Mobile Context Recommendations from Social Media through Geotopical Clustering

Elizabeth Williams¹, Jeff Gray², Brandon Dixon³

University of Alabama
Box 870290 Tuscaloosa, AL 35487
P: 205-348-6363
F: 205-348-6959
¹eawilliams2@crimson.ua.edu, ²gray@cs.ua.edu, ³dixon@cs.ua.edu

Abstract
Mobile recommender systems can improve the speed at which information is passed to a user by taking advantage of the user’s context to provide relevant data. In addition, social media can often provide relevant information to a user about what is happening in their environment. However, to obtain this knowledge, a user may be required to wade through a large amount of irrelevant data. In this paper, we describe our system, GeoContext, for providing relevant contextual information to a user based on his or her location by mining social media. We implemented GeoContext with a geotopical clustering system, a keyword query system, and a location query system. We evaluate GeoContext against a common topic modeling algorithm often used in geotopical clustering, Latent Dirichlet Allocation.

Keywords
recommender systems, social media, context-aware mobile systems
1. Introduction

With the ever-increasing use of mobile devices, users desire information faster than ever. Recommender systems for mobile devices can provide information quickly, and context-aware recommendations can take advantage of the user’s environment in order to provide even more relevant information. Recommender systems can take into account contextual information of the user and filter out data that is not useful in various scenarios [1].

Although traditional search engines are useful for many queries, they often do not provide relevant information about real-time contextual events. Users often turn to social media for questions about their context. For example, traffic and weather updates are often posted on social media due to the fact that these updates can offer critical information, and social media can generally more rapidly disseminate information to users than other mediums. Because many users utilize social media on a mobile device, posts often reflect geographical information of the user. In order to obtain information from social media, the information relevant to a user in a certain location needs to be filtered out. In our implementation of GeoContext, we used a three-faceted approach. First, we implemented our geotopical clustering system, which discovers new topics in the stream, such as high profile news stories or popular events. To implement the geotopical clustering system, we investigated two different approaches. For the first approach, we clustered the social media posts into topics, such that clearly defined subjects appeared about which many people were tweeting. We also filtered out posts that contained topics few people were discussing. Next, we determined which topics were important to different geographical regions. If a topic is determined to be especially important in a certain area, that topic is recommended for users in that area. For our second approach, we used DBSCAN, a density-based clustering algorithm, to cluster tweets by geographical location. Then, GeoContext clusters tweets within each geographical cluster by topic. Second, we also implemented our keyword query system in which the social media stream is mined based on keyword filters. The GeoContext keyword query system allows us to
discover information about topics based around a set of keywords rather than location. Third, we implemented our location query system in which the social media stream is mined based on location filters. This allows us to discover information about topics around the user’s current location, such as weather updates, air traffic delays, and traffic accidents.

We chose to utilize Twitter\(^1\) to mine environmental information for several reasons. Twitter is often utilized to quickly post news-type updates [2], which would be relevant to a user. Also, Twitter allows users to attach geographical coordinates to tweets, as well as provide location information associated with each user account. However, as we simply process information represented as JSON objects in a stream, GeoContext could easily be adapted to any social media or information provider that attaches geographical coordinates to shared information.

1.2 Overview of GeoContext

As shown in Figure 1, GeoContext receives a stream of tweets and pushes them through a pipeline in order to obtain popular topics. To create GeoContext, we used Node.js\(^2\). On the server side, we utilized the Twit framework for Node.js\(^3\) to initiate a stream of tweets. Twit allows Node.js integration for the Twitter Streaming API\(^4\), which provides a sample of tweets through an open HTTP connection. GeoContext furnishes the ability to input location (given in GPS coordinates) or keywords, in order to provide relevant information for specific scenarios. If no keywords or coordinates are given, we initiate a stream of tweets that returns a sample of all public tweets with no filters or parameters, and the stream is passed through the geotopical clustering pipeline. If keywords are given, we initiate a stream of tweets that filters the tweets by the given

\(^1\) dev.twitter.com
\(^2\) https://nodejs.org/
\(^3\) https://www.npmjs.com/package/twit
\(^4\) https://dev.twitter.com/streaming/overview
keywords. After the tweets in the keywords query system are filtered, they are passed through the keyword query pipeline. If coordinates are given, we initiate a stream of tweets that filters the tweets by the coordinates. Once the tweets are filtered, they are passed through the location query pipeline. For all three pipelines (geotopical clustering, keyword query, and location query), we also chose to filter the stream by English-language tweets only, because some of the analyses we performed on the tweets in order to achieve clustering are currently available only for English text.

![Diagram of pipelines]

**Geotopical clustering (topical clustering first) pipeline**

**Geotopical clustering (geographical clustering first) pipeline**

**Keyword query pipeline**

**Location query pipeline**

Figure 1
1.2 Summary of Contribution and Paper Overview

GeoContext provides several advantages over existing clustering approaches. First, GeoContext can process tweets immediately as they are streamed without removing stop words, (words such as “the” or “a” that are often removed before natural language processing) or any terms needing to be stemmed (returning terms to their root form). Also, because of GeoContext’s method of extracting concepts from tweets, there is no need for an initial training set.

In this paper, we present three unique contributions.

• GeoContext allows users to discover topics that people around them are tweeting about. We present a novel approach for discovering topics that are more relevant to different geographical areas.

• We examine two different approaches for geotopical clustering using DBSCAN and an adapted version of TF-IDF.

• We present our approach for utilizing cognitive computing techniques to extract topics and keywords from tweets and implement a more expansive keyword query system.

The structure of this paper is as follows: first, we provide an overview of GeoContext for recommending information via social media posts. In the following sections, we describe our implementation for the geotopical clustering system, the keyword query system, and the location query system. We evaluate the results and conclude with a discussion of existing work related to GeoContext and future directions.
2. Geotopical Clustering Implementation

GeoContext provides a new technique for geotopical clustering, which involves clustering tweets into topics for a user’s specific area and performing geographical analysis on worldwide tweets in order to find relevant information. If location coordinates and keywords are not provided, GeoContext will begin the geotopical clustering pipeline and recommend several topics that users are tweeting about in various areas.

2.1 Geolocation

In order to perform geographical analysis on the tweet stream, GeoContext needs the locations of the tweets. After starting the stream of tweets, all tweets are passed through geolocation, a step in which we extract GPS coordinates from tweets. It has been observed that only 0.87% of tweets are geotagged [3], meaning that the tweet author’s GPS coordinates are included with the tweet. More recently, Twitter has introduced Places, which allows a user to geotag posts within a geographical bounding box. In our evaluation, we discovered that 0.47% of tweets were geotagged with coordinates, while 2.16% of the remaining tweets were tagged with Places\(^5\), which gives us only a slightly rougher estimate of a user’s location. These low statistics indicate that if we only used tweets that are geotagged or tagged with Places, GeoContext would not have enough tweets to glean intelligent topics from the stream. Therefore, we need to mine locations from additional tweets that were not geotagged.

To perform the geolocation step, we first consider tweets that were geotagged. In this case, the GPS coordinates can be directly extracted from the tweets. We then consider tweets that are not geotagged directly but are tagged with Places. Again, we extract the GPS coordinates from the tweet. In this case, the GPS coordinates are represented as a bounding box, so we consider the

\(^5\) https://dev.twitter.com/overview/api/places
user’s location as the exact center of the bounding box. Lastly, we consider tweets that are not
gotagged or tagged with Places, but that have location information in the author’s description box.
Twitter allows a user to specify a location that is associated with the user’s account rather than
specific tweets. However, Twitter allows any text in the location box and does not restrict the text
to actual locations or geographical coordinates. Accordingly, many users do not include true
locations in the description box, such as “not anywhere near you,” or include locations that can be
difficult to parse, such as “California.”

To determine whether a location is valid, GeoContext performs preliminary parsing of the
text, including removing spaces, and passes the text in a query to Dbpedia\(^6\), which is a database
containing data from Wikipedia information boxes. GeoContext receives a result indicating either
that Dbpedia does not contain a page for the text or a result containing the type of the entity. We
consider the first type of result to indicate that the user’s location is either not a true location, or it is
in a format that is difficult to parse. If the result indicates that the type of the entity is anything
other than a location, we determine that the user included something that was not a true location. If
the result indicates that the entity’s type is a location, the result from the Dbpedia query also
includes GPS coordinates for that location. We consider the coordinates from the result to be the
user’s location. If GeoContext is currently running the geotopical clustering pipeline or keyword
query pipeline, the GPS coordinates are extracted from these tweets, and the tweets are
automatically placed into clusters in the geotopical clustering step.

Although this method still does not allow GeoContext to analyze all tweets received
through the stream, by including tweets whose authors have valid account locations along with the
tweets that were geotagged, we are able to increase the number of tweets that are able to be passed
to the geotopical clustering step significantly. With all three methods of geotagging tweets

(coordinates, Places, and user locations), we are able to extract locations from 11.15% of all tweets, an improvement of over 400% from utilizing only geotagged and Places-tagged tweets.

After performing the geolocation step, in order to provide a recommendation of relevant information to users in different geographic regions, we need to determine which topics mined from the Twitter stream appear in various locations. GeoContext performs this geotopical clustering in two steps: topical clustering and geographical clustering. We analyze whether GeoContext produces more accurate clusters if we perform geographical clustering first or topical clustering first.

2.2 Topical Clustering First, Geographical Clustering Second

In our first experiment, we cluster the stream of tweets by topics first, then calculate whether any topics are associated with any specific geographical locations. We refer to this implementation as TCGC.

2.2.1 Topical Clustering

After passing the tweet through the geolocation step, we begin the topical clustering step. A common method for clustering text into topics is to use Latent Dirichlet Allocation (LDA) [4], which is a topic model that takes in a corpus, or body, of text separated into documents. Each document contains words in no sorted order. The model produces a selection of topics, which are collections of words found in the documents. The topics are based on which words appear together most often. LDA can also determine what percentage of each document is composed of each topic, as well as a percentage of how each word influences each topic. However, LDA requires the number of topics to be determined beforehand, which is impractical to calculate for a real-time, worldwide system. Also, LDA only considers words that appear directly in the text, which limits the algorithms ability to detect any underlying meaning of the text. To address this problem, we
implemented GeoContext to add new topics dynamically as they appear in the Twitter stream, and prune topics if they are not tweeted about enough.

We utilize AlchemyAPI’s concept tagging and keyword extraction APIs\(^7\) to extract topics from each tweet. The topics returned from the concept tagging API are not simply terms extracted directly from the tweet, but are concepts of the tweet. For example, a tweet that contains song lyrics could result in concepts of the recording artist or the year the song was published. Keywords returned from the keyword extraction API mine important words directly from the tweet. We elect to use both APIs because, although there is some overlap between concepts and keywords extracted from a tweet, both provide useful information about the content of the tweet.

After a tweet’s topics and keywords are determined, we begin the clustering process. If there are no existing clusters of topics, we create a new topic cluster containing the topics extracted from the tweet. If there are existing topics (any case except for the first tweet), we calculate how similar the tweet’s topics are to the topics contained in existing clusters. As shown in Figure 2, we calculate a similarity score between each concept extracted from the tweet and each existing topic cluster. The highest score between the tweet’s concepts and an existing topic cluster determines in which cluster the tweet is placed.

We calculate the similarity score by first checking whether the current tweet and the tweets in existing concepts have any hashtags in common. Hashtags are tags that can indicate a user is posting about a specific topic. Hashtags can provide important metadata about the topic of the tweet. We choose to compare hashtags because they can often express a popular topic [5]. If the current tweet contains hashtags that match hashtags present in another topic, the tweet is added to that topic. Hashtags that are not exactly the same but that refer to the same event often end up in the same topic due to appearing together in tweets. Users can include multiple hashtags in each

\(^7\) alchemvapi.com
If no hashtags are matched, we then compare the concepts and keywords returned from AlchemyAPI of the current tweet against concepts and keywords of tweets in the existing topic cluster. Each concept and keyword is associated with a relevance score from the concept tagging and keyword extraction APIs. The relevance score is a percentage that indicates how much each concept or keyword influences the tweet. We compare each concept and keyword in the current topic to each concept and keyword of each tweet in existing topics. We begin with a score of 1. If a keyword or concept matches, we multiply the score by the average of the corresponding relevance score of the two tweets that are being compared. This way, tweets are matched only with topic clusters that contain only similar topics with high relevancy scores, rather than matching the secondary topics of the tweet with low relevancy scores. For example, in Figure 2 the streamed tweet has a concept “Orlando” with a relevancy score of 0.822. A tweet in an existing topic cluster has the same concept, but with a relevancy score of 0.139. If a simple keyword matching algorithm were used, these two tweets might end up in the same topic cluster even though one is talking about a job in Orlando and the other is talking about police information in Orlando. By taking the relevancy score into account, we avoid matching these two tweets into the same topic cluster.
There are many cases in which a tweet may not belong in any existing cluster. A tweet’s topics may not be related enough to tweets that have already been processed and clustered. We can determine this scenario if the concepts and keywords of the tweet have below average similarity scores with the topics of the existing clusters. Relevancy scores range from 0 to 1. Therefore our calculated similarity score will range from 0 to 1. We chose 0.50 as our threshold similarity score, because this value represents topics that are of average similarity. If a tweet’s topics do not contain similarity scores above 0.50 with any existing topic cluster, a new cluster will be created and the tweet will be placed into the new cluster. If there is at least one existing topic cluster whose similarity score to the current tweet is above 0.5, the tweet will be added to that cluster. This process is shown in Figure 3. Pseudocode for our algorithm for clustering tweets is shown in Listing 1.

![Figure 2](image-url)
2.2.2 Geographical Clustering

After the list of topic clusters reaches a multiple of 1000 topics, we prune the list. We chose to prune every 1000 topics because the list cannot be too small or else many topic clusters will have only one tweet. In the pruning process, we remove any topic clusters from the first half of the list that only contain one tweet. We only prune the first half of the list so as to retain topic clusters that only contain one tweet but are too new to have any other tweets. When we prune tweets, we also calculate topic recommendations for various geographical locations. The goal of this process is to determine, for each topic cluster, whether the tweets are clustered in one or more geographic location or spread out across a larger geographic area, for instance, the entire United States. Using this goal, we can recommend topics to users that are more specific to their location.

![Figure 3](image-url)
To perform geographical clustering, we adapted the TF-IDF algorithm for our process. TF-IDF stands for Term Frequency-Inverse Document Frequency, and is a statistic that determines how important, or meaningful, a word is to a document [6]. The statistic eliminates words that might occur many times in the document but are not meaningful to the document. For example, in an English-language text, the word “the” probably occurs many times. However, it does not add much meaning to a piece of text. TF-IDF calculates the meaningfulness of a word by determining that the word “the” also occurs many times in all pieces of text, indicating that it is common across all text and is not particularly important to one piece of text. In our case, using an adapted version of TF-IDF, we can discover whether a location occurs commonly throughout all topic clusters of tweets, indicating that that location simply has a higher population, or whether it is occurring more within a specific topic cluster, indicating that that topic cluster is important to that location.

GeoContext considers each topic cluster to be a document and each geographic location of each tweet in that topic cluster to be a word. In this way, GeoContext can sense whether a location is clustered more within a certain topic, or whether it occurs commonly throughout all topics. We decided to use this algorithm rather than a more traditional clustering algorithm such as K-means [7] or DBSCAN [8] because tweets follow a population distribution. More tweets are posted in locations where the population is higher, and thus using a traditional clustering algorithm would simply cluster tweets around population centers. We are interested in also discovering topic clusters around geographic locations with varying population densities. Using adapted TF-IDF allows us to discover topics that are influencing a certain geographic area, even if the population density of the area is small.

The TF-IDF statistic is computed by the formula shown in Formula 3. Term frequency is shown in Formula 1 and is calculated as the number of times a word \( t \) appears in a document \( d \). Following our assumptions outlined previously, we calculate term frequency as the number of times a certain geographic location appears in a topic cluster. Inverse document frequency is shown in
Formula 2 and is calculated as the logarithmically scaled fraction of the number of documents in the corpus $N$ divided by the number of documents $d$ in the corpus $D$ that contain the word $t$. Again, we calculate inverse document frequency by dividing the total number of topic clusters by the number of topic clusters that contain the geographic location and taking the logarithm. The TF-IDF statistic is then determined by taking the product of the term frequency and the inverse document frequency, as shown in Formula 3. We calculate the TF-IDF for each geographic location in each topic cluster, and if the result is above a threshold value, we can infer that the geographic location occurs more often in that topic cluster than other topic clusters. We chose 0.2 as our threshold value, because after inspection of the results, this value represents locations that occur several times in one topic cluster but few times in all other topic clusters. Therefore, if a location has a TF-IDF value of higher than 0.2, we recommend that topic cluster for the geographic location. Using our adapted TF-IDF statistic allows us to possibly discover multiple geographic locations that are important for a topic cluster, meaning that people are tweeting about a topic clustered in multiple locations.

$$tf(t, d) = f(t, d)$$  \hfill (Formula 1)

$$idf(t, D) = \log \left( \frac{N}{|\{d \in D : t \in d\}|} \right)$$  \hfill (Formula 2)

$$tfidf(t, d, D) = tf(t, d) \times idf(t, D)$$  \hfill (Formula 3)

### 2.3 Geographical Clustering First, Topical Clustering Second

In our second implementation, we cluster tweets geographically first, then determine the topics of each geographical cluster. We refer to this implementation as GCTC.
2.3.1 Geographical Clustering

As with the implementation described previously where we clustered tweets by topic first, in this new option, we first perform geolocation on the stream of tweets and then begin the clustering process. To perform geographical clustering on the stream of tweets, we utilize DBSCAN, a popular density-based clustering algorithm. As tweets are streamed, they are passed through the geolocation step just as with the TCGC implementation, and then we save their coordinates. When the list of coordinates reaches a multiple of 1000, we cluster the list of coordinates. Our parameters to the DBSCAN algorithm are 0.5 for the cluster radius and 5 for the minimum number of points to form a cluster. We chose 0.5 for the radius because a radius of 0.5° is approximately the size of an average city, so DBSCAN will cluster tweets within cities. We chose 5 for the minimum number of geographical coordinates to place in a cluster because a location with less than 5 tweets does not generally have enough tweets to successfully cluster topically in the next step in the pipeline.

The DBSCAN algorithm clusters points that are densely packed together and consider points in low-density regions to be outliers. This process differs from K-means and other clustering algorithms that cluster points based on closeness to a mean point. We chose DBSCAN because GeoContext requires tweets to be clustered by the density of the tweets in various geographical areas so that it can discover areas where tweets are occurring the most. DBSCAN returns a list of clusters and lists of all points within those clusters. It also analyzes which points do not belong within a cluster (classified as “noise”) and returns a list of those points.

2.3.2 Topical Clustering

After the geographical clusters are formed through DBSCAN, we utilize the same topical clustering system as described previously for the TCGC implementation. For each geographical cluster, we create topic clusters using the same topical clustering implementation described for
TCGC. Each tweet’s concepts and keywords are compared against tweets in existing topic clusters, and if the similarity scores indicate that the tweets have similar topics, they are clustered together. Otherwise, the tweet is placed in a new topic cluster within the geographical cluster.

We present the results of our evaluation of both methods of geotopical clustering in section 5.

3. Keyword Query Implementation and Location Query Implementation

In this section we describe our implementations for the GeoContext keyword query pipeline and location query pipeline. If any parameters are provided to GeoContext, they are sent through one of these two pipelines.

3.1 Keyword Query Implementation

In addition to the overall geotopical clustering system, we implemented a keyword query system. In the keyword query system, GeoContext can accept keywords as input. This system is useful, for example, if a user is interested in getting updated information about a certain topic such as weather. As mentioned in (Sakaki, Okazaki, & Matsuo, 2013), users can sometimes tweet about major weather events before official sources can report the events. The keyword “weather” can be provided and any relevant weather information will be displayed. Any provided keywords are sent through keyword expansion.

In GeoContext, a user can optionally input keywords to allow filtering of tweets. The Twitter Streaming API allows filtering by keyword phrases. Any number of keywords can be used, and the API will return tweets that match any of the keyword phrases present. If a keyword phrase contains more than one term separated by spaces, the Twitter API will match tweets containing all of the terms in the phrase, even if the terms are not in order. Punctuation and case is ignored in the tweet matching a keyword phrase.
We are interested in concept matching for GeoContext, rather than the more traditional keyword matching implemented by the Twitter Streaming API. Concept matching allows a user to input a keyword or keyword phrase, such as “weather,” and GeoContext will track tweets that not only contain the specific word “weather,” but also tweets relevant to the concept “weather.” This might include tweets that contain the terms “rain,” “thunderstorms,” or “hail.”

To implement concept matching, GeoContext expands any keywords the user has provided. We realize this keyword expansion by utilizing cognitive computing techniques. For each keyword in the comma-separated input list, we pass the keyword to the JoBimText distributional semantics framework, described in [9]. The JoBimText framework receives a corpus of text and analyzes the structure of the text through methods such as a dependency parser. JoBimText extracts pairs of terms from the corpus, a word and another term that describes the context of the word. After obtaining a count throughout the corpus of each pair of terms, which are denoted a Jo (the word) and a Bim (the contextual term), a frequency significance measure is calculated for each unique pair of terms. Terms with the highest significance measure are clustered into sense clusters, such that each word has a sense cluster containing terms that are conceptually related to the word. For example, the word “weather” might have a sense cluster containing the terms “rain,” “thunderstorm,” and “wind.” These clusters can be used to find other words that are similar to a term, as well as to disambiguate a term. Disambiguation involves determining the meaning of a term based on a context, because words can have different meanings in different sentences.

For keyword expansion, we used JoBimText’s similarity score feature to calculate terms that are related to each keyword provided to GeoContext. After observation, we consider terms that have a score of at least 40 because terms below the threshold of 40 are less conceptually related. We then pass each similar term as well as the original keyword term to the Twitter Streaming API to track.
After initializing the stream of tweets with the list of expanded keywords as parameters, the
tweets are passed through the geolocation step and then to the topical clustering step, both described
in Section 3, in the pipeline. The stream of tweets is also passed through the geographical
clustering step and topic clusters can be recommended for certain locations.

3.2 Location Query Implementation

We also implemented a location query system to allow a user to provide a specific location
in geographical coordinates. GeoContext will pass the coordinates to the Twitter stream and begin
to receive tweets that are located around the coordinates. Twitter supplies both tweets that are
g etagged and tweets that have been tagged with Places whose bounding boxes intersect with the
coordinates given. The location query system is useful for discovering topics of tweets that are
being discussed in a specific geographical region. For example, events occurring around a city can
be discovered using the location query system. In our evaluation discussed in Section 5,
GeoContext discovered events occurring around the University of Alabama.

After initializing the stream of tweets, the tweets are sent through the topical clustering
pipeline described in section 3. There is no need for geographical clustering with the location query
system because the tweets come only from a specific geographical area.

4. Evaluation of GeoContext

We evaluated both approaches of the GeoContext geotopical clustering system. To perform
this evaluation, we analyze the topic clusters provided by GeoContext and compare the clusters to
those produced by LDA, which is commonly used in other geotopical clustering research. In order
to effectively evaluate the same tweets in our system and LDA, we used the same set of streamed
tweets from August 2015\(^8\). We streamed the tweets through GeoContext’s topical clustering first implementation and then ran the same tweets through the geographical clustering first implementation and LDA.

With both the TCGC implementation and the GCTC implementation, we discovered several topic clusters whose topics were trending on Twitter. The most populous topics discovered consisted of tweets advertising job openings, followed by two topics consisting of tweets talking about 5 Seconds of Summer and Justin Bieber (popular musicians), respectively, and a topic consisting of tweets talking about football in Europe. As might be expected, however, these topic clusters are spread across large geographic areas, not clustered around one or several locations.

Using our adapted TF-IDF algorithm, we extracted topic cluster recommendations for various locations using both the TCGC and GCTC approaches. We extracted the ten most populous topic clusters that were recommended to various locations and present these results for the TCGC approach in Table 1. We perform the same extraction for the GCTC approach and present the results in Table 2.

The TCGC results clearly show that different topics are important to different geographical areas. We converted geographical coordinates to city names, and the cities for which the topics are recommended are displayed in the rightmost column. By extracting topics relevant to different locations rather than simply the most popular topics, we can filter out topics that users may not care about. For example, in this evaluation GeoContext recommended tweets about a UK TV show to users in London, while tweets about an Indian TV show were recommended to users in New Delhi. Topics 1 and 3 both consist of tweets talking about a UK TV show. This is due to the fact that users were using different hashtags for the same TV show, and GeoContext was not able to recognize that

\[^8\text{Available at http://eawilliams2.students.cs.ua.edu/tweets}\]
the hashtags were related. However, both topics were recommended to users in the same location, London.

The results for the GCTC implementation are shown in Table 3. These results clearly indicate what topics are the most popular in different areas. We found that the geographical clusters were very well-defined and did not contain outliers. The topic clusters within each geographical cluster were also well-defined and contained tweets that were all closely related conceptually. The most populous topic clusters over all geographic clusters are displayed in Table 3. These topic clusters align with the most populous topic clusters found in the TCGC implementation. However, although many of the most populous topic clusters extracted are similar over different areas, we were also able to extract more location-specific events, such as topics about baseball games. As with the TCGC implementation, by recommending topics that are trending in different geographical areas, GeoContext can provide more relevant information to users in those areas, rather than topics that are important in other areas of the world.

We used Mallet\(^9\) to run LDA on the same set of tweets with varying number of topics. First, we ran LDA on the set of tweets with no prior clustering or filtering. These LDA results are shown in Table 3. Using GeoContext’s geotopical clustering algorithm, we discovered about 50 topic clusters that are of significant size (more than 10 tweets), so we ran LDA with 20, 50, and 100 topics. Due to space constraints, we display only the 5 highest-weighted topics from LDA in Table 3 for each run. As is evident from the results, LDA produces topics that are much less defined than GeoContext. We believe this is due to the fact that, although LDA removes stop words, many other words such as “I’m,” “cool,” and “nice” are not removed. These terms are common, therefore they show up within the produced topics, but they do not add significant meaning to a tweet.

\(^9\) [http://mallet.cs.umass.edu/](http://mallet.cs.umass.edu/)
Next, we clustered the tweets using DBSCAN prior to running them through LDA. Each geographical cluster was considered a document for input to LDA. The results are shown in Table 4. The topics from this approach are more defined than the non-clustered results. For example, topic 2 for 20 topics, topic 3 for 50 topics, and topic 4 for 100 topics contains terms regarding popular musicians. However, the resulting topics are still much less defined compared to topics discovered from GeoContext. Terms related to jobs and hiring are spread over several topics. Also, many topics that were discovered with GeoContext are missing with LDA. For example, there are no topics that include Celebrity Big Brother, which was one of the most popular topics discovered by GeoContext. Interestingly, for all 6 LDA runs, the highest-weighted topic is very similar in each run. This topic consists mainly of common terms used in tweets.

We also analyzed our results from the GeoContext keyword query system. We set the keyword parameter as “traffic” in the keyword query system. The keyword query system expands the term “traffic” to other terms such as “congestion,” “travel,” and “transportation.” We also initialized a stream of tweets without the keyword expansion. We extracted concepts and keywords from 2000 tweets streamed from Twitter from streams both with keyword expansion and without keyword expansion. The five topic clusters with the most tweets are displayed in Table 5 for both streams. As shown, the topics in the stream with keyword expansion were able to discover multiple topics related to all types of traffic, shown in topics 1 through 4. Topic 5 was discovered due to users tweeting about a football team being in the “top flight,” and “flight” was a term to which “traffic” was expanded. Although the results from the keyword expansion system discovered some tweets not completely related to traffic, the results from the system with no keyword expansion are much less defined. For example, Topic 1 includes any tweets with the hashtag #traffic, which included tweets from road construction to driving website traffic.

Finally, we analyzed our results from the GeoContext location query system. We set the coordinate parameters as the geographical coordinates of the University of Alabama. We extracted
concepts and keywords from 6096 tweets streamed from Twitter. The top 5 most populous topic clusters are displayed in Table 6. Topics 1, 3, and 4 are similar to some of the most populous topical clusters discovered from the geotopical clustering system. However, perhaps the most interesting result from the location query system is topic cluster #5, which consists of tweets talking about an the marching band preview night, a local event occurring on the campus of the University of Alabama. This event was not highly publicized even on the University of Alabama calendar, showing that GeoContext can be a useful tool for clustering what people are tweeting about in an area and discovering new topics that may not be able to be discovered elsewhere.

6. Related Work

Kim et al. [10] detected “hot topics” from Twitter posts by normalizing high frequency words over time. This approach allowed words with a frequency that dramatically increased in a short period of time, such as words related to holidays or major events, to appear. They also used a Louvain community detection algorithm [11] to discover in which states topics were being tweeted. Their method, however, has a drawback in that some topics may be suppressed if the topics contain mostly high frequency words.

Yin et al. [12] introduced Latent Geographical Topic Analysis, or LGTA, which is their extension of LDA to take geographical information into account within a corpus of text. Rather than cluster text by document as in traditional LDA, the LGTA algorithm uses a textual corpus clustered by region to derive topics from text. LGTA discovers topics that are grouped together by geographical region. There are several limitations to their method; namely, the fact that the number of desired geographical topics must be known beforehand. Also, parameters to the algorithm must be estimated prior to the algorithm, making it inefficient for use on a real-time system, because parameters may need to be re-estimated often.
Zhang et al. [13] described their system for clustering text by topic and geographical location. Their approach is similar to Yin et al., in that they use LDA to discover topics in the corpus, a collection of unordered textual documents. They also separate the corpus by region. The authors combine LDA with DBSCAN [8], a clustering algorithm, to produce six different topic and geographical clustering algorithms. With all algorithms, however, the number of topics and clusters must still be set beforehand. These parameters may be difficult to determine for a large-scale, real-time system.

Other research has expanded beyond LDA. Vosecky et al. [14] introduced their Multi-Faceted Topic Model, which incorporates all facets of information present in tweets, including people, location, and organization entities and a time element. Hong et al. [15] built their Content Model based on Binomial Logistic Regression. The Content Model extracts content from tweets by expanding the URLs found in many tweets. The Content Model also takes into account the number of retweets.

Son et al. [16] described their method, called Probabilistic Explicit Semantic Analysis (PESA), which compares locations for the purpose of location recommendation. The authors represent each space as a set of topics gleaned from Wikipedia. Like our work, they attempt to calculate a semantic distance between topics associated with a location in order to determine which locations are similar. However, they are not applying this work to social media and mining topics from user posts.

Sakaki et al. [17] presented their approach for detecting earthquakes and other major events by analyzing a real-time stream of tweets. The authors use a classifier to determine if a user is tweeting about an event happening in real-time or whether the tweet is not referring to a major event or is irrelevant. This work differs from our work in that Sakaki et al. are detecting only pre-defined events of a large scale by filtering by keywords related to the event. Their approach also
will not detect multiple events occurring in different locations simultaneously, while GeoContext is able to detect multiple events of any type at differing locations automatically.

Hong et al. [18] model a stream of tweets across geographical locations. Through their model, they are able to predict a location of a user given the topics of the tweet and a user’s location history. Although they are mining topics from each tweet, similar to our work, their system does not attempt to model events as they happen across Twitter.

7. Future Work and Conclusion

There are two main areas we plan to focus on in future research. First, we plan to utilize sentiment analysis of individual tweets in future work to determine the overall sentiment of a tweet cluster. This could allow the system to determine the urgency of notifying a user of a topic, such as extreme weather scenarios, which would likely have a very negative sentiment over all tweets in the cluster. Second, we plan to improve the geolocation step in the GeoContext pipeline. Previous research has focused on extracting geographical coordinates from tweets via the user’s friends and follower graphs [19] and content of the tweets [20]. We plan to combine several of these approaches to significantly improve the geolocation of streamed tweets.

In conclusion, we implemented GeoContext, a novel method for clustering streamed tweets geographically and topically. GeoContext is able to discover topics that are unique to various locations and recommend topics of interest for users in those locations. We also implemented a system for GeoContext to cluster tweets that are filtered by keywords and location coordinates. The keyword query system uses cognitive computing techniques to expand keywords into collections of keywords that represent a context. The location query system provides clusters of tweets around one specific location and was able to extract a non-publicized local event that might be of interest to users. We evaluated all resulting topics extracted from a stream of tweets, and GeoContext was
able to discover more defined topics than LDA, an algorithm commonly used in topical clustering implementations.
References


Natural Language Processing, 2013.


<table>
<thead>
<tr>
<th>Topic Num</th>
<th>Num Tweets</th>
<th>Extracted Topic</th>
<th>Example Concepts</th>
<th>Recommended Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>95</td>
<td>Celebrity Big Brother (UK TV show)</td>
<td>#CBB, TV, celeb housemates</td>
<td>London, UK</td>
</tr>
<tr>
<td>Topic 3</td>
<td>20</td>
<td>Celebrity Big Brother (UK TV show)</td>
<td>#cbb,</td>
<td>London, UK</td>
</tr>
<tr>
<td>Topic 4</td>
<td>18</td>
<td>Astrology</td>
<td>#Pisces, #Leo</td>
<td>Berlin, Germany</td>
</tr>
<tr>
<td>Topic 5</td>
<td>17</td>
<td>News</td>
<td>#news, #Iran</td>
<td>London, UK</td>
</tr>
<tr>
<td>Topic 6</td>
<td>16</td>
<td>Jhalak Dikhla Jaa (Indian TV show)</td>
<td>#InjusticeToVivianDsena</td>
<td>New Delhi, India</td>
</tr>
<tr>
<td>Topic 7</td>
<td>13</td>
<td>MSG2 trailer release (movie)</td>
<td>#MSG2TrailerLaunch, Gurmeetramrahim</td>
<td>New Delhi, India</td>
</tr>
<tr>
<td>Topic 8</td>
<td>13</td>
<td>Market research/Business</td>
<td>Profit, business, forecast</td>
<td>London, UK</td>
</tr>
<tr>
<td>Topic 9</td>
<td>12</td>
<td>Leila de Lima (Philippine Secretary of State)</td>
<td>#DeLimaBringTheTruth</td>
<td>Manila, Philippines</td>
</tr>
<tr>
<td>Topic 10</td>
<td>11</td>
<td>School</td>
<td>#tipsforyear7s</td>
<td>London, UK</td>
</tr>
</tbody>
</table>

Table 1: Topic Recommendations from TCGC Approach
<table>
<thead>
<tr>
<th>Topic Num</th>
<th>Num Tweets</th>
<th>Extracted Topic</th>
<th>Example Concepts</th>
<th>Geographical Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Topic 1</strong></td>
<td>14</td>
<td>Celebrity Big Brother (UK TV show)</td>
<td>#cbb, TV, celeb housemates</td>
<td>London, UK</td>
</tr>
<tr>
<td><strong>Topic 2</strong></td>
<td>76</td>
<td>Job Advertisements</td>
<td>#job</td>
<td>Washington, DC, USA</td>
</tr>
<tr>
<td><strong>Topic 3</strong></td>
<td>81</td>
<td>Job Advertisements</td>
<td>#job</td>
<td>Boston, USA</td>
</tr>
<tr>
<td><strong>Topic 4</strong></td>
<td>60</td>
<td>Celebrity Big Brother (UK TV show)</td>
<td>#CBB, TV</td>
<td>Manchester, UK</td>
</tr>
<tr>
<td><strong>Topic 5</strong></td>
<td>11</td>
<td>Celebrity Big Brother (UK TV show)</td>
<td>#CBB, TV</td>
<td>Sheffield, UK</td>
</tr>
<tr>
<td><strong>Topic 6</strong></td>
<td>14</td>
<td>Job Advertisements</td>
<td>#job</td>
<td>Los Angeles, USA</td>
</tr>
<tr>
<td><strong>Topic 7</strong></td>
<td>57</td>
<td>Celebrity Big Brother (UK TV show)</td>
<td>#CBB, TV</td>
<td>London, UK</td>
</tr>
<tr>
<td><strong>Topic 8</strong></td>
<td>16</td>
<td>Cameron Dallas</td>
<td>#followmecam</td>
<td>Brasilia, Brazil</td>
</tr>
<tr>
<td><strong>Topic 9</strong></td>
<td>18</td>
<td>Job Advertisements</td>
<td>#job</td>
<td>Los Angeles, USA</td>
</tr>
<tr>
<td><strong>Topic 10</strong></td>
<td>13</td>
<td>Job Advertisements</td>
<td>#job</td>
<td>Chicago, USA</td>
</tr>
</tbody>
</table>

Table 2: Topic Recommendations from GCTC Approach
<table>
<thead>
<tr>
<th>Topic Num</th>
<th>20 Topics</th>
<th>50 Topics</th>
<th>100 Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>I’m love srt don’t amp good it’s day people time follow great lol can’t make today happy back you’re work</td>
<td>I’m love srt don’t amp good it’s day people time follow great lol can’t make today happy back you’re work</td>
<td>I’m love srt don’t good amp it’s day people time follow great can’t lol make today back you’re work life</td>
</tr>
<tr>
<td>Topic 2</td>
<td>B***qualityrt greg kidding wwe ya’ll families where’s romance brick slowly cools noooo ipostprakst longtime punishment thee fleek honour receiver #arsenal</td>
<td>Hack laughed trips innovation #india treat continues Monmouth bliss arri berne #porn alisagnola dish milestone ers malibuselfies use exhibition rap</td>
<td># week makes #followmecam win happy years give morning awesome heart excited car real_liam_payne link nice cool talking football past</td>
</tr>
<tr>
<td>Topic 3</td>
<td>Happiness #nyc discount childhood recounts oxford confusing struggles elmasritrt quizthe favorites medicine cos punch chili would’ve trap horror broadcaster jared_carrabis</td>
<td>Engagement navy joey lingerie vibes ties slick peoples plastic cuffing snd bend nollywood konstantinos vid satan adwords bare brand-new threatened</td>
<td>Miserable aidancmorenort niece auction they’d berahino liam’s shelter creative weekend’s #opportunity whut thee checking supplier programme pete teachers actively lousi</td>
</tr>
<tr>
<td>Topic 4</td>
<td>Kills smiling #defiance potus how’s supplemental coat deserves spain’s fifthharmonyrt turned cycling s*** stressful subway harsh burnley suspension maya tuition</td>
<td>Acc deal money stupidest #perfect noooo eds turnt vevort reporters golfer tart Oklahoma hopes tonite chin intro byrt dramatic faze</td>
<td>Standard hrt mentions ignores allinallbeautyrfteamaddennfl kpop greedy dummies bacon ruby sporting purse dudes ruining walsh platform tyler cultural wwe</td>
</tr>
<tr>
<td>Topic 5</td>
<td>Madison court ordered similar mode facing bus h*** details islands failing habits celebs hoverboard diff insta complain colin push recording</td>
<td>Undercover California Scottish follower hopeful lawsuit corners carry gabeturner backyard degree uptown students mail basically hpa onion Leverkusen bedrooms preferably</td>
<td>Tag tebow impossible hug insane thatsabinegirl #advertising bullet s beys complaining gateway unitekcollege split Reds you’re enjoying sight silver tickets murderer</td>
</tr>
</tbody>
</table>

Table 3: LDA Results (No Clustering)
<table>
<thead>
<tr>
<th>Topic Num</th>
<th>20 Topics</th>
<th>50 Topics</th>
<th>100 Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>I’m love it’s don’t good amp time people day great lol can’t today back #job work life happy night make</td>
<td>I’m love don’t it’s amp time day good great can’t lol today back work you’re night life that’s make people</td>
<td>I’m love don’t it’s amp day people great can’t today back you’re life happy that’s I’ve live home watch year</td>
</tr>
<tr>
<td>Topic 2</td>
<td>Follow love camerondallas harry_styles justinbieber #followmecam sos real_liam_payne niallofficial day hey happy cam carterreynolds make Louis_tomlinson you’re luke_brooks smile nashgrier</td>
<td>Video people good free check hashtag youtube photo follow god world hope music person years happy news heart city hours</td>
<td>Good time video man feel free youtube big photo music person make years feeling friends top full times news real</td>
</tr>
<tr>
<td>Topic 3</td>
<td>Sosfamily tha nowplaying sound Denmark stories icemoon active break yep edit #dkshame staff Australia success task split hii japan officer</td>
<td>Love follow camerondallas harry_styles justinbieber sos #followmecam day real_liam_payne happy make cam niallofficial Louis_tomlinson hey carterreynolds smile nashgrier birthday mtv</td>
<td>#job lol work s*** game we’re time good school hate job latest girl click hot play weekend #hiring talk high</td>
</tr>
<tr>
<td>Topic 4</td>
<td>Seattle road imam fancy hero jeans portrait manhattan French led #nfl brick skills wedding education state #autocar busty falls #seattle</td>
<td>Posted photo facebook storm silence psychological ignore don hero morning #aldubgettingcloser crochet notice grand rosymmichael cherrycrush hub tablecloth values girlideas</td>
<td>Love follow harry_styles camerondallas justinbieber sos #followmecam real_liam_payne make niallofficial Louis_tomlinson happy day cam carterreynolds smile hey birthday photo nashgrier</td>
</tr>
<tr>
<td>Topic 5</td>
<td>Blessed sets drivers shopping empty legend farm lies longer ooh tuition Puerto pair earth solo leader deal studios expecting raining</td>
<td>Wind temperature rain humidity hpa kit ops barometer rising dry grow it’s sold flying challenging theory wsw Erika drugs wishing</td>
<td>Commercial gear cars campaign playoffs topic bulls*** #art delays jonahmarais hes ignoring hiring smiles freeze techcrunch lane countdown overheard turkey</td>
</tr>
</tbody>
</table>

Table 4: LDA Results (Clustering)
<table>
<thead>
<tr>
<th>Topic Num</th>
<th>Keyword Extraction</th>
<th>No Keyword Extraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>Transportation jobs</td>
<td>#traffic</td>
</tr>
<tr>
<td>Topic 2</td>
<td>Travel</td>
<td>Johor Causeway traffic</td>
</tr>
<tr>
<td>Topic 3</td>
<td>Road closures/accidents</td>
<td>Manila traffic</td>
</tr>
<tr>
<td>Topic 4</td>
<td>Items for sale</td>
<td>Portland road closure</td>
</tr>
<tr>
<td>Topic 5</td>
<td>UK Football</td>
<td>#driverdiaries</td>
</tr>
</tbody>
</table>

**Table 5: Keyword Query Results**
<table>
<thead>
<tr>
<th>Topic Num</th>
<th>Num Tweets</th>
<th>Extracted Topic</th>
<th>Example Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>153</td>
<td>hiring</td>
<td>#job, hiring</td>
</tr>
<tr>
<td>Topic 2</td>
<td>18</td>
<td>weather</td>
<td>rain, weather forecasting</td>
</tr>
<tr>
<td>Topic 3</td>
<td>16</td>
<td>WWE NXT</td>
<td>Sasha Banks, #NXTtakeover</td>
</tr>
<tr>
<td>Topic 4</td>
<td>10</td>
<td>WWE NXT Brooklyn</td>
<td>#NXTtakeoverbrooklyn, WWE</td>
</tr>
<tr>
<td>Topic 5</td>
<td>7</td>
<td>University of Alabama</td>
<td>#rolltide, MDB</td>
</tr>
</tbody>
</table>

Table 6: Location Query Results
Figures

Figure 1 – Overview of Pipeline
Figure 2 – Comparison of Tweet Topics with Existing Topic Clusters
Figure 3 – Incorporation of New Tweet into Existing Topic Clusters