NKISI-ORJI, I., WIRATUNGA, N., MASSIE, S., HUI, K.-Y. and HEAVEN, R. 2019. Ontology alignment based on word embedding and random forest classification. In Berlingerio, M., Bonchi, F., Gärtner, T., Hurley, N. and Ifrim, G. (eds.) Machine learning and knowledge discovery in databases: proceedings of the 2018 European conference on machine learning and principles and practice of knowledge discovery in databases (ECML PKDD 2018), 10-14 September 2018, Dublin, Ireland. Lecture notes in computer science, 11051. Cham: Springer [online], part I, pages 557-572. Available from: https://doi.org/10.1007/978-3-030-10925-7_34

# Ontology alignment based on word embedding and random forest classification.

NKISI-ORJI, I., WIRATUNGA, N., MASSIE, S., HUI, K.-Y., HEAVEN, R.

2018

# Ontology alignment based on word embedding and random forest classification

Ikechukwu Nkisi-Orji[1][0000−0001−9734−9978], Nirmalie Wiratunga[1], Stewart Massie[1], Kit-Ying Hui[1], and Rachel Heaven[2]

[1] Robert Gordon University, Aberdeen, UK
{i.o.nkisi-orji✉, n.wiratunga, s.massie, k.hui}@rgu.ac.uk
[2] British Geological Survey, Nottingham, UK
reh@bgs.ac.uk

**Abstract.** Ontology alignment is crucial for integrating heterogeneous data sources and forms an important component for realising the goals of the semantic web. Accordingly, several ontology alignment techniques have been proposed and used for discovering correspondences between the concepts (or entities) of different ontologies. However, these techniques mostly depend on string-based similarities which are unable to handle the vocabulary mismatch problem. Also, determining which similarity measures to use and how to effectively combine them in alignment systems are challenges that have persisted in this area. In this work, we introduce a random forest classifier approach for ontology alignment which relies on word embedding to discover semantic similarities between concepts. Specifically, we combine string-based and semantic similarity measures to form feature vectors that are used by the classifier model to determine when concepts match. By harnessing background knowledge and relying on minimal information from the ontologies, our approach can deal with knowledge-light ontological resources. It also eliminates the need for learning the aggregation weights of multiple similarity measures. Our experiments using Ontology Alignment Evaluation Initiative (OAEI) dataset and real-world ontologies highlight the utility of our approach and show that it can outperform state-of-the-art alignment systems.

**Keywords:** Ontology alignment · Word embedding · Machine classification · Semantic web.

## 1 Introduction

Arising from the need to integrate heterogeneous databases, ontology alignment (or ontology matching) deals with the discovery of semantic correspondences between the entities of different ontologies. A proliferation of ontologies of different levels of formalisation to drive the semantic web, create knowledge organisation and question answering systems, etc. have led to increased need for aligning ontologies. The utility of aligned ontologies are enhanced, enabling applications

requiring knowledge exchange. Interest in this area is reflected through the Ontology Alignment Evaluation Initiative (OAEI)[3] which provides a platform to assess and compare alignment systems. In addition, the Linking Open Data community project[4] who aims to align ontologies on a Web scale currently have over a thousand aligned datasets from different contributors in multiple domains including DBpedia, WordNet, GeoNames, and MeSH.

Ontology alignment is a challenging process especially when ontologies are of heterogeneous origins which leads to inherent differences between them. Even when of the same sub-domain, ontologies can vary widely in levels of formalisation and vocabularies mismatch. Establishing semantic correspondences between the entities of different ontologies have been the subject of various research works over the years. The predominant method for alignment is through a composition of multiple string similarity metrics [2]. Various textual features of entities (labels, definitions, etc.) in the source and target ontologies being aligned are compared and aggregated to determine when they correspond. There is a deficiency of methods for semantic matching which is crucial when the vocabulary of ontologies are different [17, 18]. Lexical databases such as WordNet have been leveraged for semantic matching but lacks sufficient coverage especially when dealing with domain-specific vocabulary. As a result, word embedding which is effective at capturing semantics have been proposed for semantic matching in ontology alignment [20, 21].

In this work, we introduce a novel matching system that relies on the integration of string-based similarity and semantic matching using word embedding to build a machine learning model, a random forest classifier. This is achieved in two stages where a subset of candidate alignments are first selected using basic matching techniques. Afterwards, several similarity indicators form feature vectors for a machine classifier with which it determines whether pairs of entities are semantic correspondences or not (crisp alignment). Our main contributions are the incorporation of vector-based similarity for semantic match discovery in the alignment process and the introduction of a set of novel features for alignment. By using background knowledge from word embedding, our approach is able to rely on minimal information from the ontologies, making it suitable for aligning knowledge-light ontological resources. We evaluate the alignment system on benchmark datasets from the OAEI and dataset from EuroVoc (EU's multilingual thesaurus)[5].

The remainder of this paper is organised as follows: section 2 reviews relevant works in literature; section 3 presents our ontology alignment approach; section 4 is an experimental evaluation which compares our approach to alternative approaches; and section 5 concludes this work and presents an outline for future work.

[3] `http://oaei.ontologymatching.org/`
[4] `http://linkeddata.org/`
[5] European Union, 2018, `http://eurovoc.europa.eu/`

## 2 Related work

Ontology alignment establishes semantic links between the entities of different ontologies which is a solution to the semantic heterogeneity problem [5, 18]. Importantly, it reduces the semantic gap between overlapping representations of a domain and trends show increasing interest in this area [17]. As ontologies differ widely, it is not unusual to encounter alignment approaches which work well for some problems and perform weakly on others [8]. Establishing correspondences between the entities of different ontologies generally follows pairwise comparisons (direct or indirect) to identify best matches. Matching entities can be element-level or structure-level [17]. Element-level matching uses intrinsic features of entities such as natural language labels and comments [9]. Instead of exact string matching, edit distance approaches such as Levenshtein and Jaro–Winkler distances are commonly used for fuzzy matching to account for spelling variations and word inflection. Structure-level matching considers the ontological neighbourhood of entities in order to determine similarity. Even when entities share little element-level features, correspondences can be discovered by similarity of structures such as having similar ancestors or descendants [16]. Due to their differences, an individual string similarity approach cannot be relied on for effective alignments [2]. Accordingly, most alignment systems combine multiple metrics (basic matchers) sequentially or in parallel [3,9,15]. This leads to a categorisation of research in ontology alignment as matching techniques or matching systems. Matching techniques deal with measures of similarity and strategies that determine the extent to which the concepts of different ontologies relate while matching systems use one or more matching techniques to align ontologies [17]. The choice of matching techniques and determining composition weights for multiple similarity metrics have been subject of several research works [6, 12].

Comparing strings become less effective for alignment when the vocabulary of ontologies differ. As a result, external knowledge resources such as WordNet and Wikipedia have leveraged used to estimate semantic similarities [8, 11]. Use of external background knowledge requires anchoring entities being compared to the external resources which is then used for inferencing [7]. Still, semantic matching is rarely used because effective integration of string-based similarity and semantic similarity remains a challenge [17, 18]. Recent experiments show that word embedding outperform lexical databases such as WordNet for semantic matching [21]. Word embedding implementations such as Word2vec use shallow neural networks to embed words in a dense continuous vector space based on linguistic contexts [13]. Word embedding preserves several linguistic regularities and have been shown to correlate well with human judgements of similarity. Word2vec models (continuous bag-of-words and continuous skip-gram) are popular implementations of word embedding using shallow neural networks to embed words based on linguistic contexts [13]. The use of word embedding is also promising for cross-lingual alignment by jointly embedding ontologies in a vector space [20]. Even more effective when using word embedding for ontology alignment is a hybrid similarity approach that incorporates string similarity using edit distance [21]. To the best of our knowledge, no other system has extended

use of word embedding for alignment beyond hybrid similarity of edit distance and vector similarity.

## 3 Classifier-based ontology alignment

Our approach is based on generating a machine classifier model using a hybrid of element-level string-based, semantic similarity features, and context-based structure-level features. The alignment process starts with the selection of candidate alignments using a range of similarity metrics. A feature vector is then generated for each candidate alignment which is passed to the machine classifier. The classifier determines whether the concept pair form an alignment or not. A high-level overview of the alignment process is presented in Figure 1 and the rest of this section describes the process in detail.

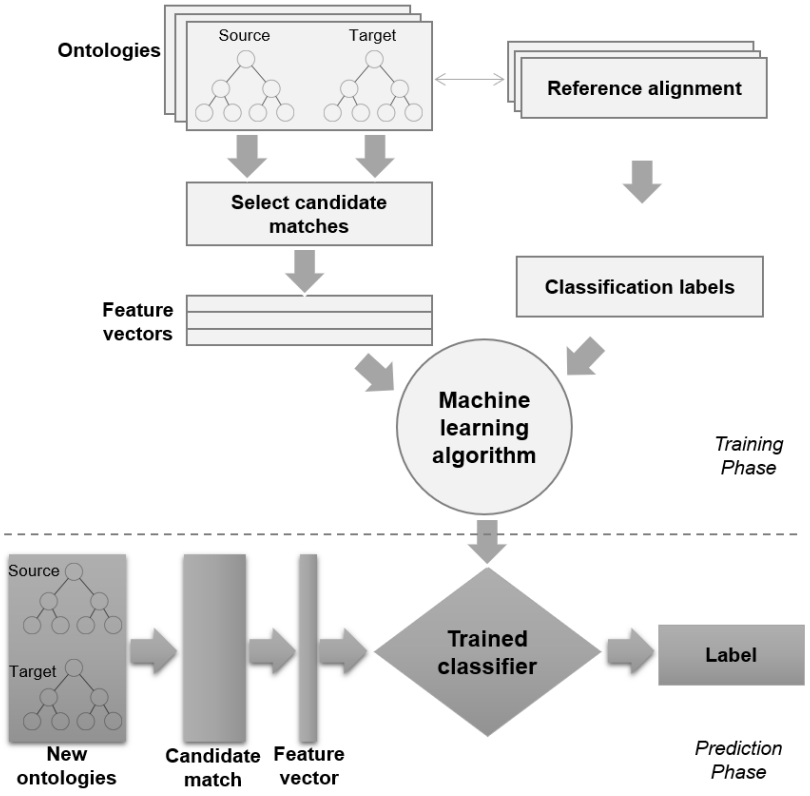


Fig. 1: Overview of ontology alignment process using machine learning.

**Notations, scope and assumptions** First, a description of the notations used in this work and assumptions we have made. An ontology, $\theta$ as specifying a set of concepts (or entities), $\theta = \{c_1, ..., c_n\}$. A concept $c \in \theta$ represents the semantic definition of a meaningful entity in a domain. Although several ontologies also specify data and object properties, we use this minimal representation to include knowledge-light ontological resources such as thesauri and controlled vocabularies. $labels(c)$ returns the set of textual labels of a concept (label, synonyms, etc.) while $labels(\theta)$ returns all labels of all concepts of $\theta$. Similarly, $tok(c)$ returns all words from a concept's label while $tok(\theta)$ returns all words in $\theta$. We assume that the ontologies being aligned specify subsumption relations (is-a or broader-than relations) between their concepts. The subsumption relation between two concepts $c_i$ and $c_j$ is represented as $c_i \prec c_j$ which means that $c_i$ is a broader concept of $c_j$.

The output of the matching process between $\theta$ and $\theta'$ is the alignment, $A$ which is a set of correspondences between semantically equivalent concepts of both ontologies. Each correspondence $a \in A$ is a 4-tuple, $a :< c, c', r, s >$ where $c$ is concept from the source ontology $\theta$, $c'$ is concept from the target ontology $\theta'$, $r$ is the alignment relation type such as *equivalent*, *subsumption*, or *disjoint*, and $s$ is the confidence of alignment correspondence in $[0.0, 1.0]$ range. For crisp alignment, confidence is either 1 (correspondence) or 0 (no correspondence). Our approach discovers equivalent relations for crisp alignment.

### 3.1 Identify alignment candidates

The alignment process begins with identifying a set of candidate correspondences between the ontologies by comparing each pair of source and target concepts using four similarity measures. The objective for selecting candidate alignments is to avoid including concept pairs that have little or no chance of being aligned in subsequent machine classification stage. A pair of concepts being compared become candidate alignments if their similarity exceeds the threshold for any of the similarity measures. Accordingly, similarity thresholds for candidate selection are kept low enough to maximise recall but not very low to select the entire similarity matrix. This avoids having to generate features for concept pairs with very low similarities and also leads to a better class balance during classification. We also use a $Max1$ selection approach for each similarity measure such that if multiple concepts in the target ontology exceed the selection threshold, we only choose the pair(s) with highest similarity value. This is commonly used to enforce a $1:1$ correspondence in alignment [18].

The similarity measures were chosen considering a range of ways in which concepts can be similar as follows.

1. Hybrid similarity (*hybrid*): combines word embedding and edit distance (Levenshtein),
2. Vector space model ($vsm$): cosine similarity of term vectors using term frequency – inverse document frequency (tf-idf) scheme,

3. Stoilos similarity (*stoilos*): string similarity metric designed for ontology alignment, and
4. Similarity of semantic context (*context*): indirectly compares concepts based on the similarity between their parent and child nodes.

**Hybrid similarity** Hybrid similarity combines use of word embeddings and edit distance measures. Hybrid similarity is expected to produce results that are at least, as good as its components [21]. After discarding words which occurred less than 10 times, we embedded a November 2016 database dump of Wikipedia English language articles in vector space of 300 dimensions using Word2vec's skip-gram model. The word embedding model was learned using an open-source deep learning library[6]. There is an abundance of literature and software tools on word embedding therefore, we will not discuss details of implementation further. We also used the Google New Corpus model[7] as an alternative word embedding for comparison. The edit distance component of our hybrid similarity is based on Levenshtein distance. Similarity between terms is based on the approach for measuring sentence similarity [10] which we describe as follows.

Let $tok(c) = \{w_1 \cdots w_n\}$ and $tok(c') = \{w'_1 \cdots w'_m\}$ be the words in labels of concepts $c$ and $c'$ being compared. Hybrid similarity is measured as shown in equation 1.

$$hybrid(c, c') = \frac{1}{max(|tok(c)|, |tok(c')|)} \cdot \sum_{w \in tok(c)} \sum_{w' \in tok(c')} max(emb(w, w'), lev(w, w')) \quad (1)$$

$emb(w, w')$ is the cosine similarity between the embedding vectors of $w$ and $w'$ while $lev(w, w')$ is normalised Levenshtein similarity. First, Levenshtein distance is normalised to $[0.0, 1.0]$ interval by dividing by the length of the longer string. Similarity is then determined as $1 -$ normalised distance. Simply put, equation 1 compares each word from one term with every word in the other term and selects the maximum similarity of either word embedding or edit distance. In our implementation, the most similar pairing is used when concepts have multiple labels (synonyms). A low hybrid similarity threshold of 0.4 was chosen in our experiments to maximise recall.

**Vector space model** The second similarity measure is based on the vector space model using cosine similarity of tf-idf weights. Each ontology forms a collection, $D$ with documents generated from labels of every concept ($D = labels(\theta)$). The tf-idf weight of each word, $w$ in a document, $d$ is determined as shown in equation 2.

[6] `https://deeplearning4j.org/word2vec.html`

[7] `https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz`

$$\text{tf-idf}(w) = f_{w,d} \cdot \log \frac{|D|}{n_w} \tag{2}$$

$f_{w,d}$ is the frequency of $w$ in $d$, and $n_w$ is the number of documents which $w$ appears in. Cosine similarity between any two documents $d$ and $d'$ is then measured using tf-idf weight vectors ($\boldsymbol{d}$ and $\boldsymbol{d'}$ respectively) as in equation 3.

$$\text{cosSim}(d, d') = \frac{\boldsymbol{d} \cdot \boldsymbol{d'}}{\|\boldsymbol{d}\|\|\boldsymbol{d'}\|} \tag{3}$$

Since each concept can have multiple documents when there are alternative labels (synonyms), TF-IDF similarity is determined as the maximum similarity of the documents of any two concepts (equation 4).

$$\text{vsm}(c, c') = \max_{\{d \in c, d' \in c'\}} (cosSim(d, d')) \tag{4}$$

By weighing terms such that frequently occurring words in an ontology contribute less to similarity, we discover alignments that may otherwise be missed as observed in [16]. Similarity threshold was set at 0.7 which is low enough for good recall.

**Stoilos similarity** The third similarity approach is a string similarity metric which was specifically designed for the purpose of aligning ontologies [19]. It is based on the intuition that the similarity between two strings is determined by the extent of their common substrings and this is offset by their differences. We use an implementation of this similarity in the Alignment API [4].

**Context similarity** When the lexical forms of textual features of a pair of concepts are entirely different, comparing their ontological neighbourhoods can discover correspondences that are missed by semantic matching. Accordingly, we indirectly measure the similarity of concepts by comparing their parent and child concepts. If the parents and children of the concepts being compared are similar, the pair are included in the alignment candidates set. Let the parent concepts of $c$ be $c_p$ and child concepts be $c_c$, we implemented context similarity as in equation 5.

$$\text{context}(c, c') = \frac{1}{2} \cdot (hybrid(c_p, c'_p) + hybrid(c_c, c'_c)) \tag{5}$$

Similarity is measured using hybrid similarity and since equation 5 is an average, we set selection threshold at half of hybrid similarity threshold.

### 3.2 Features

In the next step, feature vectors are generated for candidate alignments which will be used by a machine classifier to determine whether they are actual alignments. We introduce various novel features in addition to several similarity metrics that are commonly used for basic matching. Recall that each alignment

candidate comprises of a concept from the source ontology ($c \in \theta$) and a concept from the target ontology ($c' \in \theta'$). Features are grouped into three categories (selection, direct similarity, and context features) and summarised in Table 1.

Table 1: Feature vectors for alignment

| Feature category | Feature | Description |
|---|---|---|
| Selection | matchType | Indicates similarity method used for selection as candidate alignment |
| | sim | max(hybrid, vsm, stoilos, context) |
| | simOffset | Offset to the next highest sim |
| | hybrid | Hybrid similarity |
| | vsm | Similarity based on vector space model |
| | stoilos | String similarity |
| | context | Context similarity |
| Direct similarity | lev | Similarity based on Levenshtein distance |
| | fuzzy | Based on fuzzy string score. A point is given for every match character with bonus points awarded for subsequent matches. |
| | lcs | Similarity based on Longest Common Subsequence |
| | dice | Similarity based on Sorensen-Dice coefficient |
| | mongeElkan | Monge-Elkan similarity measure |
| | prefixOverlap | Ratio of prefix overlap to length of shorter string |
| | suffixOverlap | Ratio of suffix overlap to length of shorter string |
| | emb | Cosine similarity of word embedding vectors |
| Context | parentsOverlap | Hybrid similarity of parent concepts |
| | childrenOverlap | Hybrid similarity of child concepts |
| | contextOverlap | Hybrid similarity of all context tokens |
| | contextOverlapOffset | Offset to contextOverlap of the next most similar concept in the target ontology |
| | hasParents | Indicates whether both, one, or none of the concepts have parent nodes |
| | hasChildren | Indicates whether both, one, or none of the concepts have child nodes |
| | depthDiff | Difference in relative depths of concepts in taxonomic hierarchy |

**Selection features** These are features that are determined during selection of candidates alignments reflecting the method of similarity used for selection ($matchType$), actual similarity measure ($sim$), and similarity offset to the next most similar concept ($simOffset$) in the target ontology ($c'' \in \theta'$). $sim$ is determined as $max(hybrid(c, c'), vsm(c, c'), stoilos(c, c'), context(c, c'))$. $matchType$ is a nominal feature with values from $\{\text{type1}, \text{type2}, \text{type3}, \text{type4}\}$ to represent $hybrid$, $vsm$, $stoilos$, and $context$ similarity respectively. $simOffset$ is determined as $sim(c, c') - sim(c, c'')$. This feature captures the distinctiveness of the candidate alignment. High $sim$ and $simOffset$ values are expected to be good indicators of actual alignments. Finally, we also include each similarity method as a separate feature.

**Direct similarity features** This category of features are other similarity metrics that directly compare textual labels of concepts. They include five commonly used string-based similarity measures: Levenshtein ($lev$), Fuzzy Score[8] ($fuzzy$), Longest Common Subsequence ($lcs$), Sorensen-Dice ($dice$), and Monge-Elkan ($mongeElkan$) [2, 14]. These were chosen to provide a variation of string similarities as each algorithm makes different considerations in their algorithms. In addition, we include a similarity features for maximum prefix overlap ($prefixOverlap$), suffix overlap ($suffixOverlap$), and similarity based on word embedding alone ($emb$). Prefix and suffix overlaps are based on the number of contiguous shared characters which is normalised by the length of the shorter string. Implementation of most of the string similarity measures was based on publicly available API[9].

**Context features** Features in this category are determined by the placement of concepts on the taxonomic hierarchy. These include $parentsOverlap$ and $childrenOverlap$ which are hybrid similarities of parent and child concepts respectively of candidate nodes. We also introduce $contextOverlap$ which is the $hybrid$ similarity between all context words. That is, $contextOverlap(c, c') = hybrid((c_p + c_c), (c'_p + c'_c))$. $contextOverlapOffset$ is given as $contextOverlap(c, c') - contextOverlap(c, c'')$. Furthermore, we introduce two features ($hasParents$ and $hasChildren$) for additional insight into the neighbourhood of candidate alignments. Nominal features $\{\text{both}, \text{one}, \text{none}\}$ respectively represent when both concepts being considered have parent nodes, only one concept have parent nodes, or none have parent nodes. This is also similar for child nodes. Finally, $depthDiff$ is the difference in relative taxonomic depth of concepts being compared. Using shortest paths, the the relative depth of a concept is measured as the ratio of the number of edges from the concept to root to the total number of edges on the concept's path. Concepts with similar relative depths may increase confidence for aligning them. For the purpose of

[8] `https://commons.apache.org/proper/commons-text/apidocs/org/apache/commons/text/similarity/FuzzyScore.html`

[9] `http://github.com/tdebatty/java-string-similarity`

measuring depths, we assume an imaginary top concept (root node) when an ontology does not specify one.

### 3.3 Machine learning

The final step is the classification of candidate alignments as either true or false alignments. Decision trees have been previously shown to outperform other machine learning algorithms for aligning ontologies [15]. We use a Random Forest classifier which is an ensemble method using multiple decision trees for improved classification and to avoid overfitting [1]. Each decision tree uses a subset of features and classification is based on majority voting of decision trees' predictions. Being supervised machine classification, training data is required for learning a model. In the training data, feature vectors (as in Table 1) are generated for each candidate alignments and class labels are determined by the reference alignments. Reference alignments specify actual correspondences between source and target ontologies. The presence of a candidate alignment in the reference alignment indicates a true alignment, otherwise, it is a false alignment. In the prediction phase (when there are no reference alignments), the learned model uses generated feature vectors to determine if a candidate alignment is a true alignment.

## 4 Evaluation

### 4.1 Experiment setup

We perform experiments to evaluate the performance of our approach on two alignment datasets.

**Benchmark dataset** The first dataset is from 2016 Ontology Alignment Evaluation Initiative (OAEI) Conference track [10]. This dataset consists of 7 small to medium-sized ontologies specifying concepts in the domain of conference organisation. Although of same domain, the ontologies have heterogeneous origins resulting in differences in structure and vocabulary. The gold standard is 21 reference alignments representing all alignments between ontology pairs.

**EuroVoc dataset** The second dataset consists of two large controlled vocabularies – the European Union multilingual thesaurus (EuroVoc)[11] and the GEneral Multilingual Environmental Thesaurus (GEMET)[12]. The standard is 1,126 correspondences between equivalent concepts of both ontologies[13]. EuroVoc and GEMET describe 7,234 and 5,220 concepts respectively.

[10] http://oaei.ontologymatching.org/2016/conference/
[11] http://eurovoc.europa.eu
[12] http://www.eionet.europa.eu/gemet/en/themes
[13] http://data.europa.eu/euodp/en/data/dataset/eurovoc/resource/3430afb6-51c7-44d8-b1c7-a1e045ef5696

**Alternative alignment approaches**

- *StringEquiv*: Discovers alignments by exact string matching of concept labels. An OAEI baseline which outperforms several alignment systems in its challenges.
- *edna*: Uses edit distance for approximate string matching. Another OAEI baseline which outperforms *StringEquiv*. Edit distance is based on Levenshtein distance.
- $WordEmb$ Word embedding approach using Word2Vec's continuous skip-gram model. We embed words using Wikipedia data dump (version 20161130). Words with less than 5 occurrences were excluded and embedding vectors had dimension of 300.
- *Hybrid* Combines word embedding and edit distance to discover correspondences [21].

Our approach which we refer to as $Rafcom$, has two variants – $Rafcom_W$ and $Rafcom_G$ for Wikipedia-based and Google News word embedding models respectively. Leave-one-out approach is used for the Conference dataset such that a pair of ontologies are left out in turn while a model is trained on remaining dataset. The trained model is then used to aligned held out ontologies. For the EuroVoc dataset with a pair of ontologies only, we use ten-fold cross-validation.

Alignment performance is based on standard precision, recall and F-measure which are averaged over the dataset. Precision is the proportion of returned set of correspondences that are present in the reference alignment. Recall is the proportion of correspondences in the reference alignment that are discovered by an alignment system. F-measure is the harmonic mean of precision and recall.

### 4.2 Results and discussion

The performances of alignment approaches at best F1-measures are as shown in Tables 2 and 3 for the Conference and EuroVoc datasets respectively. Best performances for each evaluation metric are in boldface. Our approach clearly outperformed the others on the Conference dataset for all evaluation metrics with $Rafcom_G$ slightly outperforming $Rafcom_W$. About 84% of true correspondences were discovered in the candidate selection stage and the classifier achieved about 96% accuracy. Performance differences were more subtle for EuroVoc dataset. $Rafcom_W$ and $Rafcom_G$ had better precision while *edna* was best in recall. Similar to the Conference dataset, about 84% of true correspondences were discovered in the candidate selection stage. However, the classifier accuracy on was about 90%. *edna* outperformed *StringEquiv* on both datasets which is consistent with results at the OAEI challenge and previous works [2]. Also, *hybrid* outperformed its components as had been expected.

Table 2: Performances on OAEI 2016 conference track (classes only)

| | Precision | Recall | F1-measure |
|---|---|---|---|
| $StringEquiv$ | 0.878 | 0.498 | 0.635 |
| $edna$ | 0.880 | 0.537 | 0.667 |
| $WordEmb$ | 0.881 | 0.544 | 0.673 |
| $Hybrid$ | 0.880 | 0.564 | 0.687 |
| $Rafcom_W$ | 0.889 | 0.680 | 0.770 |
| $Rafcom_G$ | **0.891** | **0.695** | **0.781** |

Table 3: Performances on EuroVoc dataset (EuroVoc–GEMET alignment)

| | Precision | Recall | F1-measure |
|---|---|---|---|
| $StringEquiv$ | 0.580 | 0.746 | 0.653 |
| $edna$ | 0.572 | **0.776** | 0.659 |
| $WordEmb$ | 0.581 | 0.746 | 0.653 |
| $Hybrid$ | 0.581 | 0.768 | 0.662 |
| $Rafcom_W$ | **0.714** | 0.632 | **0.671** |
| $Rafcom_G$ | **0.714** | 0.629 | 0.669 |

Figure 2 shows results of alignment systems on the Conference dataset at the OAEI challenge ordered by F1-measure. Although the systems may have compete under a different circumstance, our results are promising when compared with the best systems at the challenge.

**Similarity types in discovery of candidate alignments** The simplest correspondences to discover are exact string matches. Any of Hybrid, Stoilos, or VSM discovers such correspondences. There are observed differences between similarity approaches when concept labels do not match as shown in Table 4. Correspondence between "Academic_Event" and "Scientific_Event" was found using the Hybrid approach because "Academic" and "Scientific" were embedded in similar vector space. Correspondene between "Track-workshop_chair" and "Workshop_Chair" was discovered using Stoilos similarity. Stoilos similarity has a greater emphasis on common substrings resulting in high similarity (0.91). The similarity between this pair is 0.6 using Levenshtein. "Conference_document" and "Document" have a high similarity of 0.94 using VSM. This is because "Conference" appeared multiple times in both ontologies (conference# and ekaw#) and as a result, has a low tf-idf weight. "conference#Conference_document" vs "ekaw#Conference" results in a similarity of 0.33 using VSM highlighting the

[14] `http://oaei.ontologymatching.org/2016/conference/eval.html`

| Matcher | Threshold | Precision | F.5-measure | F1-measure | F2-measure | Recall |
|---|---|---|---|---|---|---|
| CroMatch | 0 | 0.78 | 0.77 | 0.76 | 0.75 | 0.74 |
| AML | 0 | 0.83 | 0.8 | 0.76 | 0.72 | 0.7 |
| LogMap | 0 | 0.84 | 0.79 | 0.73 | 0.67 | 0.64 |
| XMap | 0 | 0.86 | 0.8 | 0.73 | 0.67 | 0.63 |
| LogMapBio | 0 | 0.8 | 0.76 | 0.71 | 0.67 | 0.64 |
| DKPAOMLite | 0 | 0.82 | 0.76 | 0.69 | 0.63 | 0.59 |
| DKPAOM | 0 | 0.82 | 0.76 | 0.69 | 0.63 | 0.59 |
| edna | 0 | 0.88 | 0.78 | 0.67 | 0.59 | 0.54 |
| NAISC | 0.98 | 0.85 | 0.77 | 0.67 | 0.59 | 0.55 |
| FCAMap | 0 | 0.75 | 0.72 | 0.67 | 0.63 | 0.61 |
| LogMapLt | 0 | 0.84 | 0.76 | 0.66 | 0.58 | 0.54 |
| StringEquiv | 0 | 0.88 | 0.76 | 0.64 | 0.55 | 0.5 |
| Lily | 0 | 0.59 | 0.6 | 0.61 | 0.62 | 0.63 |
| LPHOM | 0.76 | 0.89 | 0.71 | 0.55 | 0.45 | 0.4 |
| Alin | 0 | 0.89 | 0.65 | 0.46 | 0.36 | 0.31 |
| LYAM | 0.97 | 0.48 | 0.36 | 0.26 | 0.21 | 0.18 |

Fig. 2: Performance of alignment systems on OAEI 2016 conference track (classes only)[14].

reduced significance of "Conference". Also interesting is the comparison between "Paper" and "Submission" which returned low similarity scores for all direct comparisons. However, comparing their semantic neighbourhoods rightly identifies the pair as candidate alignments.

Table 4: Similarity values for some correspondences discovered

| **Source concept vs Target concept** | **Similarity approaches** | | | |
|---|---|---|---|---|
| | **hybrid** | **stoilos** | **vsm** | **context** |
| conference#Paper vs confOf#Paper | **1.0** | **1.0** | **1.0** | 0.28 |
| edas#Academic_Event vs ekaw#Scientific_Event | **0.84** | 0.61 | 0.34 | 0.72 |
| conference#Track-workshop_chair vs ekaw#Workshop_Chair | 0.56 | **0.91** | 0.42 | 0.25 |
| conference#Conference_document vs ekaw#Document | 0.57 | 0.81 | **0.94** | 0.33 |
| edas#Paper vs iasted#Submission | 0.18 | 0.0 | 0.0 | **0.76** |

**Influence of feature categories** To evaluate how the features influenced performance, we perform experiments by dropping feature categories during classification of candidate alignments. As shown in Figure 3, precision and recall values were observed for each group of feature categories. We reused previous configurations and performance was based on 10-fold cross-validation on Conference dataset.

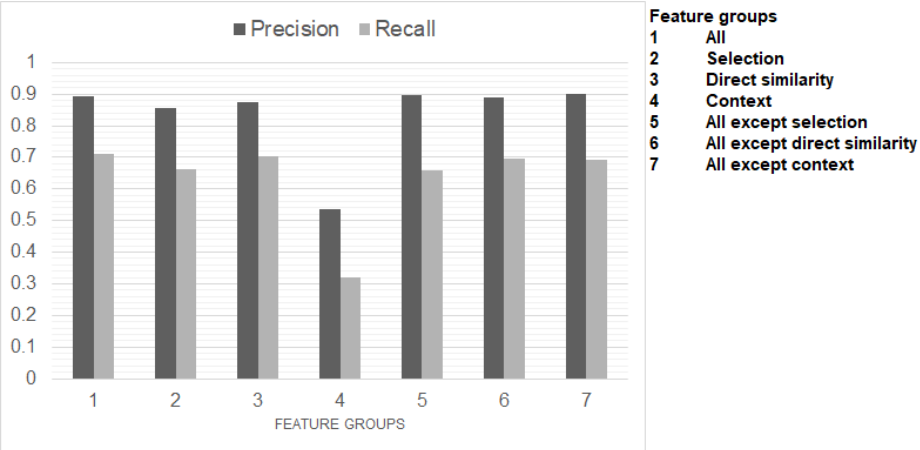


Fig. 3: Impact of excluding features categories.

Classification using all features (1) was best but only marginally better than dropping the context features (7). Context features contributed least to performance and this is further highlighted by weak performance when context features alone (4) are used for classification. We put this down to insufficient data. Analysis of candidate alignments showed that only 3% of true correspondences in the were identified using context similarity. As a result, the classifier model did not learn to effectively use context information. An interesting observation is that using direct similarity features alone (3) results in good performance. However, dropping direct features (6) is almost just as good. This suggests that several similarity features may be redundant.

## 5 Conclusion and future work

We introduced a classifier-based approach for ontology alignment based on a hybrid of string-based and semantic similarity features. Word embedding was used to generate semantic features for classification in addition to novel features which were introduced. Our experiments showed promising results and outperformed previous known approach which incorporates word embedding. Also, comparison with best-performing alignment systems at the OAEI challenge suggests that it can outperform state-of-the-art systems.

Future work will investigate a systematic determination of similarity thresholds for selecting candidate alignments. Also, the ability to transfer a trained model to a different domain will be explored. This is particularly useful in the initial stages of alignment where there are no reference alignments.

## Acknowledgement

This work was supported in part by the British Geological Survey (BGS) through the BGS University Funding Initiative (BUFI S291).

## References

1. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
2. Cheatham, M., Hitzler, P.: String similarity metrics for ontology alignment. In: International Semantic Web Conference. pp. 294–309. Springer (2013)
3. Cruz, I.F., Antonelli, F.P., Stroe, C.: AgreementMaker: Efficient matching for large real-world schemas and ontologies. Proceedings of the VLDB Endowment **2**(2), 1586–1589 (2009)
4. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The alignment api 4.0. Semantic web **2**(1), 3–10 (2011)
5. Euzenat, J., Shvaiko, P., et al.: Ontology matching, vol. 333. Springer (2007)
6. Gulić, M., Vrdoljak, B., Banek, M.: 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 (2016)
7. Husein, I.G., Akbar, S., Sitohang, B., Azizah, F.N.: Review of ontology matching with background knowledge. In: Data and Software Engineering (ICoDSE), 2016 International Conference on. pp. 1–6. IEEE (2016)
8. Jain, P., Hitzler, P., Sheth, A.P., Verma, K., Yeh, P.Z.: Ontology alignment for linked open data. In: The Semantic Web–ISWC 2010, pp. 402–417. Springer (2010)
9. Li, J., Tang, J., Li, Y., Luo, Q.: RIMOM: A dynamic multistrategy ontology alignment framework. Knowledge and Data Engineering, IEEE Transactions on **21**(8), 1218–1232 (2009)
10. Li, Y., McLean, D., Bandar, Z.A., O'shea, J.D., Crockett, K.: Sentence similarity based on semantic nets and corpus statistics. IEEE transactions on knowledge and data engineering **18**(8), 1138–1150 (2006)
11. Lin, F., Sandkuhl, K.: A survey of exploiting wordnet in ontology matching. In: IFIP International Conference on Artificial Intelligence in Theory and Practice. pp. 341–350. Springer (2008)
12. Martínez-Romero, M., Vázquez-Naya, J.M., Nóvoa, F.J., Vázquez, G., Pereira, J.: A genetic algorithms-based approach for optimizing similarity aggregation in ontology matching. In: International Work-Conference on Artificial Neural Networks. pp. 435–444. Springer (2013)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
14. Monge, A.E., Elkan, C., et al.: The field matching problem: Algorithms and applications. In: KDD. pp. 267–270 (1996)
15. Ngo, D., Bellahsene, Z.: YAM++: A multi-strategy based approach for ontology matching task. In: Knowledge Engineering and Knowledge Management, pp. 421–425. Springer (2012)

16. Ngo, D., Bellahsene, Z., Todorov, K.: Opening the black box of ontology matching. In: Extended Semantic Web Conference. pp. 16–30. Springer (2013)
17. Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. Expert Systems with Applications **42**(2), 949–971 (2015)
18. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on knowledge and data engineering **25**(1), 158–176 (2013)
19. Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: International Semantic Web Conference. pp. 624–637. Springer (2005)
20. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: International Semantic Web Conference. pp. 628–644. Springer (2017)
21. Zhang, Y., Wang, X., Lai, S., He, S., Liu, K., Zhao, J., Lv, X.: Ontology matching with word embeddings. In: Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, pp. 34–45. Springer (2014)