Effective sentiment stream analysis with self-augmenting training and demand-driven projection

Conference Paper · July 2011

DOI: 10.1145/2009916.2009981 · Source: DBLP

5 authors, including:

Ismael S. Silva
Federal University of Minas Gerais
31 PUBLICATIONS **219** CITATIONS
SEE PROFILE

Adriano Veloso
Federal University of Minas Gerais
220 PUBLICATIONS **4,526** CITATIONS
SEE PROFILE

Wagner Meira Jr.
Federal University of Minas Gerais
617 PUBLICATIONS **12,398** CITATIONS
SEE PROFILE

Renato Ferreira
Federal University of Minas Gerais
131 PUBLICATIONS **2,028** CITATIONS
SEE PROFILE

Effective Sentiment Stream Analysis with Self-Augmenting Training and Demand-Driven Projection

Ismael S. Silva, Janaína Gomide, Adriano Veloso, Wagner Meira Jr., Renato Ferreira
Computer Science Department
Federal University of Minas Gerais
31270-901 Belo Horizonte, Brazil
{ismael.silva, janaina, adrianov, meira, renato}@dcc.ufmg.br

ABSTRACT

How do we analyze sentiments over a set of opinionated Twitter messages? This issue has been widely studied in recent years, with a prominent approach being based on the application of classification techniques. Basically, messages are classified according to the implicit attitude of the writer with respect to a query term. A major concern, however, is that Twitter (and other media channels) follows the data stream model, and thus the classifier must operate with limited resources, including labeled data for training classification models. This imposes serious challenges for current classification techniques, since they need to be constantly fed with fresh training messages, in order to track sentiment drift and to provide up-to-date sentiment analysis.

We propose solutions to this problem. The heart of our approach is a training augmentation procedure which takes as input a small training seed, and then it automatically incorporates new relevant messages to the training data. Classification models are produced on-the-fly using association rules, which are kept up-to-date in an incremental fashion, so that at any given time the model properly reflects the sentiments in the event being analyzed. In order to track sentiment drift, training messages are projected on a demand-driven basis, according to the content of the message being classified. Projecting the training data offers a series of advantages, including the ability to quickly detect trending information emerging in the stream. We performed the analysis of major events in 2010, and we show that the prediction performance remains about the same, or even increases, as the stream passes and new training messages are acquired. This result holds for different languages, even in cases where sentiment distribution changes over time, or in cases where the initial training seed is rather small. We derive lower-bounds for prediction performance, and we show that our approach is extremely effective under diverse learning scenarios, providing gains that range from 7% to 58%.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.5.2 [**Pattern Recognition**]: Classifier Design and Evaluation

SIGIR'11, July 24–28, 2011, Beijing, China.

General Terms

Algorithms, Experimentation, Performance

Keywords

Sentiment Analysis, Sentiment Drift, Streams, Twitter

1. INTRODUCTION

The rise of text-based social media channels has fueled scientists with torrents of opinionated data about the most diverse topics and entities. This has spurred the proliferation of tools with the ability to analyze the sentiment expressed by the online population which visits and participates in social channels and (micro-)blogs. This population accounts for more than two-thirds of all Internet users [1], and thus sentiment analysis will eventually become a key feature of search engines, which may integrate the aggregate sentiment of the crowd into search results.

Sentiment analysis in these scenarios presents two characteristics that make it more challenging than in other previous researched scenarios. The first is that the task consists of analyzing a humongous amount of messages that are produced continuously by a large and uncontrolled number of users. The second is that these messages tend to be very short, as required by Twitter, but such practice is becoming a trend in other channels. We name this scenario *sentiment streams* and the task *sentiment stream analysis*. In this paper we focus on Twitter, but our techniques may be applicable to any channels that share the same characteristics we just mentioned. Twitter is one of the fastest-growing social media channels, and has proven itself to be an authoritative source for breaking news, some of which concerns events of huge impact world-wide [20]. Sensitive information is created almost in the same time the event is happening in the real world, and it becomes available shortly after it is created. Also, the 140-character limit is very restrictive, not providing enough space for users to explain, elaborate or get distracted from their main point. Twitter is thus a valuable niche for large scale sentiment analysis, and recent years have experienced the emergence of many tools and techniques for this task [26].

There is a growing trend in performing sentiment analysis using classification-related techniques: a process that automatically builds a classification model by learning, from a set of previously labeled messages (i.e., the training data), the underlying characteristics that distinguish one sentiment from another (i.e., happiness, madness, surprise, suspicion). The success of these classifiers rests on their ability to judge attitude by means of textual-patterns present in the messages, which usually appear in the form of (idiomatic) expressions and combinations of words. It is well accepted that the quality of the training data that is provided to the classifier is crucial to its effectiveness. Although there is no con-

sensus on how the training data should be produced, it is commonsense that the cost of manually labeling vast amounts of messages is prohibitive, since the acquisition of these example messages may require the inspection of skilled human annotators. This annotation burden motivated the emergence of a large repertoire of semi-supervised and active learning alternatives [7]. Still, these techniques assume by large that the training data is sampled from a stationary distribution, but time-varying data plays a significant role in sentiment analysis. Twitter, for instance, enables millions of users to tweet at any moment, and thus, variations in sentiment distribution may happen constantly.

We investigate sentiment analysis over Twitter real-time messages. In such scenarios, classifiers must operate with limited computing and training resources. To make things even worse, either sentiment distribution or the characteristics related to certain sentiments may change over time in almost unforeseen ways. This is known as *sentiment drift*, and it makes predictions less accurate as time passes. To prevent deterioration over time the classification model has to be constantly refreshed, meaning that classifiers must be able to automatically incorporate novel information into the training data and update the model on-the-fly, so that the predictions to come can take advantage of up-to-date information immediately. Some well-established classification strategies may become ill-suited in such hard circumstances, while alternate solutions may be more convenient.

We propose to learn sentiments using classification models composed of association rules [2]. After a small training seed is provided to the classifier, it is able to extract these rules which are essentially local mappings relating sentiments to textual-patterns in the messages. Also, two novel features make our classifiers unique in dealing with different settings of sentiment drift, while operating with limited amount of resources:

- They employ a self-augmenting training procedure, which incorporates reliable predictions as new training information, and as a result, the training data is automatically augmented as the message stream passes. The classification model is immediately refreshed by keeping rules up-to-date incrementally, so that the next message in the stream can potentially take advantage of the recently included information. Depending on the stream tightness, the classifiers can also abstain from doubtful predictions, creating a small block of messages that are temporarily waiting for a reliable prediction. These messages for which no reliable prediction is possible with only the training messages available at the time, may benefit from the training information acquired later.

- They perform demand-driven projection, which determines a specific (potentially different) training projection for each message passing through the stream. The training messages that compose each training projection are automatically selected according to the content of the message being analyzed. This incurs in a powerful strategy for tracking different types of sentiment drift, since it removes training messages that are not meaningful to the message being analyzed. We prove that the removed messages are not detrimental to prediction performance, and that the number of rules extracted from each training projection grows polynomially with the size of the vocabulary, no matter the *minimum support* value. This enables the classifier to focus on trending information that is just emerging in the stream, while staying free from a huge amount of meaningless information.

These two features were not coincidentally proposed together. In fact, they have a synergistic effect in the sense that both need each other in order to work properly. The self-augmenting training ability assures the inclusion of new training messages that are necessary to produce up-to-date training projections. At this point, the use of typical computational cost restrictions based on minimum support thresholds would compromise the entire process, since important patterns that are just emerging in the stream would be pruned, and as a result, the classification model would become obsolete and unable to respond to sentiment trends. Therefore, in order to quickly detect the appearance of novel information in the message stream, the classifier must extract rules without employing frequency restrictions based on support. As will be demonstrated, the proposed demand-driven projection approach assures that rules are efficiently extracted from the training projection (i.e., in polynomial time), even without applying restrictions on support. Furthermore, the cost associated with rule extraction is greatly amortized due to a lossless incremental approach, which drastically reduces the number of accesses to the training data.

To evaluate the effectiveness of the proposed classifiers, we performed a systematic set of experiments using sentiment-rich Twitter data collected from three important events in 2010. We employed different different learning scenarios (i.e., different sentiments, different languages, and different training seeds). To validate our claims we derived lower-bounds for classification performance, and the results show that our classifiers are extremely effective under diverse learning scenarios, with gains in prediction performance that range from 7% to 58%.

2. RELATED WORK

The ultimate goal of a classifier is to achieve the best possible prediction performance for the problem at hand. Devising effective classifiers for a specific problem is not a simple task, but there is a body of evidence suggesting that classification offers substantial advantages in several application scenarios, including sentiment analysis [26]. There has been a large amount of prior research in sentiment analysis, especially in the domain of product reviews, movie reviews, and blogs [25]. A variety of classifiers were already evaluated in many different sentiment learning scenarios, such as analyzing brand impact of microbloging [19], or learning consumer confidence and political opinion [9, 23, 24].

More recently, it becomes possible to analyze population sentiment at a large scale. Social channels like Twitter offer the necessary resource: vast amounts of opinionated content [20]. Unfortunately, there is a major bottleneck in the process: the necessity for training examples which are labeled by human annotators. In order to limit human intervention, automated alternatives that use *emoticons* (or tags) [28] and other distant supervision approaches [16] were also proposed, but (1) they are prone to error by definition, (2) they are unable to capture other types of sentiments for which no *emoticon* (or any tag) is associated, and (3) different sentiments may be associated with the same *emoticon* (or the same tag). Ramage et al. [27] characterize users and messages of Twitter using topic models. Their approach is based on labeled latent Dirichlet allocation, which is used to detect topics of words that tend to co-occur in similar tweets and from implied tweet labels (e.g. hashtags, emoticons ad replies). Other alternatives to address the annotation burden include active and semi-supervised learning approaches [7]. However, there is also a major impediment that prevent the application of these approaches: channels such as Twitter follow the data streaming paradigm [4], and thus, (1) classifiers must operate with limited amount of resources, and (2) online sentiment may change over time.

Many techniques have been proposed to handle the issues associated with analyzing data streams. An incremental approach [14]

was proposed to focus on streaming data using Hoeffding bounds. Other approaches incrementally update their classification model with new training data to cope with the evolution of the stream [10, 15]. These techniques usually require complex operations to update the model. Also, Wang et. al [31] and Fan [11] proposed ensemble techniques for stream mining.

The above techniques assume that data distribution is smooth, but actually, (concept) drifts are often hidden in the stream. Gama et al. [13] proposed to use a forgetting mechanism based on a sliding window with the most recent observations. Street and Kim [29] proposed a technique based on a ensemble of decision trees to deal with drifting streams. It splits data into batches, fits one decision tree per batch and discards the old models heuristically. Zhu et al. [36] proposed an active learning framework to selectively label instances from drifting streams. The decision tree based technique proposed in [18] keeps the model current while making the most of old data by growing an alternative subtree whenever an old one becomes questionable, and replacing the old with the new when it becomes more accurate to adapt to the current concept. An approach to concept drift [21] was proposed to create and remove weighted experts dynamically corresponding to the changes of performance. An Optional Weight Adjustment method [34, 35] utilizes the most recent data block to detect optional weighted values for the classifier, and applies a kernel mean matching method to minimize the discrepancy of data blocks in the kernel space. Two variants of bagging: ADWIN Bagging and Adaptive-Size Hoeffding Tree (ASHT) Bagging were introduced for tackling non-stationary concepts in data streams [5]. Maloof [22] proposed a new incremental rule learning algorithm, which uses heuristics to adapt the size of training window dynamically.

The advantages of the techniques to be proposed in this paper, when compared against existing techniques, are manifold. Firstly, our classifiers are able to detect exactly the pieces of the training data that must be updated. Consequently, the classification model is updated in a highly efficient way, without wasting computing resources. Secondly, our classifiers are able to project the training data on a demand-driven basis, producing a specific training frame for each message that is analyzed. The messages that compose each frame are automatically selected according to qualitative information present on the message being analyzed. We show that training messages have an expiration time, after which they become totally useless for the sake of classification. Finally, our classifiers are able to abstain from doubtful predictions. The corresponding messages are blocked until more evidence are obtained, and a reliable judgement becomes possible.

3. LEARNING SENTIMENT STREAMS

In this section we present novel classifiers for learning sentiments that are expressed in streams of Twitter messages. We start by discussing classification models based on specialized association rules. Then we describe static models with offline rule extraction, and dynamic models with self-augmenting training and demand-driven projection features.

3.1 Sentiment Stream Analysis

In our context, the task of learning sentiment streams is defined as follows. We have as input a small training seed (referred to as $\mathcal{D}$), which consists of a set of records of the form $< d, s_i >$, where d is a message (represented as a list of terms), and s_i is the sentiment implicit in d. Messages in $\mathcal{D}$ are uniquely identified and the sentiment variable s draws its values from a pre-defined and discrete set of possibilities (e.g., s_1, s_2, ..., s_k). The training seed is used to build functions relating textual patterns in the messages to their corresponding sentiments. A sequence of future messages (referred to as $\mathcal{T}$) consists of records $< t, ? >$ for which only the terms in message t are known, while the sentiment expressed in t is unknown. Classification models obtained from $\mathcal{D}$ are used to score the sentiments for each message in $\mathcal{T}$. However, messages in $\mathcal{T}$ come on streams, so that the classifier must operate with limited computing resources while producing classification models. Also, the classifier must adapt itself due to sentiment drift, being able to acquire new training information as the stream passes, and to select training messages that are relevant to each message in $\mathcal{T}$.

There are countless strategies for devising a classifier for sentiment analysis. The majority of these classification strategies, however, are not well-suited to deal with real-time data coming on streams. Some strategies [6, 8] are specifically devised for offline classification, and this is problematic because producing classification models on-the-fly would be unacceptably costly. Even updating the models in scenarios with high-speed streams would be excessively lengthy. In such hard circumstances, alternate classification strategies may become more convenient. In the following we describe classification models composed of specific association rules, and how these models are used for sentiment-scoring.

Definition 1. A sentiment rule is a specialized association rule $\mathcal{X} \rightarrow s_i$, where the antecedent $\mathcal{X}$ is a set of terms (i.e., a termset), and the consequent s_i is the predicted sentiment. The domain for $\mathcal{X}$ is the vocabulary of $\mathcal{D}$. The cardinality of rule $\mathcal{X} \rightarrow s_i$ is given by the number of terms in the antecedent, that is $|\mathcal{X}|$. The support of $\mathcal{X}$ is denoted as $\sigma(\mathcal{X})$, and is the number of messages in $\mathcal{D}$ having $\mathcal{X}$ as a subset. The confidence of rule $\mathcal{X} \rightarrow s_i$ is denoted as $\theta(\mathcal{X} \rightarrow s_i)$, and is the conditional probability of sentiment s_i given the terms in $\mathcal{X}$, that is, $\theta(\mathcal{X} \rightarrow s_i) = \frac{\sigma(\mathcal{X} \cup s_i)}{\sigma(\mathcal{X})}$.

Offline Rule Extraction

The simplest approach for sentiment learning using sentiment rules is the offline one. A set of rules is extracted from the training data $\mathcal{D}$. These rules compose the classification model.

Definition 2. The classification model is denoted as $\mathcal{R}$, and it is composed of a set of rules $\mathcal{X} \rightarrow s_i$ extracted from $\mathcal{D}$. The model is represented as a pool of entries with the form <*key, data*>, where *key*=$\{\mathcal{X}, s_i\}$ and *data*=$\{\sigma(\mathcal{X}), \sigma(\mathcal{X} \cup s_i), \theta(X \rightarrow s_i)\}$. Each entry in the pool corresponds to a rule, and the key is used to facilitate fast access to the rule properties.

The rule extraction process is divided into two steps: support counting and confidence computation. Once the support $\sigma(\mathcal{X})$ is known, it is straightforward to compute the confidence $\theta(\mathcal{X} \rightarrow s_i)$ for the corresponding rules [33]. There are several smart support-counting strategies [2, 17, 33], and many fast implementations [3] that can be used.

Usually, the support for termsets in $\mathcal{D}$ are computed in a bottom-up way, which starts by scanning all messages in $\mathcal{D}$ and computing the support of each term in isolation (i.e., 1-termsets). In the next iteration, 2-termsets (i.e., termsets of size 2) are enumerated using 1-termsets, and their support values are calculated by accessing the training data. The search for termsets proceeds, and the enumeration process is repeated until the support values for all termsets in $\mathcal{D}$ are finally computed. Obviously, the number of rules increases exponentially with the size of the vocabulary (i.e., the number of distinct terms in $\mathcal{D}$), and computational cost restrictions have to be imposed during rule extraction. Typically, the search space for rules is restricted by pruning rules that do not appear frequently in

$\mathcal{D}$ (i.e., the minimum support approach). While such restrictions make rule extraction feasible, they also lead to lossy classification models, since some rules are pruned and not be included into $\mathcal{R}$.

Sentiment Scoring

Once the classification model $\mathcal{R}$ is extracted from $\mathcal{D}$, rules are collectively used to score sentiments of messages that come later, $\mathcal{T}$. Basically, the model is interpreted as a poll, in which each rule $\{\mathcal{X} \rightarrow s_i\} \in \mathcal{R}$ is a vote given by $\mathcal{X}$ for sentiment s_i. Given a message $t \in \mathcal{T}$, a rule $\mathcal{X} \rightarrow s_i$ is only considered as a valid vote if this rule is applicable to t.

Definition 3. A rule $\{\mathcal{X} \rightarrow s_i\} \in \mathcal{R}$ is said to be applicable to message $t \in \mathcal{T}$ if $\mathcal{X} \subseteq t$. That is, if all terms in $\mathcal{X}$ are present in t.

Not all rules in $\mathcal{R}$ are applicable to a specific message $t \in \mathcal{T}$. Eventually, the model may contain many rules that are not applicable to any message in $\mathcal{T}$. These rules are said to be useless, and the set of all useless rules in $\mathcal{R}$ is denoted as $\mathcal{R}_\emptyset$.

We denote as $\mathcal{R}_t$ the set of all rules in $\mathcal{R}$ that are applicable to message $t \in \mathcal{T}$. Thus, only and all the rules in $\mathcal{R}_t$ are considered as valid votes when scoring sentiments in message t. Therefore, for m future messages in $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$, the classification model $\mathcal{R}$ can be decomposed as $\{\mathcal{R}_{t_1} \cup \mathcal{R}_{t_2} \cup \ldots \cup \mathcal{R}_{t_m} \cup \mathcal{R}_\emptyset\}$. Rules in $\mathcal{R}_\emptyset$ represent a waste of computational resources, and may pollute the classification model with irrelevant information. Ideally $|\mathcal{R}_\emptyset| = 0$.

Also, we denote as $\mathcal{R}_t^{s_i}$ the subset of $\mathcal{R}_t$ containing only rules predicting sentiment s_i. Votes in $\mathcal{R}_t^{s_i}$ have different weights, depending on the confidence of the corresponding rules. The weighted votes for sentiment s_i are averaged, giving the score for sentiment s_i with regard to message t, as shown in Equation 1:

$$s(t, s_i) = \sum \frac{\theta(\mathcal{X} \rightarrow s_i)}{|\mathcal{R}_t^{s_i}|} \tag{1}$$

Finally, the scores are normalized, as expressed by the scoring function $\hat{p}(s_i|t)$, shown in Equation 2. The scoring function estimates the likelihood of sentiment s_i being the implicit attitude of message t.

$$\hat{p}(s_i|t) = \frac{s(t, s_i)}{\sum_{j=0}^{k} s(t, s_j)} \tag{2}$$

3.2 Self-augmenting Training and Projection

The performance associated with (static) classification models tends to deteriorate over time. This is mainly due to *sentiment drift* [32], which happens when data distribution in $\mathcal{T}$ is different from that in $\mathcal{D}$. The difference usually increases over time, and at some point in time the training data may eventually become meaningless and the classification model obsolete. Drift is commonly observed in streaming environments, being evidenced either when the sentiment distribution shifts, or when the relationship between textual-patterns and sentiments changes [12]. In both cases the need for model adaptation is prevalent in order to track changing-sentiments over time.

Data Inclusion

In order to adapt the classification model accordingly, it is mandatory to gather the most current information emerging in the stream. Latest, most current training messages, may be obtained by exploiting the predictions performed using the sentiment-scoring function shown in Equation 2. These predictions may be used to assign sentiments to messages, generating labeled messages. Further, reliable predictions may be regarded as correct ones and generate reliable labeled messages, which can be included into $\mathcal{D}$.

Definition 4. Given an arbitrary message $t \in \mathcal{T}$, we say that $< t, s_i >$ is a reliable labeled message if $\hat{p}(s_i|t) \geq \delta_{min}$, where δ_{min} is a user-specified threshold ($0.0 < \delta_{min} \leq 1.0$).

The idea is to use δ_{min} as a threshold indicating the minimum reliability necessary to regard labeled message $< t, s_i >$ as a correct one, and, therefore, to include it into the training data $\mathcal{D}$. Intuitively, if reliable predictions are indeed correct ones, then the training data will be continuously augmented with novel training information, keeping the training data up-to-date as the stream evolves. However, the use of support-based pruning during rule extraction prevents the full potential of self-augmenting training, since it is highly probable that the classification model $\mathcal{R}$ will be composed only of the most general rules in $\mathcal{D}$, and most of these rules may be not applicable to future messages carrying trending (i.e., not so frequent) information.

Definition 5. Given a message $t \in \mathcal{T}$, we say that model $\mathcal{R}$ is agnostic to t, if $\mathcal{R}_t = \emptyset$. That is, if $\mathcal{R}$ does not contain rules that are applicable to t. A model $\mathcal{R}$ is said to be gnostic if $\mathcal{R}_t \neq \emptyset$ $\forall t \in \mathcal{T}$.

Unfortunately, an optimal minimum support value that guarantees a gnostic classification model, is unlikely to exist. Therefore, as we discuss in the next section, our rule extraction must be support-free, in order to produce gnostic models, and to exploit the full potential and all the benefits of self-augmenting training.

Online Rule Extraction with Data Projection

So far we have discussed offline rule extraction. Extracting rules on-the-fly, however, offers several advantages. One of these advantages is that the classifiers become able to efficiently extract rules from $\mathcal{D}$ without performing support-based pruning. The idea behind online rule extraction is to avoid completely the extraction of useless rules by projecting the training data on a demand-driven basis. More specifically, rule extraction is delayed until a message $t \in \mathcal{T}$ is given. Then, terms in t are used as a filter which configures the training data $\mathcal{D}$ in a way that only rules that are applicable to t can be extracted. This filtering process produces a projected training data, denoted as $\mathcal{D}_t$, which contains only terms that are present in message t.

Lemma 1. All rules extracted from $\mathcal{D}_t$ are applicable to t.

Proof. Since all training messages in $\mathcal{D}_t$ contain only terms that are present in message t, the existence of a rule $\mathcal{X} \rightarrow s_i$ extracted from $\mathcal{D}_t$, such that $\mathcal{X} \nsubseteq t$, is impossible. ■

Lemma 1 implies that demand-driven training projection assures that $|\mathcal{R}_\emptyset| = 0$, evidencing that only useless rules are not included into the classification model $\mathcal{R}$. The next theorem states that our classifier efficiently extracts rules from $\mathcal{D}$, no matter the minimum-support value (which can be arbitrary low). The key intuition is that the classifier works only on terms that are known to be associated to each other, drastically narrowing down the search space for rules.

Theorem 1. The number of rules extracted from $\mathcal{D}_t$ increases polynomially with the number of distinct terms in $\mathcal{D}$.

Proof. Let n be the number of distinct terms in $\mathcal{D}$. Since an arbitrary message $t \in \mathcal{T}$ contains at most l terms (with $l \ll n$), then any rule applicable to t can have at most l terms in its antecedent. That is, for any rule $\{\mathcal{X} \rightarrow s_i\}$, such that $\mathcal{X} \subseteq t$, $|\mathcal{X}| \leq l$. Consequently, the number of possible rules that are applicable to t is $l + \binom{l}{2} + \ldots + \binom{l}{l} = O(2^l) \ll O(n^l)$. Thus, the number of applicable rules increases polynomially in n. ■

Another advantage provided by demand-driven training projection comes from the fact that the projection also exploits, as a side-effect, the temporal locality associated with terms in $\mathcal{D}$. This is particularly important for dealing with sentiment drift, since by projecting the training data according to the content of a message $t \in \mathcal{T}$, the classifier is essentially concentrating the representative training information in nearly contiguous temporal data frames. However, deciding about the recency of the frame is a tricky issue, since different messages in $\mathcal{T}$ may demand training frames positioned in different points of the stream timeline. That is, some messages in $\mathcal{T}$ may demand more recent training frames, while other messages in $\mathcal{T}$ may demand older ones. Thus, instead of employing a fixed-length temporal training frame for all messages in $\mathcal{T}$, our classifier employs a different frame (i.e., $\mathcal{D}_t$) for each message in $t \in \mathcal{T}$. The recency of the training frame for an arbitrary message t is decided based upon the terms that are in the own message. Since these terms are chronologically related somehow, the projected training data $\mathcal{D}_t$ is likely to contain representative training messages for scoring the sentiments in message t.

Extending Classification Models Dynamically

With online rule extraction, we extend the classification model $\mathcal{R}$ dynamically as messages in $\mathcal{T}$ are processed. Initially $\mathcal{R}$ is empty; a sub-model $\mathcal{R}_{t_i}$ is appended to $\mathcal{R}$ every time the classifier processes a message t_i. Thus, after processing a sequence of m messages $\{t_1, t_2, \ldots, t_m\}$, the model $\mathcal{R}$ is $\{\mathcal{R}_{t_1} \cup \mathcal{R}_{t_2} \cup \ldots \cup \mathcal{R}_{t_m}\}$, and therefore, $\mathcal{R}$ is gnostic to all those m messages.

Producing a sub-model $\mathcal{R}_t$ involves extracting rules from $\mathcal{D}_t$. This operation has a significant computational cost, since it is necessary perform multiple accesses to $\mathcal{D}$. Different messages in $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ may demand different sub-models $\{\mathcal{R}_{t_1}, \mathcal{R}_{t_2}, \ldots, \mathcal{R}_{t_m}\}$, but different sub-models may share some rules (i.e., $\{\mathcal{R}_{t_i} \cap \mathcal{R}_{t_j}\} \neq \emptyset$). In this case, memorization is very effective in avoiding work replication, reducing the number of data access operations. Thus, before extracting rule $\mathcal{X} \rightarrow s_i$, the classifier first checks whether this rule is already in $\mathcal{R}$. If an entry is found with a key matching $\{\mathcal{X}, s_i\}$, then the rule in $\mathcal{R}$ is used instead of extracting it from $\mathcal{D}_t$. If it is not found, the rule is extracted from $\mathcal{D}_t$ and then it is inserted into $\mathcal{R}$. The main steps are summarized in Algorithm 1.

Algorithm 1 Online Rule Extraction

Require: message $t \in \mathcal{T}$ and $\mathcal{D}$
Ensure: $\mathcal{R}_t$ and $\mathcal{R}$

1: $\mathcal{D}_t \Leftarrow \mathcal{D}$ projected according to message t
2: $\mathcal{R}_t \Leftarrow$ rules $\{\mathcal{X} \rightarrow s_i\} \notin \mathcal{R}$, extracted from $\mathcal{D}_t$
3: append $\mathcal{R}_t$ to $\mathcal{R}$

Model Maintenance

Entries in the classification model $\mathcal{R}$ may become invalid when reliable labeled messages $< t, s_i >$ are included into $\mathcal{D}$. As a result, $\mathcal{R}$ has to be updated properly. We propose to maintain the model up-to-date incrementally, so that the updated model is exactly the same one that would be obtained by re-constructing it from scratch.

Update speed is a key issue in model maintenance, and a challenge that threatens the efficiency of our approach is that the model may be composed of a potentially large number of rules, and updating all these rules may be unacceptably costly in a streaming environment. Fortunately, not all rules in $\mathcal{R}$ have to be updated.

Lemma 2. The inclusion of a labeled message $< t, s_i >$ into $\mathcal{D}$ does not change the value of $\sigma(\mathcal{X})$, for any termset $\mathcal{X} \not\subset t$.

Proof. Since $\mathcal{X} \not\subset t$, the number of messages in $\mathcal{D}$ having $\mathcal{X}$ as a subset is essentially the same as in $\{\mathcal{D} \cup t\}$. ■

Lemma 3. The inclusion of a labeled message $< t, s_i >$ into $\mathcal{D}$ does not change the value of $\theta(\mathcal{X} \rightarrow s)$, for any rule $\{\mathcal{X} \rightarrow s\} \in \mathcal{R}$ for which $\mathcal{X} \not\subset t, \forall s \in \{s_1, s_2, \ldots, s_k\}$.

Proof. Comes directly from the fact that confidence is invariant under the null-addition operation [30]. ■

From Lemmas 2 and 3, the number of rules that have to be updated due to the inclusion of labeled message $< t, s_i >$, is bounded by the number of possible termsets in t. Since most of the messages that are included into $\mathcal{D}$ contain only a very small fraction of all possible termsets, the inclusion of an arbitrary message t corresponds to a null-addition to most of the rules in $\mathcal{R}$. The following lemma states exactly the rules in $\mathcal{R}$ that have to be updated.

Lemma 4. The only and all the rules in $\mathcal{R}$ that must be updated due to the inclusion of labeled message $< t, s_i >$ are those in $\mathcal{R}_t$.

Proof. All rules $\{\mathcal{X} \rightarrow s_i\} \in \mathcal{R}$ that have to be updated due to the inclusion of $< t, s_i >$ are those for which $\mathcal{X} \subseteq t$. By definition, $\mathcal{R}_t$ contains only and all such rules. ■

Once rules $\{\mathcal{X} \rightarrow s\} \in \mathcal{R}_t$ are retrieved from $\mathcal{R}$, updating the corresponding values for $\sigma(\mathcal{X})$ and $\theta(\mathcal{X} \rightarrow s)$ is a simple operation. It suffices to iterate on $\mathcal{R}_t$ and increment the values of $\sigma(\mathcal{X})$ and $\sigma(\mathcal{X} \cup s)$. The corresponding values for $\theta(\mathcal{X} \rightarrow s)$ are obtained by computing $\frac{\sigma(\mathcal{X} \cup s)}{\sigma(\mathcal{X})}$ $\forall s \in \{s_1, s_2, \ldots, s_k\}$. These steps are summarized in Algorithm 2.

Algorithm 2 Incremental Model Maintenance

Require: labeled message $< t, s_i >$, $\mathcal{D}$, and $\mathcal{R}_i$
Ensure: $\mathcal{R}$

1: **for all** rules $\{\mathcal{X} \rightarrow s\} \in \mathcal{R}_t$ **do**
2: increment $\sigma(\mathcal{X})$
3: increment $\sigma(\mathcal{X} \cup s_i)$
4: $\theta(\mathcal{X} \rightarrow s_i) \Leftarrow \frac{\sigma(\mathcal{X} \cup s_i)}{\sigma(\mathcal{X})}$
5: **end for**

The Sub Judice Strategy

Naturally, some predictions are not reliable enough, given certain values of δ_{min}. An alternative is to abstain from using such doubtful predictions as the classifier does not have enough evidence for a reliable judgement, that is, we do not use the corresponding labeled messages for model building and keep them *sub judice*. As new reliable labeled messages are included into $\mathcal{D}$, new sentiment

evidence is exploited, hopefully increasing the reliability of the labeled messages that were previously hold and releasing them. More specifically, when a reliable labeled message is included into $\mathcal{D}$, the classifier re-evaluates all messages that are *sub-judice*. At the end of the process, either doubtful messages become reliable ones (possibly improving prediction performance), or there is no more reliable labeled messages to be included into $\mathcal{D}$ and, therefore, the remaining messages that are *sub-judice* have to be processed normally. The process stops when all messages in $\mathcal{T}$ are processed by the classifier. The main steps are summarized in Algorithm 3.

Algorithm 3 Blocking Doubtful Predictions

Require: message $t \in \mathcal{T}$, δ_{min}
Ensure: $\mathcal{D}$

1: **if** $\hat{p}(s_i|t) < \delta_{min}$
2: keep t *sub judice* until another labeled message is included into $\mathcal{D}$
3: **else**
4: include labeled message $< t, s_i >$ into $\mathcal{D}$
5: **end if**

As will be discussed in the next section, messages in the stream are kept *sub-judice* for a certain period of time, which depends on the application (i.e., minutes, hours, days etc.). After this period, all messages are necessarily processed.

4. EXPERIMENTAL EVALUATION

In this section we empirically analyze the sentiment scoring performance of our classifiers. We employ the mean squared error (MSE) as the basic evaluation measure in our experiments, since we are primarily interested in evaluating sentiment scoring (rather than sentiment prediction). In order to evaluate the scoring performance over the time, we employ the area under the curve (AUC)[1]. We used Multinomial Naive Bayes [4] as baseline, since it is a representative of the state-of-the-art. All datasets used in our experiments were manually labeled by three to five human annotators. Unless otherwise stated, the training seed that is provided to the classifiers are composed by the first 1% of the messages. In the following we describe the datasets, and then we discuss the scoring performance of our classifiers on these datasets[2].

4.1 Brazilian Presidential Elections

The presidential election campaigns were held from June to October 2010. Candidate Dilma Rousseff launched a Twitter page during a public announcement, and she used Twitter as one of the main sources of information for her voters. The campaign attracted more than 500,000 followers and Dilma was the second most cited person on Twitter in 2010. The election came to a second round vote, and Dilma Rousseff won the runoff with 56% of the votes.

Dilma Rousseff Election Campaign. We collected 466,724 Portuguese messages referencing Dilma Rousseff in Twitter during her campaign. We randomly selected 66,643 of these messages, and we annotated them in order to track the population sentiment of approval during this period. Approval varied greatly due to several polemic statements and political attacks, and our goal is to score approval during her campaign. The dataset contains 62,089 distinct terms, and messages are grouped by day (i.e., all messages posted in the same day are placed together in the same group). Messages in the stream come in at a rate of 0.02 messages/sec.

Figure 1 shows a series of results obtained for the evaluation of our classifiers in this dataset. Figure 1-a shows a colored map which allows us to grasp the existence of temporal locality of messages passing in the stream. A message exhibits temporal locality if it is likely to be accessed again in the near future, that is, message $d \in \mathcal{D}$ becomes more likely to be in $\mathcal{D}_{t_i}$ and in $\mathcal{D}_{t_j}$ if $t_i \in \mathcal{T}$ is close in time with $t_j \in \mathcal{T}$. In the figure, messages placed in lighter colored regions are those that appeared in the projected training data of the corresponding message in the x-axis. Messages placed in darker regions, on the other hand, are those that did not appear in the projected training data of the corresponding message in the x-axis. Since messages in the x- and y-axes are chronologically ordered, we can understand how frequently these messages are used over the time. Clearly, messages are gradually less and less used as the stream passes, and future messages tend to demand messages that were just included into $\mathcal{D}$. Also, messages have an expiration time, after which they become useless for the sake of prediction. For instance, the first messages to appear in the stream become meaningless after about 30,000 messages are processed by the classifier. Our classifiers are able to automatically discard such meaningless messages while building classification models.

Figure 1-b shows stacked histograms indicating the percentage of messages in $\mathcal{T}$ that were correctly, wrongly, and not included into $\mathcal{D}$ by varying the value of δ_{min}. As expected, the percentage of messages that are included into $\mathcal{D}$ increases as δ_{min} decreases. This is because message inclusion becomes less restrictive for lower values of δ_{min}. For the same reason, the percentage of messages wrongly included into $\mathcal{D}$ increases as δ_{min} decreases. Figure 1-c shows the same analysis, but allowing the *sub judice* strategy. For this dataset, this strategy was clearly effective, since the percentage of correctly included messages always increases, while the percentage of wrongly included messages always decreases. This is because the *sub judice* strategy makes the classifier able to abstain from doubtful predictions until more evidence is gathered with the inclusion of reliable training information, and this greatly improves scoring performance. For instance, for $\delta_{min} = 0.7$ and using a seed size of 2/100, we observed that, on average, 8% of all messages were kept *sub judice* and 225 messages were inserted into $\mathcal{D}$ at each day (which was the time period for which messages are allowed to be kept *sub-judice*).

Figure 1-d shows the online population approval sentiment over the campaign. We try to approximate sentiment variations using our classifiers. As can be seen, a better sentiment approximation is obtained when the classifier is allowed to perform the *sub judice* strategy. Figure 1-e shows the error area under the curve for different values of δ_{min} and for different training seed sizes. Since our basic performance metric is MSE, the smaller the area under the curve, the better is the performance. As expected, performance increases by increasing the seed size, since in this case more training messages are available. The best performance provided by our classifier ($\delta_{min} = 0.8$) is highly competitive with the performance provided by the baseline. Figure 1-f shows the same analysis, but allowing the classifier to perform the *sub judice* strategy. Most of the results have improved greatly – in some cases the error area decreases by more than 30%, and the resulting performance is much superior than the performance provided by the baseline.

4.2 TIME's Person of the Year

At the end of every year, TIME magazine selects a person, or a

[1]Specifically, we calculate the area under the curve induced by the MSE associated with chunks of messages, that is, messages are grouped by minutes, hours or days depending on the application.
[2]We cannot redistribute the datasets due to Twitter restrictions (`http://dev.twitter.com/pages/api_terms`)

group of persons that has most influenced events during the year. The chosen person for 2010 was Mark Zuckerberg. The reader choice, however, was Julian Assange, with an overwhelming superiority of votes.

Twittersphere Battle: Zuckerberg and Assange. We collected 93,411 English messages referencing Julian Assange and Mark Zuckerberg from 12-15-2010 to 12-21-2010. We randomly selected 5,616 of these messages, and we annotated them in order to track diverse sentiments regarding the magazine's decision. Sentiments include surprise (since the reader choice was pointing to Julian Assange), approval/disapproval, and even fury. The dataset contains 7,294 distinct terms, and messages are grouped by hour. Messages in the stream come in at a rate of 0.02 messages/sec.

Figure 2 shows a series of results obtained for the evaluation of our classifier in this dataset. Figure 2-a shows the effectiveness of training augmentation. The percentage of correctly included messages is approximately the same for δ_{min} values varying from 0.5 to 0.8. However, for $\delta_{min} = 0.8$, no message was wrongly included into $\mathcal{D}$. As a result, the best scoring performance, as shown in Figure 2-b, was obtained with $\delta_{min} = 0.8$. In this case, the area under the curve is essentially the same one obtained by the baseline. However, as the figure also shows, our classifiers obtained much better results when the *sub judice* strategy is allowed. In this case we observed that, on average, 33% of all messages were kept *sub judice* and 60 messages were included into $\mathcal{D}$ at each hour. Figure 2-c shows a X-Y scatter-plot which correlates the moment in time in which a message arrives and the moment in time in which the same message is processed. As shown in the figure, messages are blocked until it becomes possible to perform reliable predictions for them. The first messages to pass through the stream are so doubtful given only the training seed, that the corresponding predictions are kept *sub judice* even for relaxed δ_{min} values (i.e., $\leq$ 0.6). As more labeled messages are included into $\mathcal{D}$, predictions become more reliable, and only predictions with reliability below $\delta_{min} = 0.8$ are kept *sub judice*.

4.3 FIFA World Cup

The 2010 Soccer World Cup involved 32 teams competing for the title. The Brazilian team was defeated by the Dutch team on 07-02-2010, after a controversial match. The Brazilian team scored first, but soon after the Dutch team scored twice and won the match. A specific player, Felipe Melo, had decisive participation (for better or worse) in all three goals.

The Brazilian Defeat. We collected 12,020 messages referencing Felipe Melo. We randomly selected 4,646 of these messages, and we annotated them in order to track the sentiment of appreciation for the participation of Felipe Melo. This resulted in two datasets, the first one containing 3,214 annotated messages in Portuguese (8,101 distinct terms), and the second one containing 1,432 annotated messages in English (4,962 distinct terms). For these datasets, messages are grouped by minute. Messages in the stream come in at a rate of 1.12 messages/sec.

Figures 3 and 4 show a series of results obtained for the evaluation of our classifiers in these datasets. We start by discussing the results regarding the dataset composed of messages in Portuguese. Figure 3-a concerns the temporal locality associated with the messages in this dataset. Old messages are gradually less and less used as the stream passes. For instance, the first messages to appear in the stream become meaningless after about 2,000 messages are processed by the classifier. It is also clear that messages exhibit temporal locality, since, as can be seen, a given message is more likely to be used again in the near future.

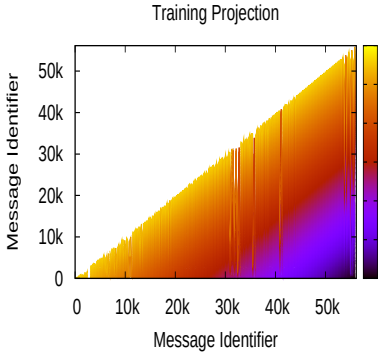
Figure 3-b shows the cumulative MSE as the stream passes. The static classifier (i.e., the training data is never augmented with new training messages) offers the worst performance, and the dynamic classifier reaches its maximum performance for $\delta_{min} = 0.5$. This δ_{min} value was used to track appreciation sentiment over time, as shown in Figure 3-c. As can be seen, there was a sudden appreciation increase in the beginning of the match. This is because the queried player, Felipe Melo, made an assistance to a goal. There was also a sudden decrease in appreciation after one hour of match, and this has happened because the same queried player failed two consecutive times, allowing the adversary to score twice, winning the match consequently. As shown in Figure 3-c, the dynamic classifier offers a much better approximation, when compared against the static classifier. The superiority is due to the inclusion of new training messages into $\mathcal{D}$, as shown in Figure 3-d. For $\delta_{min} = 0.5$, the percentage of correctly included messages surpasses 95%.

Figure 3-e shows the error area under the curve obtained for different values of σ_{min}. For most of δ_{min} values, our classifier was able to provide a better performance than the performance provided by the baseline. The performance increases even more if the classifier is allowed to perform the *sub judice* strategy, as shown in the same figure. We finish the analysis of the results obtained for this dataset by inspecting the *sub judice* strategy. Figure 3-f correlates the time in which a message arrives with the time in which the same message is processed. As can be seen, messages are blocked until it becomes possible to perform reliable predictions for them. A large number of messages were blocked exactly just after one hour of match, when there was a sudden sentiment drift. This shows that the *sub judice* strategy is effective for dealing with sentiment drift (as shown in Figure 3-e), even if the drift is huge, such as the one depicted in Figure 3-c.

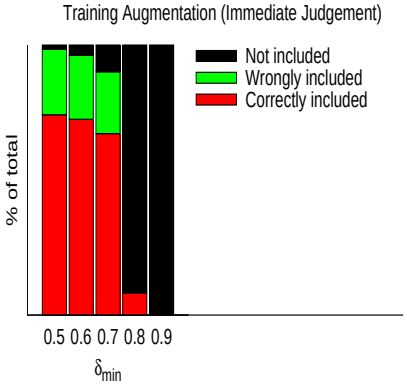
The last set of experiments concerns the evaluation of the same event, but using the dataset composed of messages in English. Figure 4-a shows the cumulative MSE as the stream passes. Again, the static classifier offers the worst results. Performance improves greatly when $\delta_{min} \leq 0.8$. In these cases, the number of correctly included messages surpasses 92%, as shown in Figure 4-b. As expected, the percentage of wrongly included messages increases as δ_{min} decreases, and thus, the best performance is achieved for $\delta_{min} = 0.7$, as shown in Figure 4-c. Again, the *sub judice* strategy offers a substantial improvement, and when it actually came into action, it is able to decrease the error area by at least 15%.

5. CONCLUSIONS

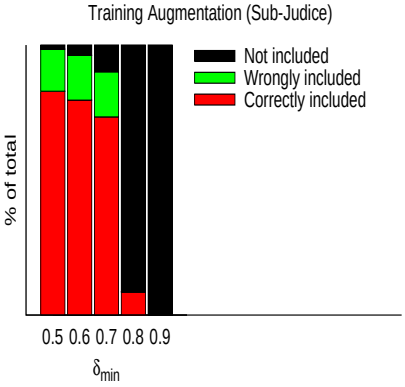
This paper focused on the important problem of sentiment analysis in streaming environments. We have introduced new classifiers based on sentiment rules. The proposed classifiers are able to exploit reliable predictions in order to augment the training data. The self-augmentation training procedure keeps the classifier up-to-date automatically. Furthermore, sentiment rules are extracted from the training data on a demand-driven basis, by projecting the search space for sentiment rules according to qualitative information in future messages, allowing an efficient extraction. Also, by projecting the training data, the classifier eliminates irrelevant and outdated information from consideration. This happens as a side effect, because messages coming in the stream exhibits temporal locality, and old messages are unlikely to be demanded by recent messages passing through the stream. We also show that training messages have an expiration time, after which they are totally useless for the sake of classification. Our classifier are able to discard from consideration all such meaningless information. A systematic



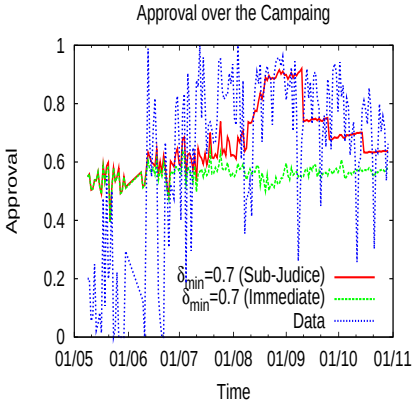
(a) Projected Training Data for each message in $\mathcal{T}$. Given a message x in the x-axis, the figure shows the corresponding $\mathcal{D}_x$ in the y-axis. Lighter colored regions indicate the presence of the corresponding message in $\mathcal{D}_x$.



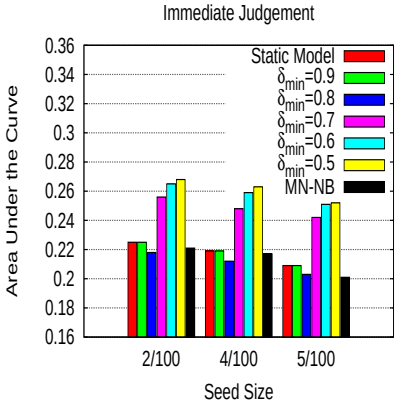
(b) Stacked histograms showing the percentage of messages in $\mathcal{T}$ that were correctly/wrongly/not included into $\mathcal{D}$, for different values of δ_{min} and using seed size of 2/100. No abstention is allowed (i.e., immediate judgement).



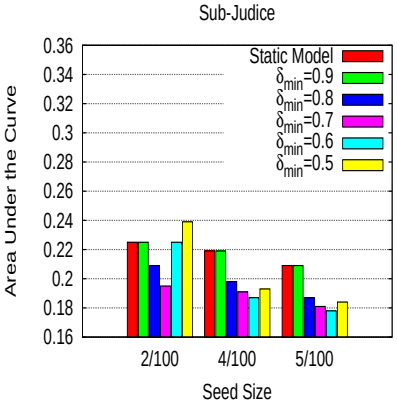
(c) Stacked histograms showing the percentage of messages in $\mathcal{T}$ that were correctly/wrongly/not included into $\mathcal{D}$, for different values of δ_{min} and using seed size of 2/100. Abstention is allowed (*sub judice*).



(d) Tracking population approval sentiment over the campaign.

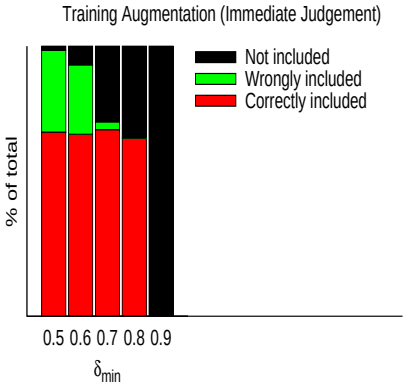


(e) Area under the curve for different values of δ_{min}. Abstention is not allowed (i.e., immediate judgement).

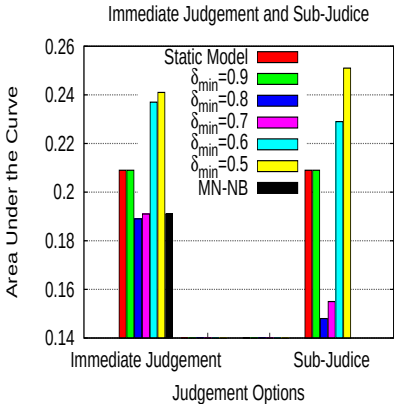


(f) Area under the curve for different values of δ_{min}. Abstention is allowed (i.e., *sub judice*).

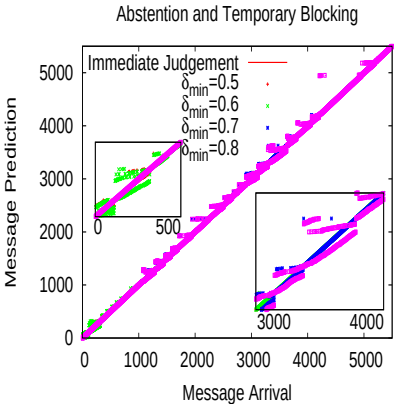
Figure 1: Dilma Rousseff's Presidential Campaign (Portuguese).



(a) Percentage of messages in $\mathcal{T}$ that were correctly/wrongly/not included into $\mathcal{D}$, for different values of δ_{min}.

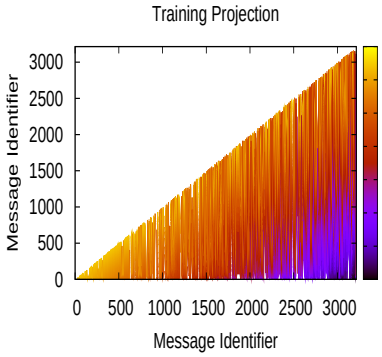


(b) Area under the curve for different values of δ_{min}.

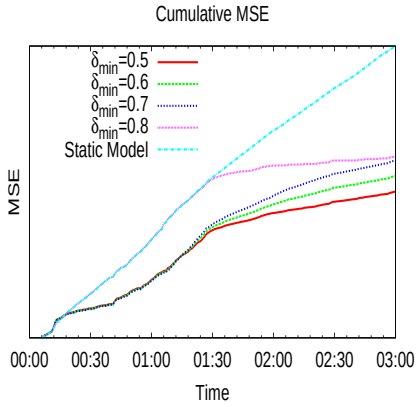


(c) X-Y scatter plot correlating message arrival and prediction, for different values of δ_{min}.

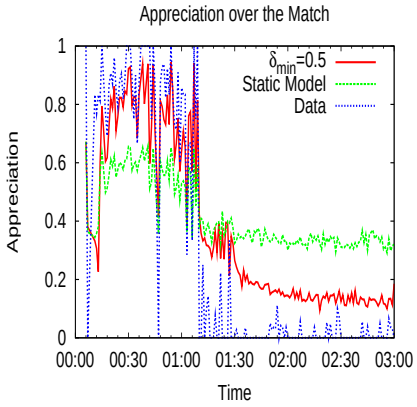
Figure 2: Person of the Year (English).



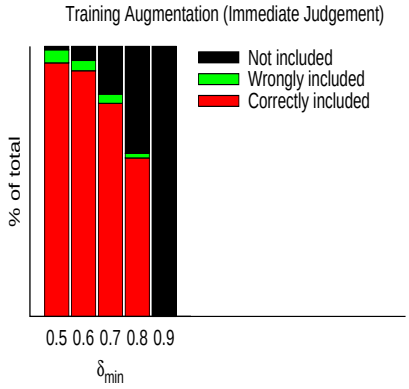
(a) Projected Training Data for each message in $\mathcal{T}$.



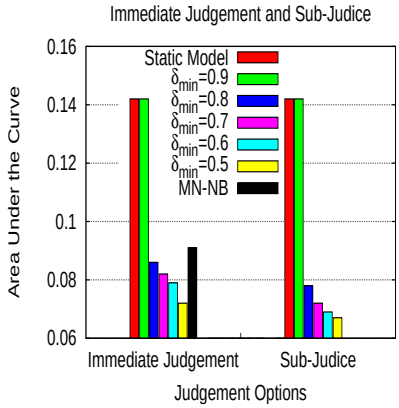
(b) Cumulative MSE for different values of δ_{min}. Messages are grouped by minute, and a MSE value is calculated for each group.



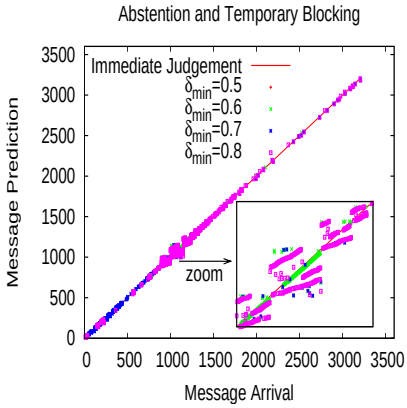
(c) Tracking appreciation sentiment over the match.



(d) Percentage of messages in $\mathcal{T}$ that were correctly/wrongly/not included into $\mathcal{D}$, for different values of δ_{min}.

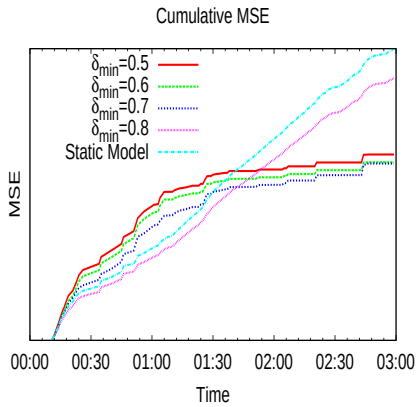


(e) Area under the curve for different values of δ_{min}.

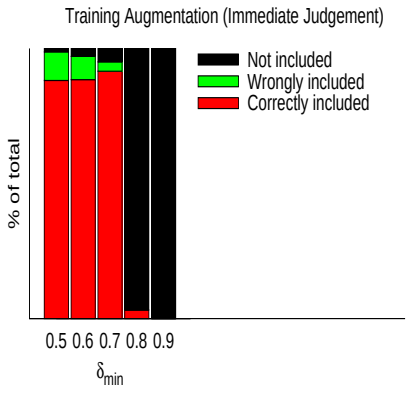


(f) X-Y scatter plot correlating message arrival and prediction, for different values of δ_{min}.

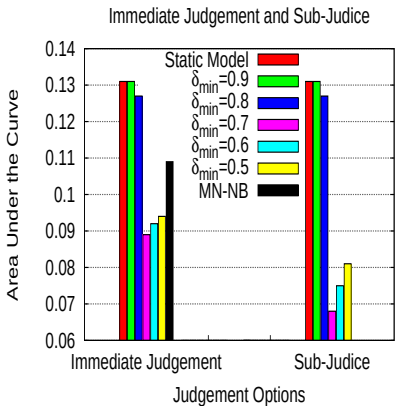
Figure 3: Brazilian Defeat in the FIFA Cup (Portuguese).



(a) Cumulative MSE for different values of δ_{min}. Messages are grouped by minute, and a MSE value is calculated for each group.



(b) Percentage of messages in $\mathcal{T}$ that were correctly/wrongly/not included into $\mathcal{D}$, for different values of δ_{min}.



(c) Area under the curve for different values of δ_{min}.

Figure 4: Brazilian Defeat in the FIFA Cup (English).

evaluation involving major 2010 events demonstrated the effectiveness of the proposed classifiers, which were compared against a representative of the state-of-the-art.

We proposed to further improve the prediction performance of these classifiers by introducing the innovative *sub-judice* strategy, which makes the classifiers able to abstain from doubtful predictions, and to temporarily block these predictions until more evidence is gathered due to the inclusion of new reliable training information. Experimental evaluation has shown that the *sub judice* strategy substantially boosts prediction performance providing gains up to 58%.

6. ACKNOWLEDGMENTS

This research was sponsored by UOL (www.uol.com.br) through its UOL Bolsa Pesquisa program process number 20110215215201, CNPq, Capes, Fapemig and Inweb - the Brazilian National Institute of Science and Technology for the Web.

7. REFERENCES

[1] Nielsen online report. social networks & blogs now 4th most popular online activity. 2009.

[2] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216. ACM, 1993.

[3] R. Bayardo, B. Goethals, and M. Zaki, editors. *Workshop on Frequent Itemset Mining Implementations*, volume 126, 2004.

[4] A. Bifet and E. Frank. Sentiment knowledge discovery in twitter streaming data. In *Discovery Science*, pages 1–15, 2010.

[5] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavalda. New ensemble methods for evolving data streams. In *SIGKDD*, pages 139–148, 2009.

[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and regression trees. *Wadsworth Intl.*, 1984.

[7] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, 2006.

[8] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[9] N. Diakopoulos and D. Shamma. Characterizing debate performance via aggregated twitter sentiment. In *CHI*, pages 1195–1198, 2010.

[10] P. Domingos and G. Hulten. Mining high-speed data streams. In *SIGKDD*, pages 71–80, 2000.

[11] W. Fan. Streamminer: A classifier ensemble-based engine to mine concept-drifting data streams. In *VLDB*, pages 1257–1260, 2004.

[12] G. Forman. Tackling concept drift by temporal inductive transfer. In *SIGIR*, pages 252–259, 2006.

[13] J. Gama, R. S. ao, and P. Rodrigues. Issues in evaluation of stream learning algorithms. In *SIGKDD*, pages 329–338, 2009.

[14] J. Gama, R. Rocha, and P. Medas. Accurate decision trees for mining high-speed data streams. In *SIGKDD*, pages 523–528, 2003.

[15] J. Gehrke, V. Ganti, R. Ramakrishnan, and W. Loh. Boat-optimistic decision tree construction. In *SIGMOD*, pages 169–180, 1999.

[16] A. Go, L. Huang, and R. Bhayani. Twitter sentiment classification using distant supervision. In *CS224N Project Report*, 2009.

[17] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining Knowledge Discovery*, 8(1):53–87, 2004.

[18] G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *SIGKDD*, pages 97–106, 2001.

[19] B. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Micro-blogging as online word of mouth branding. In *CHI*, pages 281–290, 2009.

[20] B. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Twitter power: Tweets as electronic word of mouth. *JASIST*, 60(11):2169–2188, 2009.

[21] J. Kolter and M. Maloof. Dynamic weighted majority: an ensemble method for drifting concepts. *Journal of Machine Learning Research*, 8:2755–2790, 2007.

[22] M. A. Maloof. Incremental rule learning with partial instance memory for changing concepts. In *IJCNN*, pages 2764–2769. IEEE, 2003.

[23] B. O'Connor, R. Balasubramanyan, B. Routledge, and N. Smith. From tweets to polls: Linking text sentiment to public opinion time series. In *ICWSM*, pages 122–129, 2010.

[24] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Intl. Conf. on Language Resources and Evaluation*, pages 1320–1326, 2010.

[25] B. Pang, L. Lee, , and S. Vaithyanathan. Thumbs up? sentiment classiŕcation using machine learning techniques. In *EMNLP*, pages 79–86, 2002.

[26] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2007.

[27] D. Ramage, S. Dumais, and D. Liebling. Characterizing microblogs with topic models. In *ICWSM*, 2010.

[28] J. Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *ACL Student Research Workshop*, pages 43–48, 2005.

[29] W. Street and Y. Kim. A streaming ensemble algorithm (SEA) for large-scale classification. In *SIGKDD*, pages 377–382, 2001.

[30] P. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *SIGKDD*, pages 32–41, 2002.

[31] H. Wang, W. Fan, P. Yu, , and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *SIGKDD*, pages 226–235, 2003.

[32] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[33] M. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *SIGKDD*, pages 283–286, 1997.

[34] P. Zhang, X. Zhu, and Y. Shi. Categorizing and mining concept drifting data streams. In *SIGKDD*, pages 812–820, 2008.

[35] P. Zhang, X. Zhu, J. Tan, and L. Guo. Classifier and cluster ensembles for mining concept drifting data streams. In *ICDM*, pages 1175–1180, 2010.

[36] X. Zhu, P. Zhang, X. Lin, and Y. Shi. Active learning from data streams. In *ICDM*, pages 757–762, 2007.