DE GRUYTER OPEN

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **18**, No 1
Sofia • 2018 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2018-0010

Linking Datasets Using Semantic Textual Similarity

John P. McCrae, Paul Buitelaar

Insight Centre for Data Analytics, National University of Ireland Galway, Galway H91 A06C, Ireland
E-mails: john@mccr.ae paul.buitelaar@insight-centre.org

***Abstract*:** *Linked data has been widely recognized as an important paradigm for representing data and one of the most important aspects of supporting its use is discovery of links between datasets. For many datasets, there is a significant amount of textual information in the form of labels, descriptions and documentation about the elements of the dataset and the fundament of a precise linking is in the application of semantic textual similarity to link these datasets. However, most linking tools so far rely on only simple string similarity metrics such as Jaccard scores. We present an evaluation of some metrics that have performed well in recent semantic textual similarity evaluations and apply these to linking existing datasets.*

***Keywords*:** *Linked data, link discovery, ontology alignment, semantic textual similarity, structural similarity, NLP architectures.*

1. Introduction

With the increasing amount of big datasets available from a wide variety of sources, the problem of integrating heterogeneous datasets is increasingly important for a wide variety of commercial and public sector applications. By datasets, we refer to a broad class of datasets describing named entities in the real world including lexicons, thesauri, ontologies and terminologies. One of the most important steps in this is the identification of similar elements between the content or schemas of two datasets, a process that is called *link discovery*. Link discovery can rely on both linguistic information in the form of the labels, descriptions and other textual data attached to elements of a dataset and structural information in terms of the organization of entities in a dataset as well as any semantic information that may be attached to the dataset. For the linguistic information, there has been significant research into estimating the similarity of two strings and in the context of *semantic textual similarity*, for which the annual SemEval task evaluates systems. Similarly for structural similarity, *ontology alignment* has been defined as a task that aligns two highly structured models, and has been supported by the Ontology Alignment Evaluation Initiative (OAEI). However, most real world tasks involve a mix of linguistic and structural information and the combination of this has not been studied

yet. In this paper we present Nearly Automatic Integration of SChemas (NAISC) ("Naisc" means "links" in Irish and is pronounced "nashk"), an architecture that combines semantic similarity at the structural and textual levels and we show that our implementation has competitive performances for both the SemEval STS and OAEI benchmarks. We then consider the case of linking WordNet and Wikipedia, a data integration task that requires exploiting both the rich linguistic information in each resource as well as the structure of both resources and show that we can significantly improve the performance on this task by combining both structural and linguistic similarity approaches. We find that by applying combinations of state-of-the-art semantic textual similarity we can achieve stronger results for link discovery than by simple single feature approaches, and furthermore we introduce a simple modification to Jaccard index that for some datasets produces the best overall performance. We also look into computing using constraints on the mappings and show that greedy approaches often seem to outperform more exact solutions to the logical constraints.

2. Related work

The nature of linking datasets requires understanding the meaning and context of the entities in an ontology or similar dataset and it is very rare that this can be done without reading the labels and descriptions of the entities in each dataset. As such the task of *Semantic Textual Similarity* (STS) as exemplified by the tasks at SemEval [2] should be of critical importance. In these tasks, the goal is to take two texts and produce a judgement of the similarity on a scale of 0 to 5. There have been a number of attempts to solve this issue using monolingual alignment that is attempting to match words between the two sentences and use the quality of this matching to predict the alignment. In [25], a monolingual alignment was created based on matching named entities and on the use of a database of paraphrases [9] and then the overall quality was estimated by the harmonic mean of the matching precisions. More recently, the use of deep neural networks, such as the model employed in [26], whereby two 'Siamese' networks are learnt simultaneously, has produced strong results for textual similarity. These have formed the basis of systems that perform well at this task, however the use of lexico-semantic resources such as WordNet [4] are still very important for this task [24].

A survey of link discovery frameworks is given in [19] and we will briefly discuss some of the major systems that already exist for this task. Silk [27] is a tool for discovering links between datasets that relies primarily on declarative mapping languages called the Silk-Link Specification Language (Silk-LSL). CODI [21] similarly provides a mapping methodology, but in this case it is based on Markov Logic [23], which is a probabilistic logical framework allowing for soft constraints in the reasoning. Rule Miner [22] also employs a rule based approach to matching concepts but also uses a semi-supervised approach to learn the rules based on an expectation-maximization approach. A notable bottleneck in link discovery is that the comparison of every pair of elements can be quite intractable for large datasets as this quadratic in size and two systems have focussed on solving this issue. The

LIMES system [20] uses the triangle inequality in order to eliminate unnecessary pair-wise comparisons, however it is not the case in general that this inequality holds for all metrics so its application is limited. LogMap [11] applies a strategy based on lexical overlap and as such is very efficient but cannot create links where the labels are dissimilar. In general, all these methods support string based similarity measure such as Jaccard but none of them explore the combination of multiple metrics or the use of techniques from semantic textual similarity.

3. Architecture for link discovery

Our tool (NAISC) is implemented as a Java application that takes two structured datasets and returns a proposed linking between these two datasets. In addition, there is the functionality to train the model in a supervised manner and to evaluate the quality of an alignment relative to a gold standard. The input to NAISC may be in several formats; however they are implicitly converted into the NAISC internal format which models a dataset as a set of *entities.* Each of these entities may have the following:

- Labels in one or more languages
- Descriptions, that is short sentences describing the entity in one or more languages
- Relations to other entities, which must be from one of a list of supported relation types.
- A type from a short list of choices including both ontological types (Class, Property, Individual) as well as lexical types (Noun, Verb, Adjective).

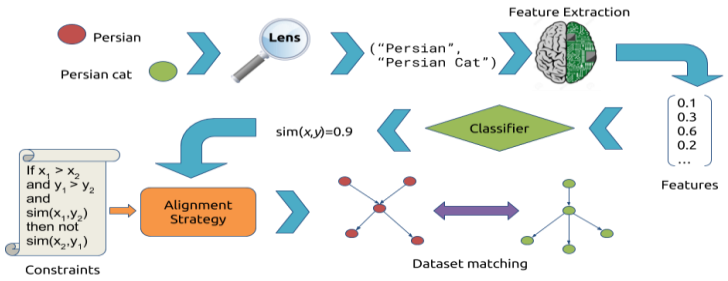

Fig. 1. The architecture of NAISC

The architecture of the system is shown in Fig. 1 and consists of a pipeline of tools controlled by the *alignment strategy*. A JSON configuration file is used to describe the particular pipeline that is used for a single run of the system. These configurations have the advantage of declaratively describing an experiment and thus enhance the reproducibility of NAISC runs. The configuration describes the language(s) of the experiment and then the elements that will be used. The *lenses* of the experiment are used to describe the elements of a pair of entities that should be examined. In the case of the example (Fig. 2) below we are looking only at the label and description strings, but we could also look at other structural features (see Section 5.1). Then for each

lens, we apply a *feature extractor*, in order to extract a numeric value, for example in this case we are using the string similarity features described in Section 4.1. This creates a fixed-length vector of features, which in a supervised learning paradigm we learn using a classifier, for which currently the only implementation is a wrapper around the Weka Toolkit [7]. The lenses, feature extractors and classifier combine to find a function that takes two entities in the dataset and produces a score related to how likely they are to be linked. The *aligner* is then tasked with using these scores in order to produce an overall linking between the datasets. Note the aligner also decides when and if to calculate similarity between entities and whether to apply any *blocking* strategies to reduce the complexity of the alignment (These are not used in the experiments reported in this paper, but the system can be configured to match only elements that are of the same type or have the same property value). In the example in Fig. 2, we see it is configured to use the *exhaustive* strategy that outputs all links that have a similarity over some threshold. In addition, this component is configured to use *negative sampling* when training, which means that it creates a number of examples by choosing pairs not found in the gold standard alignment and treats them as unaligned values, in this case creating five times as many negative examples as in the gold standard. This is necessary as datasets for link discovery typical only contain known linking and do not have negative examples. It is generally correct to assume that all entities in a dataset alignment are not linked if they are not in the set of known linked entities; however the size of this negative set grows quadratically, while the positive set tends to only grow linearly, and as such these sets can be very imbalanced. Negative sampling is thus a solution that allows us to control the ratio of positive to negative examples maintaining effective performance at test time.

```
{
   "languages": ["en"],
   "lenses": [{
      "name": "basic.Label"
   },{
      "name": "basic.Description"
   }],
   "featureExtractors": [{
      "name": "basic.Basic"
   }],
   "similarityClassifier": {
      "name": "basic.Weka",
      "path": "test.sim"
   },
   "aligner": {
     "name": "basic.Exhaustive",
     "path": "test.align",
     "params": {
       "negSampling": 5.0,
       "threshold": 0.5
            }
   }
}
```

Fig. 2. An example configuration of NAISC

4. Semantic textual similarity

In order to establish similarity between entities, an essential step in nearly all methodologies is to compare the textual labels and descriptions between these entities. As such, a major part of NAISC is the development of string similarity metrics.

4.1. String Similarity

NAISC implements a collection of string similarity metrics as follows:

- **Longest Common Substring:** The length in characters of the longest substring shared between two strings
- **Jaccard, Dice and Containment**: We consider the two strings both as a set of words and a set of character *n*-grams and compute the following functions

$$J(A, B) = |A \cap B| / |A \cup B|,$$
$$D(A, B) = 2|A \cap B| / (|A| + |B|),$$
$$C(A, B) = |A \cap B| / \min(|A|, |B|).$$

- **Length Ratio**: The ratio of the number of tokens in each sentence. For symmetry this ratio is defined as $\rho(x, y) = \min(x, y) / \max(x, y)$.
- **Average Word Length Ratio**: The average length of each word in the text is also compared as above.
- **Negation**: 1 if both texts or neither text contain a negation word ("not", "never", etc.), 0 otherwise.
- **Number**: 1 if all numbers (e.g, 6) in each text are found in the other, 0 otherwise.

We have experimented with other string metrics (including edit distance-based metrics) but have not found them to be useful in our experiments, so have reduced the set of string similarity metrics to those that capture distinct differences in the input texts well.

4.2. Smoothed jaccard

On the evaluation datasets, we have found that extracting only one feature, namely Jaccard, can produce 60-90% of the performance of our overall system; however this feature can be very poor on short texts. Our intuition is that the total number of matching words is also important, so that matching 1 out of 3 words is less important than matching 4 out of 12 words. We model this by applying a nonlinear function to the counts as follows:

$$J_\sigma(A, B) = \frac{\sigma\left(\left|A \cap B\right|\right)}{\sigma\left(\left|A\right|\right) + \sigma\left(\left|B\right|\right) - \sigma\left(\left|A \cup B\right|\right)},$$

where $\sigma(x) = 1 - \exp(-\alpha x)$. It can easily be shown that as $\alpha \to 0$ then this value is equivalent to the standard Jaccard as in this case $\sigma(x) \to x$.

4.3. Word alignment

The basic idea of many more sophisticated methods for evaluating the similarity of words is to first create an alignment between the two input strings and then use this

alignment in order to estimate the similarity of the strings. As such we assume that we are able to construct a $N \times M$ matrix, S, of similarity values in $\{0, 1\}$, where N and M are the length of the two strings. We use the following methods to extract a single similarity score from this matrix, which can produce similar scores regardless of the length of the input strings, and thus M and N. This is important as in general the length of the two strings we wish to compare varies and most obvious metrics cannot work on arbitrarily sized matrices.

Firstly, we consider the precision of the mapping, in terms of how many words are aligned with similarity greater than one half:

Forward Precision:

$$f = \frac{1}{N}\left|\left\{ i \mid \exists j : \frac{s_{ij}}{\sum_{j'} s_{ij'}} > 0.5 \right\}\right|.$$

Backward Precision:

$$b = \frac{1}{M}\left|\left\{ j \mid \exists i : \frac{s_{ij}}{\sum_{i'} s_{i'j}} > 0.5 \right\}\right|.$$

These two precisions can be combined by a harmonic mean; this method is due to Sultan, Bethard and Sumner [25].

Harmonized Alignment Mean:

$$m = \frac{2fb}{f+b}.$$

We also look at the normalized column/row maximums as a feature. This will be larger if we tend to have one value in the similarity matrix that is much larger and smaller is all values are approximately equal.

Column/Row Mean *p*-Max:

$$c_{f,p} = \frac{1}{N}\left(\sum_i \max_j \left(\frac{s_{ij}}{\sum_{j'} s_{ij'}} \right)^p \right)^{\frac{1}{p}},$$

$$c_{b,p} = \frac{1}{M}\left(\sum_j \max_i \left(\frac{s_{ij}}{\sum_{i'} s_{i'j}} \right)^p \right)^{\frac{1}{p}}.$$

Column/Row Mean *p*-Norm:

$$\mathrm{col}_{f,p} = \frac{1}{N}\sum_i \left(\sum_j s_{ij}^p \right)^{\frac{1}{p}},$$

$$\mathrm{col}_{b,p} = \frac{1}{M}\sum_j \left(\sum_i s_{ij}^p \right)^{\frac{1}{p}}.$$

Finally, inspired by Hurley and Rickard [10] on matrix sparsity, we used

sparsity metrics to measure the quality of an alignment. Our preliminary experiments suggested that the best metric for this would be Gaussian entropy diversity as follows:

Gaussian Entropy Diversity:

$$H_{\mathrm{G}} = \frac{1}{MN}\sum_i\sum_j -\log\left(s_{ij}^2\right).$$

4.3.1. Word embeddings

Word embeddings [17] are a well-established method for estimating word similarity and we use these to construct a similarity matrix for the above as follows, where v_i and v_j are the vectors or the i-th and j-th word respectively.

$$s_{ij} = \frac{v_i^{\mathrm{T}} v_j}{\left\|v_i\right\|\left\|v_j\right\|}.$$

4.3.2. Monolingual alignment

Monolingual alignment [25] is a process of finding which words in two similar sentences correspond and it has been shown that the proportion of words that can be aligned is directly related to the similarity of the sentence. We conduct a monolingual alignment based on a simplified method of Sultan's work as follows:

1. First align all words that exactly match in the string.
2. Apply named entity recognition (we use Stanford NER [6]) and align all words in these named entity recognition if either one word is an abbreviation of at least one word matches in the named entity. To test for abbreviations we take the first letter of each word in the named entity and compare the result string to the other form, accepting it if 75% of the characters match, e.g., "United States of America" would be shortened to "usoa" and match "USA".
3. Finally, for each word we calculate the similarity as:

$$\mathrm{sim}(w_i, w_j) = \alpha\times\mathrm{wordSim}(w_i, w_j) + (1 - \alpha)\times\mathrm{contextSim}(w_i, w_j),$$

where

$$\mathrm{contextSim}(w_i, w_j) = \frac{1}{\left|W_i\right|\left|W_j\right|}\sum_{m\in W_i}\sum_{n\in W_j}\mathrm{wordSim}\left(W_{i,m}, W_{j,n}\right),$$

and W_i is a fixed window of words around w_i and wordSim is a word similarity function. We then choose the most likely mapping for a given word in a greedy manner.

Once the word alignment has been completed we convert it into a matrix as above using 1 for alignments that we have extracted and 0 for all other values.

4.3.3. WordNet similarity

In addition, we can also use Princeton WordNet [4] as a basis to construct the similarity of two words. A number of methods have been proposed and we have implemented and tested the methods of shortest path [15], Wu and Palmer [28], Leacock and Chodorow [13], and Li and Bandar [14].

4.4. Deep learning

The use of recurrent neural networks has been shown recently to produce some of the strongest results for sentence similarity [1]. We use the method of T a i, S o c h e r and M a n n i n g [26] in which two representations of a sentence, h_L and h_R, are learnt and they are combined to produce a similarity vector as follows:

$$h_t = h_L \circ h_R,$$
$$h_p = |h_L - h_R|,$$
$$h_s = \sigma(W_t h_t + W_p h_p + b),$$

where σ is the sigmoid function and W_t, W_p and b are parameters to be learnt. In the original paper, this is then combined by means of a softmax layer to match the observed similarity scores, however this only works if the sentences are graded on an integral scale (in this case 1, 2, .., 5). Instead, we assume h_s is a one-dimensional vector and directly gives a similarity score in [0, 1]. The deeper representations are learnt by means of recurrent neural networks or LSTMs.

5. Structural similarity

A key part of deducing the similarity of two datasets is the structural similarity of the models and we approach this in two manners. Firstly, we define lenses to extract similarity between parts of the ontology, on which we can apply the string similarity techniques described above. Secondly, we define global constraints and scoring on matching between datasets and apply search to find matchings that are valid for the constraints and maximized the scoring function.

5.1. Features for structural similarity

Our structural similarity methods are principally based on finding lenses that select labels around an entity. For example, we use the *Superterms* lens to extract all labels that are labels of a superentity, that is an entity that is linked by means of some broader, superclass, superproperty or similar link. We then select a pair of labels, one from each entity that has the minimal (character-based) edit distance. NAISC uses its own property list, and the mapping from other models, such as OWL or WordNet is an implementation detail, that in all cases is straightforward. We use the following structural properties:

- Superterms: Any entity that is transitively *broader*.
- Direct Superterms: Any entity that is at most one step *broader*.
- Subterms/Direct Subterms: As superterms but using *narrower* links.
- Related: Any term that is marked as related using the *related* link.
- Range, Domain, Related Class, Properties of Class: The labels of classes that give the *range* or *domain* of a property (the property is the entity). “Related Class” is the combination of labels of ranges and domains and “Properties of Class” is any property whose range or domain is the entity.

Once we have extracted all these labels, we use the string similarity metrics defined in Section 4 to extract similarity.

5.2. Bipartite matching

Given that we have constructed a similarity function as described in Section 3, we need to find an overall best matching. A simple strategy, which we call *exhaustive*, is to accept all pairs if their similarity is above a fixed threshold; however this gives very poor results in general. A stronger assumption is the *bipartite* assumption that states that no entity can be linked to more than one entity in the other dataset. In this case, the global optimal solution can be found in polynomial time using the Hungarian Algorithm [12] and we implement this for bipartite matching. In addition to this exact solution, we have also implemented a *greedy solver*, which starts with no alignment and iteratively adds new alignments unless adding this alignment would violate the bipartite assumption. The algorithm continues until no more alignments can be added. Finally, we also implemented a system that aims to find a good solution quickly, namely a *beam search*, which searches using a beam (a fixed length array that contains only the highest scoring partial solutions). The beam search, as the greedy solver, takes each alignment ordered by its similarity score and then for each element in the beam adds a new candidate solution including this candidate, thus up to doubling the number of partial solutions in the beam.

6. Evaluation

The NAISC system is intended to work across a number of existing tasks and as such we evaluate over multiple datasets from different tasks including semantic similarity, ontology alignment and link discovery. We will briefly describe the construction of our benchmarks, whose size in terms of the number of entities in the two datasets we are linking and the total number of gold standard links is given in Table 1. Given our broad definition of link discovery there are a large number of datasets available, however these datasets have some significant weaknesses. We derive datasets from the two major complementary evaluation the Ontology Alignment Evaluation Initiative (**http://oaei.ontologymatching.org/**) and Semantic Textual Similarity task at TexEval (**http://ixa2.si.ehu.es/stswiki/index.php/Main_Page**). These tasks are complementary with one focusing on similarity of sentence length texts and the other on graph entities with short labels of 1-3 words. In order to provide a more balanced evaluation we look at the linking of Princeton WordNet and Wikipedia, both of which have a mixture of short text labels and longer descriptions.

Table 1. Size of datasets used in this work

Dataset	Entities in first dataset	Entities in second dataset	Links
STS (Train)	13,597	13,597	13,597
STS (Test)	1,186	1,186	1,186
WWIM (Train)	33,188	9.993	7,582
OAEI Anatomy	1,497	1,509	1,516
OAEI Conferences	921	921	305
OAEI Sabine	737	1,129	338
200 NS	202,680	117,659	173

6.1. Datasets

6.1.1. SemEval STS

SemEval's Semantic Textual Similarity (STS) task has occurred in most editions of SemEval and as such a significant amount of data has been collected. The goal of this task is to estimate the similarity of two sentences such as:

Sentence 1: What are the bus (*coach*) *connections from Thessaloniki, Greece to Tbilisi, Georgia?*

Sentence 2: Is there a bus from Tbilisi, Georgia to Thessaloniki, Greece?

Similarity: 4/5.

We view this as a special case of dataset alignment, where we have no structural features and each entity has only a single description with no label. Furthermore, we have a blocking strategy we can apply that matches each sentence in a one-to-one manner, thus removing the need for a structural match. We followed the structure of the 2,016 SemEval STS evaluation using the data from tasks before 2,016 as the training set and evaluating on the test data from the 2,016 task.

6.1.2. OAEI

The Ontology Alignment Evaluation Initiative [3] aims to focus on the alignment of two ontologies. This task has often emphasized the use of structural constraints and most of the entities in the ontologies used in this task have no labels at all. This can be seen as an opposite case to SemEval where we have mostly structural constraints and little linguistic information. For this reason, we mostly have to infer the label of terms from the URI of the term; this is done by de-camelcasing the fragment or, if there is no fragment, the final filename in the path. Further, we map the relations in the OWL ontologies to our list of relations. We chose a few datasets from this evaluation based on their suitability to a linking and how much linguistic information is useful for this task. For one of the datasets (Conferences) the data is split into multiple individual ontologies and the NAISC system is aware of this and does not consider linking between different subsets of the data.

6.1.3. WordNet-Wikipedia

As neither of the major evaluations above contain both linguistic and structural information we looked to find a dataset for which we could provide gold standard mappings. One of the most obvious candidates is the mapping of WordNet and Wikipedia, which has been conducted by a number of authors including as part of large resources such as BabelNet [18]. We note that the datasets we have created link to Wikipedia URLs, but that these can be quickly aligned to DBpedia or WikiData URLs, therefore in this work we do not distinguish between these resources. In particular, we used the mapping created by Fernando and Stevenson [5], where 200 synsets were annotated for equivalence to Wikipedia articles (200 NS). As a training set we also constructed a manual linking between WordNet and Wikipedia called the WordNet-Wikipedia Instance Matching (WWIM) [16]. For these resources, we use the article title as the label and the first line of the article as the description.

6.2. Evaluation

As the nature of the STS data is different from the OAEI and the WordNet-Wikipedia alignment task we use a different metric to analyse the results. In the case of the STS data we use the Pearson correlation coefficient between the predicted similarity of the entities and the gold standard similarity. Pearson correlation measures the accuracy of fitting a linear best-fit line, with +1 representing an exact positive correlation, –1 an exact negative correlation and 0 meaning no correlation at all. For the other datasets we use the F-measure at α=0.5, which is defined as follows:

$$\text{Precision}_\alpha(A, G) = \frac{\left|\{(x, y) \in A : \text{sim}(x, y) > \alpha \wedge (x, y) \in G\}\right|}{\left|\{(x, y) \in A : \text{sim}(x, y) > \alpha\}\right|} \quad [1], [1]$$

$$\text{Recall}_\alpha(A, G) = \frac{\left|\{(x, y) \in A : \text{sim}(x, y) > \alpha \wedge (x, y) \in G\}\right|}{\left|\{(x, y) \in A : (x, y) \in G\}\right|} \quad [2], [2]$$

$$\text{F-M}_\alpha = \frac{2 \times \text{Precision}_\alpha \times \text{Recall}_\alpha}{\text{Precision}_\alpha + \text{Recall}_\alpha} \quad [3]. [3]$$

We use this as it allows us to evaluate mappings such as the exhaustive mapping strategy that output very many results. When there were multiple alignments in the same dataset, we used a micro-average, i.e., we merged all the alignments as if they were in one dataset. If the datasets contained many entities that were not aligned in the gold standard, we allowed our system to map to these entities, but did not count any mappings from our system where both entities are not in the gold standard. This affects our F-Measure in comparison to other authors, however better represents a realistic use case, where there are many confounding entities and alignments are missing from the gold standard because the alignment is not known.

We first evaluated the system by comparing the effect of different algorithms for finding the mapping as described in Section 5.2. We used only the basic features described in Section 4.1 and trained a single system on the WWIM dataset and evaluated on the OAEI and 200 NS datasets. The results of this are given in Table 2. These results suggested that we could use the greedy approach for our experiments with feature extraction.

Table 2. Comparison of matching strategies

Strategy	Anatomy	Conferences	Sabine	200 NS
Exhaustive	0.199	0.265	0.090	0.286
Exact	0.708	0.305	0.438	0.454
Greedy	0.747	0.596	0.538	0.607
Beam	0.746	0.503	0.536	0.607

As there are a large number of components to NAISC, we wish to evaluate their combined performance as such we performed the following experimental settings. Unless specified otherwise we used the basic feature set (in Section 4.1) with lenses on the label and description and using the bipartite matching (Section 5.2). We thus used the following settings:

1. Longest Common Subsequence as the only feature
2. Jaccard as the only feature
3. Length Ratio as the only feature
4. Average Word Length Ratio as the only feature

5. Longest Common Subsequence, Jaccard, Dice, Containment, Length Ratio, and Average Word Length Ratio as the only features
6. All the basic features (this is the default setting)
7. Using smoothed Jaccard as the only feature with α=1
8. Using smoothed Jaccard as the only feature with α=2
9. As 6, using word embeddings and forward precision as an extra feature
10. As 9 with backward precision (instead of forward precision)
11. As 9 with harmonized alignment mean
12. As 9 with column mean *p*-Max, *p*=1
13. As 9 with column mean *p*-Norm, *p*=2
14. As 9 with Gaussian entropy diversity
15. As 9 using all metrics from Section 4.3
16. As 15 but using monolingual alignment from Section 4.3.2
17. As 15 but using *shortest path* WordNet similarity (Section 4.3.3)
18. As 15 but using Wu and Palmer [28] WordNet similarity (Section 4.3.3)
19. As 15 but using Leacock and Chodorow [13] WordNet similarity (Section 4.3.3)
20. As 15 but using Li [14] WordNet similarity (Section 4.3.3)
21. The combination of all features from 6, 7, 15 and 20
22. As 6 Using recurrent neural networks as an additional feature (Section 4.4)

Table 3. Results for matching datasets using various feature configuration. **Bold** indicates the best result; * – this computation did not complete due to cost of applying Stanford NER to all the entities

Experiment	STS Correl	Anatomy F-M	Conferences F-M	Sabine F-M	200NS F-M
1	0.605	0.763	0.623	0.747	0.558
2	0.448	0.792	0.622	0.749	0.504
3	0.113	0.016	0.050	0.014	0.009
4	0.195	0.021	0.052	0.037	0.034
5	0.587	0.736	0.581	0.630	0.561
6 (Default)	0.608	0.747	0.596	0.538	**0.607**
7	0.483	**0.863**	**0.655**	**0.759**	0.575
8	0.341	0.859	0.652	0.751	0.510
9	0.605	0.673	0.582	0.484	0.604
10	0.608	0.781	0.616	0.680	0.600
11	0.606	0.690	0.598	0.559	0.607
12	0.616	0.705	0.585	0.567	0.570
13	0.603	0.758	0.603	0.661	0.604
14	0.614	0.679	0.583	0.535	0.585
15	**0.649**	0.686	0.566	0.747	0.598
16	0.644	0.690	0.299	0.465	*
17	0.648	0.710	0.291	0.432	0.455
18	0.611	0.734	0.302	0.432	0.460
19	0.628	0.720	0.311	0.437	0.437
20	0.624	0.717	0.306	0.449	0.465
21	0.621	0.653	0.559	0.733	0.588
22	0.611	0.693	0.588	0.528	0.601

The results of these experiments are given in Table 3. In general we found that the combination of multiple features seems to help, as shown by multi-feature configurations 6 and 21 providing good results. However the usage of a single strong feature, namely the smoothed Jaccard ratio produced very strong results. We believe that this is partially due to the fact that we were training on a different dataset to those that we were testing on. As such, the issue of transferring learning from one dataset

to another is quite challenging. This explains why we see that multiple combinations of features is better on the 200 NS dataset, which is closer to the WWIM dataset used in training as these are both alignments between WordNet and Wikipedia. We also see that while the use of monolingual alignment is helpful for the task of semantic similarity, it is less useful for the case of the OAEI datasets, as these mostly only had labels, which are considerably shorter than the texts in the STS dataset. We found that conventional WordNet similarity metrics do not seem to be useful for this task. Finally, we attempted to use deep learning methods in experiment 22, however we found disappointing results and this was due to the nature of deep learning, where hyperparameters have to be carefully tuned in order to avoid overfitting. We experimented with various settings and methods such as dropout [8], however we found that we obtained very high training set correlations and comparatively poor test set correlations, with various settings.

We provide a more detailed analysis of some of the errors made by the system. These results are from configuration 7 run on the 200 NS dataset. A common source of errors is that the wrong part-of-speech is mapped to, in this case that system predicts mapping to the noun, but the gold standard (likely erroneously) maps to a verb synset (Table 4).

Table 4

Wikipedia	Tickling
WordNet Predicted	tickled, tickling, titillation (i36171, Noun)
WordNet Gold	titillate, tickle, velicate (i32357, Verb)

Many mappings are missed due to there not being an overlapping lemma and even small differences in spelling can cause obvious pairs to be missed such as in Table 5.

Table 5

Wikipedia	Cicindela
WordNet Gold	family Cicindelidae, Cicindelidae (i46843)

Finally, the large number of candidates in Wikipedia and WordNet can lead to spurious mappings, in this case the word “purple” has many subtle shades of meaning in WordNet related to the colour as a noun, verb, adjective and other related meanings “purple” as royal). In contrast, Wikipedia has only one concept related to the colour but several related to “Purple” as names for songs, albums or even stage name of artists. The system matched the primary sense of “purple” in Wikipedia to the adjective “purple” (i2123) instead of the noun (i63106), and then generates many spurious links for specific concepts (Table 6).

Table 6

Wikipedia	Purple
WordNet Predicted	purple, violet, purplish (i2123)
Wikipedia	Purple (Baroness album)
WordNet Predicted	purple (i112846)
Wikipedia	Purple (song)
WordNet Predicted	purple, empurpled, over-embellished (i11052)
Wikipedia	Purple.com
WordNet Predicted	purple, royal, imperial, majestic, regal (i8715)
Wikipedia	Babyshambles [from Purple (rapper)]
WordNet Predicted	empurple, purpurate, purple (i23142)
Wikipedia	Purple (album)
WordNet Predicted	purple, purpleness (i63106)

7. Conclusion

We have considered the task of linking datasets and have considered the use of state-of-the-art methods from the NLP task of semantic textual similarity and how to combine them with structural similarity of ontology alignment. Our results show that combining these features is of great value, however the task of training these models is still very tricky and they are prone to overfitting. However, we do see a correlation between the quality of the semantic textual similarity scores and the quality of linking between datasets, suggesting that with better semantic similarity we could obtain much higher alignment scores. In the case of dataset linking the labels of concepts are much shorter than the sentences used in semantic textual similarity and much less linguistically complex. As such, we believe that further work to develop specialized features for term equivalence would be of great benefit to this task. Finally, we notice that a significant obstacle to the task of link discovery is the diversity of the datasets, in terms of the nature of texts, the existence of additional information useful for linking, such as properties, descriptions, etc., and the ability to transfer learning from one task to another is still a significant challenge.

References

1. Agirre, E., C. Banea, D. M. Cer, M. T. Diab, A. Gonzalez-Agirre, R. Mihalcea, G. Rigau, J. Wiebe. SemEval-2016. Task 1: Semantic Textual Similarity, Monolingual and Cross-Lingual Evaluation. – In: SemEval@ NAACL-HLT, 2016, pp. 497-511.
2. Cer, D., M. Diab, E. Agirre, I. Lopez-Gazpio, L. Specia. SemEval-2017. Task 1: Semantic Textual Similarity-Multilingual and Cross-Lingual Focused Evaluation. arXiv Preprint arXiv:1708.00055, 31 July 2017.
3. Euzenat, J., C. Meilicke, H. Stuckenschmidt, P. Shvaiko, C. Trojahn. Ontology Alignment Evaluation Initiative: Six Years of Experience. – Journal on Data Semantics, Vol. **XV**, 2011, Berlin, Heidelberg, Springer, pp. 158-192.
4. Fellbaum, C. WordNet. In Theory and Applications of Ontology. Computer Applications. Netherlands, Springer, 2010, pp. 231-243.
5. Fernando, S., M. Stevenson. Mapping WordNet Synsets to Wikipedia Articles. – In: Proc. of 8th International Conference on Language Resources and Evaluation, 2012, pp. 590-596.
6. Finkel, J. R., T. Grenager, C. Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. – In: Proc. of 43nd Annual Meeting of the Association for Computational Linguistics (ACL'05), 2005, pp. 363-370
7. Frank, E., M. Hall, G. Holmes, R. Kirkby, B. Pfahringer, I. H. Witten, L. Trigg. Weka-a Machine Learning Workbench for Data Mining. – In: Data Mining and Knowledge Discovery Handbook. US, Springer, 2009, pp. 1269-1277.
8. Gal, Y., Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. – In: Proc. of International Conference on Machine Learning, 2016, pp. 1050-1059.
9. Ganitkevitch, J., B. Van Durme, C. Callison-Burch. PPDB: The Paraphrase Database. – In: HLT-NAACL, 9 Jun 2013, pp. 758-764.
10. Hurley, N., S. Rickard. Comparing Measures of Sparsity. – IEEE Transactions on Information Theory, Vol. **55**, October 2009, No 10, pp. 4723-4741.
11. Jiménez-Ruiz, E., B. C. Grau, Y. Zhou. LogMap 2.0: Towards Logic-Based, Scalable and Interactive Ontology Matching. – In: Proc. of 4th International Workshop on Semantic Web Applications and Tools for the Life Sciences, 2011, pp. 45-46.

12. K u h n, H. W. The Hungarian Method for the Assignment Problem. – Naval Research Logistics Quarterly, Vol. **2**, 1955, pp. 83-97.
13. L e a c o c k, C., M. C h o d o r o w. Combining Local Context and Wordnet Similarity for Word Sense Identification. – An Electronic Lexical Database, 1998, pp. 265-283.
14. L i, Y., Z. M. D. B a n d a r. An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. – Transactions on Knowledge and Data Engineering, Vol. **15**, 2003, No 4, pp. 871-882.
15. L i n, F, K. S a n d k u h l. A Survey of Exploiting Wordnet in Ontology Matching. – Artificial Intelligence in Theory and Practice, Vol. **II**, 2008, pp. 341-350.
16. M c C r a e, J. P. Mapping WordNet Instances to Wikipedia. – In: Proc. of 2018 Global WordNet Conference, 2018.
17. M i k o l o v, T, W. T. Y i h, G. Z w e i g. Linguistic Regularities in Continuous Space Word Representations. – In: HLT-NAACL, Vol. **13**, 9 Jun 2013, pp. 746-751.
18. N a v i g l i, R, S. P. P o n z e t t o. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. – Artificial Intelligence, Vol. **193**, 1 December 2012, pp. 217-250.
19. N e n t w i g, M., M. H a r t u n g, A. C. N g o n g a-N g o m o, E. R a h m. A Survey of Current Link Discovery Frameworks. – Semantic Web, Vol. **8**, 1 January 2017, No 3, pp. 419-36.
20. N g o n g a-N g o m o, A. C., S. A u e r. LIMES – A Time-Efficient Approach for Large Scale Link Discovery on the Web of Data. – In: Proc. of 22nd Joint International Conference on Artificial Intelligence, 2011, pp. 2313-2317.
21. N i e p e r t, M., C. M e i l i c k e, H. S t u c k e n s c h m i d t. A Probabilistic-Logical Framework for Ontology Matching. – In: Proc. of 24th AAAI Conference on Artificial Intelligence, 2010, pp. 1413-1418.
22. N i u, X., S. R o n g, H. W a n g, Y. Y o n g. An Effective Rule Miner for Instance Matching on the Web of Data. – In: Proc. of 21st ACM International Conference on Information and Knowledge Management, 2012, pp. 1085-1094.
23. R i c h a r d s o n, M., P. D o m i n g o s. Markov Logic Networks. – Machine Learning, Vol. **62**, 2006, No 1-2, pp. 107-136.
24. R y c h a l s k a, B, K. P a k u l s k a, K. C h o d o r o w s k a, W. W a l c z a k, P. A n d r u s z k i e w i c z. Samsung Poland NLP Team at SemEval-2016. Task 1: Necessity for Diversity; Combining Recursive Autoencoders, WordNet and Ensemble Methods to Measure Semantic Similarity. – In SemEval@ NAACL-HLT, 2016, pp. 602-608.
25. S u l t a n, M. A., S. B e t h a r d, T. S u m n e r. Back to Basics for Monolingual Alignment: Exploiting Word Similarity and Contextual Evidence. – Transactions of the Association for Computational Linguistics, Vol. **2**, 31 May 2014, pp. 219-230.
26. T a i, K. S., R. S o c h e r, C. D. M a n n i n g. Improved Semantic Representations from Tree-Structured Long Short-Term Memory Networks. – arXiv Preprint arXiv:1503.00075, 28 February 2015.
27. V o l z, J., C. B i z e r, M. G a e d k e, G. K o b i l a r o v. Discovering and Maintaining Links on the Web of Data. – The Semantic Web-ISWC'09, 2009, pp. 650-665.
28. W u, Z., M. P a l m e r. Verb Semantics and Lexical Selection. – In: 32nd Annual Meeting of the Association for Computational Linguistics, New Mexico State University, Las Cruces, New Mexico, 1994, pp. 133-138.

Received 29.09.2017; *Second Version 19.11.2017*; *Accepted 30.11.2017*