DeepAlignment: Unsupervised Ontology Matching With Refined Word Vectors

Prodromos Kolyvakis[1], **Alexandros Kalousis**[2], **Dimitris Kiritsis**[1]
[1]École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
{prodromos.kolyvakis,dimitris.kiritsis}@epfl.ch
[2]Business Informatics Department, University of Applied Sciences,
Western Switzerland Carouge, HES-SO, Switzerland
alexandros.kalousis@hesge.ch

Abstract

Ontologies compartmentalize types and relations in a target domain and provide the semantic backbone needed for a plethora of practical applications. Very often different ontologies are developed independently for the same domain. Such "parallel" ontologies raise the need for a process that will establish alignments between their entities in order to unify and extend the existing knowledge. In this work, we present a novel entity alignment method which we dub *DeepAlignment*. DeepAlignment refines pre-trained word vectors aiming at deriving ontological entity descriptions which are tailored to the ontology matching task. The absence of explicit information relevant to the ontology matching task during the refinement process makes DeepAlignment completely unsupervised. We empirically evaluate our method using standard ontology matching benchmarks. We present significant performance improvements over the current state-of-the-art, demonstrating the advantages that representation learning techniques bring to ontology matching.

1 Introduction

Translation across heterogeneous conceptual systems is an important challenge for cognitive science (Goldstone and Rogosky, 2002; Stolk et al., 2016). Ontology Matching constitutes the task of establishing correspondences between semantically related entities (i.e. classes and properties) from different ontologies, as illustrated in Figure 1. Similarly, ontology matching is crucial for accomplishing a mutual understanding across heterogeneous artificial cognitive agents (Taylor, 2015). However, despite the many proposed solutions, it is widely accepted that there is no solution robust enough to deal with the high ontological linguistic variability (Shvaiko and Euzenat, 2008, 2013); hampering, thus, the discovery of shared meanings.

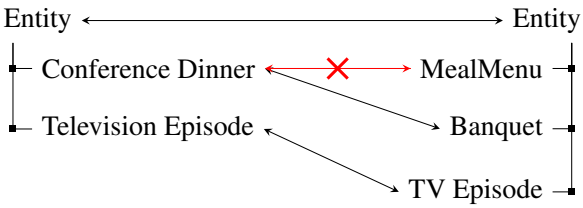

Figure 1: Example of alignments (black lines) and misalignments (red crossed lines) between ontologies.

Research in automatic ontology matching has focused on engineering features from terminological, structural, extensional (ontology instances) and semantic model information extracted from the ontological model. These features are then used to compute ontological entity similarities that will guide the ontology matching. Deriving such features for a given problem is an extremely time consuming task. To make matters worse, these features do not transfer in other domains. As Cheatham and Hitzler (2013) have recently shown, the performance of ontology matching based on different textual features varies greatly with the type of ontologies under consideration.

At the same time, machine learning research is characterised by a shift from feature engineering based approaches to feature and representation learning as a result of the performance improvements brought by deep learning methods. A by now classical example is the unsupervised learning of semantic word representations based on the *distributional hypothesis* (Harris, 1954), i.e. the assumption that semantically similar or related words appear in similar contexts (Deerwester et al., 1990; Bengio et al., 2003; Mikolov et al., 2013a,c; Pennington et al., 2014). Word vectors have the potential to bring significant value to on-

Proceedings of NAACL-HLT 2018, pages 787–798
New Orleans, Louisiana, June 1 - 6, 2018.

tology matching given the fact that a great deal of ontological information comes in textual form.

One drawback of these semantic word embeddings is that they tend to coalesce the notions of *semantic similarity* and *conceptual association* (Hill et al., 2016b). For instance, the word "harness" is highly related to the word "horse", as they share strong associations, i.e. a harness is often used on horses (Lofi, 2016). From an ontological point of view, however, these types should not be similar. Moreover, as unsupervised learning requires even larger text corpora, the learned vectors tend to bring closer words with similar frequency instead of similar meaning (Faruqui et al., 2016). Clearly, word representations that reflect frequency instead of meaning is an undesired feature if we seek to exploit word vectors for ontology matching; alignment based on such representations will reflect similar frequency instead of similar meaning.

A number of lightweight vector space representation refining techniques were introduced recently in an effort to correct these biases (Faruqui et al., 2015; Mrkšić et al., 2016). They use synonymy and antonymy constraints extracted from semantic lexicons to refine the learned word representations and make them better suited for semantic similarity tasks. Such methods are a way to inject domain-specific knowledge to tailor the learned word representations to a given task. As a result, we can exploit the synonymy/antonymy constraints to learn semantic word representations that are better candidates for ontology matching.

In this paper we learn representations of ontological entities instead of feature engineering them. We use the learned representations to compute the entities' semantic distances and to subsequently perform the ontology matching task. In order to represent the ontological entities, we exploit the textual information that accompanies them. We represent words by learning their representations using synonymy and antonymy constraints extracted from general lexical resources and information captured implicitly in ontologies. We cast the problem of ontology matching as an instance of the Stable Marriage problem (Gale and Shapley, 1962) using the entities semantic distances.

Our approach has a number of advantages. The word embeddings we establish are tailored to the domains and ontologies we want to match. The method relies on a generic unsupervised representation learning solution which is important given the small size of training sets in ontology matching problems. We evaluate our approach on the Conference dataset provided by the Ontology Alignment Evaluation Initiative (OAEI) campaign and on a real world alignment scenario between the Schema.org and the DBpedia Ontologies. We compare our method to state-of-the-art ontology matching systems and show significant performance gains on both benchmarks. Our approach demonstrates the advantages that representation learning can bring to the task of ontology matching and shows a novel way to study the problem in the setting of recent advances in NLP.

2 Related Work

2.1 Selecting Features for Ontology Matching

The vast majority of ontology matching research follows the feature engineering approach (Wang and Xu, 2008; Cruz et al., 2009; Khadir et al., 2011; Jiménez-Ruiz and Grau, 2011; Fahad et al., 2012; Ngo and Bellahsene, 2012; Gulić et al., 2016). Features are generated using a broad range of techniques (Anam et al., 2015; Harispe et al., 2015), ranging from the exploitation of terminological information, including structural similarities and logical constraints, such as datatype properties, cardinality constraints, etc.

Ontology matching is done by acting on the aforementioned features in different ways. Heuristic methods that rely on aggregation functions, such as *max, min, average, weighted sum*, etc., to fuse the information found in these features are quite popular (Anam et al., 2015). Other approaches use first order logic and cast ontology matching as a satisfiability problem (Giunchiglia et al., 2004; Jiménez-Ruiz and Grau, 2011).

Several works exploit supervised machine learning for Ontology Matching. Mao et al. (2011) cast ontology mapping as a binary classification problem. They generate various domain independent features to describe the characteristics of the entities and train an SVM classifier on a set which provides positive and negative examples of entity alignments. In general, the number of real alignments is orders of magnitude smaller than the number of possible alignments which introduces a serious class imbalance problem (Mao et al., 2008) hindering learning. Since we only use supervision to refine the word vector representations we avoid altogether the class imbalance problem.

2.2 Deep Learning for Ontology Matching

Deep learning has so far limited impact on ontology matching. To the best of our knowledge, only two approaches, (Zhang et al., 2014; Xiang et al., 2015), have explored the use of unsupervised deep learning techniques. Zhang et al. (2014) are considered to be the first ones that use word vectors in ontology matching. They train *word2vec* (Mikolov et al., 2013a) vectors on Wikipedia. They use the semantic transformations to complement the lexical information, i.e. names, labels and comments, describing entities. Their entity matching strategy is based on maximum similarity; for every entity e in the source ontology O, the algorithm finds the most similar entity e' in the target ontology O'. Their experiments on the OAEI benchmarks show that their techniques, even when combined with classical NLP techniques, could not outperform the state-of-the-art. In contrast, we refine pre-trained word embeddings with the intention of leveraging a new word vector set that is tailored to the ontology matching task.

Xiang et al. (2015) propose an entity representation learning algorithm based on Stacked Auto-Encoders (Bengio et al., 2007). To describe an entity they use a combination of its class ID, labels, comments, properties descriptions and its instances' descriptions. The entities' similarity is computed with a fixed point algorithm. They perform the entity matching using the Stable Marriage algorithm. Training such powerful models with so small training sets is problematic. We overcome this by using a transfer learning approach, known to reduce learning sample complexity (Pentina and Ben-David, 2015), to adapt pre-trained word vectors to a given ontological domain.

3 DeepAlignment

We present an ontology matching approach that uses information from ontologies and additional knowledge sources to extract synonymy/antonymy relations which we use to refine pre-trained word vectors so that they are better suited for the ontology matching task. We represent each ontological entity as the bag of words of its textual description, which we complement with the refined word embeddings. We match the entities of two different ontologies using the Stable Marriage algorithm over the entities' pairwise distances. We compute the aforementioned distances using a variant of a document similarity metric.

3.1 Preliminaries

Before we proceed with the presentation of the method, we will provide a formal definition of what an entity correspondence is. Given two ontologies O and O', we define the correspondence between two entities $e \in O$ and $e' \in O'$ as the five-element tuple:

$$cor_{e,e'} =< id, e, e', r, n > \quad (1)$$

where r is a matching relation between e and e' (e.g., equivalence, subsumption) and $n \in [0, 1]$ is the degree of confidence of the matching relation between e and e' (Euzenat and Shvaiko, 2013). The id holds the unique identifier of the mapping. Unlike the majority of ontology alignment systems which discover one-to-one equivalence mappings (Anam et al., 2015), we focus on discovering many-to-many mappings. We will also introduce some additional notation used in the paper. Let $u_1, u_2 \in \mathbb{R}^d$ be two d-dimensional vectors, we compute their cosine distance as follows: $d(u_1, u_2) = 1 - cos(u_1, u_2)$. For $x \in \mathbb{R}$, we define the *rectifier* activation function as: $\tau(x) = \max(x, 0)$.

3.2 Learning Domain Specific Word Vectors

The *counter-fitting* method (Mrkšić et al., 2016) uses synonymy and antonymy relations extracted from semantic lexicons to refine and adapt pre-trained word embeddings for given semantic similarity tasks. We broaden the concept of antonymy relations and allow for a larger class of ontology relations to define antonymies. This allows us to inject domain knowledge encoded in ontologies and produce more appropriate word vectors for the ontology matching task. In the rest of the section we revise the main elements of the counter-fitting method and describe how we can exploit it for learning domain specific word embeddings.

Let $V = \{v_1, v_2, \dots v_N\}$ be an indexed set of word vectors of size N. The counter-fitting method transforms a pretrained vector set V into a new one $V' = \{v'_1, v'_2, \dots v'_N\}$, based on a set of synonymy and antonymy constraints S and A, respectively. This is done by solving the following non-convex optimization problem:

$$\min_{V'} \kappa_1 AR(V') + \kappa_2 SA(V') + \kappa_3 VSP(V, V')$$

The $AR(V')$ function defined as:

$$AR(V') = \sum_{(u,w)\in A} \tau(1 - d(v'_u, v'_w))$$

is called *antonym repel* and pushes the refined word vectors of "antonymous" words to be away from each other. As we already mentioned, we extend the notion of antonymy relations with respect to its more narrow traditional linguistic definition. We consider that two entities in a given ontology are "antonymous" if they have not been explicitly stated as equivalent, in the sense of a logical assertion or a synonymy relation found in a semantic lexicon.

The $SA(V')$ function defined as:

$$SA(V') = \sum_{(u,w)\in S} d(v'_u, v'_w)$$

is called *synonym attract* and brings closer the transformed word vectors of synonyms. In order to extract synonymy information we search for paraphrases in semantic lexicons. Concretely, let $\omega_1 = \{word_1^1, word_2^1, \ldots, word_m^1\}$, $\omega_2 = \{word_1^2, word_2^2, \ldots, word_n^2\}$ be the textual information of two entities from different ontologies. If the combination $\{word_i^1, word_j^2\}$ or $\{word_j^2, word_i^1\}$ for some $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$ appears as a paraphrase in any semantic lexicon then we add the synonymy information (u, w) in the set S of synonymy constraints.

The $VSP(V, V')$ function defined as:

$$VSP(V, V') = \sum_{i=1}^{N} \sum_{j\in N(i)} \tau(d(v'_i, v'_j) - d(v_i, v_j))$$

forces the refined vector space to reflect the original word-vector distances. $N(i)$ is the set of words that lie within ρ distance from the i-th word vector in the original vector-space. The experiments show that the value of ρ does not affect significantly the performance of the whole algorithm, so for computational efficiency we fix it to $\rho = 0.05$. We minimize the objective function with stochastic gradient descent (SGD). We use as a convergence criterion the norm of the gradient. We continue updating the model until this is smaller than 10^{-5}. In our experiments we typically observe convergence with less than 25 iterations.

3.3 Semantic Distance Between Entities

As before, let V' be the refined word vectors and $\omega_1 = \{word_1^1, word_2^1, \ldots, word_m^1\}$, $\omega_2 = \{word_1^2, word_2^2, \ldots, word_n^2\}$ be the textual information that describes two entities from different ontologies. The textual information of an entity can be extracted from different sources, such as the entity's name, label, comments, etc. We replace the appearance of a word with its refined word vector. Hence, we end up with two sets of word vectors Q and S, respectively. In order to do the matching of the entities of two ontologies we use a semantic distance over the entities' representations, here the set of word vectors associated with each entity.

There have been many ways to compute the semantic similarity of two word sets, such as the *Word Moving Distance* (Kusner et al., 2015) and the *Dual Embedding Space Model* (DESM) (Nalisnick et al., 2016). We will base our semantic distance δ on a slight variation of the DESM similarity metric. Our metric δ computes the distance of two sets of word vectors Q and S as follows:

$$\delta(Q, S) = \frac{1}{|Q|} \sum_{q_i \in Q} d(q_i, \bar{S}) \qquad (2)$$

where $\bar{S} = \frac{1}{|S|} \sum_{s_j \in S} \frac{s_j}{\|s_j\|}$ is the normalised average of the word embeddings that constitute the set of words S.

Hence, one of the word vectors' sets is represented by the centroid of its normalized vectors. The overall set-to-set distance δ is the normalized average of the cosine distance d between the computed centroid and the other's set word vectors. A first observation is that the introduced distance is not symmetric. Ideally, we would expect the semantic distance of two word sets to be irrelevant of the order of the inputs. To make it symmetric, we redefine the distance between two sets of word vectors as:

$$dis(\omega_1, \omega_2) = \max(\delta(Q, S), \delta(S, Q)) \qquad (3)$$

It is important to note that $dis(\omega_1, \omega_2)$ is not a proper distance metric as it does not satisfy the triangle inequality property. Despite this fact, it has proved to work extremely well on all the ontology matching scenarios.

3.4 Ontology Matching

Similar to the work in (Xiang et al., 2015) we use the extension of the Stable Marriage Assignment

problem to unequal sets (Gale and Shapley, 1962; McVitie and Wilson, 1970). The stable marriage algorithm computes one-to-one mappings based on a preference $m \times n$ matrix, where m and n is the number of entities in ontologies O and O', respectively. Note that the violation of the triangle inequality by our semantic distance (equation 3) is not an impediment to the Stable Marriage algorithm (Gale and Shapley, 1962).

The majority of the ontology matching systems produce equivalence mappings with cardinality one-to-one. Hence, one entity e in ontology O can be mapped to at most one entity in e' in O' and vice versa. According to a recent review (Anam et al., 2015) only two out of almost twenty ontology matching systems provide solutions to detect many-to-many mappings. However, ontology designers focus on different degrees of granularity, so it is expected that one entity from some ontology can correspond to more than one entities in another ontology and vice-verca.

To address this problem, we present an algorithm that extends the one-to-one mappings of the previous step to many-to-many. The basic idea is that some alignments that were omitted by the Stable Marriage solution were very close to the optimal alignment and they should also be included in the final alignment set. However, despite the use of refined word vectors, we cannot completely avoid the problems that come from the semantic similarity and conceptual association coalescence. The solution of this problem comes from the observation that we can add the constraint that the mapping should be extended only in the case that the new entity that will be added will share a subsumption relation with the existing one. Below we give a more formal definition of what we will call an ϵ-optimal mapping between two entities e and e' that belong to two different ontologies O and O' respectively.

Definition 1 *Let $e \rightarrow e'$ be the optimal mapping - produced by the Stable Marriage Solution - from the entity $e \in O$ to the entity $e' \in O'$, where O and O' are two different ontologies. Let $e \rightarrow e''$ be another mapping, where $e'' \in O'$. Given an $\epsilon > 0$, we call the mapping $e \rightarrow e''$ ϵ-optimal with respect to the mapping $e \rightarrow e'$ if and only if the following two hold:*

- $|dis(\omega_1, \omega_2) - dis(\omega_1, \omega_3)| < \epsilon$, *where ω_1, ω_2, ω_3 is the textual information of entities e, e' and e'', respectively.*

Algorithm 1 extendMap(e, h, $\mathcal{O}'$, P_e, $i_{e'}$, n, ϵ, r)

Require: source entity: e
hash function from integers to entities: h
subsumption's transitive closure: $\mathcal{O}'$
sorted (increasingly) preference matrix: P_e
index of optimal solution: $i_{e'}$
number of target's ontology entities: n
$\epsilon-$optimality value: ϵ
number of relatives: r

Ensure: sequence of the $\epsilon-$optimal mappings

```
Initialization: list = ∅
opt = P_e[i_{e'}]
e' = h(i_{e'})
for i = min(i_{e'} + 1, n) to min(i_{e'} + r, n) do
    tmp = P_e[i]
    if abs(opt − tmp) < ε then
        e_i = h(i)
        if (e_i, e') ∈ O' or (e', e_i) ∈ O' then
            list.append(e → e_i)
        end if
    end if
end for
```

- *e' and e'' should be logically related with a subsumption relation. Equivalently, there must be either a logical assertion that e' is subclass of e'' or e'' is subclass of e'.*

The *subsumption restriction* requires that the extended alignments share a taxonomic relation, in order to avoid matchings between entities that are conceptually associated. We iteratively search for ϵ-optimal mappings according to the algorithm 1 to extend the established one-to-one mappings to many-to-many. For efficiency reasons, we do not check all the entities, but only the r closest entities according to the $dis(\omega_1, \omega_2)$ distance. As a final step, we iteratively pass through all the produced alignments and we discard those with $dis(\omega_1, \omega_2)$ greater than a hyperparameter value $thres$.

4 Experiments

In this section, we present the experiments we performed on the OAEI conference dataset and in one real word alignment scenario between the Schema.org and DBpedia ontologies. One of the main problems that we have encountered with the comparative evaluation of our algorithm is that even though numerous ontology matching algorithms exist, for only a very small portion of them either the respective software or the system's out-

put is publicly available. To the best of our knowledge, among all the systems tested in the conference dataset only AML (Cruz et al., 2009) and LogMap (Jiménez-Ruiz and Grau, 2011) are publicly available. As it happens these are two of the state-of-the-art systems. Moreover, AML offers solutions to detect many-to-many alignments (Faria et al., 2015) and, thus, constitutes a competitive baseline against which we will compare the performance of extendMap which also provides many-to-many alignments.

When training to refine the vector representations an unbalanced proportion of synonymy and antonymy constraints sets can cause problems; the set with the lower cardinality will have limited impact on the final word representations. To overcome this problem, we run an additional step of the counter-fitting procedure, using only a small random subset of the supernumerary constraints and all constraints of the minority set. We randomly undersample the larger set and reduce its cardinality to that of the smaller set. We call this additional step the *recounter-fitting* process. To demonstrate the importance of the recounter-fitting process and test the behavior of the pre-trained word vectors in the absence of synonymy and/or antonymy relations, we have conducted additional experiments which we also present.

In all of our experiments we have applied the counter-fitting process upon the Paragram-SL999 word vectors provided by Wieting et al. (2015). With respect to the textual information extracted for each entity, we have only used the entity's ID (rdf:ID). To estimate the precision, recall and $F1$ measure of all the systems, that we consider for testing, and check for the statistical significance of the results we use an approximate randomization test with 1048576 shuffles, as described in Yeh (2000).

4.1 Semantic Lexicons

Let $\omega_1 = \{word_1^1, word_2^1, \ldots, word_m^1\}$, $\omega_2 = \{word_1^2, word_2^2, \ldots, word_n^2\}$ be the textual information that accompanies two entities from different ontologies. We extracted the synonymy and antonymy constraints that we used in the experiments from the following semantic lexicons:

WordNet: a well known lexical database for the English language (Miller, 1995). In our experiments we did not use WordNet synonyms. Instead, we have included WordNet antonymy pairs together with the "antonymy" relations extracted by the ontologies. The strategy that we have followed in order to create the WordNet's antonymy pairs is that every two words with antonymous word senses, we have considered them as antonyms.

PPDB 2.0: the latest release of the Paraphrase Database (Pavlick et al., 2015). We have used this database in two different ways. We have used the largest available single-token terms (XXXL version) in the database and we have extracted the *Equivalence* relations as synonyms, and the *Exclusion* relations as antonyms. Additionally, we have searched the whole XXXL version of PPDB for paraphrases based on the words appeared in two entities from different ontologies. Namely, our strategy was the following: If the pair $(word_i^1, word_j^2)$ or the pair $(word_j^2, word_i^1)$ appeared on the PPDB and their type of relation was not *Exclusion*, we considered it as synonym.

WikiSynonyms: a semantic lexicon which is built by exploiting the Wikipedia redirects to discover terms that are mostly synonymous (Dakka and Ipeirotis, 2008). In our experiments we have used it only on the Schema.org[1] - DBpedia[2] scenario. Our strategy was the following: we search if there exist synonyms in the WikiSynonyms for the ω_1 and ω_2. If this is the case, we extract them and we stop there. In the opposite case we extract the synonyms for each $word_i^1$ and $word_j^2$.

4.2 Hyperparameter Tuning

We tuned the hyperparameters on a set of 100 alignments which we generated by randomly sampling the synonyms and antonyms extracted from WordNet and PPDB. We chose the vocabulary of the 100 alignments so that it is disjoint to the vocabulary that we used in the alignment experiments, described in the evaluation benchmarks, in order to avoid any information leakage from training to testing. We tuned to maximize the $F1$ measure. In particular, we did a coarse grid search over a parameter space for $\kappa_1, \kappa_2, \kappa_3$, r, ϵ and $thres$. We considered $\kappa_1, \kappa_2 \in [0.35, 0.45]$ and $\kappa_3 \in [0.1, 0.2]$ with common step 0.01, $r \in [1, 10]$ with step 1, $\epsilon \in [0.01, 0.1]$ with step 0.01 and $thres \in [0.3, 0.7]$ with step 0.05. We trained for 25 epochs for each hyperparameter using SGD.

[1] https://github.com/schemaorg/schemaorg/blob/sdo-callisto/data/releases/3.2/schema.ttl

[2] http://downloads.dbpedia.org/2014/dbpedia_2014.owl.bz2

The best values were the following: $\kappa_1 = 0.4$, $\kappa_2 = 0.4$, $\kappa_3 = 0.1$, $r = 8$, $\epsilon = 0.07$ and $thres = 0.5$. We used the selected configuration on all the alignment scenarios described below.

4.3 Evaluation Benchmarks

One of our evaluation benchmarks comes from the Ontology Alignment Evaluation Initiative (OAEI), which organizes annual campaigns for evaluating ontology matching systems. The external to OAEI evaluation benchmark comes from the provided alignments between the Schema.org and the DBpedia ontologies. We provide some further details for each dataset below:

OAEI Conference Dataset: It contains 7 ontologies addressing the same domain, namely the conference organization. These ontologies are suitable for ontology matching task because of their heterogeneous character of origin. The overall performance (micro-precision, micro-recall, micro-F1) of the systems is tested upon 21 different test cases. Specifically, we summed up the individual true positives, false positives and false negatives based on the system results for the different ontology matching tasks and, in the next step, we computed the performance metrics. The original reference alignment is not closed under the alignment relation, so the transitive closure should be computed before proceeding on the evaluation of the systems.

Schema.org - DBpedia Alignment: It corresponds to the incomplete mapping of the Schema.org and DBpedia ontologies. Schema.org is a collaborative, community activity with a mission to create, maintain, and promote schemas for structured data on the Internet, on web pages, in email messages, and beyond. On the other hand, DBpedia is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. This alignment corresponds to a real case scenario between two of the most widely used ontologies in the web today.

4.4 Experimental Results

All the systems presented in the Conference dataset experiments (Table 1) fall into the category of feature engineering. CroMatch (Gulić et al., 2016), AML (Cruz et al., 2009), XMap (Djeddi and Khadir, 2010) perform ontology matching based on heuristic methods that rely on aggregation functions. LogMap and LogMapBio (Jiménez-Ruiz and Grau, 2011) use logic-based reasoning over the extracted features and cast the ontology matching to a satisfiability problem.

4.4.1 OAEI Conference Dataset

Table 1 shows the performance of our algorithm compared to the five top performing systems on the Conference 2016 benchmark, according to the results published in OAEI[3]. DeepAligment achieves the highest micro-$F1$ measure and the highest recall. We were able to perform statistical significance test only for the two systems that were publicly available. DeepAlignment is significantly better than both of them with a $p\text{-}value \leq 0.05$. In order to explore the performance effect of the many-to-many mappings that DeepAlignment produces we also did experiments where our extendMap algorithm was not used, thus generating only one-to-one alignments. We give these results under the DeepAlignment$_*$ listing. It can be seen that DeepAlignment$_*$ achieves the same level of recall as the state-of-the-art systems and this with no feature engineering. When we compare the performance of DeepAlignment$_*$ and DeepAlignment we see that the use of extendMap generates correct many-to-many alignments and thus it does not produce large numbers of false positives. In any case, however, we retain a small precision which indicates a semantic similarity and conceptual association coalescence.

System	Precision	Recall	Micro-F1
DeepAlignment	0.71	**0.80**	**0.75**
CroMatch	0.76	0.69	0.72
AML	0.79	0.65	0.71
DeepAlignment$_*$	0.68	0.68	0.68
XMap	0.81	0.58	0.67
LogMap	0.79	0.58	0.66
LogMapBio	0.75	0.58	0.65
StringEquiv	0.83	0.50	0.62

Table 1: Results on Conference OAEI dataset. StringEquiv corresponds to ontology matching by simple string equivalence check.

We perform additional experiments to investigate the importance of the counter-fitting step, which are summarized in Table 2. In all of these experiments, we have applied the extendMap algorithm. The last row of Table 2, corresponds to the best result reported in Table 1. The first row

[3] http://oaei.ontologymatching.org/2016/

gives the results of executing the algorithm without the counter-fitting process, just by providing the Paragram-SL999 word vectors.

Parameters		Precision	Recall	Micro-F1
Synonyms	**Antonyms**			
No	No	0.63	0.55	0.59
No	Yes	0.67	0.51	0.58
Yes	No	0.69	0.72	0.71
Yes	Restricted	0.65	0.78	0.71
Yes	Yes	**0.71**	**0.80**	**0.75**

Table 2: Experiments on Conference OAEI dataset.

The results support the importance of the counter-fitting process, which succeeds in tailoring the word embeddings to the ontology matching task. By injecting only antonymy information (second row), we observe an increase in precision, but a decrease in recall. This behavior is due to the fact that the antonym repel factor imposes an orthogonality constraint to the word vectors, leading to higher values of the dis distance. In absence of synonymy information, the majority of words tend to become "antonymous". The third row of Table 2 gives the performance when we also include synonyms extracted from PPDB but no antonymy information. We can see that this leads to a large increase of all the recorded performance metrics. Finally, we also include antonymy information only from the Cmt and the Conference ontologies found in the Conference dataset. This has two effects: an increase in recall, but a decrease in precision. This can be explained by the fact that even though all ontologies describe the same domain the description granularity provided by each of them is not capable of giving all the antonymy relations needed to provide more refined alignments.

4.4.2 Schema.org - DBpedia Alignment

Table 3 summarizes the obtained results from the matching of the Schema.org and DBpedia ontologies. The fact that the alignment is incomplete restricts us on testing the performance only on the recall. To make the comparison as fair as possible, we did not apply the extendMap algorithm. We should highlight that we have applied the recounter-fitting process because the synonyms that we have extracted from the PPDB and WikiSynonyms were very few compared to the constructed "antonyms". The results of the LogMap system show a quite similar behavior with the experiments conducted in the conference dataset. However the recall of AML is zero. It

System	Recall
DeepAlignment$_*$	0.82
LogMap	0.5
AML	0

Table 3: Results on aligning Schema.org and DBpedia ontologies.

discovers none of the available alignments even though it manages to recall other quite reasonable matchings, which, however, are not included in the ground truth. According to our understanding, this might be an indication of the absence of domain transferability of the extracted features as well as of the implemented metrics. We summarize in Table 4 the results of the experiments we did on the two domains to study the effect of counter-fitting and recounter-fitting. As we can see, even without the counter-fitting, the semantic embeddings show quite good results. This provides evidence on the importance of using representation learning techniques instead of the classical feature engineering choice. By injecting only antonymy information (second row), we observe a different behavior in the recall metric compared to the one presented in Table 2. This can be explained by the fact that while the antonym repel factor imposes an orthogonality constraint, its effect is by no means universal to the whole word vector space. Therefore, a misalignment can be pushed far away leaving the space open for a true alignment to be detected. With the addition of the extracted synonyms, we observe an increase of 0.13 in the recall. However, the insertion of the extracted "antonyms" leads to lower performance. This shows practically the importance of applying the recounter-fitting process that allows both the synonym attract and the antonym repel factors to affect the word vectors.

Parameters			Recall
Recounter-fitting	**Synonyms**	**Antonyms**	
No	No	No	0.71
No	No	Yes	0.76
No	Yes	No	0.84
No	Yes	Yes	0.76
Yes	Yes	Restricted	**0.82**

Table 4: Experiments on aligning Schema.org and DBpedia ontologies. Restricted indicates that we choose only a small random subset of the antonymy constraints.

4.5 Further Analysis

DeepAlignment vs. initial word vectors. To investigate the impact of the initial pre-trained word vectors on DeepAlignment's performance, we carried out two additional experiments, this time using a set of word2vec vectors (Mikolov et al., 2013b), trained on the Google news dataset[4]. We report and compare the obtained results to the ones produced by the use of Paragram-SL999 vectors in Table 5. In the absence of counter-fitting,

Counter fitting	Word Vectors	Conference			Schema.org DBpedia
		P	R	Micro-F1	R
No	word2vec	0.64	0.52	0.58	0.74
No	Paragram	0.63	0.55	0.59	0.71
Yes	word2vec	0.67	0.75	0.71	0.75
Yes	Paragram	0.71	0.80	0.75	0.76

Table 5: Dependency of DeepAlignment's performance on the choice of the initial word vectors[6].

the word2vec vectors achieve better results on the Schema.org - DBpedia scenario, however, they exhibit lower performance on the conference dataset. This observation is in accordance with recent studies (Hill et al., 2016a) which show that different word vectors optimization objectives yield representations tailored to different applications and domains. After the application of the counter-fitting process, the use of Paragram-SL999 vectors leads to a better performance. This fact provides additional evidence that word vectors which reflect semantic similarity are better candidates for being further tailored to the ontology matching task.

DeepAlignment vs. resources' coverage. The choice and coverage of the different lexical resources may have a determining factor on the performance of DeepAlignment. For that reason, we present in Table 6 a set of experiments where we exclude a part of the synonymy/antonymy relations from the various semantic lexicons. For both the matching scenarios, we experimented with excluding all the antonyms from PPDB and WikiSynonyms. For the conference dataset, we additionaly experimented with including only a subset of PPDB synonyms (50% coverage). Finally, we carried out one experiment where we excluded all the synonymy information extracted from WikiSynonyms for the Schema.org - DBpedia scenario. The resulted performance is presented in the rows 1, 4, 2, 5 of Table 6, respectively. The reported results provide evidence that the greater the coverage of synonyms and antonyms, the greater the performance of DeepAlignment will be.

Dataset	Experiment Setting	P	R	Micro-F1
Conference	With no antonyms from PPDB & WikiSynonyms	0.67	0.76	0.71
	With only a subset of the PPDB synonyms	0.67	0.76	0.71
	With all the available synonyms/antonyms	0.71	0.80	0.75
Schema.org DBpedia	With no antonyms from PPDB & WikiSynonyms	-	0.76	-
	With no synonyms from WikiSynonyms	-	0.73	-
	With all the available synonyms & antonyms	-	0.76	-

Table 6: Dependency of DeepAlignment's performance on the external resources' coverage[6].

5 Conclusion

In this paper, we propose the refinement of pre-trained word vectors with the purpose of deriving ontological entity descriptions which are tailored to the ontology matching task. The refined word representations are learned so that they incorporate domain knowledge encoded in ontologies as well as knowledge extracted from semantic lexicons. The refinement procedure does not use any explicit information relevant to the ontology matching task making the entity representation task completely unsupervised. We perform ontology matching by applying the Stable Marriage algorithm over the entities' pairwise distances. Our experimental results demonstrate significant performance gains over the state-of-the-art and show a novel way to study the problem of ontology matching under the setting of NLP.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments on the paper. This project was supported by the Swiss State Secretariat for Education, Research and Innovation SERI (SERI; contract number 15.0303) through the European Union's Horizon 2020 research and innovation programme (grant agreement No 688203; bIoTope). This paper reflects the authors' view only, and the EU as well as the Swiss Government is not responsible for any use that may be made of the information it contains.

[4] https://code.google.com/p/word2vec

[6] For the Schema.org - DBpedia scenario's experiments, the recounter-fitting process has not been applied.

References

Sarawat Anam, Yang Sok Kim, Byeong Ho Kang, and Qing Liu. 2015. Review of ontology matching approaches and challenges. *International journal of Computer Science and Network Solutions* 3(3):1–27.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. 2007. Greedy layer-wise training of deep networks. *Advances in neural information processing systems* 19:153.

Michelle Cheatham and Pascal Hitzler. 2013. String similarity metrics for ontology alignment. In *International Semantic Web Conference*. Springer, pages 294–309.

Isabel F Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. 2009. Agreementmaker: efficient matching for large real-world schemas and ontologies. *Proceedings of the VLDB Endowment* 2(2):1586–1589.

Wisam Dakka and Panagiotis G Ipeirotis. 2008. Automatic extraction of useful facet hierarchies from text databases. In *2008 IEEE 24th International Conference on Data Engineering*. IEEE, pages 466–475.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41(6):391.

Warith Eddine Djeddi and Mohammed Tarek Khadir. 2010. Xmap: a novel structural approach for alignment of owl-full ontologies. In *Machine and Web Intelligence (ICMWI), 2010 International Conference on*. IEEE, pages 368–373.

Jérôme Euzenat and Pavel Shvaiko. 2013. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2nd edition.

Muhammad Fahad, Nejib Moalla, and Abdelaziz Bouras. 2012. Detection and resolution of semantic inconsistency and redundancy in an automatic ontology merging system. *Journal of Intelligent Information Systems* 39(2):535–557.

Daniel Faria, Catarina Martins, Amruta Nanavaty, Daniela Oliveira, Booma Sowkarthiga, Aynaz Taheri, Catia Pesquita, Francisco M Couto, and Isabel F Cruz. 2015. Aml results for oaei 2015. In *OM*. pages 116–123.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 1606–1615. http://www.aclweb.org/anthology/N15-1184.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. In *Proc. of the 1st Workshop on Evaluating Vector Space Representations for NLP*. http://aclweb.org/anthology/W/W16/W16-2506.pdf.

David Gale and Lloyd S Shapley. 1962. College admissions and the stability of marriage. *The American Mathematical Monthly* 69(1):9–15.

Fausto Giunchiglia, Pavel Shvaiko, and Mikalai Yatskevich. 2004. S-match: an algorithm and an implementation of semantic matching. In *European semantic web symposium*. Springer, pages 61–75.

Robert L Goldstone and Brian J Rogosky. 2002. Using relations within conceptual systems to translate across conceptual systems. *Cognition* 84(3):295–320.

Marko Gulić, Boris Vrdoljak, and Marko Banek. 2016. Cromatcher: An ontology matching system based on automated weighted aggregation and iterative final alignment. *Web Semantics: Science, Services and Agents on the World Wide Web* 41:50–71.

Sebastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. 2015. Semantic similarity from natural language and ontology analysis. *Synthesis Lectures on Human Language Technologies* 8(1):1–254.

Zellig S Harris. 1954. Distributional structure. *Word* 10(2-3):146–162.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016a. Learning distributed representations of sentences from unlabelled data. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. pages 1367–1377. http://aclweb.org/anthology/N/N16/N16-1162.pdf.

Felix Hill, Roi Reichart, and Anna Korhonen. 2016b. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics* .

Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. 2011. Logmap: Logic-based and scalable ontology matching. In *International Semantic Web Conference*. Springer, pages 273–288.

MT Khadir, A Djeddai, and W Djeddi. 2011. Xmap++: A novel semantic approach for alignment of owl-full ontologies based on semantic relationship using wordnet. In *Innovation in Information & Communication Technology (ISIICT), 2011 Fourth International Symposium on*. IEEE, pages 13–18.

Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Kilian Q Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*. pages 957–966.

Christoph Lofi. 2016. Measuring semantic similarity and relatedness with distributional and knowledge-based approaches. *Database Society of Japan (DBSJ) Journal* 14(1):1–9.

M. Mao, Y. Peng, and M. Spring. 2008. Ontology mapping: As a binary classification problem. In *2008 Fourth International Conference on Semantics, Knowledge and Grid*. pages 20–25. https://doi.org/10.1109/SKG.2008.101.

Ming Mao, Yefei Peng, and Michael Spring. 2011. Ontology mapping: as a binary classification problem. *Concurrency and Computation: Practice and Experience* 23(9):1010–1025.

DG McVitie and Leslie B Wilson. 1970. Stable marriage assignment for unequal sets. *BIT Numerical Mathematics* 10(3):295–309.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781. http://arxiv.org/abs/1301.3781.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013c. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pages 142–148. http://www.aclweb.org/anthology/N16-1018.

Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. 2016. Improving document ranking with dual word embeddings. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, pages 83–84.

Duy Hoa Ngo and Zohra Bellahsene. 2012. Yam++:(not) yet another matcher for ontology matching task. In *BDA'2012: 28e journées Bases de Données Avancées*. pages N–A.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, Beijing, China, pages 425–430. http://www.aclweb.org/anthology/P15-2070.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Anastasia Pentina and Shai Ben-David. 2015. Multi-task and lifelong learning of kernels. In *International Conference on Algorithmic Learning Theory*. Springer, pages 194–208.

Pavel Shvaiko and Jérôme Euzenat. 2008. Ten challenges for ontology matching. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, pages 1164–1182.

Pavel Shvaiko and Jérôme Euzenat. 2013. Ontology matching: state of the art and future challenges. *IEEE Transactions on knowledge and data engineering* 25(1):158–176.

Arjen Stolk, Lennart Verhagen, and Ivan Toni. 2016. Conceptual alignment: How brains achieve mutual understanding. *Trends in cognitive sciences* 20(3):180–191.

Julia M Taylor. 2015. Mapping human understanding to robotic perception. *Procedia Computer Science* 56:514–519.

Peng Wang and Baowen Xu. 2008. Lily: Ontology alignment results for oaei 2008. In *Proceedings of the 3rd International Conference on Ontology Matching-Volume 431*. CEUR-WS. org, pages 167–175.

John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics* 3:345–358.

Chuncheng Xiang, Tingsong Jiang, Baobao Chang, and Zhifang Sui. 2015. Ersom: A structural ontology

matching approach using automatically learned entity representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 2419–2429. http://aclweb.org/anthology/D15-1289.

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, pages 947–953.

Yuanzhe Zhang, Xuepeng Wang, Siwei Lai, Shizhu He, Kang Liu, Jun Zhao, and Xueqiang Lv. 2014. Ontology matching with word embeddings. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, Springer, pages 34–45.