GEOCENT

WHITE PAPER

Performing Sentiment Analysis with Twitter and Microsoft Dynamics CRM 2011

By: Nikola Kocic, nikola.kocic@geocent.com

Date: June, 2011

Contents

Introduction

Over the last five years, Twitter has grown to be a major influence in our daily lives. For fewer than 140 characters or less, users can pretty much post anything – from what they think, how they feel, or just to give status updates to friends and family members. Companies have began to see the importance of Twitter and its impact of society, and have even started including in their own advertisements directions on how to follow them on Twitter. These companies most probably use some sort of Customer Relationship Management tool, such as Microsoft Dynamics CRM 2011, to maintain their customer relationships. CRM 2011 is an xRM platform (eXtended Relationship Management), which can be extended and customized to fit an organization's needs, and offers out-of-the box functionality such as reporting, sales, marketing, service, and customer management. With the popularity of and effectiveness of Twitter, some startups began providing services which determine if the context of a tweet is positive or negative, which is also known as sentiment analysis. Sentiment analysis is the means of applying natural language processing methods and determining subjective information in source text (Sentiment Analysis). A good example of a website offering such service is Twitter Sentiment (http://twittersentiment.appspot.com).

Abstract

The combination of Twitter and sentiment analysis allows one to determine what the general public or user is feeling pertaining to any subject. These analytics can be used to measure how people feel about a particular service or product. The combination of this already powerful combination of "technology meets applicative method" with the already existing powerful and flexible customer relationship management platform (CRM 2011) will result in a socially-driven customer relationship management tool, which gives further insight into what customers think or feel about a particular product or service. The following white paper outlines the steps taken in creating such a system, which includes including the following components:

- Sentiment analysis service
- Twitter client service
- Windows service which mediates between Twitter, sentiment analysis, and CRM
- CRM 2011 customizations

Technical Overview

The following tools were used for the development of the system:

- Microsoft Visual Studio 2010 SP1
- .NET Framework 4.0, C#
- Entity Framework 4.1
- Microsoft Dynamics CRM 2011
- Microsoft Dynamics CRM 2011 SDK 5.0.3

- SQL Server 2008 R2
- LINQ to Twitter (http://linqtotwitter.codeplex.com)

System Architecture

The following diagram illustrates the overall system architecture.

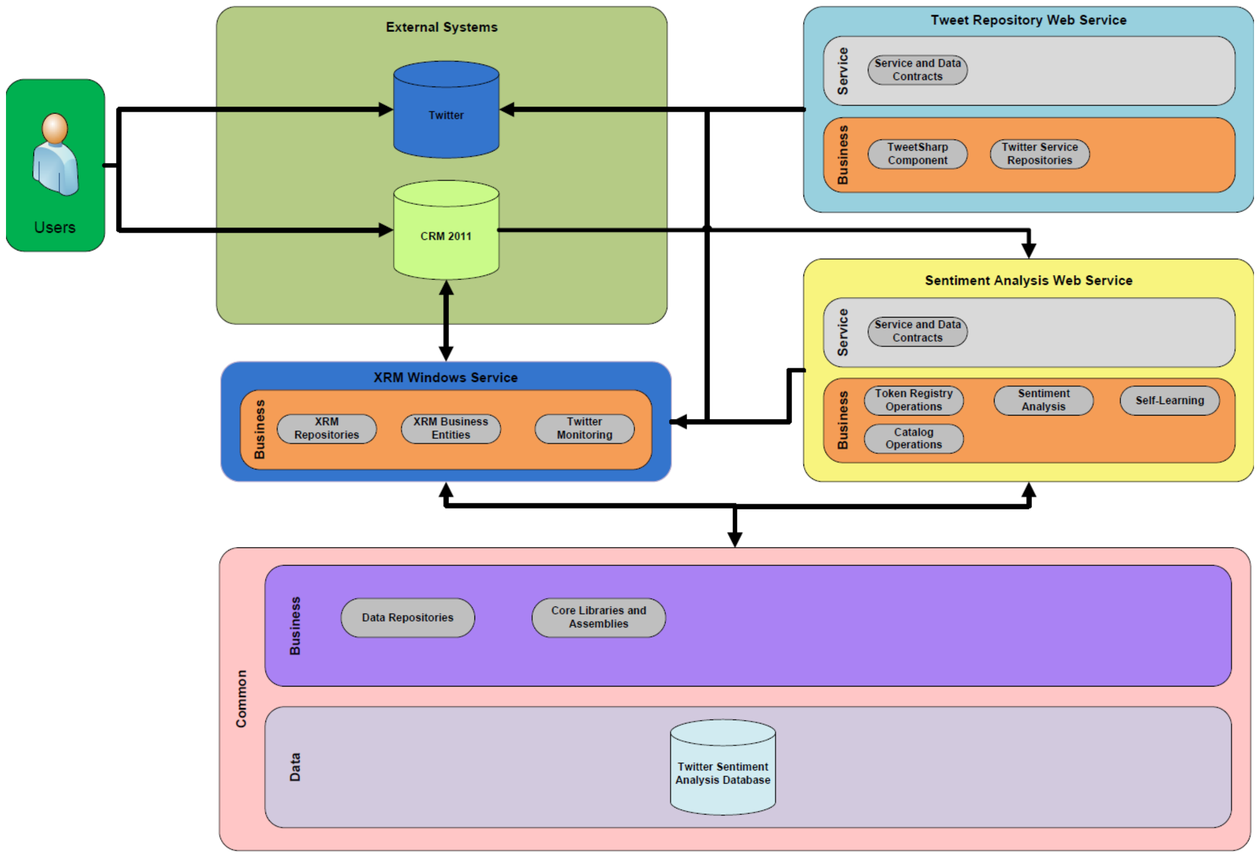


Figure 1 - High Level System Architecture

Sentiment Analysis Service

The sentiment analysis service created for this system is a simple WCF web service which is responsible for performing sentiment analysis on a text. Additionally, it provides the option to update the underlying corpuses and token registry, as well as a mechanism for it to "teach" itself.

At the start of the development, research was done to locate suitable libraries that already had built-in support for sentiment analysis. Initially, Python and the Natural Language Toolkit (NLTK) were considered for usage. NLTK uses the Naïve Bayes classifier, which is based on the Bayes Theorem (Natural Language Toolkit Development). Further research was done to locate a suitable .NET implementation of the Naïve Bayes theorem and an application which implements a NET spam filter (Kester) was used as a starting point. Kester's implementation is also based on Paul Graham's implementation of an algorithm for a spam filter.

Building a Corpus

Kester's implementation of the corpus and the Naïve Bayes classifier was used as a basis of the sentiment analysis service. However, the training of the corpus as well as retaining of the training data was segregated to a database. To build the corpus, a sub-set of a data set containing approximately 3,800,000 tweets was used to train the classifier (Chang, Caverlee, Lee). This subset was imported into database at which point two types of queries were executed; one which would return tweets containing "positive" key words and one that would return tweets containing "negative" keywords. The following keywords were used to return tweets that would be used to train the classifier. Both training sets had a sub-set of 50,000 records.

Positive Key Words: *love, like, beautiful, sweet, beautiful, pretty, joy*

Negative Key Words: *hate, dislike, bad, shit, fuck, horrible, ugly, disgusting*

As a result of checking the contents of both training sets, it was concluded that tweets containing one or more of specified key words contained other words of similar nature. In many cases, tweets contained both positive and negative key words and a combination of other words of the same nature.

In order to build permanent positive and negative corpuses, several database tables were created. In addition to the corpuses, a token registry was built which serves as a common repository for all words extracted from the training data. Following Kester's implementation of the corpus, both training sets were loaded respectively. During the load of each training set, tokens (words that don't start with a number) were extracted and their overall occurrence count was measured. Once both corpuses were loaded, the token registry was built. The following database diagram illustrates the corpus and registry tables.

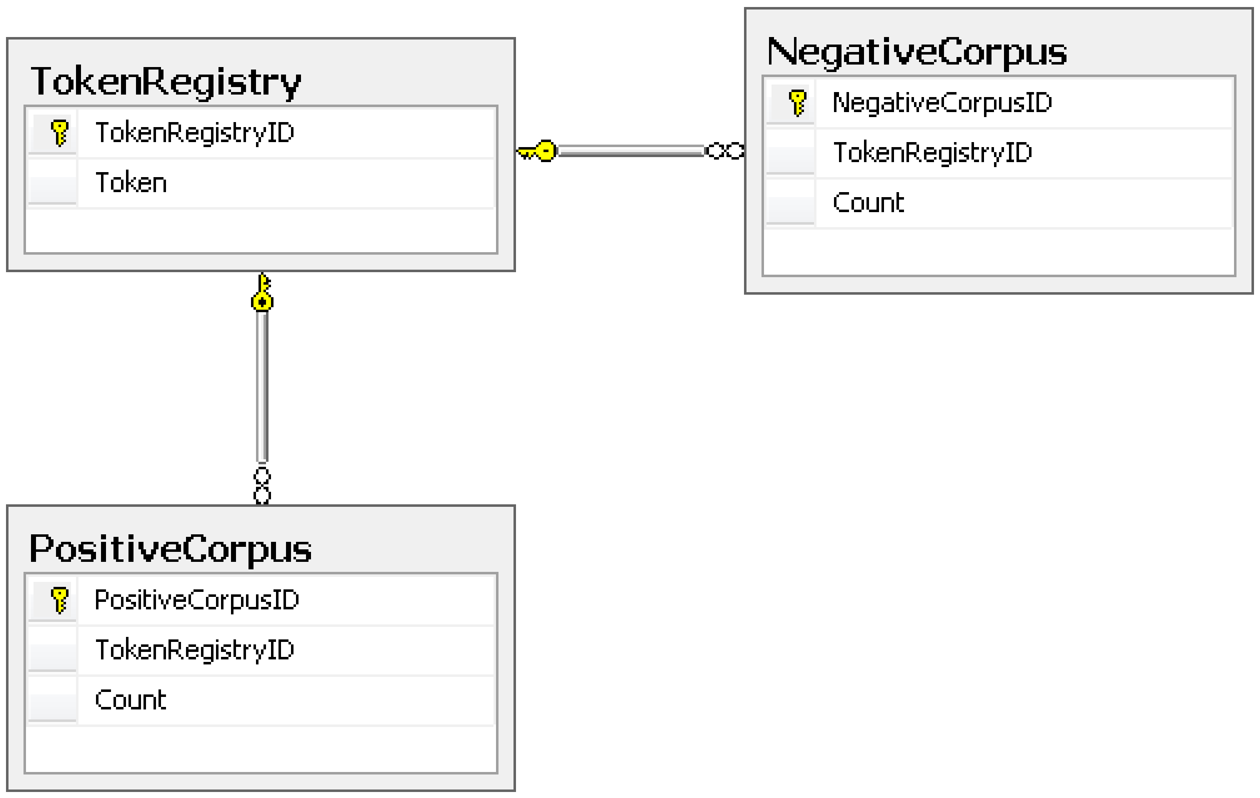
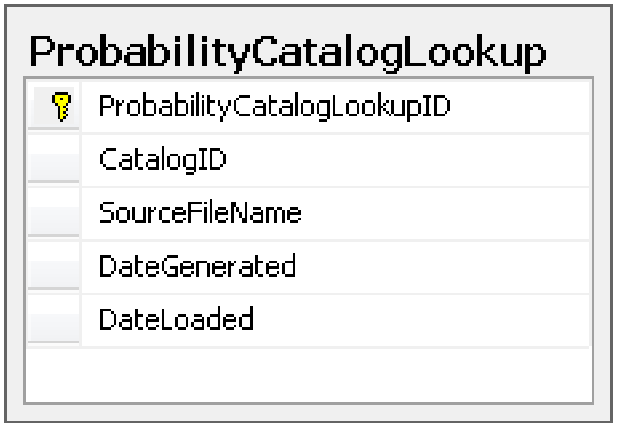


Figure 2 - Corpuses and Token Registry

The resulting table structure is designed so that both corpuses reference an individual token from the registry and keep a count of number of occurrences of the token.

Training the Classifier

For the calculation of probabilities that a word in the token registry will either be "positive" or "negative", Kester used Graham's implementation of a spam filter algorithm to train the classifier. The same approach was used in the training of the classifier and building a probability catalog. In order to retain the training results, the data was serialized into a JSON string and stored in a text file. The output files were being tracked in another database table, called *ProbabilityCatalogLookup*. The following diagram illustrates this table:



ProbabilityCatalogLookup

🔑	ProbabilityCatalogLookupID
	CatalogID
	SourceFileName
	DateGenerated
	DateLoaded

Figure 3 - Probability Catalog Lookup table

The generated output file contains key/value pairs of all words in the registry and their calculated probability. The scale used to determine if the word is "positive" or "negative" is the following:

- Positive: 0 – 0.49
- Neutral: Greater than 0.49 and less than or equal to 0.51
- Negative: Greater than 0.51 and less than or equal to 1

A "neutral" indicator was assigned to words with a probability between 0.49 and 0.51. Whenever the web service is started, it de-serializes the contents of the probability catalog and loads the values into memory.

Tweet Parsing

Before an appropriate sentiment value can be assigned to a Tweet, the contents of the tweet must first be parsed and tokenized. Hash tags, user references (@), URIs, re-tweets, and stop words are removed. This was similar to the approach taken by Pak and Paroubek. Additionally, tokens that start with a numeric character are ignored and not processed (Kester). Stop words specified by Brahaj were used in the processing of the text. The resulting tokens are then passed to the classifier which calculates the probability and the result is returned to the caller.

Self-Learning Capability

In order to ensure that the underlying token registry and corpuses can be updated with new words and their respective counts, the service has a built-in self-learning mechanism. This mechanism works in the following manner:

1. Sentiment analysis is performed on text and appropriate sentiment value is determined. In result is “neutral”, it is completely ignored.
2. All non-existing tokens are added to the token registry; existing token counts are incremented in respective corpus by the new number of occurrences
3. Learning can be invoked through the web service at which point the following occurs:
    a. Probability catalog is rebuilt from the current corpuses in the database
    b. New probability catalog is loaded in memory and web service uses it from that point on.

Twitter Repository Web Service

The Twitter repository web service is WCF web service which is responsible for performing making calls to Twitter and retrieving data with provided criteria. For handling the under-the-hood connections to the Twitter API, LINQ to Twitter was used. In addition, a Twitter developer account was created and the application was registered. This was done so that the underlying Twitter calls are made through OAuth and the maximum calls per hour are 350, instead of 150 if done anonymously.

Data Retrieval

Simple data retrieval mechanisms were implemented so that Tweets can be retrieved from the Twitter API seamlessly. The following describe these mechanisms.

- Retrieve Tweets by User
    - Retrieve first page of a user. This returns 100 recent tweets.
    - Retrieve tweets for user since a provided Tweet.
- Retrieve Tweets by a Hash tag
    - Retrieve first page of all tweets containing provided hash tag

In addition, the service always keeps track of how many calls are remaining for that hour.

CRM Customizations

In order to store Twitter data into CRM and specify criteria, new customizations were made to the default CRM Solution. The following are the customizations that were made:

Tweets

This entity is responsible for storing information pertaining to a Tweet, including its sentiment value.

Fields:

- ***Original ID***; Single Line of Text; Primary Field – Stores the original Tweet identifier from the Twitter Web Service API
- ***Post Date;*** Date and Time; – The date and time when the Tweet was posted
- ***Sentiment;*** Option Set; - Positive, Neutral, or Negative indicates the sentiment value
- ***Sentiment Value*;** Decimal (hidden) – Indicates the numerical value of the sentiment. Acceptable values in range from 0.00 to 1.00
- ***Text***; Single Line of Text – Stores the text value of the Tweet
- **Twitter Author**; Single Line of Text – The name of the author of the Tweet.

Relationships:

- N:1 Relationship with Twitter User

Twitter User

This entity is responsible for tracking which user is to be followed; if that user's Tweets will be stored in CRM; and if that user's tweets are to have sentiment analysis performed on them.

Fields:

- ***First Name:*** Single Line of Text; Stores the first name of the user being followed.
- ***Last Name:*** Single Line of Text; Stores the last name of the user being followed.
- ***User Name:*** Single Line of Text; Primary Field - Stores the actual Twitter username.
- ***Enable Sentiment Analysis;*** Option Set; - Yes, No. Indicates whether or not to enable sentiment analysis for user's tweets.
- **Twitter Page**; Single Line of Text – The URL of the user's Twitter page.

Relationships:

- 1:N Relationship with Tweet

Hashtag

This entity is responsible for tracking which hash tags to pull tweets for.

Fields:

- ***Name:*** Single Line of Text Primary Field - Stores the name of the hash tag to track.
- ***Enable Sentiment Analysis****;* Option Set; - Yes, No. Indicates whether or not to enable sentiment analysis for user's tweets.
- ***Is Tracked****;* Option Set; - Yes, No. Indicates whether or not to track the hash tag.

Relationships:

- N:N Relationship with Tweet

XRM Windows Service

The XRM windows service is standalone windows service client which is responsible for pulling the data from Twitter; applying sentiment analysis to a tweet; and storing the data in CRM. This service was configured to make calls to Twitter every 90 seconds.

CRM SDK

In order to perform data operations with XRM, the early-bound entities approach was used. Early-bound entities were generated as specified in the CRM 2011 SDK.

Algorithm

As stated previously, this service is responsible for interacting with XRM, Twitter, and the sentiment analysis service. Interaction to Twitter is done through the Twitter web repository service. The following describes the general algorithm used to pull Tweets from Twitter; perform sentiment analysis; and store and create Tweets in CRM.

```
var userResults = FetchFollowedActiveTwitterUsersInXrm();

For Each User in userResults
      var tweet;
      var Tweets;

      If User Has Tweets
            tweet = GetRecentTweet(User.UserId)
            Tweets = GetTweetForUserSince(tweet.OriginalId);
      Else
            Tweets = GetFirstPageForUser(User.UserId);
      End For Each

      For Each fetchedTweet in Tweets
            XrmTweet newTweet;

            If User.IncludeInSentimentAnalysis = True

                  SentimentReult result = 
SentimentAnalysisService.Analyze(fetchedTweet.Text);

                  newTweet = CreateXrmTweet(tweet, result);
            Else
                  newTweet = CreateXrmTweet(tweet);
            End If

            CreateXrmRelationshipToTwitterUser(user, newTweet);

            SaveToXrm();

      End For Each
End For Each

var hashTagResults = FetchActiveTrackedHashTagsInXrm();

For Each hashTag in hashTagResults;
      var Tweets;

      Tweets = GetFirstPageForHashTag(hashTag.Name);

      For Each fetchedTweet in Tweets

            XrmTweet newTweet;

            If hashTag.IncludeInSentimentAnalysis = True

                  SentimentReult result = 
SentimentAnalysisService.Analyze(fetchedTweet.Text);

                  newTweet = CreateXrmTweet(fetchedTweet, result);
            Else
                  newTweet = CreateXrmTweet(fetchedTweet);

            End If

            CreateXrmRelationshipToTwitterHashTag(hashTag, newTweet);
            SaveToXrm();

      End For Each
End For Each
```

Conclusion

This paper has shown how the Microsoft Dynamics CRM 2011 platform, Twitter, and sentiment analysis can be used in creating a socially-driven customer relationship management tool which can be used to deliver metrics on what customers are thinking or feeling about a particular product or service. CRM 2011's reporting capabilities can also be used to generate reports and dashboards which would give users a visual insight into what people are feeling about a certain product.

Acknowledgements

Key Contributors:
Dr. Vlajko Kocic, Xavier University, Professor of Mathematics, vkocic@xula.edu

Geocent CRM IPT Members:
Greg Holmes, Greg.Holmes@geocent.com
Barry Moulds, Barry.Moulds@geocent.com
Samac Richardson, Samac.Richardson@geocent.com
Jeremy Langley, Jeremy.Langley@geocent.com
James Lancaster, James.Lancaster@geocent.com

References

Brahaj, Armand. "List of English Stop Words." (n.d.). 14 Apr. 2009. Web. 13 Feb. 2001. <http://armandbrahaj.blog.al/2009/04/14/list-of-english-stop-words/>

Z. Cheng, J. Caverlee, and K. Lee. "You Are Where You Tweet: A Content-Based Approach to Geo-locating Twitter Users. " In Proceeding of the 19th ACM Conference on Information and Knowledge Management (CIKM),Tonronto, Oct 2010. (Bibtex)

Graham, Paul. "A Plan for Spam." (n.d.). Web. 20 Jan. 2011. <http://www.paulgraham.com/spam.html>

Kester, James. "A Naïve Bayesian Spam Filter for C#." *The Code Project*. 6 Feb. 2008. Web. 20 Jan. 2011.

Alexander Pak, Patrick Paroubek. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining."

"Natural Language Toolkit Development." (n.d.) Web. 24 Jan. 2011. <http://code.google.com/p/nltk/>

"Sentiment Analysis." *Wikipedia, the Free Encyclopedia.* Wikimedia Foundation, Inc. 15 Apr. 2011. Web. 19 Apr. 2011.