UBC_UOS-TYPED: Regression for Typed-similarity

Eneko Agirre
University of the Basque Country
Donostia, 20018, Basque Country
e.agirre@ehu.es

Nikolaos Aletras
University of Sheffield
Sheffield, S1 4DP, UK
n.aletras@dcs.shef.ac.uk

Aitor Gonzalez-Agirre
University of the Basque Country
Donostia, 20018, Basque Country
agonzalez278@ikasle.ehu.es

German Rigau
University of the Basque Country
Donostia, 20018, Basque Country
german.rigau@ehu.es

Mark Stevenson
University of Sheffield
Sheffield, S1 4DP, UK
m.stevenson@dcs.shef.ac.uk

Abstract

We approach the typed-similarity task using a range of heuristics that rely on information from the appropriate metadata fields for each type of similarity. In addition we train a linear regressor for each type of similarity. The results indicate that the linear regression is key for good performance. Our best system was ranked third in the task.

1 Introduction

The typed-similarity dataset comprises pairs of Cultural Heritage items from Europeana[1], a single access point to digitised versions of books, paintings, films, museum objects and archival records from institutions throughout Europe. Typically, the items comprise meta-data describing a cultural heritage item and, sometimes, a thumbnail of the item itself. Participating systems need to compute the similarity between items using the textual meta-data. In addition to general similarity, the dataset includes specific kinds of similarity, like similar author, similar time period, etc.

We approach the problem using a range of similarity techniques for each similarity types, these make use of information contained in the relevant meta-data fields.In addition, we train a linear regressor for each type of similarity, using the training data provided by the organisers with the previously defined similarity measures as features.

We begin by describing our basic system in Section 2, followed by the machine learning system in Section 3. The submissions are explained in Section 4. Section 5 presents our results. Finally, we draw our conclusions in Section 6.

2 Basic system

The items in this task are taken from Europeana. They cannot be redistributed, so we used the urls and scripts provided by the organizers to extract the corresponding metadata. We analysed the text in the metadata, performing lemmatization, PoS tagging, named entity recognition and classification (NERC) and date detection using Stanford CoreNLP (Finkel et al., 2005; Toutanova et al., 2003). A preliminary score for each similarity type was then calculated as follows:

- General: cosine similarity of **TF.IDF** vectors of tokens, taken from all fields.
- Author: cosine similarity of **TF.IDF** vectors of dc:Creator field.
- People involved, time period and location: cosine similarity of **TF.IDF** vectors of location/date/people entities recognized by NERC in all fields.
- Events: cosine similarity of **TF.IDF** vectors of event verbs and nouns. A list of verbs and nouns possibly denoting events was derived using the WordNet Morphosemantic Database[2].
- Subject and description: cosine similarity of **TF.IDF** vectors of respective fields.

IDF values were calculated using a subset of Europeana items (the Culture Grid collection), available internally. These preliminary scores were im-

[1] http://www.europeana.eu/

[2] urlhttp://wordnetcode.princeton.edu/standoff-files/morphosemantic-links.xls

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task*, pages 132–137, Atlanta, Georgia, June 13-14, 2013.

proved using **TF.IDF** based on Wikipedia, UKB (Agirre and Soroa, 2009) and a more informed time similarity measure. We describe each of these processes in turn.

2.1 TF.IDF

A common approach to computing document similarity is to represent documents as Bag-Of-Words (BOW). Each BOW is a vector consisting of the words contained in the document, where each dimension corresponds to a word, and the weight is the frequency in the corresponding document. The similarity between two documents can be computed as the cosine of the angle between their vectors. This is the approached use above.

This approach can be improved giving more weight to words which occur in only a few documents, and less weight to words occurring in many documents (Baeza-Yates and Ribeiro-Neto, 1999). In our system, we count document frequencies of words using Wikipedia as a reference corpus since the training data consists of only 750 items associated with short textual information and might not be sufficient for reliable estimations. The **TF.IDF** similarity between items a and b is defined as:

$$sim_{\text{tf.idf}}(a,b) = \frac{\sum_{w \in a,b} \text{tf}_{w,a} \times \text{tf}_{w,b} \times \text{idf}_w^2}{\sqrt{\sum_{w \in a} (\text{tf}_{w,a} \times \text{idf}_w)^2} \times \sqrt{\sum_{w \in b} (\text{tf}_{w,b} \times \text{idf}_w)^2}}$$

where $\text{tf}_{w,x}$ is the frequency of the term w in $x \in \{a, b\}$ and idf_w is the inverted document frequency of the word w measured in Wikipedia. We substituted the preliminary general similarity score by the obtained using the **TF.IDF** presented in this section.

2.2 UKB

The semantic disambiguation UKB[3] algorithm (Agirre and Soroa, 2009) applies personalized PageRank on a graph generated from the English WordNet (Fellbaum, 1998), or alternatively, from Wikipedia. This algorithm has proven to be very competitive in word similarity tasks (Agirre et al., 2010).

To compute similarity using UKB we represent WordNet as a graph $G = (V, E)$ as follows: graph nodes represent WordNet concepts (synsets) and dictionary words; relations among synsets are represented by undirected edges; and dictionary words are linked to the synsets associated to them by directed edges.

Our method is provided with a pair of vectors of words and a graph-based representation of WordNet. We first compute the personalized PageRank over WordNet separately for each of the vector of words, producing a probability distribution over WordNet synsets. We then compute the similarity between these two probability distributions by encoding them as vectors and computing the cosine between the vectors. We present each step in turn.

Once personalized PageRank is computed, it returns a probability distribution over WordNet synsets. The similarity between two vectors of words can thus be implemented as the similarity between the probability distributions, as given by the cosine between the vectors.

We used random walks to compute improved similarity values for author, people involved, location and event similarity:

- Author: UKB over Wikipedia using person entities recognized by NERC in the dc:Creator field.
- People involved and location: UKB over Wikipedia using people/location entities recognized by NERC in all fields.
- Events: UKB over WordNet using event nouns and verbs recognized in all fields.

Results on the training data showed that performance using this approach was quite low (with the exception of events). This was caused by the large number of cases where the Stanford parser did not find entities which were in Wikipedia. With those cases on mind, we combined the scores returned by UKB with the similarity scores presented in Section 2 as follows: if UKB similarity returns a score, we multiply both, otherwise we return the square of the other similarity score. Using the multiplication of the two scores, the results on the training data improved.

2.3 Time similarity measure

In order to measure the time similarity between a pair of items, we need to recognize time expressions in both items. We assume that the year of

[3] http://ixa2.si.ehu.es/ukb/

creation or the year denoting when the event took place in an artefact are good indicators for time similarity. Therefore, information about years is extracted from each item using the following pattern: $[1|2][0-9]\{3\}$. Using this approach, each item is represented as a set of numbers denoting the years mentioned in the meta-data.

Time similarity between two items is computed based on the similarity between their associated years. Similarity between two years is defined as:

$$sim_{year}(y_1, y_2) = max\{0, 1 - |y1 - y2| * k\}$$

k is a parameter to weight the difference between two years, e.g. for $k = 0.1$ all items that have difference of 10 years or more assigned a score of 0. We obtained best results for $k = 0.1$.

Finally, time similarity between items a and b is computed as the maximum of the pairwise similarity between their associated years:

$$sim_{time}(a, b) = max_{\substack{\forall i \in a \\ \forall j \in b}}\{0, sim_{year}(a_i, b_j)\}$$

We substituted the preliminary time similarity score by the measure obtained using the method presented in this section.

3 Applying Machine Learning

The above heuristics can be good indicators for the respective kind of similarity, and can be thus applied directly to the task. In this section, we take those indicators as features, and use linear regression (as made available by Weka (Hall et al., 2009)) to learn models that fit the features to the training data.

We generated further similarity scores for general similarity, including Latent Dirichlet Allocation (LDA) (Blei et al., 2003), UKB and Wikipedia Link Vector Model (WLVM)(Milne, 2007) using information taken from all fields, as explained below.

3.1 LDA

LDA (Blei et al., 2003) is a statistical method that learns a set of latent variables called topics from a training corpus. Given a topic model, documents can be inferred as probability distributions over topics, θ. The distribution for a document i is denoted as θ_i. An LDA model is trained using the training set consisting of 100 topics using the *gensim* package[4]. The hyperparameters (α, β) were set to $\frac{1}{num\_of\_topics}$. Therefore, each item in the test set is represented as a topic distribution.

The similarity between a pair of items is estimated by comparing their topic distributions following the method proposed in Aletras et al. (2012; Aletras and Stevenson (2012). This is achieved by considering each distribution as a vector (consisting of the topics corresponding to an item and its probability) then computing the cosine of the angle between them, i.e.

$$sim_{LDA}(a, b) = \frac{\vec{\theta_a} \cdot \vec{\theta_b}}{|\vec{\theta_a}| \times |\vec{\theta_b}|}$$

where $\vec{\theta_a}$ is the vector created from the probability distribution generated by LDA for item a.

3.2 Pairwise UKB

We run UKB (Section 2.2) to generate a probability distribution over WordNet synsets for all of the words of all items. Similarity between two words is computed by creating vectors from these distributions and comparing them using the cosine of the angle between the two vectors. If a words does not appear in WordNet its similarity value to every other word is set to 0. We refer to that similarity metric as UKB here.

Similarity between two items is computed by performing pairwise comparison between their words, for each, selecting the highest similarity score:

$$sim(a, b) = \frac{1}{2}\left(\frac{\sum_{w_1 \in a} \arg\max_{w_2 \in b} UKB(w_1, w_2)}{|a|} + \frac{\sum_{w_2 \in b} \arg\max_{w_1 \in a} UKB(w_2, w_1)}{|b|}\right)$$

where a and b are two items, $|a|$ the number of tokens in a and $UKB(w_1, w_2)$ is the similarity between words w_1 and w_2.

3.3 WLVM

An algorithm described by Milne and Witten (2008) associates Wikipedia articles which are likely to be relevant to a given text snippet using machine learning techniques. We make use of that method to represent each item as a set of likely relevant Wikipedia

[4] http://pypi.python.org/pypi/gensim

articles. Then, similarity between Wikipedia articles is measured using the Wikipedia Link Vector Model (WLVM) (Milne, 2007). WLVM uses both the link structure and the article titles of Wikipedia to measure similarity between two Wikipedia articles. Each link is weighted by the probability of it occurring. Thus, the value of the weight w for a link $x \rightarrow y$ between articles x and y is:

$$w(x \rightarrow y) = |x \rightarrow y| \times \log\left(\sum_{z=1}^{t} \frac{t}{z \rightarrow y}\right)$$

where t is the total number of articles in Wikipedia. The similarity of articles is compared by forming vectors of the articles which are linked from them and computing the cosine of their angle. For example the vectors of two articles x and y are:

$$x = (w(x \rightarrow l_1), w(x \rightarrow l_2), ..., w(x \rightarrow l_n))$$
$$y = (w(y \rightarrow l_1), w(y \rightarrow l_2), ..., w(y \rightarrow l_n))$$

where x and y are two Wikipedia articles and $x \rightarrow l_i$ is a link from article x to article l_i.

Since the items have been mapped to Wikipedia articles, similarity between two items is computed by performing pairwise comparison between articles using WLVM, for each, selecting the highest similarity score:

$$sim(a,b) = \frac{1}{2}\left(\frac{\sum_{w_1 \in a} \arg\max_{w_2 \in b} WLVM(w_1, w_2)}{|a|} + \frac{\sum_{w_2 \in b} \arg\max_{w_1 \in a} WLVM(w_2, w_1)}{|b|}\right)$$

where a and b are two items, $|a|$ the number of Wikipedia articles in a and $WLVM(w_1, w_2)$ is the similarity between concepts w_1 and w_2.

4 Submissions

We selected three systems for submission. The first run uses the similarity scores of the basic system (Section 2) for each similarity types as follows:

- General: cosine similarity of **TF.IDF** vectors, IDF based on Wikipedia (as shown in Section 2.1).
- Author: product of the scores obtained obtained using **TF.IDF** vectors and UKB (as shown in Section 2.2) using only the data extracted from dc:Creator field.
- People involved and location: product of cosine similarity of **TF.IDF** vectors and UKB (as shown in Section 2.2) using the data extracted from all fields.
- Time period: time similarity measure (as shown in Section 2.3).
- Events: product of cosine similarity of **TF.IDF** vectors and UKB (as shown in Section 2.2) of event nouns and verbs recognized in all fields.
- Subject and description: cosine similarity of **TF.IDF** vectors of respective fields (as shown in Section 2).

For the second run we trained a ML model for each of the similarity types, using the following features:

- Cosine similarity of **TF.IDF** vectors as shown in Section 2 for the eight similarity types.
- Four new values for general similarity: **TF.IDF** (Section 2.1), LDA (Section 3.1), UKB and WLVM (Section 3.3).
- Time similarity as shown in Section 2.3.
- Events similarity computed using UKB initialized with the event nouns and verbs in all fields.

We decided not to use the product of **TF.IDF** and UKB presented in Section 2.2 in this system because our intention was to measure the power of the linear regression ML algorithm to learn on the given raw data.

The third run is similar, but includes all available features (21). In addition to the above, we included:

- Author, people involved and location similarity computed using UKB initialized with people/location recognized by NERC in dc:Creator field for author, and in all fields for people involved and location.
- Author, people involved, location and event similarity scores computed by the product of **TF.IDF** vectors and UKB values as shown in Section 2.2.

5 Results

Evaluation was carried out using the official scorer provided by the organizers, which computes the Pearson Correlation score for each of the eight similarity types plus an additional mean correlation.

Team and run	General	Author	People_involved	Time	Location	Event	Subject	Description	Mean
UBC_UOS-RUN1	0.7269	0.4474	0.4648	0.5884	0.4801	0.2522	0.4976	0.5389	0.5033
UBC_UOS-RUN2	0.7777	0.6680	0.6767	0.7609	0.7329	0.6412	0.7516	0.8024	0.7264
UBC_UOS-RUN3	**0.7866**	**0.6941**	**0.6965**	**0.7654**	**0.7492**	**0.6551**	**0.7586**	**0.8067**	**0.7390**

Table 1: Results of our systems on the training data, using cross-validation when necessary.

Team and run	General	Author	People_involved	Time	Location	Event	Subject	Description	Mean	Rank
UBC_UOS-RUN1	0.7256	0.4568	0.4467	0.5762	0.4858	0.3090	0.5015	0.5810	0.5103	6
UBC_UOS-RUN2	0.7457	0.6618	0.6518	**0.7466**	0.7244	0.6533	0.7404	0.7751	0.7124	4
UBC_UOS-RUN3	**0.7461**	**0.6656**	**0.6544**	0.7411	**0.7257**	**0.6545**	**0.7417**	**0.7763**	**0.7132**	**3**

Table 2: Results of our submitted systems.

5.1 Development

The three runs mentioned above were developed using the training data made available by the organizers. In order to avoid overfitting we did not change the default parameters of the linear regressor, and 10-fold cross-validation was used for evaluating the models on the training data. The results of our systems on the training data are shown on Table 1. The table shows that the heuristics (RUN1) obtain low results, and that linear regression improves results considerably in all types. Using the full set of features, RUN3 improves slightly over RUN2, but the improvement is consistent across all types.

5.2 Test

The test dataset was composed of 750 pairs of items. Table 2 illustrates the results of our systems in the test dataset. The results of the runs are very similar to those obtained on the training data, but the difference between RUN2 and RUN3 is even smaller. Our systems were ranked #3 (RUN 3), #4 (RUN 2) and #6 (RUN 1) among 14 systems submitted by 6 teams. Our systems achieved good correlation scores for almost all similarity types, with the exception of author similarity, which is the worst ranked in comparison with the rest of the systems.

6 Conclusions and Future Work

In this paper, we presented the systems submitted to the *SEM 2013 shared task on Semantic Textual Similarity. We combined some simple heuristics for each type of similarity, based on the appropriate metadata fields. The use of lineal regression improved the results considerably across all types. Our system fared well in the competition. We submitted three systems and the highest-ranked of these achieved the third best results overall.

Acknowledgements

This work is partially funded by the PATHS project (http://paths-project.eu) funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 270082. Aitor Gonzalez-Agirre is supported by a PhD grant from the Spanish Ministry of Education, Culture and Sport (grant FPU12/06243).

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th conference of the European chapter of the Association for Computational Linguistics (EACL-2009)*, Athens, Greece.

Eneko Agirre, Montse Cuadros, German Rigau, and Aitor Soroa. 2010. Exploring knowledge bases for similarity. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC10). European Language Resources Association (ELRA). ISBN: 2-9517408-6-7. Pages 373–377.".*

Nikolaos Aletras and Mark Stevenson. 2012. Computing similarity between cultural heritage items using multimodal features. In *Proceedings of the 6th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 85–93, Avignon, France.

Nikolaos Aletras, Mark Stevenson, and Paul Clough. 2012. Computing similarity between items in a digital library of cultural heritage. *J. Comput. Cult. Herit.*, 5(4):16:1–16:19, December.

R. Baeza-Yates and B. Ribeiro-Neto. 1999. *Modern Information Retrieval*. Addison Wesley Longman Limited, Essex.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November.

D. Milne and I. Witten. 2008. Learning to Link with Wikipedia. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM'2008)*, Napa Valley, California.

D. Milne. 2007. Computing semantic relatedness using Wikipedia's link structure. In *Proceedings of the New Zealand Computer Science Research Student Conference*.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.