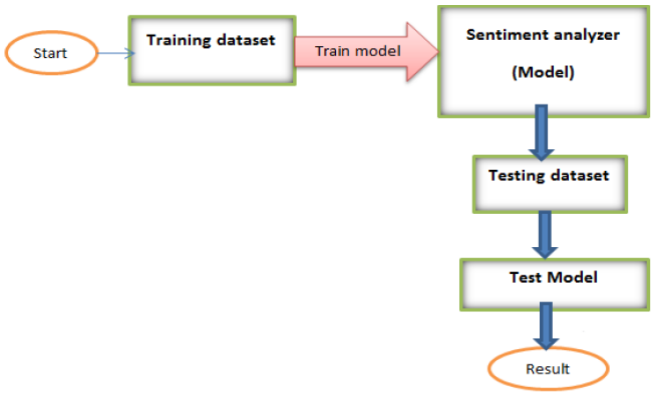

Figure 1 Work flow of Sentiment Analysis Model

III. BLOG MINING

A blog (short for weblog) is a personal online journal that is frequently updated and intended for general public consumption. Blogs have recently attracted a lot of research interest. There are currently two major research directions on blog analysis. One direction is to make use of links or URLs in Blogspace. Kumar et al, (2003) build a time graph for Blogspace, and develop views of the graph as a function of time. By observing the evolving behavior of the time graph, burst defined as a large sequence of temporally focused documents with plenty of links between them can be traced. Efron (2004) describes a hyperlink-based method to estimate the political orientation of Web documents. By estimating the likelihood of co-citation between a document of interest and documents with known orientations, the unknown document is classified to either left- or right-wing community. Gruhl et al(2005) prove that there is a strong correlation between blog mentions and sales rank; they therefore it can be believed that the temporal information of topics may help to forecast spike patterns in sales rank.

The other direction focuses on analyzing the contents of blogs. Mei et al (2006) considers blog as a mixture of unigram language models, with each component corresponding to a distinct subtopic or theme. To analyze spatiotemporal theme patterns from blogs, location variable and theme snapshot for each given time period are integrated in the model. Dalli (2006) builds a system for large-scale spatiotemporal analysis of online news and blogs. He utilizes an embedded hierarchical geospatial database to distinguish geographical named entities, and provides results for an extremely fine-grained analysis of items in news contents. Figure 3. Blog mining process model. This figure illustrates the process of blog mining.

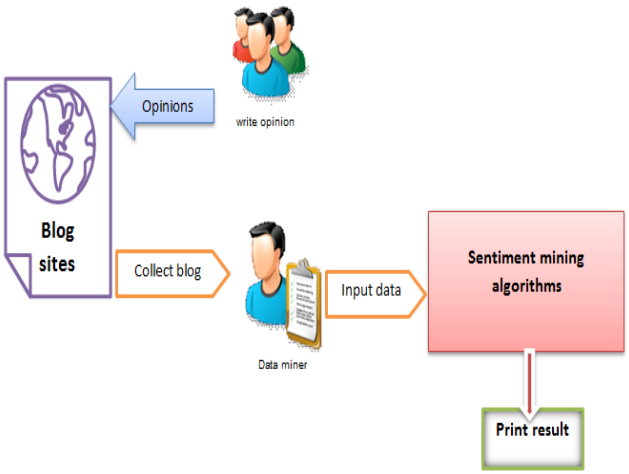

Figure 3 Blog Mining Process Model

IV. HADOOP FRAMEWORK

Hadoop is an open source framework for writing and running data intensive distributed applications that process large amounts of data. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. It was derived from Google's MapReduce and Google File System (GFS) papers.

Hadoop cluster is a set of commodity machines networked together in one location. Data storage and processing all occur within this "cloud" of machines. For computationally intensive processing data can be moved, but for data-intensive processing, the size of data becomes too large to be moved around easily. Hadoop focuses on moving code to data instead of vice versa. The clients send only the MapReduce programs to be executed, and these programs are usually small (often in kilobytes). More importantly, the move-code-to-data philosophy applies within the Hadoop cluster itself. Data is broken up and distributed across the cluster, and as much as possible, computation on a piece of data takes place on the same machine where that piece of data resides.

This move-code-to-data philosophy makes sense for the type of data-intensive processing Hadoop is designed for. The programs to run ("code") are orders of magnitude smaller than the data and are easier to move around. Also, it takes more time to move data across a network than to apply the computation to it. Let the data remain where it is and move the executable code to its hosting machine. Hadoop consists of two primary components: HDFS and MapReduce. HDFS, the Hadoop Distributed File System, is the primary storage system of Hadoop, and is responsible for storing and serving all data stored in Hadoop. MapReduce is a distributed processing framework designed to operate on data stored in HDFS.

V. HDFS

HDFS stands for Hadoop Distributed File System, which is the storage system used by Hadoop. Figure 4 is a high-level architecture that explains how HDFS works.

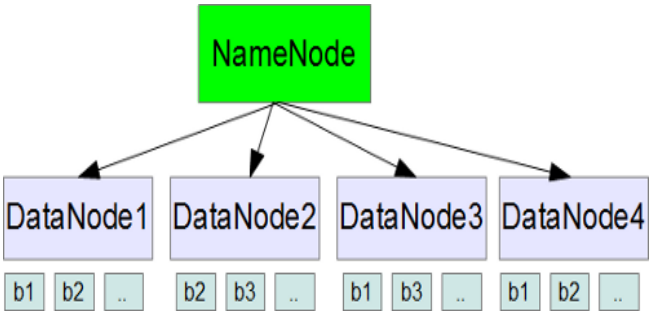

Figure 4. High-Level Architecture of HDFS

Hadoop employs master/slave architecture for both distributed storage and distributed computation. There is one Name Node (master), and multiple Data Nodes (workers) in a given HDFS.

The Name Node is the master of HDFS that directs the slave DataNode daemons to perform the low-level I/O tasks. The NameNode is the bookkeeper of HDFS; it keeps track of how your files are broken down into file blocks, which nodes store those blocks, and the overall health of the distributed file system. The function of the NameNode is memory and I/O intensive. As such, the server hosting the NameNode typically doesn't store any user data or perform any computations for a MapReduce program to lower the workload on the machine. This means that the NameNode server doesn't double as a DataNode or a TaskTracker.

There is unfortunately a negative aspect to the importance of the NameNode—it's a single point of failure of your Hadoop cluster. For any of the other daemons, if their host nodes fail for software or hardware reasons, the Hadoop cluster will likely continue to function smoothly or you can quickly restart it. Not so for the NameNode.

Each slave machine in the cluster will host a DataNode daemon to perform the grunt work of the distributed file system—reading and writing HDFS blocks to actual files on the local file system. When we want to read or write a HDFS file, the file is broken into blocks and the NameNode will tell the client which DataNode each block resides in. The client communicates directly with the DataNode daemons to process the local files corresponding to the blocks. Furthermore, a DataNode may communicate with other DataNodes to replicate its data blocks for redundancy.

In Figure 3 indicates data blocks. Each block has three replicas. The content of the files are distributed among the DataNodes. For example, b1 is replicated over the three rightmost DataNodes. This ensures that if any one DataNode crashes or becomes inaccessible over the network, you'll still be able to read the files.

DataNodes are constantly reporting to the NameNode. Upon initialization, each of the DataNodes informs the NameNode of the blocks it's currently storing. After this mapping is complete, the DataNodes continually poll the NameNode to provide information regarding local changes as well as receive instructions to create, move, or delete blocks from the local disk.

- **MapReduce**: MapReduce is a parallel programming model that is used to retrieve the data from the Hadoop cluster. In this model,

The library handles lot of messy details that programmers doesn't need to worry about. For example, the library takes care of parallelization, fault tolerance, data distribution, load balancing, etc. This splits the tasks and executes on the various nodes parallely, thus speeding up the computation and retrieving required data from a huge dataset in a fast manner. This provides a clear abstraction for programmers. They have to just implement (or use) two functions: map and reduce. The data are fed into the map function as key value pairs to produce intermediate key/value pairs. Once the mapping is done, all the intermediate results from various nodes are reduced to create the final output.

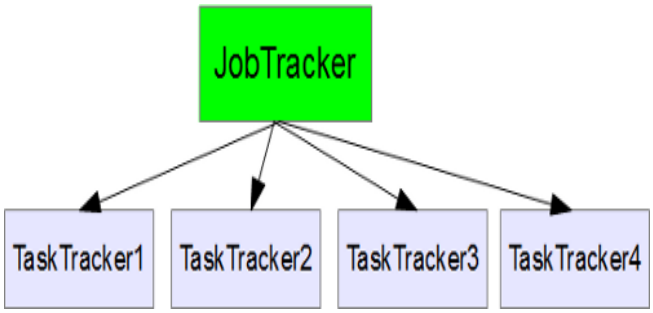

Figure 5 A high-level architecture of MapReduce

The JobTracker daemon is the liaison between our application and Hadoop. Once we submit our code to our cluster , the Job Tracker determines the execution plan by deter mining which files to process, assigns nodes to different tasks, and monitors all tasks as they're running. Should a task fail, the Job Tracker will automatically relaunch the task, possibly on a different node, up to a predefined limit of retries. There is only one Job Tracker daemon per Hadoop cluster. It's typically run on a server as a master node of the cluster.

VI. SENTIMENT PLSA

Mining opinions and sentiments present unique challenges that cannot be handled easily by traditional text mining algorithms. This is mainly because the opinions and sentiments, which are usually written in natural languages, are often expressed in complex ways. Moreover, sentiments are often multifaceted, and can differ from one another in a variety of ways, including polarity, orientation, graduation, and so on. Therefore, it would be too simplistic to just classify the sentiments expressed in a blog as either positive or negative. For the purpose of sales prediction, a model that can extract the sentiments in a more accurate way is needed.

To this end, a probabilistic model called Sentiment Probabilistic Latent Semantic Analysis (S-PLSA), in which a blog can be considered as being generated under the influence of a number of hidden sentiment factors. The use of hidden factors allows us to accommodate the intricate nature of sentiments, with each hidden factor focusing on one specific aspect of the sentiments. The use of a probabilistic generative model, on the other hand, enables us to deal with sentiment analysis in a principled way. In its traditional form, PLSA [9] assumes that there are a set of hidden semantic factors or *aspects* in the documents, and models the relationship among these factors, documents, and words under a probabilistic framework. With its high flexibility and solid statistical foundations, PLSA has been widely used in many areas, including information retrieval, Web usage mining, and collaborative filtering. In this study PLSA model is used for sentiment mining.

Now it is a time to formally present S-PLSA. Suppose we are given a set of blog entries $B = \{b1, \ldots, bN\}$, and a set of words (appraisal words) from a vocabulary $W = \{w1, \ldots, wM\}$. The blog data can be described as a $N \times M$ matrix, $D = (c(bi, wj))ij$, where $c(bi, wj)$ is the number of times wi appears in blog entry bj. Each row in D is then a frequency vector that corresponds to a blog entry. Let us consider the blog entries as being generated from a number of hidden sentiment factors, $Z = \{z1, \ldots, zK\}$. We expect that those hidden factors would correspond to blogger's complex sentiments expressed in the blog review.

- S-PLSA can be considered as the following generative model.

1. Pick a blog document b from B with probability $P(b)$;
2. Choose a hidden sentiment factor z from Z with probability $P(z|b)$;
3. Choose a word from the set of appraisal words W with probability $P(w|z)$.

The end result of this generative process is a blog-word pair (b,w), with z being integrated out. The joint probability can be factored as follows:

$$P(b, w) = P(b)P(w \mid b),$$

Where

$$P(w \mid b) = \sum_{z \in Z} P(w \mid z)P(z \mid b).$$

Assuming that the blog entry b and the word w are conditionally independent given the hidden sentiment factor z, we can use Bayes rule to transform the joint probability to the following:

$$P(b, w) = \sum_{z \in Z} P(z)P(b \mid z)P(w \mid z).$$

To explain the observed (b,w) pairs, we need to estimate the model parameters $P(z)$, $P(b|z)$, and $P(w|z)$. To this end, we seek to maximize the following likelihood function:

$$L(B,W) = \sum_{b \in B} \sum_{w \in W} c(b, w) \log P(b, w),$$

Where $c(b,w)$ represents the number of occurrences of a pair (b,w) in the data.

A widely used method to perform maximum likelihood parameter estimation for models involving latent variables (such as our S-PLSA model)

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

is the Expectation-Maximization (EM) algorithm, which involves an iterative process with two alternating steps: (1) an expectation step (E-step), where posterior probabilities for the latent variables (in our case, the variable z) are computed, based on the current estimates of the parameters; (2) a maximization step (M-step), where estimates for the parameters are updated to maximize the complete data likelihood.

In S-PLSA model, with the parameters $P(z)$, $P(w|z)$, and $P(b|z)$ suitably initialized, we can show that the algorithm requires alternating between the following two steps:

- In E-step compute

$$P(z \mid b, w) = \frac{P(z)P(d \mid z)P(w / z)}{\sum_{z' \in Z} P(z')P(d \mid z')P(w \mid z')};$$

- In M-Step update the parameters of the model with:

$$P(w \mid z) = \frac{\sum_{b \in B} c(b, w)P(z / b, w)}{\sum_{b \in B} \sum_{w' \in W} c(b, w')P(z \mid b, w')},$$

$$P(b \mid z) = \frac{\sum_{w \in W} c(b, w)P(z / b, w)}{\sum_{b' \in B} \sum_{w \in W} c(b', w)P(z \mid b', w)},$$

$$P(z) = \frac{\sum_{b \in B} \sum_{w \in W} c(b, w)P(z / b, w)}{\sum_{b \in B} \sum_{w \in W} c(b, w)}.$$

It is shown that the iteration above monotonically increases the complete data likelihood, and the algorithm converges when a local optimal solution is achieved. Once the parameter estimation for the model is completed, we can compute the posterior probability $P(z|b)$ using the Bayes rule:

$$P(z \mid b) = \frac{P(b \mid z)P(z)}{\sum_{z \in Z} P(b \mid z)P(z)}$$

Intuitively, $P(z|b)$ represents how much a hidden sentiment factor $z(\in Z)$ "contributes" to the blog document b. Therefore, the set of probabilities $\{P(z|b)|z \in Z\}$ can be considered as a succinct summarization of b in terms of sentiments. As will be shown in the next section, this summarization can then be used in the predication of future product sales.

VII. EXPERIMENT SETTINGS

The movie data used in the experiments consists of two components. The first component is a set of blog documents on movies of interest collected from the Web, and the second component contains the corresponding daily box office revenue data for these movies.

Blog entries were collected for movies released from March 1st -31st 2011, in the United States. For each movie, using the movie name and a date as keywords, we composed and submitted queries to Google's blog search engine, and retrieved the blogs entries that were listed in the query results. For a particular movie, we only collected blog entries that had a timestamp ranging from one week before the release to three weeks after, as we assume that most of the reviews might be published close the release date. Through limiting the time span for which we collect the data, we are able to focus on the most interesting period of time around a movie's release, during which the blog discussions are generally the most intense. As a result, the amount of blog entries collected for each movie ranges from 127,000 (for Diary of a Wimpy Kid: Rodrick Rules) to 8,700,000 (for Battle: Los Angeles). In total, 21,610,000 blog entries that comment on 10 different movies were collected. Then extracted the title, perm link, free text contents, and time stamp from each blog entry, and indexed them using Apache Lucene.

Table 1 the number of blog entries and daily gross revenue data from March 1 to 31, 2011 collected manually from IMDB and Google blogger websites.

Table 1 Number of blog entries and box office revenue data

Number	Movie Title	Total Gross in one Month	Number of Blog entries in one month
1	Rango	$123,477,607	2,600,000
2	Battle: Los Angeles	$83,552,429	8,700,000
3	Limitless	$79,249,455	3,180,000
4	The Adjustment Bureau	$62,495,645	608,000
5	The Lincoln Lawyer	$58,009,200	855,000
6	Diary of a Wimpy Kid: Rodrick Rules	$52,698,535	127,000
7	Red Riding Hood	$37,662,162	2,070,000
8	Paul	$37,412,945	1,290,000
9	Sucker Punch	$36,392,502	1,710,000
10	Beastly	$27,865,571	470,000
		Total	21,610,000

In addition the gross box office revenue data for the 10 movies from the IMDB website has been manually collected. For each movie, the number of blog entries in US for 1 month period of time starting from the release date till four weeks after the release was collected. In each run of the experiment, the following procedure was followed:

- Randomly choose half of the movies for training set, and the other half for testing; the blog entries and box office revenue data are correspondingly partitioned into training and testing data sets.

- Using the training blog entries, train an S-PLSA model. For each blog entry *b*, the sentiments towards a movie are summarized using a vector of the posterior probabilities of the hidden sentiment factors, $P(z \mid b)$.
- Feed the probability vectors obtained in step 2, along with the box office revenues of the preceding days, into the SAAR model, and obtain estimates of the parameters.
- Evaluate the prediction performance of the SAAR model by experimenting it with the testing data set.

In this report, the *mean absolute percentage error* (MAPE) is used to measure the prediction accuracy:

$$MAPE = \frac{1}{n}\sum_{i=1}^{N}\frac{|\Pr ed_i - True_i|}{True_i},$$

Where *n* is the total amount of predictions made on the testing data, $Pred_i$ is the predicted value, and $True_i$ represents the true value of the box office revenue. All the accuracy results reported herein are averages of 10 independent runs.

VIII. PARAMETER SELECTION

In the SAAR model, there are several user-chosen parameters that provide the flexibility to well tune the model for best performance. They include the number of hidden sentiment factors in S-PLSA which is *K*, and the orders of the SAAR model parameters, *p* and *q*. We now study how the choice of these parameter values that affects the prediction accuracy.

First vary *K*, with fixed *p* and *q* values (*p* = 7, and *q* = 1). As shown in Figure 7-1, as *K* increases from 1 to 4, the prediction accuracy improves, and at *K* = 4, *SAAR* achieves an MAPE of 11%. That implies that representing the sentiments with higher dimensional probability vectors allows S-PLSA to more fully capture the sentiment information, which leads to more accurate prediction. On the other hand, as shown in the graph, the prediction accuracy decline once *K* gets past 4. The explanation here is that a large *K* may cause the problem of overfitting , i.e., the SPLSA might fit the training data better with a large *K*, but its generalization capability on the testing data might become poor. Some tempering algorithms have been proposed to solve the overfitting problem , but it is out of the scope of our study(Hofmann.1999). Also, if the number of appraisal words used to train the model is *M*, and the number of blog entries is *N*, the total number of parameters which must be estimated in the S-PLSA model is $K(M + N + 1)$. This number grows linearly with respect to the number of hidden factors *K*. If *K* gets too large, it may incur a high training cost in terms of time and space.

Then vary the value of *p*, with *K* = 4 and *q* = 1 to study how the order of the autoregressive model affects the prediction accuracy. We observe from Figure 7, that the model achieves it best prediction accuracy when *p* = 7. This suggests that *p* should be large enough to factor in all the significant influence of the preceding days' box office performance, but not too large to let irrelevant information in the more distant past to affect the prediction accuracy.

Using the optimal values of *K* and *p*, we vary *q* from 1 to 5 to study its effect on the prediction accuracy. As shown in Figure 8, the best prediction accuracy is achieved at *q* = 1, which implies that the prediction is most strongly related to the sentiment information captured from blog entries posted on the immediately preceding day.

Figure 6, 7, & 8 illustrates the effect of user defined parameters *K*, *P*, & *q* on prediction accuracy.

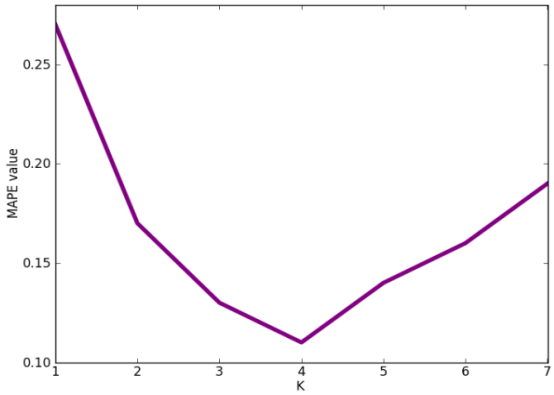

Figure 6. The effect of parameter K on prediction accuracy

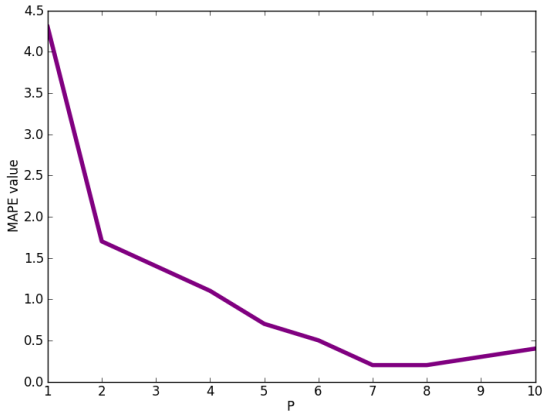

Figure 7 the effect of parameter P on prediction accuracy

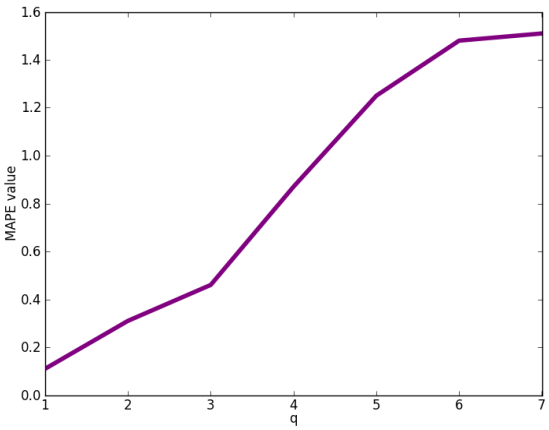

Figure 8 the effect of parameter *q* on prediction accuracy

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

To better illustrate the effects of the parameter values on the prediction accuracy, the experimental results presented on a particular movie, *Red Riding Hood* on Figure 9,10, and 11. For each parameter *k*, *p* and *q* we plot the predicted box office revenues and the true values for each day using different values of the parameter. It is evident from the plots that the responses to each parameter are similar to what is observed from Figure 6, 7, and 8 . Also note that the predicted values using the optimal parameter settings are close to the true values across the whole time span. Similar results are also observed on other movies, demonstrating the consistency of the proposed approach for different days.

Figure 9, 10, and 11 illustrates the effects of the parameter values on the prediction accuracy and experimental results presented on a particular movie, *Red Riding Hood.*

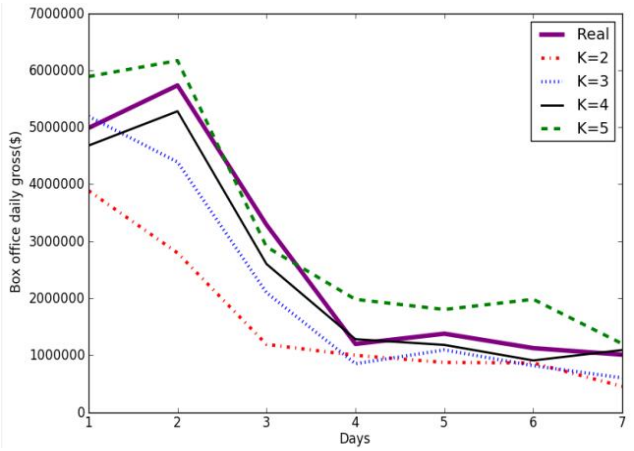

Figure 9 the effect *K* for the movie *Red Riding Hood*

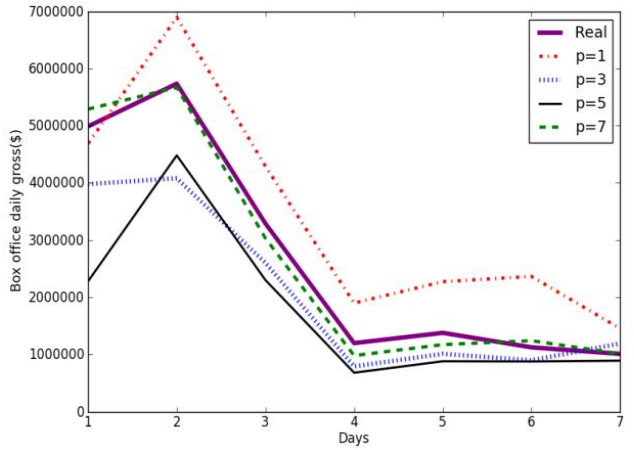

Figure 10 the effect P for the movie Red Riding Hood

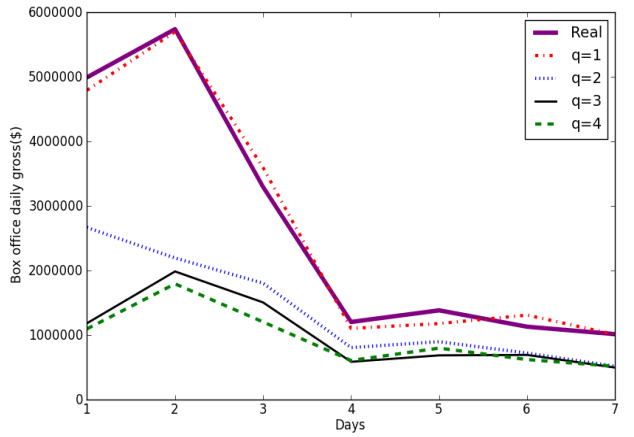

Figure 11 the effect q for the movie Red Riding Hood

IX. COMPARISON WITH ALTERNATIVE METHODS

To prove that the sentiment information captured by the S-PLSA model plays an important role in box office revenue prediction, SAAR is compared with two alternative methods which do not take sentiment information into consideration. First some experiments are conducted to compare SAAR against the pure autoregressive (AR) model without any terms on sentiments, i.e. $y_t = \sum_{i=1}^{p} \phi_i y_{t-i} + \in_t$. The results are shown in Figure 12. It is observed that the behavior of the two models as *p* ranges from 3 to 7. Apparently, although the accuracy of both methods improves with increasing *p*, SAAR constantly outperforms the AR model by a factor of 2 to 3.

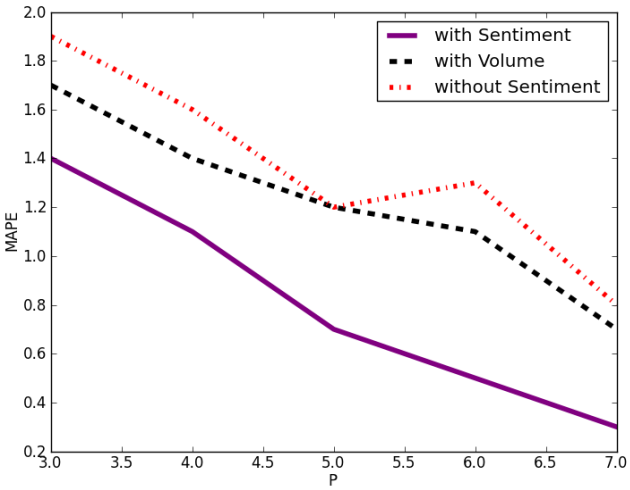

Figure 12 SAAR vs. Other models

This study then proceeds to compare SAAR with an autoregressive model that factors in the volume of blog mentions in prediction. Previously, the study illustrated the characteristics of the volume of blog mentions and its connection to the sales performance with an example, showing that although there exist a correlation between the volume of blog mentions and the sales performance, this correlation may not be strong enough to enable prediction. To further demonstrate this, we experiment with the following autoregressive model that utilizes the volume of blogs mentions. In contrast to SAAR, where we use a multi-dimensional probability vector produced by S-PLSA to represent bloggers' sentiments, this model uses a scalar (number of blog mentions) to indicate the degree of popularity. The model can be formulated as

$$y_t = \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{i=1}^{q} p_i v_{t-i} + \in_t$$

Where y_t's are obtained in the same way as in SAAR, v_{t-i} denotes the number of blog mentions on day $t-i$, and ϕ_i & ρ_i are parameters to be learned.

This model can be trained using a procedure similar to what is used for SAAR. Using the same training and testing data sets as what are used for SAAR, we test the performance of this model and compare it with SAAR. The results are shown in Figure 12. Observe that although this method yields a moderate performance gain over the pure AR model (which proves that the volume data do have some predictive power), its performance is still dominated by the SAAR model.

X. CONCLUSIONS

The wide spread use of blogs as a way of conveying personal views and comments has offered an unique opportunity to understand the general public's sentiments and use this information to advance business intelligence. This study have explored the predictive power of blogs using movies as a case study, and studied the problem of predicting sales performance using sentiment information mined form blogs. A center piece of our work is the proposal of S-PLSA, a generative model for sentiment analysis that helps us move from simple "negative or positive" classification towards a deeper comprehension of the sentiments in blogs. Using S-PLSA as a means of "summarizing" sentiment information from blogs, SAAR is developed, a model for predicting sales performance based on the sentiment information and the products past sales performance. The accuracy and effectiveness of our model have been confirmed by the experiments on the movie data set. Equipped with the proposed models, companies will be able to better harness the predictive power of blogs and conduct businesses in a more effective way. It is worth noting that although to use SPLSA for the purpose of prediction in this work, it is indeed a model general enough to be applied to other scenarios.

REFERENCES

1. Bar-Ilan, J. (2004). An outsider's view on "topic-oriented blogging". In WWW Alt. '04, pages 28–34..
2. Blei, D. Ng, A. & Jordan, M. (2003). "Latent dirichlet allocation". Journal of Machine Learning Research.
3. Dalli, A. (2006). "System for spatio-temporal analysis of online news and blogs". In WWW '06, pages 929–930,
4. Dempster, A. P., Laird, N. M., & Rubin D. B.(1977). "Maximum likelihood from incomplete data via the em algorithm". Journal of Royal Statistical Society, B(39):1–38, 1977.
5. Efron, M.(2004). "The liberal media and right-wing conspiracies: using co-citation information to estimate political orientation in web documents". In CIKM '04, pages 390–398, 2004.
6. Enders W.(2004). Applied Econometric Time Series. Wiley, New York, 2nd edition, 2004.
7. Gruhl, D., Guha, R., Kumar, R., Novak, J. & Tomkins, A.(2005) "The predictive power of online chatter". In KDD '05, pages 78–87.
8. Gruhl, D., Guha, R. ,Liben-Nowell, D., & Tomkins A.(2004) "Information diffusion through blogspace". In WWW '04, pages 491–501.
9. Hofmann T.(1999). Probabilistic latent semantic analysis. In UAI'99.
10. Jank, W., Shmueli, G., & Wang S.(2006) Dynamic, real-time forecasting of online auctions via functional models. In KDD '06, pages 580–585.
11. Kamps, J. & Marx, M.(2002) "Words with attitude." In Proc. of the First International Conference on Global WordNet, pages 332–341, 2002.
12. Kumar R., Novak, J., Raghavan, P. & Tomkins, A.(2003). On the bursty evolution of blogspace. In WWW '03, pages 568–576, 2003.
13. Kumar R., Novak, J., Raghavan, P. & Tomkins, A.(2004) "Structure and evolution of blogspace." Commun. ACM, 47(12):35–39.
14. Li, Z. Wang, B., Li, M., & Ma, W.(2005) "A probabilistic model for retrospective news event detection." In SIGIR '05, pages 106–113.
15. Liu, B., Hu, M. & Cheng, J.(2005). "Opinion observer: analyzing and comparing opinions on the web." In WWW '05, pages 342–351, 2005.
16. Mei, Q., Liu, C. ,Su, H. & Zhai, C.X.(2006) "A probabilistic approach to spatiotemporal theme pattern mining on weblogs". In WWW '06, pages 533–542.
17. Mei, Q. & Zhai,, C.X.(2006) "A mixture model for contextual text mining". In KDD '06, pages 649–655.
18. Pang, B. & Lee, L.(2004) "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts." In ACL '04, pages 271–278 .
19. Pang, B. & Lee, L. (2005). "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales". In ACL '05, pages 115–124.
20. Pang, B., Lee L., & Vaithyanathan S.(2002). "Thumbs up? sentiment classification using machine learning techniques". In Proc. of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP).
21. Technorati (2007). URL:http://technorati.com/about/. Retrieved on January 27.
22. Tseng, B. L., Tatemura, J., & Wu, Y.(2005). "Tomographic clustering to visualize blog communities as mountain views". In Proc. Of 2nd Annual Workshop on the Weblogging Ecosystem.
23. Turney, P.D. (2001). "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews". In ACL '02, pages 417–424.
24. Whitelaw, C., Garg, N., & Argamon S.(2005). "Using appraisal groups for sentiment analysis", In CIKM '05, pages 625–631.
25. Zhang, Z. and Varadarajan, B.(2006). "Utility scoring of product reviews". In CIKM '06, pages 51–57.
26. Foutz, N.Z. & Jank, W.(2007 "The Wisdom of Crowds: Pre-Release Forecasting via Functional Shape Analysis of the Online Virtual Stock Market," Technical Report 07-114 Marketing Science Inst. Of Reports.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication