Clustering Tweets Containing Ambiguous Named Entities Based on the Co-occurrence of Characteristic Terms

Maike ERDMANN, Gen HATTORI, Kazunori MATSUMOTO, and Yasuhiro TAKISHIMA †

† KDDI R&D Laboratories, Inc. 2-1-15 Ohara, Fujimino, Saitama, 356-8502 Japan

E-mail: † (ma-erdmann, gen, matsu, takisima) @kddilabs.jp

Abstract Social media platforms such as Twitter are an invaluable source of information. However, one of the problems that arise when analyzing Twitter messages is the ambiguity of many named entities. Named entity disambiguation is usually performed by comparing the text surrounding the occurrence of the ambiguous term to the text in a knowledge base such as Wikipedia. However, texts published via social media are usually very short and written in an informal way, thus the overlap of terms is too small for accurate entity matching. Apart from that, the usage of a term in social media can differ greatly from the entities represented in the knowledge base. Therefore, we propose an unsupervised and domain independent tweet clustering method based on co-occurring terms in the tweets. Our method extracts characteristic keywords for a named entity from the tweets and adds terms proposed by Google Autocomplete. Then, it clusters all keywords according to the entity they represent and assigns one of the keyword categories to each tweet. In an experiment with ambiguous company names, car names and TV show titles, our proposed method achieved both a higher precision and a higher recall than two named entity disambiguation methods matching named entities in the tweets to corresponding Wikipedia entities.

Keywords Keyword extraction, text clustering, social media analysis

1. Introduction

Social media platforms such as Twitter are an invaluable source of information for understanding what people think about various topics such as products and companies, celebrities or news and events. Unfortunately, many named entities are ambiguous. The company name “Apple”, for instance, is also the name of a fruit. The term “Golf” can be the name of a car or the name of a sport. Therefore, it is necessary to distinguish the different meanings of a named entity.

Named entity disambiguation is usually performed by comparing the text surrounding the occurrence of the ambiguous term to the text in a knowledge base such as Wikipedia. However, texts published via social media are usually very short and written in an informal way. This leads to a small overlap of terms, making accurate entity matching difficult. Apart from that, the usage of a term in social media can differ greatly from the entities represented in the knowledge base. For instance, there are 45 Wikipedia articles related to the term “Apple”. Most of those meanings, such as the “Apple River” or the annual football match “Apple Cup”, are rarely discussed in social media. On the other hand, meanings that are commonly discussed in social media are sometimes not listed in Wikipedia, particularly the general meaning of a term. Lin et al. [1] report that when they tried to assign Wikipedia articles to entity mentions on the Internet, no suitable article could be found for one third of the entities.

Instead of performing named entity disambiguation, we propose an unsupervised tweet clustering method based on co-occurring terms in the tweets. Our method extracts characteristic keywords from tweets containing an ambiguous named entity as well as terms proposed by Google Autocomplete, an API that provides a list of autocomplete suggestions for a search term. For the term “Apple”, for instance, terms such as “iPhone”, “iOS”, “pie” or “cinnamon” can be extracted.

After that, the proposed method clusters all keywords according to the entity they represent and assigns the tweets to one of the keyword categories. Our method can be applied to any kind of named entity, such as product, company, person or location names. We will describe an experiment that we have conducted to compare the accuracy of our proposed method to DBPedia Spotlight[1] and AIDA[2], both popular tools for matching named entities in texts to corresponding Wikipedia articles.

2. Related Work

A lot of research has been conducted on named entity disambiguation (NED) [2]. Many recent approaches try to match ambiguous named entities with entities in knowledge bases such as Wikipedia [3]. Two popular implementations of such a system are DBPedia Spotlight [4] and AIDA [5]. However, most existing research aims at named entity disambiguation of much longer texts than

[1] http://spotlight.dbpedia.org
[2] https://gate.d5.mpi-inf.mpg.de/webaida/

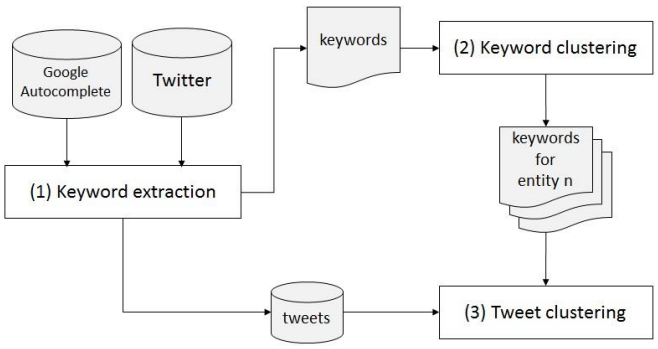

Fig.1. System Overview

those published on social media platforms.

Named entity disambiguation has also been performed on Twitter, which is much more challenging, since tweets are short and informal, making it difficult to categorize them through word overlap calculation. Several approaches have been proposed to disambiguate specific types of named entities, such as company names [6][7][8], person names [9] or TV show titles [10][11]. Depending on the domain, various external resources such as Wikipedia articles, company Web sites or electronic TV program guides are used for determining characteristic keywords.

Habib van Keulen [12] proposed a general named entity disambiguation method for Twitter that uses the Yago knowledge base consisting of entries from Wikipedia, WordNet and GeoNames. In addition, they crawl Web pages representing the entities by using the Google search engine. Then, they assign the named entities to both Wikipedia articles and Web pages, but their experiment indicates that using only Wikipedia articles achieves the best results.

Instead of performing NED, we propose an unsupervised and domain independent method for clustering tweets according to the meaning of the contained named entity. The method uses information on co-occurring terms and is therefore is suitable for any type of named entity. Besides, it does not depend on a manually compiled knowledge base, since those do not sufficiently represent all meanings of an ambiguous term in social media.

3. Clustering Tweets Using Co-occurring Keywords

In this section, we will introduce an unsupervised method for clustering tweets of short and informal text such as Twitter messages, which does not depend on a manually created external knowledge base. Our method consists of three steps, visualized in Figure 1, which we will describe in the following subsections.

3.1. Keyword extraction

In the first step, characteristic keywords are extracted from all tweets containing the ambiguous named entity. Since the tweets are not disambiguated, the keywords of different meanings are mixed up (e.g. “iPhone” and “cinnamon” for “Apple” are not separated) and thus need to be clustered based on which meaning they represent.

Unfortunately, the meanings of the named entity are usually not represented equally in the tweets. The majority of tweets containing the term “Apple”, for instance, are about the company Apple. Because of that, keywords of minority meanings are underrepresented in the keyword lists, making proper categorization difficult. If we decide to extract more keywords to ensure each meaning is covered appropriately, we face the problem that the quality of the keywords will decrease drastically.

In order to solve that problem, we decided to collect keywords using two different methods, which we will introduce in the following, as well as a combination of both methods.

The first method is a modified version of the tf-idf algorithm. At first, the top x keywords (e.g. top 10) are extracted from all tweets using tf-idf and added to a keyword repository. Then, all tweets containing the top y keywords with $y \leq x$ (e.g. top 1) are deleted from the corpus of collected tweets, before collecting the next set of top x keywords. The reason for deleting the tweets is that one of the entities in the tweets is usually dominant and keywords of this entity are overrepresented. Therefore, deleting all tweets containing the top y keywords emphasizes the tweets of minority entities and help ensure that all entities are represented by a sufficient number of keywords. In the case of “Apple”, the top keyword extracted from the collected tweets by tf-idf was “iPhone”. By removing all tweets containing that term, the percentage of tweets about the fruit increases, making it more likely to extract keywords representing them in subsequent keyword sets. The keyword collection process is repeated until enough keywords have been collected (e.g. 100 keywords).

The second method collects keywords from Google Autocomplete rather than analyzing the terms contained in tweets. Google has published an API that provides a list of autocomplete suggestions for a term, consisting of words that frequently co-occur in search queries. We selected

Table 1. Test terms

Companies	Cars	TV shows
Amazon, Apple, Citizen, Coach, Converse, Fox, Jaguar, Sharp, Subway	Civic, Escape, Explorer, Focus, Golf, Legacy, Pilot, Polo, Soul	Arrow, Castle, Empire, Nashville, Perception, Reign, Revenge, Scandal, Suits

Google Autocomplete keywords, because Web search queries usually correspond to terms that appear in tweet texts much more than entities in a knowledge base.

When combining both methods, the keywords extracted by the modified tf-idf algorithm and the keywords extracted from Google Autocomplete are simply merged.

3.2. Keyword clustering

In order to cluster the keywords based on their meaning, we estimate the semantic similarity of the keywords to each other. For each pair of keywords k_i and k_j extracted for an ambiguous term, the number of tweets containing both k_i and k_j is calculated (co-occurrence score). The idea behind this method is that similar keywords frequently co-occur in tweets whereas different keywords co-occur less frequently (e.g. “iPhone” and “iOS” co-occur much more frequently than “cinnamon” and “iOS”). After calculating the co-occurrence score of each keyword pair, the keywords are clustered based on these scores. In order to do that, the keywords are represented as nodes of a graph and the co-occurrence as weighted edges connecting two nodes, then the edge weight is set according to the co-occurrence score. A graph clustering algorithm such as CNM [13] can be used to cluster the keywords automatically. Each cluster in the graph represents one meaning of the named entity.

4. Tweet clustering

After all keywords have been assigned to a category, the tweets containing those keywords are assigned to the keyword categories. However, if a tweet contains multiple keywords of different clusters, it cannot be clustered. After all tweets containing characteristic keywords have been processed, it is possible to cluster tweets that contain no or conflicting keywords based on their similarity to the tweets that are already clustered. However, this last step is not described in this paper.

5. Experimental Results

In order to show that our proposed method works for different kinds of named entities, we conducted an experiment with 27 named entities in the categories companies, cars and TV shows. For each category, we selected 9 popular ambiguous named entities. The test terms are listed in Table 1. For the company category, we used terms from the WePS-3 workshop [14]. For the car category, we selected ambiguous terms from Web sites that rank cars by popularity. For the TV show category, we chose the most popular, currently running TV shows listed on an online TV program guide. After compiling a list of ambiguous terms for each category, we started to extract tweets for them and selected the 9 terms for which the largest number of tweets could be collected over a period of three weeks. On average, about 1 million tweets were collected per test term. Besides, we ensured that at least 100,000 tweets were collected for each of the terms.

5.1. Keyword extraction and categorization

In the first part of the experiment, we extracted keyword sets by four different methods and compared the results.

Tf-idf

The keywords in this method are extracted using the standard tf-idf algorithm.

Modified tf-idf

In this method, the tf-idf algorithm is modified according to the algorithm described in Section 3.1.

Google Autocomplete (GA)

Only the keywords collected through the Google Autocomplete API are used.

Modified tf-idf + Google Autocomplete (GA)

The keywords extracted by the modified tf-idf algorithm are merged with the keywords collected from Google Autocomplete.

For tf-idf and modified tf-idf, we extracted the 100 top ranked keywords. The global document set for calculating the inverse document frequency consisted of a large set of randomly collected tweets. The Google Autocomplete API suggests 10 keywords for each letter of the alphabet and each number from 0 to 9. We only used the keywords for each letter of the alphabet, resulting in 260 keywords. We created a keyword co-occurrence graph according to the method described in Section 3.2, but disregarded all keywords that did not appear in the tweets or did not co-occur with any other keywords. Then, we clustered all keywords using the CNM algorithm [13] and deleted all categories to which fewer than 10 keywords had been assigned.

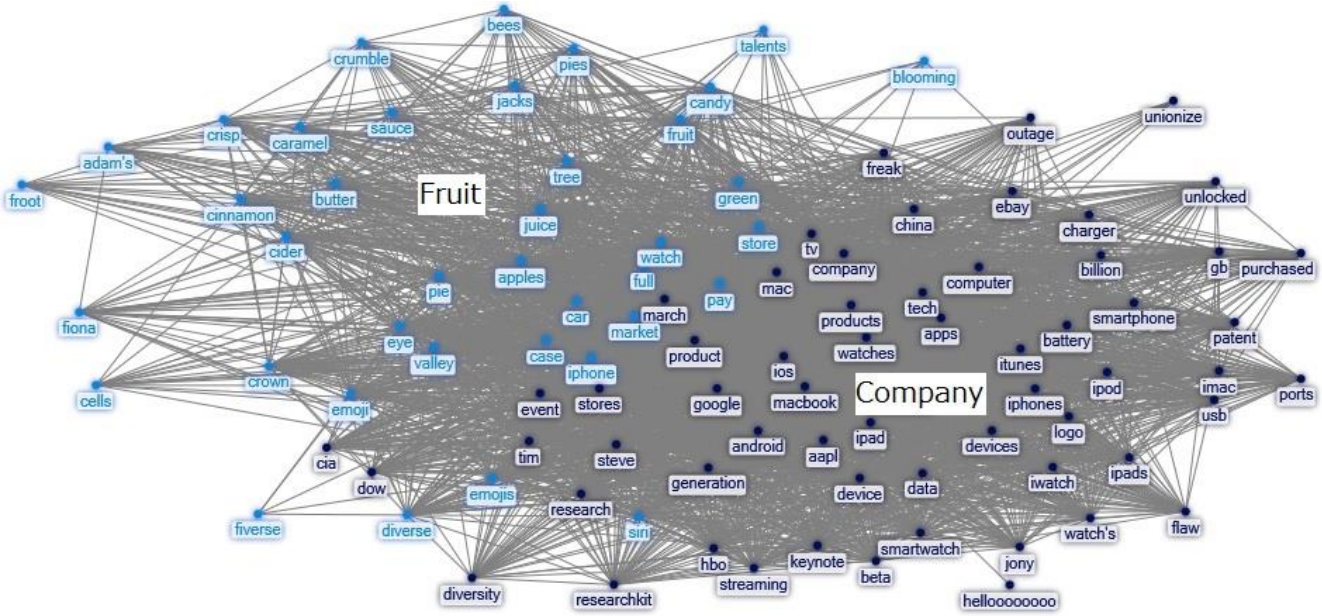

Fig. 2. Tweet clustering example (Apple)

Table 2. Keyword clustering results

	Companies	Cars	TV shows
number of terms	9	9	9
correctly clustered			
tf-idf	3	3	2
GA	3	3	2
modified tf-idf	6	4	4
modified tf-idf + GA	8	4	6

Figure 2 shows an example of a cluster structure which was constructed for the term “Apple” and visualized with the Harel-Koren Fast Multiscale layout algorithm. The graph is divided into one cluster representing the company (dark blue nodes) and another cluster representing the fruit (light blue nodes).

As shown in the graph, not all keywords have been assigned to the correct category. The keywords “watch” and “siri” have accidentally been placed in the fruit category, even though they are company related. The keyword “adam's” should be placed in none of the two categories. And for other keywords (e.g. “china” and “march”), it is unclear whether the category assignment makes sense without further examination of the tweets. However, the majority of keywords was clustered correctly.

We clustered all 27 test terms shown in Table 1 and manually evaluated the categories for each term. The numbers of terms for which keyword clustering was successful are shown in Table 2. In order to consider the keyword clustering of a term to be successful, each major entity of the term had to be represented in a separate cluster and the meaning of each cluster had to be clear for the judge when seeing the top ranked keywords assigned to the cluster. The clustering was considered to be unsuccessful if either the cluster for an important entity was missing or the entity represented by a cluster could not be recognized from the keywords (e.g. many unknown or mislabeled keywords). The major entities for each term were manually defined after manually analyzing 100 sample tweets per term prior to the experiment.

As expected, standard tf-idf did not perform well, since minority meanings were not represented in the keywords for most terms used in the experiment. Google Autocomplete also did not perform well, facing the same problem as tf-idf. Slightly better results were achieved with our modified version of tf-idf, since the method ensures more diverse keywords. The best clustering performance was achieved by combining keywords from modified tf-idf and Google Autocomplete. Comparing both keyword sets, we realized that the keywords extracted by Google Autocomplete often emphasized a different meaning than the ones extracted from modified tf-idf. For the term “Coach”, for instance, the keywords extracted from Google Autocomplete were mostly company related, while the keywords extracted from tweets using modified tf-idf carried the meaning of the term in the sports domain. For that reason, Google Autocomplete keywords are useful to complement underrepresented keyword meanings. Only for the car domain, the keywords for less than half of the test terms could be clustered successfully. This is because

Table 3. Tweet clustering results

		Companies	Cars	TV shows	Average
AIDA	categories	4.25	0.67	1.67	2.19
	precision	65.33	100.0	90.86	85.40
	recall	11.75	0.89	8.67	7.10
	f-measure	0.20	0.02	0.16	0.13
DBPedia Spotlight	categories	28.78	30.67	30.11	29.85
	precision	61.08	50.18	53.24	54.83
	recall	50.94	31.09	25.80	35.94
	f-measure	0.56	0.38	0.35	0.43
m.tf-idf +GA	categories	2.56	3.33	2.78	2.89
	precision	68.69	51.77	61.59	60.68
	recall	54.64	53.10	53.89	53.87
	f-measure	0.61	0.52	0.57	0.57

the general meaning of the terms (e.g. “Escape”, “Focus”, “Pilot”) is so dominant that car related keywords were rarely extracted.

5.2. Tweet clustering

After having determined the keyword categories, we assigned all tweets to the category of the keywords they contain. We disregarded the tweets that did not contain any keywords or that contained keywords in multiple categories. To evaluate our proposed method, we compared our results to the annotation results of DBPedia Spotlight and AIDA. For each of the 27 test terms, we randomly selected 1,000 tweets and manually evaluated the assigned categories. The results of the tweet categorization process are shown in Table 3.

AIDA and DBPedia Spotlight clustered the tweets into more fine grained categories than our method. For instance, tweets related to the company “Apple” were assigned to entities such as “Apple iPhone 4S”, “Apple iPad”, “Apple MacBook Pro” or “Apple Inc.”. However, assuming that these subcategories can likely be grouped automatically by analyzing the Wikipedia link structure, we manually merged them in our experiment.

The average precision of the proposed method (modified tf-idf + GA) was 60.68%, whereas DBPedia Spotlight achieved only an average precision of 54.83%. For both methods, the precision varied significantly depending on the test terms. For the proposed method, it decreased drastically when important keywords were assigned to the wrong category. In case of the term “Apple”, for instance, the keyword “watch” was assigned to the fruit category, resulting in all tweets about the “Apple watch” being mislabeled. For DBPedia Spotlight, the precision decreased when characteristic keywords were missing in the tweets and thus unspecific terms extracted from the tweets were matched with terms in unrelated Wikipedia articles. The best accuracy of 85.4% was achieved by AIDA. Apparently, the algorithm classifies only entities where the classification confidence is high.

Our proposed method achieved an average recall of 53.87% with the simple keyword matching approach. In the next step, we are planning cluster the remaining tweets based on the text similarity to the already clustered tweets in order to further increase the recall. For DBPedia Spotlight, the average recall was significantly lower with 35.94% and the recall of AIDA was extremely low with only 7.1%. Besides, the number of tweets clustered with DBPedia Spotlight and AIDA greatly varied with the different categories. While relatively many tweets were clustered for the company domain, the recall for cars and TV shows was very low. The main reason was that terms in those categories are often general terms, which are not covered in Wikipedia. For instance, no Wikipedia article exists for the general meaning of the company name “Sharp”.

We also realized that for some terms, the tweet clustering process performed well even though the keyword clustering was not successful. In those cases, the unknown or mislabeled keywords apparently did not have a significant impact on the tweet classification result. However, if we cannot understand the entities from the keywords, we have to manually evaluate some of the tweets to identify the entity represented by a cluster.

6. Conclusion and Future Work

In this paper, we proposed an unsupervised method for clustering Twitter messages according to the meaning of a named entity they contain based on only co-occurring terms. Our method extracts characteristic keywords from tweets containing an ambiguous named entity and Google Autocomplete, clusters the keywords according to the entity they represent and then assigns keyword categories to the tweets. Our method can be applied to any kind of named entity, such as product names, company names person names or location names.

We have conducted an experiment with 27 ambiguous terms in the categories companies, cars and TV shows, in order to compare the accuracy of our proposed method to DBPedia Spotlight and AIDA, two popular tools for matching named entities in texts to corresponding Wikipedia articles. Our proposed method achieved a precision of 60.68% and a recall of 53.87%, whereas DBPedia Spotlight achieved only a precision of 54.83% and a recall of 35.94%. AIDA achieved a precision of

85.4%, but the tradeoff is a recall of only 7.1%.

In the future, we will increase the recall of our proposed method by clustering the currently unassigned tweets based on their similarity to the already clustered tweets. This can be achieved using, for instance, a machine learning algorithm trained on characteristic keywords extracted from the already clustered tweets. Apart from that, we will further refine our keyword extraction and keyword clustering algorithms to improve the accuracy of our proposed method, and conduct a more comprehensive experiment.

References

[1] T. Lin, Mausam and O. Etzioni, "Entity Linking at Web Scale", In Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, pp. 84-88, 2012.

[2] D.Nadeau and S. Sekine, "A survey of named entity recognition and classification", Lingvisticae Investigationes, 30, 1, pp. 3-26, 2007.

[3] J. Hoart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum, "Robust disambiguation of named entities in text", In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 782-792, 2011.

[4] P.Mendes, N. and Jakob, Max and García-Silva, Andrés and Bizer, Christian, "DBpedia Spotlight: Shedding Light on the Web of Documents", In Proceedings of the International Conference on Semantic Systems (I-Semantics), pp. 1-8, 2011.

[5] Mohamed Amir Yosef and Johannes Hoffart and Ilaria Bordino and Marc Spaniol and Gerhard Weikum, "AIDA: An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables", In Proceedings of the International Conference on Very Large Databases (VLDB), pp. 1450-1453, 2011.

[6] F. Perez-Tellez and D. Pinto and J. Cardiff and P. Rosso, "On the Difficulty of Clustering Microblogging Texts for Online Reputation Management", In Proceedings of the ACL-HLT Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA), pp. 146-152, 2011.

[7] S. Zhang, J. Wu, D. Zheng, Y. Meng, Y. Xia, and H. Yu, "Two stages based organization name disambiguity", In Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing (CICLing), pp. 249-257, 2012.

[8] Zhang, Shu, Jianwei Wu, Dequan Zheng, Yao Meng, and Hao Yu, "An Adaptive Method for Organization Name Disambiguation with Feature Reinforcing", In Proceedings of the Pacific Asia Conference on Language, Information and Computation (PACLIC), pp. 237-245, 2012.

[9] S. R. Yerva, Z. Miklós, and K. Aberer. "Quality-aware similarity assessment for entity matching in web data", Information Systems, 37(4): pp. 336-351, 2012.

[10] O. Dan, J. Feng, and B. D. Davison, "A Bootstrapping Approach to Identifying Relevant Tweets for Social TV", In Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM), pp. 462-465, 2011.

[11] B. Renger and J. Feng and O. Dan and H. Chang and L. Barbosa, "VoiSTV: Voice-Enabled Social TV", In Proceedings of the International World Wide Web Conference (WWW), pp. 253-256, 2011.

[12] M. Habib and M. van Keulen, "A generic open world named entity disambiguation approach for tweet", In Proceedings of the International Conference on Knowledge Discovery and Information Retrieval (KDIR), pp. 267-276, 2013.

[13] M. E. J. Newman and M. Girvan, "Finding community structure in very large networks", Physical Review E, 70(6), 2004.

[14] E. Amigó, J. Artiles, J. Gonzalo, D. Spina, B. Liu, and A. Corujo, "Weps-3 evaluation campaign: Overview of the online reputation management task", In Proceedings of the Web People Search Evaluation Workshop (WePS), pp. 1-13, 2010.