Making Social Media Analysis More Efficient Through Taxonomy Supported Concept Suggestion

Fabio Cardoso Coutinho, Alexander Lang, Bernhard Mitschang

IBM Deutschland Research & Development
Schoenaicher Str. 220
71032 Boeblingen
fabio.coutinho@gmail.com
alexlang@de.ibm.com
bernhard.mitschang@ipvs.uni-stuttgart.de

Abstract: Social Media sites provide consumers the ability to publicly create and shape the opinion about products, services and brands. Hence, timely understanding of content created in social media has become a priority for marketing departments, leading to the appearance of social media analysis applications. This article describes an approach to help users of IBM Cognos Consumer Insight, IBM's social media analysis offering, define and refine the analysis space more efficiently. It defines a *Concept Suggestion Component (CSC)* that suggests relevant as well as off-topic concepts within social media, and tying these concepts to taxonomies typically found in marketing around brands, products and campaigns. The CSC employs data mining techniques such as term extraction and clustering, and combines them with a sampling approach to ensure rapid and high-quality feedback. Initial evaluations presented in this article show that these goals can be accomplished for real-life data sets, simplifying the definition of the analysis space for a more comprehensive and focused analysis.

1 Introduction

According to the study in [Cor09], 500 billion impressions about products and services are annually shared among clients in social networks, while 78% of consumers trust peer recommendations. These statistics show that news of great products and services, along with their experiences, have the potential to rapidly define a product's success or failure. Therefore, the fast understanding of user experience via users' direct feedback or customer conversations about their products and services on social network web sites has become crucial to effective marketing.

Aiming to aid marketing teams in exploring social networks as marketing channels, *social media analysis* has drawn a great deal of attention recently. It can be defined as the process of measuring, analyzing, and interpreting the results of interactions and associations among people, topics and ideas through data mining techniques. IBM Cognos Consumer Insight (CCI) [IBM11] is a social media analysis application that helps companies to gain insight into consumer opinions and spot trends related to products and brands. The chal-

lenge in setting up a relevant social media *analysis space* is two-fold:

1. Homonyms, in which a monitored term has multiple meanings, causing the retrieval of unwanted posts for the analysis.
2. Missing terms, in which important terms that could be used to further refine the analysis are not present in the analysis space.

This article describes an approach to tackle above challenges through the suggestion of relevant as well as off-topic concepts to the user setting up the analysis space. This suggestion is based on term extraction from a subset of social media documents, combined with a relevancy classification of these terms according to a user-provided taxonomy.

The “on-topic" vs. “off-topic" classification of terms helps social media analysts to rapidly uncover homonyms, spot missing concepts, and rapidly refine the analysis space. The result is a significantly improved user experience and a decrease in “time to value", i.e., the time needed until end users can uncover results from social media.

To ensure both rapid response times as well as high quality of the concept suggestions, the *Concept Suggestion Component* described in this article combines several techniques such as term ranking, sampling and adaptions to well-known k-means clustering in a new way. The initial results shown in this paper seem to validate the effectiveness of our approach.

2 The Social Media Analysis Process in IBM Cognos Consumer Insight

IBM Cognos Consumer Insight (CCI) is a social media analytics application that enables marketing professionals to

- Measure the effectiveness of marketing campaigns by analyzing the volume, influence and sentiment of consumer responses across multiple social media channels.
- Anticipate consumer reaction to new products and services by examining affinity relationships and evolving topics.
- Pinpoint key influencers and social channels to engage consumer communities through both traditional and digital marketing activities.

CCI accesses a wide range of social media content, including twitter, facebook, blogs, message boards, video comments and product reviews, and provides analysis capabilities over a wide range of languages, including (but not limited to) English, German, French, Spanish and Chinese.

Figure 1 portrays one of the many available analysis in CCI based on the monitored social media content - in this case, allowing the differentiation between the content of posts from authors who own a certain product vs. those from authors who just speculate about

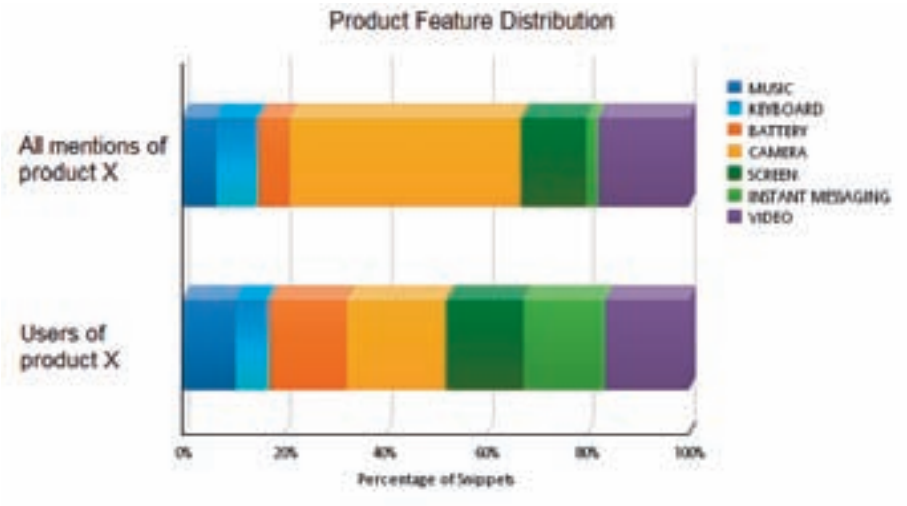

Figure 1: Using CCI to understand the importance of certain features for a particular product

it. These sort of analysis found on IBM Cognos Consumer Insight are typically used by employees in different roles within an organization:

- *Social Media Analysts* define the *analysis space* provided by CCI. They develop strategies for engagement in social media, and regularly feed back insights gained from social media monitoring into the marketing organization, to help them evolve their strategies in a timely fashion.
- *Social Media Consumers*, such as brand managers, product managers or call center staff, explore the analysis space through pre-configured dashboards in their daily business as one information channel. Another important recipient of this information are Chief Marketing Officers or the Heads of Digital Marketing.

The Social Media Analyst uses the *CCI Administration Portal* to define the analysis space in two steps:

1. A set of *content queries* that define the "raw" social media content to be pulled from several social media sources. Content queries are defined like "typical" web queries, using a keyword search syntax with operators like *and, or, not* or *proximity*.
2. A *taxonomy* of the relevant concepts that should be monitored. Top-level entries of this taxonomy are typically brands, companies grouped by their market, product lines and campaigns. Within each top-level entry, the analyst defines *concepts* such as brand and company names, products, product features, spokespeople for a certain brand (often celebrities such as sports stars), as well as marketing campaign messages. The analyst provides one or more terms or regular expressions that define

each concept, which are used by CCI's analysis platform to extract the information from social media content.

The definition is typically done iteratively:

1. The Social Media Analyst starts with a small taxonomy and initial content queries
2. The analyst runs the extraction process
3. The analyst reviews the results, refines the definitions further or adds new definitions
4. Unless the configuration is sufficient for the use case, the next iteraton starts at Step 2.

Once the definitions are both stable and comprehensive enough, the Social Media Consumers navigate within this taxonomy in the *CCI Dashboards* to answer their business questions. They only see the concepts defined in the taxonomy, and don't need to know the content queries or the taxonomy definitions being used. CCI ensures that the content queries are regularly re-executed and the taxonomy rules re-applied to make sure new information is available to the Social Media Consumers.

This approach is depicted in Figure 2, which shows the two steps described previously and the underlying components used in producing the information to be consumed by CCI users.

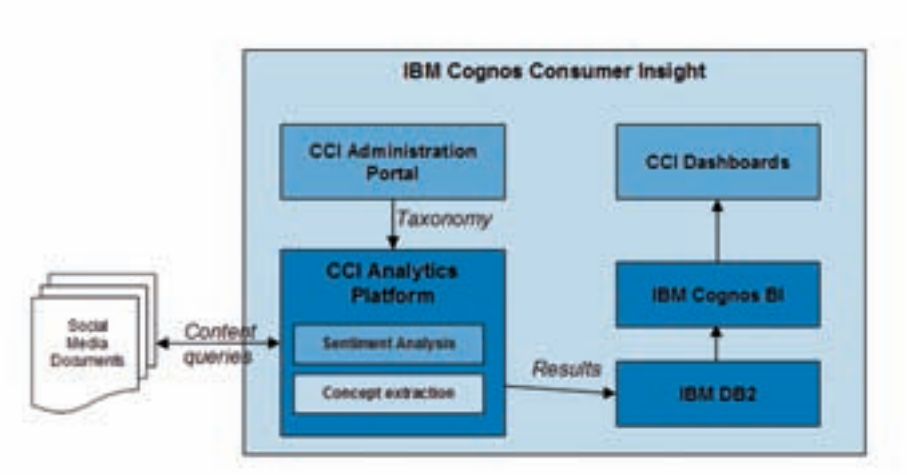

Figure 2: IBM Cognos Consumer Insight Architecture Overview

The two-step approach in defining the analysis space allows analysts to be very broad in their set of content queries (e.g., *"everything that mentions company X"*), and use the power of regular expressions within the taxonomy for very fine-grained analysis (e.g.,

identifying and extracting authors who own a certain product vs. authors who just speculate about it, or have an intention to buy, as seen in Figure 1).

Still, the configuration of social media analysis presents challenges on two fronts:

- **Ambiguous brand, product or feature names** In many cases, product names also have a meaning outside their product domain. This is typically caused by product names that are used by different companies (for example, both adidas as well as Ferrari have a product called *F50*), or product names that are a "real world" concept, for example, HTC's smartphone called *Desire*. One way to remove the ambiguity is to use content queries that contain the brand name, e.g., *adidas F50*. However, this approach can miss a lot of social media content, where users may use other terms from the domain to "implicitly" disambiguate the term - for example, a tweet like *Just played soccer with my new F50 - great!*. Hence, it's typically more accurate to disambiguate content queries through negation, such as *F50 -Ferrari*. In both cases, analysts are required to manually go through the results of the content queries to pick the right approach, as well as the right terms to further improve the content queries - a potentially time-intensive process.
- **Missing concepts in the analysis space** Not all concepts can be anticipated in advance - especially when monitoring campaigns or new developments in social media. Hence, Cognos Consumer Insight provides feedback on dynamically evolving topics through clustering of social media content. Still, enriching the taxonomy with relevant concepts as part of the definition and refinement of the analysis space makes it easier to integrate the analysis results in business processes or alerts. The challenge for the analyst is to identify the concepts that are present in the "raw" documents, but are missing from the taxonomy and its definitions. A system that automatically suggests relevant concepts can greatly cut down the time required to arrive at a comprehensive taxonomy.

The goal of the Concept Suggestion Component is to support the analyst at step 3 of the analysis space definition process by:

- Suggesting disambiguations of ambiguous content queries
- Suggesting new concepts, which will be reflected as additional content queries as well as enhancements to the taxonomy

3 The Concept Suggestion Component

Existing approaches for automatic term extraction, an obvious solution for the problem at hand, suffer from two main drawbacks. Simple and fast approaches, such as word counts shown within a word cloud, often provide too much "raw" information to be useful. Moreover, they do not provide a clear way to classify the terms according to their relevance to the user query. On the other hand, more sophisticated approaches, such as document

clustering, provide more concise topic keywords, but solve the classification problem only partially: these approaches do not allow the relation to a user-defined taxonomy, which leaves the distinction between relevant and off-topic concepts to the user. On top of that, the “time-to-feedback” for these latter approaches is very high. Considering that the configuration process typically takes several iterations, and that the alternative for automated feedback is manually sifting through results, users do accept feedback that takes longer to create than the typical time span acceptable for “interactive" user interfaces (8 seconds), but shouldn’t “block” the user for several minutes in each iteration either.

This work defines a middle-out approach to this problem that is fast enough to be acceptable, but uses techniques beyond simple counting to improve its usefulness and allow term classification. The resulting *Concept Suggestion Component (CSC)* combines good time-to-feedback with result quality, based on the following steps:

1. Downsampling of the document corpus to reduce the execution time of the analysis.
2. Extraction of the most important terms from the downsampled documents.
3. Clustering of the downsampled documents based solely on the extracted terms.
4. Suggest the Relevant and Off-Topic Concepts based on the classification of clusters into Relevant or Off-Topic according to user taxonomy. This classification is based on the distance of the clusters to a control cluster formed by the terms in the taxonomy.

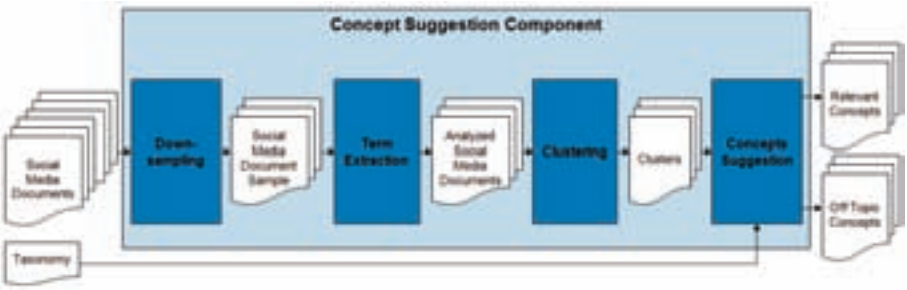

Figure 3: CSC Architecture

As seen in Figure 3, CSC is comprised of four main modules that execute sequentially to provide the desired output. Each of these modules are further detailed in the next sections.

3.1 Downsampling Module

When executed upon a very large set of documents, a clustering algorithm can take a prohibitive amount of time to finish, being then not responsive enough for user interaction.

To mitigate this problem, the first module to execute within CSC is responsible for the reduction of the document corpus via document sampling.

Sampling is a well known statistical method to analyze big populations. As defined in [JKLV99], Simple Random Sampling is commonly used in document analysis and information retrieval. This approach makes use of the fact that, if the whole document corpus is analyzed, its term structure is not embodied by any single document. Thus, if a random sample of a big enough size is taken, the characteristics of the entire corpus can be estimated from this smaller set of documents. Examples of this approach applied to information retrieval can be seen in [CCD99, Bro97, CKPT92, RGA04].

Assuming a normal distribution, the size of a sample can be determined for a given margin of error and confidence level. Figure 4 shows a statistical analysis of the inherent similarity of a document corpus that will be used for one of the case studies presented further in this article.

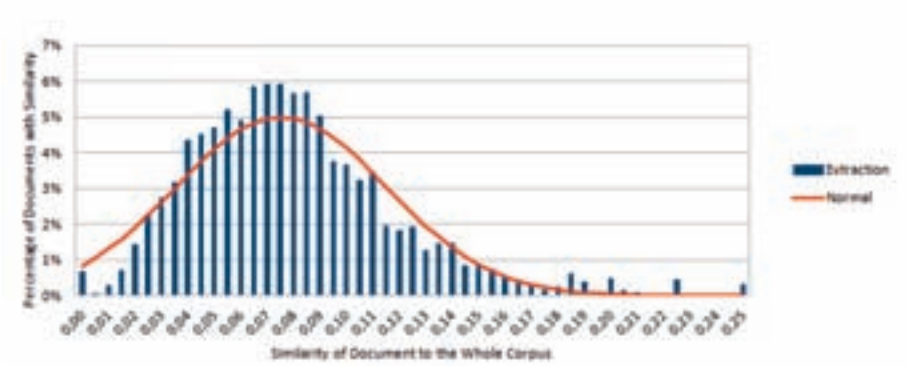

Figure 4: Document Corpus Statistical Analysis

The image shows a histogram of the similarities between the most important terms of each document when compared to the most important terms extracted from the whole of the document corpus. As it is made clear by the picture, the distribution of the similarities of the document corpus fairly resembles a normal curve. Thus, we can use the following sampling method:

For a margin of error *ME* and confidence level α, the number of samples *n* needed to estimate its mean value is determined by:

$$n = \frac{N * 0,25z^2}{(N-1) * ME^2 + (0,25z^2)} \tag{1}$$

where z is the critical standard score for $1 - \frac{\alpha}{2}$, N is the total number of documents and the margin of error is the percentual error allowed for the mean of the similarities of the document corpus when analyzing the sample instead of the whole amount of documents.

Figure 5 depicts the behavior of the chosen downsampling function with a 95% of confidence level for a 1% margin of error .

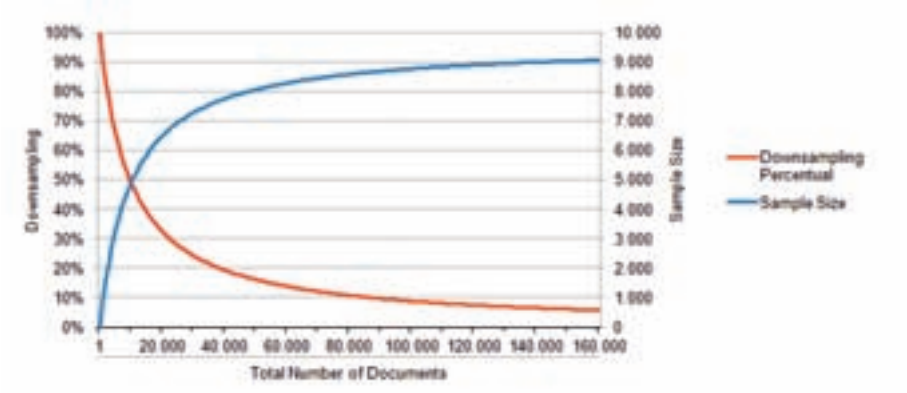

Figure 5: Downsampling Function Behavior

It is clear from the picture that the chosen sampling technique assures that only a reasonable number of documents are processed, even when a very large document corpus is analyzed - the maximum number of processed documents is smaller than 10.000 documents for an analysis with 95% confidence level with a 1% margin of error. Therefore, the use of the proposed downsampling technique allows a substantial improvement on the overall performance of the algorithm.

3.2 Term Extraction Module

In the Term Extraction Module, each document is separately analyzed and has a group of terms extracted from its contents. These extracted terms aim to provide a complete enough description of each document, so that the clustering can be performed against these words instead of the whole text.

The term extraction algorithm implemented in this work is the TextRank algorithm presented in [MT04] - an unsupervised extraction technique that is totally context independent and, therefore, each document can be analyzed independently without needing an analysis of the entire document corpus to extract its terms. The chosen algorithm has also a better performance against a big set of documents when compared to other state-of-the-art techniques.

As presented in [MT04], the generic graph-based ranking algorithm for texts consists of the following steps:

1. Identify text units that best define the task at hand, and add them as vertices in the graph.
2. Identify relations that connect such text units, and use these relations to draw edges between vertices in the graph.

3. Iterate the graph-based ranking algorithm until convergence.

4. Sort vertices based on their final score. Use the values attached to each vertex for ranking/selection decisions.

By default, the algorithm uses all the nouns found in the document analyzed as vertices of the graph. The selection of nouns provides a simple filter that typically captures brands, products and features. However, users can also select adjectives or simple noun phrases as vertices. The latter are sequences of nouns, excluding determiners or modifiers. In this case, compound terms are also treated as noun sequences. For example, *Akkulaufzeit* (German for *battery life*) is split into the two nouns *Akku* and *Laufzeit*. This split removes phenomena such as so-called *Fugenmorpheme* in German to ensure that the split yields valid nouns. For example, *Dämpfungssohle* (cushion sole) yields *Dämpfung* + *Sohle*.

As stated in the algorithm, the next step is identifying the relations that connect each of the vertices, and use them to draw the edges of the graph. TextRank uses the co-occurrence of terms to define the vertices connections - the distance between the occurrence of two terms determines if these vertices are linked or not. Following the configuration that has shown the best results in [MT04], in this implementation, the window of co-occurrence that determines a link between two vertices is of two words and the links are weighted by the amount of times that the vertices co-occur. Therefore, if a pair of words co-occur more often, the importance of this recommendation is taken into consideration in the algorithm.

Having the graph constructed, the value of each vertex can be calculated. As proposed by [MT04], the equation that determines the score of a vertex can be determined by:

$$Rank(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} Rank(V_j) \quad (2)$$

where $In(V_i)$ are the vertices that point to the vertex V_i and $Out(V_j)$ the vertices that V_j points out to. The w_{ji} is the weight of the of the link from vertex j to vertex i and the factor d is a constant set to 0,85 as suggested by [MT04].

Using this recursive formula, the score of each vertex in the graph is recalculated until convergence, and then the vertices are ranked respectively in terms of their importance to the document. Since the social media documents analyzed by CCI typically show a fairly concise nature, the use of a number greater than 10 terms to represent them is found to impact the rest of the component's performance as well as leaving their summarization excessively verbose. Therefore, in this solution, a maximum of 10 terms is used to represent each of the documents being analyzed.

Having finished the selection of the top ranked terms, the document can then be viewed as a vector represented in a maximum of 10 dimensions space as proposed by [SWY75]. The author proposes the Vector Space Model, in which a document is represented by a vector $[w_{t0}, w_{t1}, ...w_{tn}]$, where $[t_0, t_1, ...t_n]$ is a set of words and w_{ti} expresses the importance of t_i to the document. Therefore, from here on, whenever a document *D* is mentioned, it should not be interpreted as a text, but as a term vector as defined below:

$$D = [w_{t0}, w_{t1}, ...w_{t9}] \tag{3}$$

3.3 Clustering Module

Having each document mapped in a group of terms as an input from Term Extraction Module, these descriptive terms are used to separate the document corpus into a group of clusters in the Clustering Module. The clustering algorithm performed in the proposed component is a K-Means algorithm with some improvements based on the algorithm presented in [CKPT92].

As defined in [MRS08], a known issue in K-Means clustering is determining the number of clusters. However, as is the case in the study presented in [Wei06], the real output of the component is not influenced by the decision of cluster cardinality. The clusters in this solution are intermediate structures that allow the comparison of documents to the taxonomy, and therefore, the number of clusters does not influence critically on the final output of the algorithm.

Having the greatest setback of K-Means mitigated, this algorithm can be adapted to the reality of social media document clustering using the scatter gather algorithm proposed by [CKPT92] as a guide for the design of the solution to the problem at hand.

As defined in [JMF99], the typical K-Means clustering consists of the following steps:

1. Choose *k* cluster centers to coincide with *k* randomly-chosen patterns or *k* randomly defined points inside the hypervolume containing the pattern set.
2. Assign each pattern to the closest cluster center.
3. Recompute the cluster centers using the current cluster memberships.
4. If a convergence criterion is not met, go to step 2.

This section has been organized according to the previous mentioned steps. First, the selection of initial cluster centers is discussed, followed by the description of how each document is assigned to each cluster. Then methods to refine the clusters are given and, finally, the convergence of the algorithm is measured.

3.3.1 Selection of Initial Cluster Centers

Several examples on how to enhance initial cluster selection are found in document clustering literature, such as in [CKPT92, Wei06]. The main motivation behind it is due to the fact that, if one outlier document is chosen as an initial seed, no other vector is assigned to it during subsequent iterations, leading to an isolated cluster.

Seeing that, during the cluster refinement step, a noise reduction optimization is performed that avoids the clustering of outliers, these implementations are not adopted as part of the algorithm.

Therefore, the selection of initial cluster centers is made by randomly choosing *k* documents from the entire corpus to be the *k* initial centers. Hence the cluster is also represented by a vector with a maximum of 10 dimensions. This representation is used to determine the similarity between the documents and the clusters, as defined next.

3.3.2 Assigning Documents to Clusters

With the initial centers chosen, the algorithm proceeds then to assign the documents to each cluster. As determined by the typical K-Means algorithm, each document should be associated to the cluster with the closest center. However, to find the closest cluster, a distance function must be determined first.

According to [TC92], the most common distance metric used in the vector space model is the cosine similarity between the document vectors, as seen in [CKPT92, Wei06]. The cosine measure is a robust technique that measures whether two vectors are pointing towards the same direction in the space determined by the term-document matrix formed by both document's vectors. The cosine is used as an approximation of the angle between both vectors, and is used in place of the angle, since it is easier to compute. It is then defined that the similarity *Sim(x,y)* is determined by the cosine of the angle between the vectors *x* and *y*.

With the similarity formula defined, it is easy to determine the distance function between documents. [Bro97] determines that the similarity *Sim(x, y)* of two documents, can be measured by $Dist(x, y) = 1 - Sim(x, y)$. Hence, the proposed clustering algorithm uses, as distance function, the following formula:

$$Dist(x, y) = 1 - Sim(x, y) = 1 - \cos(x, y) = 1 - \frac{x \cdot y}{\parallel x \parallel \parallel y \parallel} \tag{4}$$

where $x \cdot y$ is the dot product between vectors *x* and *y*, $\parallel x \parallel$ is the norm of vector *x* and $\parallel y \parallel$ is the norm of vector *y*.

Having defined the distance property, the definition of the cluster assignment algorithm is very straightforward. Each document in the corpus has its distance to every one of the *k* cluster centers computed. The document is then assigned to the nearest one.

Since both the clusters and the documents are represented by a maximum of 10 dimensions, there is a chance that a given document is orthogonal to every cluster. Such documents are then entirely dissimilar to the entire cluster group. When this situation occurs, these documents are assigned to a special cluster - the Noise Cluster - which groups all of these dissimilar documents.

3.3.3 Cluster Refinement

With all the documents assigned to their respective clusters, the algorithm can then use the clusters' own information to refine the quality of the result. Some refinement algorithms proposed by [CKPT92] are adapted to the problem here at hand.

Cluster Center Updating [CKPT92] proposes that the most important words from the cluster's documents are a short description of the contents of the cluster, serving thus the purpose of its center. The formula that determines the term importance in a given cluster is defined as:

$$Importance(t, D) = \sum_{d_i \in D} \begin{cases} Rank(t) & \text{if } t \in d_i \\ 0 & \text{if } t \notin d_i \end{cases} \quad (5)$$

where *D* is the set of documents in the cluster and *Rank(t)* is the ranking of the term *t* in the document *d*.

Having calculated the importance of all cluster's terms, the proposed algorithm substitutes the old center for the new 10 word vector comprised of the most important words. The number of terms to describe the cluster's new center is kept to 10, so that the the cluster's description remains fairly concise and that the algorithm maintains its performance in the new iterations.

Cluster Joining In [CKPT92], the authors propose merging document clusters that are not usefully distinguished by their cluster centers. In this implementation, the clusters are compared against one another, and, if there is an overlapping of more than 50% of the cluster centers' terms, they are merged. The newly created cluster inherits all documents from both clusters and has its center updated by the same algorithm presented before.

Even though this optimization reduces the number of initial clusters, the total number of clusters is kept due to additions taken place in the Noise Reduction section of the algorithm.

Noise Reduction As explained before, the last refinement in the clustering algorithm aims to reduce the amount of documents not assigned to any cluster. In order to assure that a relevant cluster is created from the unassigned documents, some computation is necessary before creating the new clusters. Therefore, a noise reduction algorithm is proposed, as summarized in the following steps:

1. Rank the Noise Cluster's terms according to their importance.
2. Randomly choose from the Noise Cluster one document that contains the most important term and create a new cluster with it as the center.
3. Repeat step 2 until the number of clusters has once again reached *k* clusters.

4. Compare the number of documents of the smallest cluster with the number of times that the most important term on the Noise Cluster appears on the Noise Cluster's documents.

5. If this term's frequency is greater than the smallest cluster size, randomly choose from the Noise Cluster one document that contains this most important term and create a new cluster with it as the center substituting the smallest cluster.

6. Repeat steps 4 and 5 until the smallest cluster size is bigger than the frequency of the most important term remaining in the Noise Cluster.

This algorithm reduces dramatically the Noise Cluster size and, what is more important, makes sure that the remaining documents have actually no similarity with any other documents. On top of that, the use of documents containing highly important terms, makes sure that any bad initial cluster setup has a great chance of being overcome with the creation of noise-based clusters.

3.3.4 Convergence Check

As stated in [DBE99], the Davies Bouldin index takes into account both cluster dispersion and the distance between cluster means. Therefore, well separated compact clusters are preferred when converging according to this index. According to [DB79], the index "has the significance of being the system-wide average of the similarity measures of each cluster with its most similar cluster. The best choice of clusters, then, is that which minimizes this average similarity."

The proposed clustering algorithm aims then to minimize Davies-Bouldin index while it iterates through the Cluster Assignment and Clusters Refinement steps. Using such metric, the algorithm will converge to a better overall clustering result, assuring that the clusters are compact and far from each other.

3.4 Concepts Suggestion Module

The next step on the CSC information flow is the ranking of each cluster according to its relevance to the user provided taxonomy. As stated before, CSC expects a group of terms that describes the interest of the user when performing a query. After the execution of this analysis, every cluster has a ranking of how relevant it actually is to the user.

The next step in the algorithm is then determining the distance of each cluster to the taxonomy's terms. Therefore, the taxonomy is translated into a Control Cluster defined by these user-defined terms. All terms in the Control Cluster are ranked with a standard ranking of 1. Having defined this cluster as a bias, the algorithm then measures the distance of the k clusters to this control cluster. According to their results, the clusters are then classified in three categories, as presented below:

1. If $Dist(C_i, ControlCluster) = 1$, then the clusters have no terms in common. Therefore, C_i is not relevant.
2. If $\mu + \sigma < Dist(C_i, ControlCluster) < 1$, then the cluster is an outlier and, thus, considered ambiguous.
3. If $Dist(C_i, ControlCluster) \leq \mu + \sigma$, then $Cluster_i$ is considered relevant.

In these inequations, the μ stands for the mean of all non orthogonal clusters' distance to the Control Cluster, whereas σ stands for their standard deviation.

With the classified clusters as an input, the concept suggestion algorithm aims to extract, from each of these cluster groups, words that serve as a good identifier to their content. Loosely based on the *tf*idf* metric for term extraction that favors terms that distinguish certain individual documents from the remainder of the collection, the importance of a term in a cluster group is given by defining *tf* as the term's importance to the group, and *idf* as a term that varies inversely with its importance to the remainder of the groups. Thus, the term-weighting formula for a given group is defined as:

$$w_t(t, d, D) = Importance(t, d) * (\frac{Frequency(t, d)}{Frequency(t, D - d)})^n \quad (6)$$

In the above definition, *d* stands for the documents of a group, *D* the entire document corpus and *n* determines the relevance of the exclusivity factor on the term weighing.

This weighing procedure allows the ranking of the terms and present to the user the concepts that best define each one of the categorized groups. Thus, as an output of this module and of the component itself, two lists of concepts are provided:

- List of Relevant Concepts, extracted from Relevant Clusters.
- List of Off-Topic Concepts, extracted from Off-Topic Clusters.

4 Implementation and Evaluation

4.1 CCI-Integration

As stated before, the main goal of the proposed solution is to enhance CCI's usability, making it an easier task to the user. An overview of the component's integration with CCI is presented in Figure 6.

Figure 6 shows CSC as an add-on to CCI's architecture. This component receives two inputs - the Taxonomy and the Social Media Documents - and, as output, provides a list of concepts to the user via the Administration Portal. These concepts are already classified as relevant or off-topic, so that the user can then, in a timely manner, refine his set-up. The refinement can be either made by introducing into the set-up relevant concepts previously left out, or by excluding off-topic terms not relevant to his interests.

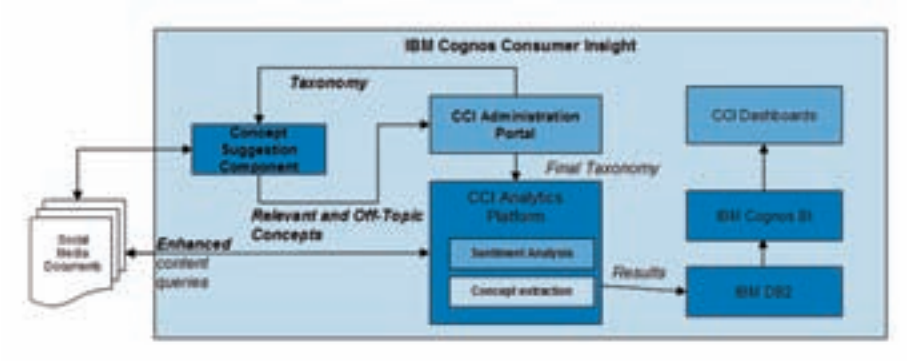

Figure 6: Solution Architecture

As seen in Figure 6, the software receives the Taxonomy directly from CCI. These are the terms that are used to determine whether the suggested concepts are relevant or not to the user query. The second input the component receives are the Social Media Documents. These are the results from the user's query in CCI's Administration Portal. These documents are analyzed by the algorithm proposed in the previous section and the output is extracted.

This output is then, as mentioned beforehand, provided to CCI's Administration Portal. If CSC manages to suggest these concepts rapidly enough, the end user is able to refine the set-up parameters in a much faster manner, thus improving the overall usability of CCI's analysis set-up.

This service provided by the component is made available to CCI as a REST web-service with JSON message format. The chosen architecture provides a non intrusive development, that keeps the current flow of user operations.

4.2 Evaluation

To perform the component's evaluation two case studies are proposed in which the ambiguity of monitored concepts are known to be a good example for the use of CSC. After analyzing the results of the case studies, CSC's performance is briefly analyzed.

4.2.1 Case Study - Adidas F50 Football Boots

The first case study is the monitoring of the *f50* concept, which is the name of a known brand of Adidas' football boots, as well as a known sports car brand produced by Ferrari. The social media content used in this case study is drawn from blogs, discussion forums, microblogs (e.g., Twitter), video comments and product reviews. It focuses on English content posted between January 1st and March 1st, 2012. The taxonomy used by the

Concept Suggestion Component contains two top-level entries:

1. *Shoe Brands*, which includes a single concept, *adidas*
2. *Football Shoes*, which includes a single concept, *F50*, as a product name.

Figure 7 presents the Relevant Concepts for the performed query. As expected, all terms suggested are somehow football related and can be used to further refine the set-up of CCI.

Figure 7: Relevant Concepts for *f50*

Figure 8 presents the Off-Topic Concepts for the performed query. The most important off-topic concept is *ferrari*. Other important terms retrieved by the algorithm - such as *currency* and *vs* - appear due to posts related to Fujitsu's F50 currency dispensers.

As the taxonomy is very simple, some terms that were identified as off-topic could be relevant to the user - such as *cristiano* and *ronaldo*, the name of a football player sponsored by Nike. In this case, the user is free to enhance the taxonomy with an additional top-level entry *Brand Spokespeople*, and add *Christiano Ronaldo*.

Figure 8: Off-Topic Concepts for *f50*

In order to improve the initial definitions, the user can select *ferrari* and *camcorder* from the off-topic terms, and add them as so-called *exclude terms* to the *F50* concept. The CSC will automatically update the CCI content query from *F50* to *f50 -ferrari -fujitsu*. The user can also create new concepts from on-topic terms such as *micoach*, or enhance the definition of *F50* with the product name variant *adizero*.

4.3 Case Study - Nike and Adidas Football Boots

In order to illustrate the behavior of CSC when analyzing more than one concept in a query, the previous example is extended to monitor not only the *f50* boot, but also Nike's *Mercurial* football shoe. The query is changed to monitor both terms. The taxonomy is enhanced with the following:

1. The concept *Nike* is added to *Shoe Brands*
2. The concept *Mercurial* is aded to *Football Shoes*

and, in the taxonomy, the term *nike* is added as a brand. This query's execution provides the output presented in figures 9 and 10.

Figure 9: Relevant Concepts for *f50* and *mercurial*

Figure 10: Off-Topic Concepts for *f50* and *mercurial*

As expected, the Relevant Concepts are still all football-related terms like *vapor* and *superfly* which refer to brands of nike football boots. However, some new terms appear when compared to the previous analysis. The same behavior is found on the Off-Topic Concepts. Some of the most important terms from the previous query remain in the extraction, while new terms are added that are related to the term *mercurial*.

In this example, it is clear that the addition of the term *mercurial* to the query introduced some documents that are not relevant to the user. This is due to the fact that *mercurial* is thorougly used as an adjective to define personality of people, which brings various groups of documents relating to various subjects such as football players behavior and political and cultural discussions. The off-topic concept suggestion allows nevertheless the identification of terms such as *control* and *repository*, which relate to a cross-platform, distributed revision control tool for software developers that is also named mercurial.

4.4 Performance Evaluation

In this section, the performance of CSC is evaluated. The key focus here is "time to insight" for end users, as this determines whether the approach can really provide interactive and high-value feedback.

To evaluate the overall performance, the two queries proposed in the previous section are analyzed. For each of these queries, two timespans are proposed - the 2 first months of

2012, and the 12 months of 2011. The performance of the algorithm is presented in Table 1. The results are presented with the entire document corpus and also for a downsampling of the corpus with a 95% of confidence level for a 1% margin of error.

	CCI-Concepts	
	f50	*f50* and *mercurial*
2 months, entire corpus	5.083 documents analyzed in 52 seconds	18.962 documents analyzed in 120 seconds
12 months, entire corpus	27.851 documents analyzed in 284 seconds	69.265 documents analyzed in 852 seconds
2 months, downsampling	3.325 documents analyzed in 37 seconds	6.376 documents analyzed in 66 seconds
12 months, downsampling	7.142 documents analyzed in 81 seconds	8.436 documents analyzed in 78 seconds

Table 1: Concept Suggestion Component's Performance Evaluation

As seen on Table 1, when executed against the entire document corpus for a whole year, the algorithm's execution time can reach up to almost 15 minutes. Such a long time is prohibitive for interactive use. However, the use of the downsampling technique enables its usage due to its great enhancement in overall performance.

As seen on the analysis, the downsampling enables CSC to execute up to 10x faster, allowing a fast enough output for the user, even when a very demanding query is performed. Seeing that, for the analyzed set-up, the maximum number of processed documents is smaller than 10.000 documents, the downsampling should allow in every situation a responsive enough execution.

Figures 11 and 12 compare the extracted concepts when the entire document corpus is analyzed and when they are downsampled. The pictures portray the most downsampled scenario - the 12 months extraction of both *f50* and *mercurial* CCI-Concepts.

Figure 11: Comparison of Relevant Concepts Extraction With Downsampling

The figures show that, in fact, the huge gain in performance does not substantially impact the quality of the extraction. As expected, the most important terms for both the Relevant and Off-Topic Concepts are found even when the downsampling is performed. In the

Figure 12: Comparison of Off-Topic Concepts Extraction With Downsampling

presented example, for both the Relevant and Off-Topic Concepts, the 10 most important extracted terms were the same when performing downsampling - 100% precision when comparing both approaches. When all extracted terms are considered, the downsampled execution achieved a precision of 77% and 93% for the Relevant and Off-Topic Concepts respectively.

5 Conclusions and Future Research

This work proposes an approach that provides Social Media Analysts with relevant and off-topic concepts within their analysis domain to effectively support them in configuring Social Media Analysis dashboards for their stakeholders.

The core of this approach is the Concept Suggestion Component (CSC). It is comprised of four main modules. First, the documents are randomly sampled to enable the algorithm's execution in a responsive manner. Then, a term extraction module extracts, from each analyzed document, the terms that best define its content. Next, a clustering module uses these extracted terms to group the documents according to their content, forming well defined clusters. Finally, the concept suggestion module compares each cluster to the user taxonomy and suggests Relevant and Off-Topic concepts. The implementation of CSC has been evaluated in several case studies, which have shown that CSC improves the Social Media Analysis workflow of IBM Cognos Consumer Insight. As shown by these examples, its use leads to a better user experience on the definition and refinement of the analysis space, resulting in an analysis that is both more focused as well as more comprehensive.

The results presented in this article show that, in fact, our approach to term extraction and classification is faster than traditional document clustering. On top of that, the modularization of the solution allows the application of parts of the proposed algorithm to other analysis scenarios that can benefit from term extraction and document clustering, such as comparing the vocabulary and themes used by different social media authors about the same topic or product.

References

[Bro97] A. Z. Broder. On The Resemblance And Containment of Documents. In *Compression and Complexity of Sequences*, pages 21–29, Salerno, Italy, June 1997. IEEE Computer Society Press.

[CCD99] Jamie Callan, Margaret Connell, and Aiqun Du. Automatic Discovery of Language Models for Text Databases. In *In Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 479–490. ACM Press, 1999.

[CKPT92] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J.W. Tukey. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1992.

[Cor09] Sean Corcoran. The Broad Reach Of Social Technologies. Technical report, Forrester Research, 2009.

[DB79] D.L. Davies and D.W. Bouldin. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:224–227, 1979.

[DBE99] A. Demiriz, K. Bennett, and M.J. Embrechts. Semi-Supervised Clustering Using Genetic Algorithms. In *In Artificial Neural Networks in Engineering (ANNIE-99*, pages 809–814, 1999.

[IBM11] IBM. IBM Cognos Consumer Insight. Published online at http://www.ibm.com/software/analytics/cognos/analytic-applications/consumer-insight, 2011.

[JKLV99] Fan Jiang, Ravi Kannan, Michael L. Littman, and Santosh Vempala. Efficient Singular Value Decomposition via Improved Document Sampling. Technical report, DEPT. OF COMPUTER SCIENCE, DUKE UNIVERSITY, 1999.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Comput. Surv.*, 31, 1999.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[MT04] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. *Proceedings of EMNLP*, 2004.

[RGA04] J. Ruoming, A. Goswami, and G. Agrawal. Fast and Exact Out-of-core and Distributed K-Means Clustering. *Knowledge and Information Systems*, 10, 2004.

[SWY75] G. Salton, A. Wong, and C.S. Yang. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 1975.

[TC92] H. R. Turtle and W. B. Croft. A Comparison of Text Retrieval Models. *Comput. J.*, 35(3):279–290, 1992.

[Wei06] D. Weiss. *Descriptive Clustering as a Method for Exploring Text Collections*. PhD thesis, Poznań University of Technology, 2006.