Results of PropMatch in OAEI 2023

Guilherme Sousa[1], Rinaldo Lima[2] and Cassia Trojahn[1]

[1]IRIT: Institut de Recherche en Informatique de Toulouse, France
[2]Universidade Rural de Pernambuco, Recife, Brazil

Abstract
This paper presents the results from the PropMatch system in the OAEI 2023 campaign. PropMatch is a system dedicate to the generation of alignments between ontology properties. It combines word and sentence embeddings with alignment extension. The system has participated in the Conference track. This is the first participation of PropMatch in the OAEI campaigns.

Keywords
Ontology Matching, Property Alignment, Machine Learning, Embeddings

1. Presentation of the system

PropMatch is a property matching system that combines TF-IDF measures with language models to measure property similarities. Additionally, alignment extension [1] and similarity reinforcement are applied to increase the number of generated correspondences.

1.1. State, purpose, general statement

Property matching is an important task in ontology matching, dealing with aligning heterogeneous knowledge resources by identifying semantically equivalent properties across different ontologies or knowledge graphs.

Nowadays, with the advancements in Natural Language Processing (NLP) and Machine Learning (ML) [2], there is increasing adoption of language models and embeddings in ontology matching. These language models, such as BERT, can be used to generate embedded representations of textual information that the names of properties are based on, and with that representation, measure the similarity between them.

Most of the works however focus on the application of these resources for class matching, and their exploitation for property matching remains under-explored. This paper presents a property-matching approach that combines TF-IDF and language models for property matching. Additionally, alignment extension and similarity reinforcement techniques are applied to increase the number of correspondences generated.

OM 2023: The 18th International Workshop on Ontology Matching collocated with the 22nd International Semantic Web Conference ISWC-2023 November 7th, 2023, Athens, Greece
*Corresponding author.
†These authors contributed equally.
guilherme.santos-sousa@irit.fr (G. Sousa); rinaldo.jose@ufrpe.br (R. Lima); cassia.trojahn@irit.fr (C. Trojahn)
0000-0002-2896-2362 (G. Sousa); 0000-0002-1388-4824 (R. Lima); 0000-0003-2840-005X (C. Trojahn)

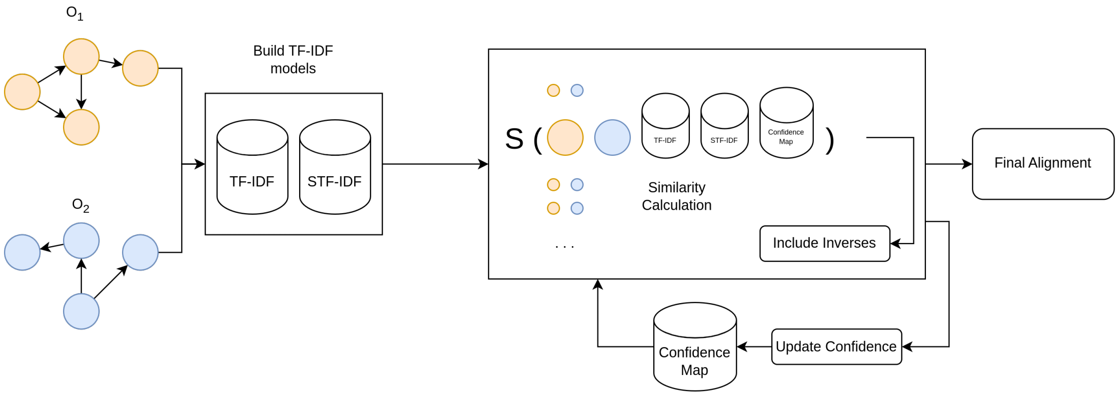

Figure 1: PropMatch architecture.

This matcher extends the work of [3] and more details can be found in [4].

1.2. Specific techniques used

PropMatch[1] is based on four main techniques, introduced below. The overview of the system architecture is presented in Figure 1.

1.2.1. TF-IDF models

The TF-IDF and Soft TF-IDF models are created before the matching process. Each entity from the ontologies is represented by a virtual document that is a piece of text containing entity information. Ontology class virtual documents use the class labels and comments, while properties virtual documents include the property labels, domain, and range. These documents are then tokenized and converted to lowercase. After that, the frequency models, vocabulary, and IDF are generated from this set of documents using Scikit-learn [2]. Cosine similarity is then used to compute the final similarity between the embeddings of the property pairs. For property labels, the system employs the Soft TF-IDF approach, using the Jaro-Winkler metric with a threshold of 0.8.

After constructing the TF-IDF models, the system calculates the similarity score for each property pair. The final score is the minimum of three confidence values based on domain, range, and property label similarities. This ensures similarity only when all three exceed the threshold. If the metric yields zero similarity, an embedding similarity is used as an alternative measure.

1.2.2. Embedding Models

When TF-IDF models generate pairs of properties with low similarity, the system instead uses word embeddings for domain similarity and sentence embeddings for property label similarity.

[1]https://gitlab.irit.fr/melodi/ontology-matching/propmatch
[2]https://scikit-learn.org/

For domain similarity, we employ pre-trained word embeddings from the Finnish Internet Parsebank [5] and this is only applied to single-word domain entities otherwise similarity remains zero. The word embedding is also used for domain similarity since it can capture more semantic relations than TF-IDF captures.

In property label similarity, we first remove the last word if it matches the first word in the range label.

Similar to the domain similarity approach, we apply a fallback strategy for property labels. This happens when domain and range similarity exceeds 0.9, but label similarity falls below 0.1. These parameters are found to perform better in the Conference track and are hyperparameters of the system that can be adjusted. We use a sentence embedding similarity model from the HuggingFace [3] repository to generate embeddings for property labels. The property label with the range labels is composed in a sentence that is fed to the model to generate an embedding used for the similarity calculation.

1.2.3. Alignment Extension

A common practice in matching is to use existing alignments to find new correspondences, following the ”locality principle” [1], which states that new correspondences are often found among previously aligned entities. Based on this principle, the inverse of properties with high similarity is also included in the final alignment set since they are more likely to be aligned given that their inverses are similar. In order to keep a simple correspondence set, if there exist multiple correspondences for the same property, only the pair with the highest similarity is retained.

1.2.4. Similarity reinforcement

Following the locality principle, since the system can use information from previous alignments to find new correspondences, we apply a process of similarity reinforcement. Assuming that the system can find a subset of the final alignment set, repeating the matching process with the previous alignment found by the system can increase its confidence in new similarity measures given the previously discovered correspondences.

In order to keep track of the previous correspondences, a key-value store is used named Confidence-Map, where keys represent pairs of classes, and values represent the similarity between them. When a property correspondence is found, the domains in each property form a pair in the Confidence Map. In the next iterations, the system checks if the domain pair of the evaluated properties is present in the Confidence Map. If found, the domain confidence in the similarity computation is increased by 0.66. Multiple iterations are needed to fully reflect the Confidence Map’s influence, and the number of iterations is a system hyperparameter.

The similarity reinforcement works as the following. Consider the triples:

- Ontology 1: (Paper, hasTitle, Title), (Author, writes, Paper)
- Ontology 2: (Contribution, hasTitle, Title), (Author, contributes, Contribution)

[3] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2 consulted at 07/09/2023.

In the first iteration, if the similarity measure between 2 properties is higher than the threshold t, their domains are added to the map. (Paper, Contribution):0.66 is added to the map because the similarity between o_1 : *hasTitle* and o_2 : *hasTitle* is higher than t. In a second iteration, the map entries are taken into account for calculating the property similarity. In the example, the similarity between o_1 : *writes* and o_2 : *contributes* will have a higher range similarity since the pair (Paper, Contribution) is present in the map.

1.3. Managing of complex constructors

In the Conference track some property domains are composed of complex constructors. in order to match properties with those domains and to compare their similarity the first step of the matching process consists of converting the complex entities into simple entities that have all labels of the complex entity concatenated. For example, complex constructors such as `owl:UnionOf` can be handled. For example, the property `hasTitle` which has a complex domain containing two entities Conference and Paper. After the processing, the labels of the Conference and Paper are concatenated to generate a single label Conference_Paper that the matcher can use to measure the similarity between domains.

2. Results

PropMatch was evaluated on Conference in the modality M2 which is based only on the alignment of properties. In these results, ALIN, AMD, LSMatch, and SORBETMatch do not produce any property alignments. While using language models, PropMatch is not fine-tuned and does not require any reference alignments as input. **ra1** is the original reference alignment. **ra2** is an entailed reference alignment created through transitive closure from the original reference alignment (ra1). And **rar2** is a violation-free version of reference alignment ra2 as described in the result page of the OAEI [4].

Table 1 presents the results of the systems that participated in the OAEI 2023 in the ra1-M2 (property matching only) modality. PropMatch achieved the best results in all metrics in this modality.

Tables 2 and 3 present the results of the systems in the modalities ra2-m2 and rar2-m2 respectively. In both ra2-m2 and rar2-m2 modalities PropMatch still stays at first winning in all metrics while improving its precision by 0.03 and recall by 0.02 in the ra2-m2 modality compared to the ra1 modality, and also improving its precision by 0.03 and recall by 0.04 in the rar2-m2 modality compared to the ra1 modality.

3. General comments

3.1. Comments on the results

PropMatch achieves the best results in the M2 modality in all metrics. This corroborates the interest in combining embeddings with classical matching metrics for property matching. It

[4]https://oaei.ontologymatching.org/2023/results/conference/eval.html

Matcher	Threshold	Precision	F.5-measure	F1-measure	F2-measure	Recall
PropMatch	0.00	**0.83**	**0.74**	**0.64**	**0.56**	**0.52**
Matcha	0.78	0.48	0.48	0.47	0.46	0.46
GraphMatcher	0.91	0.72	0.55	0.40	0.32	0.28
LogMap	0.79	0.62	0.50	0.39	0.31	0.28
OLaLa	0.76	0.39	0.35	0.30	0.26	0.24
LogMapLt	0.00	0.24	0.24	0.23	0.22	0.22
TOMATO	0.00	0.17	0.18	0.18	0.19	0.20
edna	0.00	0.21	0.18	0.14	0.12	0.11
StringEquiv	0.00	0.07	0.05	0.03	0.02	0.02

Table 1
Results of the systems in the ra1-m2 modality.

Matcher	Threshold	Precision	F.5-measure	F1-measure	F2-measure	Recall
PropMatch	0.00	**0.86**	**0.77**	**0.66**	**0.58**	**0.54**
Matcha	0.78	0.48	0.48	0.47	0.46	0.46
GraphMatcher	0.91	0.72	0.55	0.40	0.32	0.28
LogMap	0.79	0.62	0.50	0.39	0.31	0.28
OLaLa	0.76	0.39	0.35	0.30	0.26	0.24
LogMapLt	0.00	0.24	0.24	0.23	0.22	0.22
edna	0.00	0.21	0.18	0.14	0.12	0.11
TOMATO	0.00	0.15	0.15	0.16	0.17	0.17
StringEquiv	0.00	0.07	0.05	0.03	0.02	0.02

Table 2
Results of the systems in the ra2-m2 modality.

Matcher	Threshold	Precision	F.5-measure	F1-measure	F2-measure	Recall
PropMatch	0.00	**0.86**	**0.78**	**0.68**	**0.60**	**0.56**
Matcha	0.93	0.67	0.57	0.47	0.40	0.36
GraphMatcher	0.91	0.72	0.56	0.41	0.33	0.29
LogMap	0.79	0.62	0.51	0.40	0.32	0.29
OLaLa	0.76	0.39	0.35	0.30	0.26	0.24
LogMapLt	0.00	0.24	0.24	0.23	0.22	0.22
TOMATO	0.00	0.15	0.16	0.16	0.17	0.18
edna	0.00	0.21	0.18	0.14	0.12	0.11
StringEquiv	0.00	0.07	0.05	0.03	0.02	0.02

Table 3
Results of the systems in the rar2-m2 modality.

gives the system an increased recall while preserving higher precision.

3.2. Improvements

The embeddings in the system are still not the main source of similarity computation. In order to improve the system performance, moving to a full embedding approach could improve the capacity of finding more correspondences. Along with that, better models that aggregate more context information into embeddings can also improve the system's performance. Another improvement is extending the approach to class matching since the confidence map built for the property alignment could contribute to finding class correspondences. Lastly, the discovery of complex correspondences between properties could be addressed.

Furthermore, since the representation of properties can change between tracks, the system still has difficulties to be evaluated in all tracks. Improving the system's capacity to find the properties in the ontology structure and also dealing with complex domain and ranges (composed of multiple entities) can help the system's generalization and ability to run in other tracks.

4. Conclusions

This paper presented the PropMatch system and its results in the OAEI 2023 campaign. This year, we have participated only in the Conference track. Next year, we plan to participate in more tracks.

References

[1] E. Jiménez-Ruiz, B. C. Grau, LogMap: Logic-Based and Scalable Ontology Matching, in: Proceedings of the 10th International Semantic Web Conference, 2011, pp. 273–288.

[2] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics, 2019, pp. 4171–4186.

[3] M. Cheatham, P. Hitzler, The properties of property alignment, in: Proceedings of the 9th International Workshop on Ontology Matching, October 20, 2014, 2014, pp. 13–24.

[4] G. Sousa, R. Lima, C. Trojahn, Combining word and sentence embeddings with alignment extension for property matching, in: P. Shvaiko, J. Euzenat, E. Jiménez-Ruiz, O. Hassanzadeh, C. Trojahn (Eds.), Proceedings of the 18th International Workshop on Ontology Matching (OM 2023) co-located with the 22th International Semantic Web Conference (ISWC 2023), Athens, Greece, November 7th, 2023, CEUR Workshop Proceedings, CEUR-WS.org, 2023. URL: https://disi.unitn.it/~pavel/om2023/papers/om2023_STpaper6.pdf.

[5] J. Luotolahti, J. Kanerva, V. Laippala, S. Pyysalo, F. Ginter, Towards universal web parsebanks, in: Proceedings of the Third International Conference on Dependency Linguistics, DepLing 2015, August 24-26 2015, Uppsala University, Uppsala, Sweden, 2015, pp. 211–220.