Using Knowledge Graphs to Explain Entity Co-occurrence in Twitter

Yiwei Wang*
Hong Kong University of Science and Technology
Hong Kong, China
wangyw_seu@foxmail.com

Mark James Carman
Monash University
Caulfield, VIC, Australia
mark.carman@monash.edu

Yuan-Fang Li
Monash University
Clayton, VIC, Australia
yuanfang.li@monash.edu

ABSTRACT

Modern Knowledge Graphs such as DBPedia contain significant information regarding Named Entities and the logical relationships which exist between them. Twitter on the other hand, contains important information on the popularity and frequency with which these entities are mentioned and discussed in combination with one another. In this paper we investigate whether these two sources of information can be used to complement and explain one another. In particular, we would like to know whether the logical relationships (a.k.a. semantic paths) which exist between pairs of known entities can help to explain the frequency with which those entities co-occur with one another in Twitter. To do this we train a ranking function over semantic paths between pairs of entities. The aim of the ranker is to identify the path that most likely explains why a particular pair of entities have appeared together in a particular tweet. We train the ranking model using a number of lexical, graph-embedding and popularity-based features over semantic paths containing a single intermediate entity and demonstrate the efficacy of the model for determining why pairs of entities occur together in tweets.

KEYWORDS

Microblog; Information Retrieval; Importance Ranking; Machine Learning; DBPedia; Knowledge Graphs; Twitter

1 INTRODUCTION

On-line social networks have become an inalienable part of many people's lives allowing them to communicate effectively with friends and colleagues. Currently about 500 million tweets are posted on Twitter per day[1]. This mountain of data provides useful information about the popularity of various named entities (people, places, products, etc.) which are also described in knowledge graphs such as DBPedia [2]. Many tweets contain more than one named entity and knowledge graphs can provide semantic relations (paths) between the entities to explain their co-occurrence. For example, Fig. 1 shows a tweet referring to two entities, Adidas and Nike Inc., and the semantic relations containing one intermediate entity between them present in DBPedia, as shown by RelFinder[2].

Figure 1: A tweet referring to the entities *Nike* and *Adidas* (above), and semantic relations linking them (below).

For any given pair of named entities, many possible paths may link them in the knowledge graph. The issue we investigate in this paper is how best to rank these relations, as represented by the intermediate entities that lie along the path between the entities, for the purpose of explaining their co-occurrence. To the best of our knowledge, this is the first research work aimed at ranking the semantic relations between popular entities in Twitter. Our method can be described as follows:

- We propose an approach for automatically labelling semantic paths for building a large training corpus of labelled semantic paths, alleviating the need for manual labelling and allowing us to scale to larger training quantities: a dataset of approx. 10 million tweets with 4 hundred thousand pairs of co-occurring entities.
- We propose several features for predicting the importance of different paths based on lexical information, knowledge graph embeddings and on-line popularity information.
- We cast the problem as a rank learning problem and train a RankSVM model to rank paths.
- Our preliminary evaluation using a human-labelled dataset in terms of NDCG@k shows promising results. Our analysis also identified features most important to the ranking algorithm.

2 METHOD

We now describe methods used for data collection, labelling and feature extraction.

*Work was conducted while on placement at Monash University, and was supported partly by the China Scholarship Council.

[1] http://www.internetlivestats.com/twitter-statistics/

[2] http://www.visualdataweb.org/relfinder/

CIKM'17, November 6–10, 2017, Singapore, Singapore

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133161

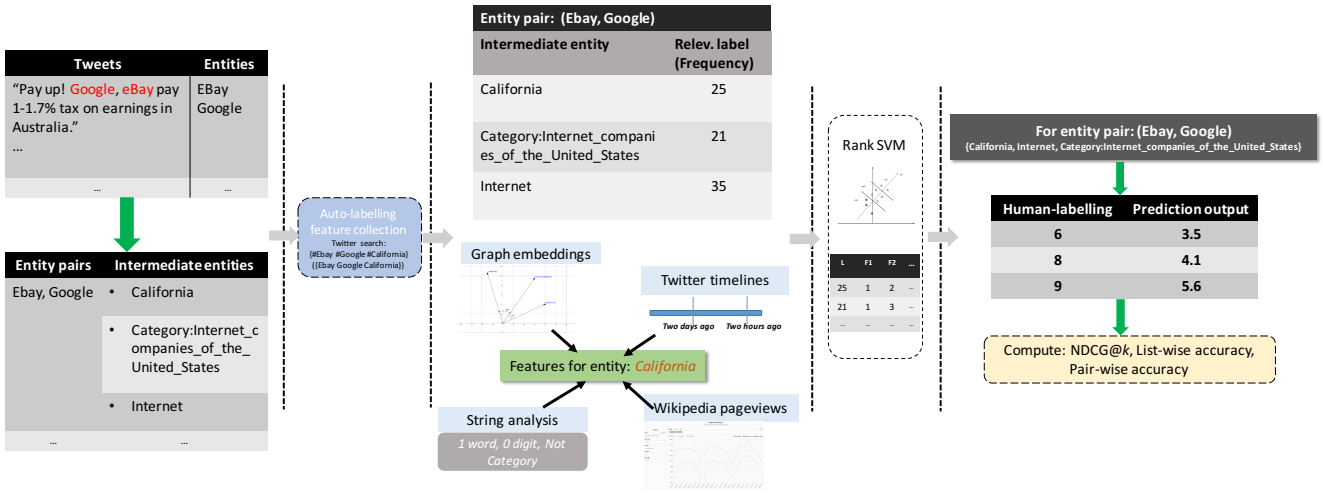

Figure 2: The data processing pipeline of our system.

2.1 Data Collection

Fig. 2 shows the data processing pipeline for the system. Twitter's streaming API is used to collect tweets, from which named entities that are present in DBPedia are extracted using DBPedia Spotlight [5], with confidence set to 0.8 and support set to 20. Twitter's streaming API requests query terms, so we used the top 400 words from a list of the 500 most frequent words on Twitter[3] as queries to ensure maximum coverage of tweets (more than 91% of tweets contain at least one word from this list). Tweets containing at least two entities are stored for further analysis.

For each pair of entities that co-occur in a tweet, queries are executed against a DBPedia SPARQL endpoint to collect the set of intermediate entities that appear in semantic relations between them (e.g., the entities *Clothing*, *Sports equipment* and *Fashion accessory* that connect *Nike, Inc.* and *Adidas* in Fig. 1).

We limited SPARQL queries to only return paths containing exactly one intermediate entity linking a pair of co-occurring entities. Semantic relations directly connecting the observed entities through a single predicate (i.e., without an intermediate entity) were very rare[4] and thus insufficient for explaining the co-occurrence in most cases. Longer relations containing two or more intermediate entities were so prolific (due to DBpedia's high branching factor) that they were infrequently meaningful and were ignored in this work. Pairs of entities with no paths between them in DBPedia are removed from further analysis. Future work will investigate the ranking of paths of varied length.

Over a two-month period more than 10 million tweets were collected, from which 689,336 occurrences of pairs of entities were extracted. These occurrences cover 90,981 unique entities and 383,234 unique pairs of entities. Fig. 3 shows a histogram of the number of paths between pairs of entities (y-axis) that contain a certain number of intermediate entities (x-axis). As can be seen, though most pairs do not have more than 100 intermediate entities, a huge number of intermediate entities do exist between some pairs, which demonstrates the challenging nature of the proposed research.

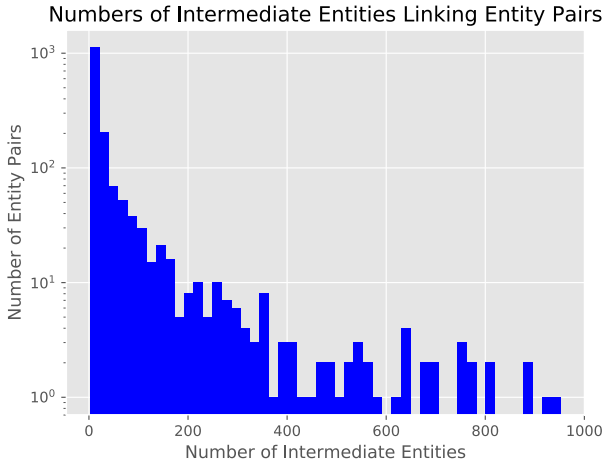

Figure 3: Number of intermediate entities between pairs of co-occurring entities in tweets.

In addition, we also collect graph embeddings of these entities from RDF2Vec [7] (with dimension of vectors set to 200), as well as pageview data of Wikipedia pages corresponding to these entities to use as features, which we will discuss in a following subsection.

2.2 Automatic labelling of training data

Effective training of a rank learning model requires a large quantity of labelled training data consisting of queries (observed pairs of entities) for which each document (semantic relation or intermediate entity linking them) has been assigned a relevance label (or rank). In this preliminary work we treat the intermediate entity as sufficient representation of the semantic relation between the

[3]http://techland.time.com/2009/06/08/the-500-most-frequently-used-words-on-twitter/
[4]The DBPedia graph contains nearly 4 million entities but has an average degree of only 7.0, see: http://konect.uni-koblenz.de/networks/dbpedia-all/.

observed entities. Future work will investigate the ranking of relations involving different predicates but the same intermediate entity/entities.

Human labelling is both tedious and time-consuming, and hence infeasible for generating the quantity of training data required to train a rank learning model. Thus we instead developed a method of automatically labelling data that uses co-occurrence information between the intermediate entity and the observed entity pair to estimate the relevance of the intermediate entity. Specifically, given a pair of co-occurring entities (e_l, e_r) and an intermediate entity e_i, we use Twitter's search functionality to find the 20 most recent tweets that contain *all three* of the entities, in the form of hashtags (e.g., `#Adidas` for the entity *Adidas*). We append the hashtag prefix # to each entity's name since if the name is used as a hashtag it is more likely to be an actual reference to the the entity (consider a tweet containing the word "apple" versus the hashtag `#apple`). For some pairs of entities (especially those with longer names), queries using the three hashtags return zero results for almost all intermediate entities. If this is the case, we remove the hashtag prefix # from each of the entity names and repeat the search without it. We do this whenever the set of counts across the intermediate entities has variance less than 5.[5]

Finally, the average frequency for each intermediate entity (calculated as the inverse of average time interval between tweets containing all three entities) is used as its relevance label (rank), since it approximates the popularity of the intermediate entity in the context of the pair of entities (i.e., $P(e_i|e_l, e_r)$).

2.3 Features

We have designed a set of seven features, organised into three broad categories: lexical, graph embeddings based, and popularity based features.

Lexical Features. We propose three features that relate an entity to its name.

- **F1** We conjecture that the number of words in an intermediate entity's name may influence its importance, since long names can indicate that the entity denotes a more specific topic.
- **F2** Similar to the number of words, number of digits in an entity's name could correlate with its specificity.
- **F3** Whether the entity is a category is chosen as the third feature. Wikipedia uses Category pages to classify entities, these classes are not mutually exclusive but concrete.

Graph embeddings features. DBpedia's network structure reflects relationships of different entities in the world. RDF2Vec [7] is a recently proposed model that translates entities in an RDF-based knowledge graph (in this case DBPedia) to distributed vector embeddings. Represented as vectors, different entities can be compared using cosine similarity between the corresponding vectors. Let $\vec{e}$ be the vector representation of entity e in RDF2Vec. Given an intermediate entity e_i between a pair of entities e_l and e_r, we propose two features.

- **F4** Sum of the cosine similarities: $cos(\vec{e_i}, \vec{e_l}) + cos(\vec{e_i}, \vec{e_r})$. Intuitively, this feature measures how similar the intermediate entity is with both e_l and e_r.

[5]The value of the threshold was set empirically by inspecting the results.

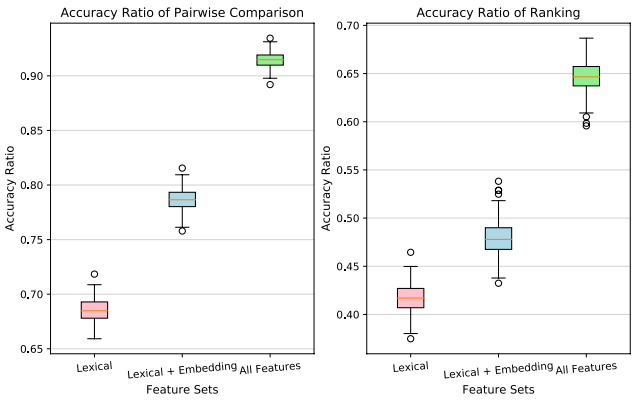

Figure 4: Accuracy ratio of pair-wise comparison and list-wise ranking.

- **F5** Absolute difference between the cosine similarities: $|cos(\vec{e_i}, \vec{e_l}) - cos(\vec{e_i}, \vec{e_r})|$. Intuitively, this feature measures the *difference* in similarity of e_i with e_l and e_r respectively.

On-line Popularity Features. We also propose two features to represent the popularity of entities.

- **F6** The log of number of pageviews of the Wikipedia page corresponding to an entity, during March 2017.
- **F7** Average time interval between consecutive co-occurrences of pairs of entities (e_i, e_l) and (e_i, e_r).

2.4 Training

After performing min-max normalisation of feature values, we train a SVM^{rank} [4] model with linear kernel function to rank intermediate entities. Three thousand two hundred pairs of most popular entities (co-occurring in most number of tweets) are chosen for training our ranker. These pairs of entities have in total more than 25,000 relations among them extracted from DBPedia. The dataset is randomly split 4:1 for training and testing, and repeated 600 times.

As described in the previous subsection, we proposed three groups of features, and these groups are increasingly expensive to calculate. The last group, on-line popularity features, is also time-sensitive.

To understand the effect of the three different groups of features, we trained ranking models with (1) lexical features only, (2) lexical and graph embeddings features, and (3) all features. The accuracy values of pair-wise comparison and list-wise ranking of these models are shown in Fig. 4. As can be seen, the more features are included, the higher the accuracy values are.

3 EVALUATION

We perform evaluation using human-labelled data to evaluate the effectiveness of our technique using the performance metric Normalised Discounted Cumulative Gain (NDCG@k).

We extracted from the pairs of entities co-occurring in Twitter those which were linked by between 8 to 10 unique intermediate

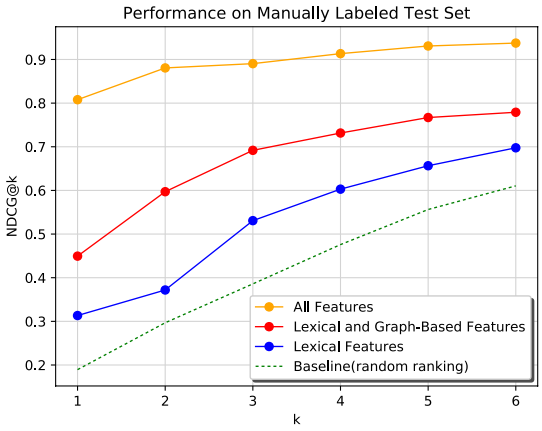

Figure 5: NDCG@k versus k (using ground truth labels).

entities in DBPedia[6], and randomly selected 50 pairs for use in the evaluation. Five participants were shown all of the intermediate entities for each of the 50 pairs, and asked to rate each intermediate entity on a scale of 0 to 10 of how "important" it was given the pair. The scores for each intermediate entity were then aggregated (summed) and used as graded relevance judgements on the scale from 0 to 50 to compute NDCG for each query pair. The NDCG@k values (for k from 1 to 6) on the manually labelled test dataset is shown in Fig. 5. For comparison random ranking is shown as well.

In human evaluation, similar to what is observed in Fig. 4, we can observe that performance (as measured by NDCG@k) improves with the addition of features. Moreover, models trained on all three groups of features outperform the random ranking baseline. The model trained on all features is the most accurate, significantly outperforming the other models, with NDCG values of at least 0.8 for all k values. It suggests that the seven features all contribute positively to ranking performance.

To understand the effects of different features on the result, we perform an ablation study to compare the drop in NDCG@k when each of the 7 features is deleted, for $k = 1$ and $k = 5$. Table 1 shows the result. Note that with the full set of features, NDCG@1 = 0.812, and NDCG@5 = 0.934.

Table 1: Ablation study - list of features, ordered from most important to least.

Feature	Δ NDCG@k	
	$k = 1$	$k = 5$
F7: Entity popularity in Twitter	0.341 (42%)	0.162 (19%)
F6: Num. of Wikipedia pageviews	0.036 (4%)	0.022 (3%)
F1: Num. of words in entity name	0.025 (2%)	0.014 (2%)
F5: Difference of entity similarity	0.024 (2%)	0.013 (2%)
F4: Sum of entity similarity	0.018 (1%)	0.008 (2%)
F2: Num. of digits in entity name	0.014 (1%)	0.004 (1%)
F3: Whether entity is category	0.013 (1%)	0.003 (1%)

[6]Note that this means 8 to 10 unique paths each with a different intermediate entity, but not paths of length 8 to 10.

The results show that F7, which is the Twitter popularity of the intermediate entity, is the most important, and the drop in NDCG@k is significantly larger than any of the other features.

4 RELATED WORK

There has been ongoing research into ranking relationships and explaining relatedness between entities *on* knowledge graphs [1, 3, 6, 8], and most of these existing work only makes use structured data encoded in knowledge graphs. In contrast, our work is concerned with the problem of ranking semantic relations between pairs of entities co-occurring in the same tweet. Moreover, we are first in incorporating novel features based on RDF graph embeddings [7].

5 CONCLUSION

Microblogging services such as Twitter have become an indispensable part of modern life. Many tweets contain multiple entities, and their co-occurrence could be explained by one or more semantic relations between these entities. In this paper, we address this problem of ranking such relations to explain co-occurrence of pairs of entities in tweets using a rank learning approach.

We propose a novel and sophisticated framework to automatically obtain labelled data for training. We propose several features for predicting the importance of different relations based on lexical information, knowledge graph structure and on-line popularity information. Our preliminary evaluation shows promising ranking accuracy as measured by NDCG@k.

Our future work plan includes identifying additional features and ranking learning algorithms to further improve ranking performance. We also plan to generalise our work to (1) support longer, arbitrary-length relations (paths) and (2) incorporate information about edges (predicates) and not only intermediate entities.

REFERENCES

[1] Kemafor Anyanwu, Angela Maduko, and Amit Sheth. 2005. SemRank: Ranking Complex Relationship Search Results on the Semantic Web. In *Proceedings of the 14th International Conference on World Wide Web (WWW '05)*. ACM, New York, NY, USA, 117–127. https://doi.org/10.1145/1060745.1060766

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. 2008. DBpedia: A Nucleus for a Web of Open Data, In Proceedings of 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC+ASWC 2007). *The Semantic Web*, 722–735. http://dx.doi.org/10.1007/978-3-540-76298-0_52

[3] Gong Cheng, Daxin Liu, and Yuzhong Qu. 2016. Efficient Algorithms for Association Finding and Frequent Association Pattern Mining. In *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I (Lecture Notes in Computer Science)*, Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil (Eds.), Vol. 9981. 119–134. https://doi.org/10.1007/978-3-319-46523-4_8

[4] Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 217–226.

[5] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia spotlight: shedding light on the web of documents. In *I-SEMANTICS (ACM International Conference Proceeding Series)*, Chiara Ghidini, Axel-Cyrille Ngonga Ngomo, Stefanie N. Lindstaedt, and Tassilo Pellegrini (Eds.). ACM, 1–8.

[6] Giuseppe Pirrò. 2015. Explaining and suggesting relatedness in knowledge graphs. In *International Semantic Web Conference*. Springer, 622–639.

[7] Petar Ristoski and Heiko Paulheim. 2016. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference*. Springer, 498–514.

[8] Stephan Seufert, Klaus Berberich, Srikanta J. Bedathur, Sarath Kumar Kondreddi, Patrick Ernst, and Gerhard Weikum. 2016. ESPRESSO: Explaining Relationships Between Entity Sets. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16)*. ACM, New York, NY, USA, 1311–1320. https://doi.org/10.1145/2983323.2983778