Kanopy4Tweets: Entity Extraction and Linking for Twitter

Pablo Torres-Tramón   Hugo Hromic   Brian Walsh   Bahareh R. Heravi   Conor Hayes
Insight Centre for Data Analytics @ NUI Galway
{fistname.secondname}@insight-centre.org

ABSTRACT

Named Entity rEcognition and Linking (NEEL) from text is an essential task in many Natural Language Processing (NLP) applications because it enables a better understanding of the content. However in the context of Social Media, NEEL is challenging due to the higher level of writing mistakes, fast language dynamics and often lack of context. To this end, we adapted KANOPY – an unsupervised graph-based topic disambiguation system – to be used for the task of NEEL in the domain of Twitter, a fast-paced micro-blogging platform. We describe the design of our solution and report the results obtained by our system using the official corpus of Tweets for the NEEL 2016 Challenge [10].

Keywords

Information Extraction, Twitter, Entity Linking

1. INTRODUCTION

We designed our system, KANOPY4TWEETS, inspired on KANOPY [7], an approach that interlinks text documents with a knowledge base such as DBPEDIA using relations between concepts and their neighbouring graph structure. KANOPY is able to discover rich knowledge that is not directly extracted from the text itself without the need of a training phase. In NEEL [10], there are two main steps: (1) named entity recognition and (2) entity linking. The first is well understood in the literature [2, 4, 5, 6], however the latter remains challenging and we use KANOPY for this purpose. However, it was not designed for Twitter text, hence we need to adapt the approach to be used for this popular micro-blogging platform.

2. KANOPY4TWEETS

Our system must deal with Tweets, very noisy and short pieces of text. KANOPY was not intended for this type of content, hence we need to adapt it. KANOPY has three stages: Name Entity Recogniser (NER), Entity Linking and Entity Disambiguation. We need to adapt each one of them in order to process tweets.

2.1 Named Entity Recognition

We integrated a NER specially designed for Twitter into KANOPY based on the well-known GATE NLP Framework [4]. In GATE, pipelines for information extraction (IE) can be built using combinations of processing resources (PR) such as tokenisers, named entities extractors, POS taggers, language detectors, ontologies, gazetteers and many more.

In recent versions, GATE now includes a ready to use Twitter-specific pipeline for IE called TWITIE [2]. It uses fine-tuned PRs for Twitter text, with the NER PR being the most heavily adapted for Twitter data. This NER is based on the state of the art Stanford NER [6] – which is based on Conditional Random Fields (CRF) models – and also incorporates many findings from the work of Ritter et al. [9]. The most challenging Twitter text processing tasks addressed by TWITIE are normalisation of slang, mis-spellings, emoticons and entity disambiguation [5].

To integrate GATE into KANOPY4TWEETS we use an embedded engine in an optimised multi-threading REST API server. This server operates in two recognition modes: (1) the TWITTER mode uses the above described TWITIE pipeline for text processing, and (2) the NORMAL mode uses the standard ANNIE pipeline included in GATE for regular text processing. Those two modes are used during the on-line Tweet processing and the off-line DBPEDIA indexing stages respectively. More details are in Section 3.

2.2 Named Entity Linking

We built a DBPEDIA index using a selection of datasets in order to find suitable resource candidates for each extracted named entity. The datasets used in our system are: [1] DBPEDIA *ontology*, *mapping based types*, *mapping based properties*, *titles*, *short abstracts*, *article categories*, *categories(labels)*, *categories(skos)* and *redirects*.

We stored the DBPEDIA datasets in a single binary file using the HDT RDF format [1]. This is a binary representation of RDF data that uses compact structures while allowing fast search functionality without the need of decompression. This type of storage has a number of benefits. For example, HDT allows to quickly browse the datasets for a particular object, subject or predicate at a glance. We processed all the mentioned DBPEDIA datasets and indexed a subset of resource properties (e.g. title, abstract, redirect, disambiguation, etc.) that best describe them.

Published as part of the #Microposts2016 Workshop proceedings, available online as CEUR Vol-1691 (http://ceur-ws.org/Vol-1691)

#Microposts2016, Apr 11th, 2016, Montréal, Canada.

[1] From: http://dbpedia.org/Downloads2015-04/

After the index is built, we use the named entities found by our NER component to query DBpedia. For each entity we obtain a list of resource candidates using the following top-down strategy: first, we search a resource with exactly the same title property as the name entity. If a result is found, we use it immediately for the next stage as a candidate, otherwise we query again the index looking for a resources with a title similar to the named entity. Again, if we find some candidates we use them for the next stage. Each time we query the index we use a query that is more general than the previous one. The last query finds matches of the named entity in the abstract property. This process stop when at least one candidate is selected. If we found no candidates, the named entity is assgined as a `NIL` resource. [2]

One challenge we found using this approach is that the number of found resource candidates tends to be large, increasing the processing time for the disambiguation stage. In such situations, we reduce the number of candidates by ranking them according to the document score assigned by the indexing engine (details in Section 3) and selecting the top-k elements.

2.3 Named Entity Disambiguation

The selected resource candidates for each named entity found in a Tweet are processed by Kanopy in an unsupervised graph-based approach for joint disambiguation that combines the same datasets mentioned in Section 2.2, resulting in a graph of *linked resources* [7]. In this graph, nodes are DBpedia resources and edges are weighted by the exclusivity of the DBpedia link type with respect to both the source and target resource nodes.

The disambiguation process takes as input a list of named entities, each containing a list of candidate DBpedia resources after the linking stage. The output is the selection of the best candidate resource for each input named entity. For this, *relatedness* of the candidates with respect to the candidate resources of all the other entities is calculated based on the number of paths between the resources weighted by the exclusivity of the edges of these paths [8].

The input named entities are then jointly disambiguated and linked to the candidate resources with the highest combined relatedness. This process is fully detailed in [7]. The score assigned by during the indexing step is used as a tie breaker in the case of multiple combinations of candidates scoring equal relatedness or in the case of tweets containing a single entity.

2.4 NIL Resources Clustering

When no resource in the knowledge base can be found to be linked to a named entity, a `NIL` resource is assigned. However, we might have sets of unlinked named entities that can be similar under some criteria. Hence we are interested on further *clustering* all the found `NIL`-linked named entities.

For this clustering, we use the following *hierarchical* and *incremental* simple approach: we iterate over each `NIL`-linked entity and aggregate them into clusters one by one. The first element is assigned into an initial cluster, then the next item is compared to the previous ones using the Monge-Elkan similarity measure [3]. This process is repeated for all the `NIL`-typed entities. If the similarity between an item and a clusters centroid is above a fixed threshold, the item is added to the cluster with the highest similarity. If no current cluster can be found with a similarity above the threshold, a new cluster for the item is then created. The choice of threshold was empirically determined.

3. IMPLEMENTATION

Kanopy4Tweets has two different pipelines as shown in Figure 1. The first pipeline is used during a one-time initial off-line processing, while the other is used during the normal on-line processing of input Tweets.

Off-line Pipeline.

As seen in Figure 1, the off-line pipeline starts with a list of input datasets to create a single HDT file. The number of datasets depends on the context of the task. In our case, we selected a small number of datasets that are mostly composed by text but we discarded those containing RDF literals. Furthermore, each output RDF triple is filtered in order to decide whether it should be included for indexing or not. For example, titles, redirections and disambiguation links are among the key properties considered.

One fundamental resource property used is the *short abstract*. For this attribute, we use our NER service in NORMAL mode to extract named entities to be used for indexing. On the other hand, the HDT file is also the input for the graph building. As mentioned in Section 2.2, this graph contains all the DBpedia resources that are indexed.

Both processes, the index and graph creation, are slow and usually take a few hours to complete. However the off-line pipeline is only required to run if the knowledge base is modified. For the indexing engine we use Elastic [3] and for handling the graph we use Neo4j [4]. Both solutions offer high performance and robustness.

On-line Pipeline.

The on-line pipeline is the main process that reads an input `TSV` file, extracts a list of Tweets and generates an ouput `TSV` file with the found linked named entities, all as shown in Figure 1. Each input Tweet is sent to our NER service in TWITTER mode and the returned list of named entities are labelled according to their function of in the text. Next, each named entity is used to search the index and a list of candidates is generated. Later, the Tweet is sent to the disambiguation service that determines which candidate DBpedia resources are best suited for the Tweet and named entity. If the link to an entity can not be determined, the named entity is goes to the `NIL` clustering component where it assigned a numbered `NIL` cluster.

NER Component.

Regardless of the NER operation mode (which is defined per instance), the REST API exposes a single POST method for entity extraction from text. This method accepts an array of texts to process and returns an array of the same texts including a per-text array of found named entities. These entities returned contain the position indices in the text where they are located, their class and any meta-data that the GATE pipeline identified, i.e. first and last names for a `Person` class or additional data from a gazetteer for `Location`s, among others. This meta-data can be later used to

[2] A place-holder resource for *unknown subjects.*

[3] `http://www.elastic.co`

[4] `http://neo4j.com`

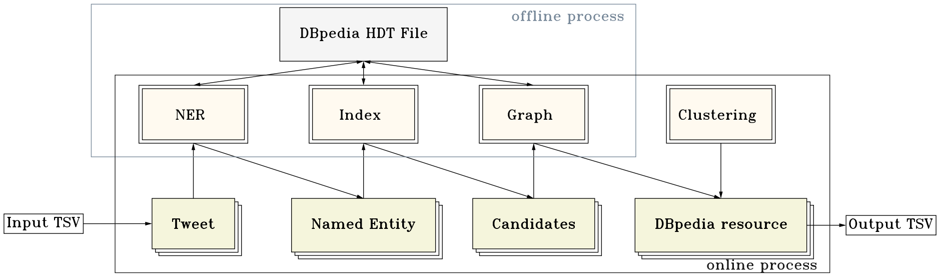

Figure 1: Diagram of the solution proposed

refine and/or further disambiguate the initially found named entities. The returned entities classes belong to the set of annotations available in the TwitIE and ANNIE pipelines: `Hashtag`, `UserID`, `URL`, `Address`, `Date`, `Identifier`, `Location`, `Money`, `Organization`, `Percent`, `Person` and `Phone.` When operating in NORMAL mode (using ANNIE), the `Hashtag`, `UserID` and `URL` classes are never returned.

4. RESULTS AND CONCLUSION

We evaluated Kanopy4Tweets using the **NEEL 2016** **`dev`** and `training` datasets. The results for each dataset are shown in the table 1. The three measures used to evaluate our system are: *i*) **Strong-typed Mention Match (SMM)** is a micro-averaged evaluation of entity mentions including the entity offset and type, *ii*) **Strong Link Match (SLM)** is a micro-averaged evaluation of links, *iii*) **Mention Ceaf (MC)** is based on a one-to-one alignment between system and gold entity clusters.

Table 1: Precision (Pr), recall (Re) and F1-score (F1) for each dataset and measure.

Dataset	Pr	Re	F1	Pr	Re	F1
		SMM			*SLM*	
`dev`	0.334	0.355	0.344	0.491	0.324	0.390
`training`	0.435	0.459	0.447	0.386	0.330	0.355
		MC				
`dev`	0.802	0.793	0.798			
`training`	-	-	-			

We presented Kanopy4Tweets, a system for NEEL of Twitter data. Our approach uses Kanopy, an unsupervised system that exploits the graph structure of DBpedia in order to disambiguate a list of candidate resources for named entities. Furthermore, our system implements a hierarchical clustering approach for grouping unlinked named entities according to their textual similarity.

Our results reported a relatively poor performance, however we believe this is mostly because of the small number of entities found in the Tweets and the hardness of lack of context in Twitter data. In addition, hashtags (words used by users to tag their Tweets) and user mentions require further special considerations in the NER phase. Finally, we want to investigate better approaches for traversing the disambiguation graph.

5. ACKNOWLEGMENT

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Numbers SFI/12/RC/2289 and 14/TIDA /2419, and National University of Ireland Galway under the Hardiman Scholarship.

6. REFERENCES

[1] M. Arias et al. Hdt-it: Storing, sharing and visualizing huge rdf datasets. In *10th International Semantic Web Conference (ISWC 2011)*, 2011.

[2] K. Bontcheva et al. Twitie: An open-source information extraction pipeline for microblog text. In *RANLP*, pages 83–90, 2013.

[3] W. Cohen et al. A comparison of string metrics for matching names and records. In *Kdd workshop on data cleaning and object consolidation*, volume 3, pages 73–78, 2003.

[4] H. Cunningham. Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254, 2002.

[5] L. Derczynski et al. Microblog-genre noise and impact on semantic annotation accuracy. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, pages 21–30. ACM, 2013.

[6] J. R. Finkel et al. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370, 2005.

[7] I. Hulpuş et al. Kanopy: Analysing the semantic network around document topics. In *Machine Learning and Knowledge Discovery in Databases*, pages 677–680. Springer, 2013.

[8] I. Hulpuş et al. *The Semantic Web - ISWC 2015, Proceedings*, chapter Path-Based Semantic Relatedness on Linked Data and Its Use to Word and Entity Disambiguation, pages 442–457. 2015.

[9] A. Ritter et al. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1104–1112. ACM, 2012.

[10] G. Rizzo, M. van Erp, J. Plu, and R. Troncy. Making Sense of Microposts (#Microposts2016) Named Entity rEcognition and Linking (NEEL) Challenge. In *6th Workshop on Making Sense of Microposts (#Microposts2016)*, pages 50–59, 2016.