Fine-TOM Matcher Results for OAEI 2021

Leon Knorr[1][0000−0003−4117−2629] and Jan Portisch[1,2][0000−0001−5420−0663]

[1] SAP SE, Walldorf, Germany
[2] Data and Web Science Group, University of Mannheim, Germany
jan@informatik.uni-mannheim.de

Abstract. In this paper, the *Fine-Tuned Transformes for Ontology matching* (Fine-TOM) matching system is presented along with the results it achieved during its first participation in the Ontology Alingment Evaluation Initiative (OAEI) campaign (2021). The system uses the publicly available *albert-base-v2* model, which has been fine-tuned with a training dataset that includes *20%* of each reference alignment from the *Anatomy*, *Conference*, and *Knowledge Graph* track, as well as a wide variety of generated false examples. The model is then used by a separate matching pipeline which calculates a confidence score for each correspondence. In the submitted docker container, only the matching pipeline with an already fine-tuned model is included.

Keywords: Ontology Matching · Ontology Alignment · Language Models · Transformers · Fine-Tuning.

1 Presentation of the System

1.1 State, purpose, general statement

Fine-Tuned Transformers for Ontology Matching (Fine-TOM) is a transformer-based matching system. It consists of two separate pipelines, a pipeline for generating training data and model training, and a *matching pipeline* which performs the actual matching task. Both can be executed individually or in a row. Each pipeline uses predefined components, which are included in the *Matching Evaluation Toolkit (MELT)* [6], a framework for ontology matching and evaluation. In particular, the the new transformer extension of MELT [7] is used.
For the submission, only the matching pipeline was packaged in a docker container using the *Melt Web Interface*[3], where a fine-tuned *albert-base-v2* model is included. This model was fine-tuned beforehand with a training set that included *20%* of the reference alignments of the *Anatomy*, *Conference*, and *Knowledge Graph* track, as well as generated negative examples. This year's submission marks the first introduction of the Fine-TOM system to the OAEI.

[3] https://dwslab.github.io/melt/matcher-packaging/web#web-interface-http-matching-interface

1.2 Specific Techniques Used

Transformer-based language models One possible solution to solving NLP problems is the use of transformers. The initial transformer was introduced by Google in 2017 and uses a, so called, *Self-Attention Architecture* [13], which is said to be more parallelizable and requires significantly less time to train. Today, the NLP domain mostly adapted the use of transformers and they became the de facto standard for most NLP tasks like text translation and classification [13,4]. As a result, today, there are many different transformer models available, e.g. *bert-base-cased* [4] and *gpt-2* [11]. All of them are using different variations of the initial self-attention architecture.

Fine-Tuning In order to achieve good results a transformer needs to be initially trained on a large amount of training data. This process is also called pre-training. As it requires a vast amount of data as well as processing power to pre-train a transformer model, most models are pre-trained on a specific task, like next sentence prediction and then uploaded to huggingface[4] [14] where they are available for download as well and can be tested in web demos. This initial training process has a great impact on how the selected model will perform later on. As most transformers are trained for conventional tasks like text summarization, next sentence prediction, or review classification [14,13], they are not suitable for other tasks, in this case ontology matching, right out of the box. Therefore, transformers can be retrained or fine-tuned to perform other or similar tasks. This process is usually computationally cheaper than the pre-training process. However, quality training data is needed, which has to consist of positive as well as negative examples. Because training data is currently not available, Fine-TOM includes a *training pipeline*, which generates training data based on a fraction of already known reference alignments.
During the development of Fine-TOM different BERT models were fine-tuned and evaluated on the *Anatomy* [1], *Conference* [2], and *Knowledge Graph* [8,5] track. Based on the data gathered, the best performing configuration was determined which uses the *albert-base-v2* model and is further explained in the following.

1.3 Fine-TOM architecture

The Fine-TOM matching system consists of two individual pipelines, as shown in Figure 1:

- A trainging pipeline, which handles the Fine-Tuning process of a transformer and saves it to the disk
- a matching pipeline, which will perform the actual matching task and is based on the architecture presented in the TOM paper [9].

This architecture can also be used to run transformers with a zero shot approach, by only executing the matching pipeline with a pre-trained model.

[4] https://huggingface.co

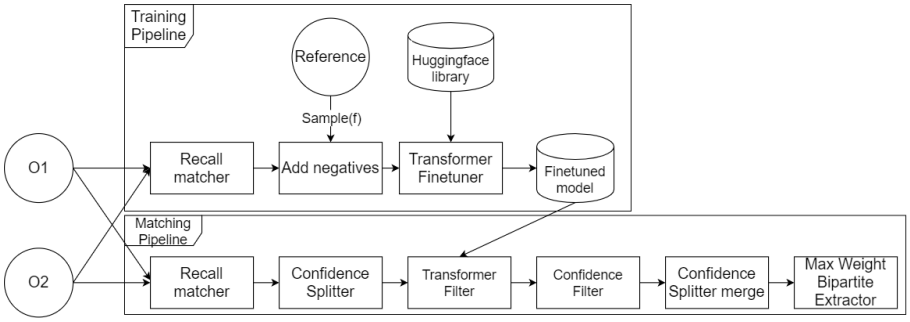

Fig. 1. High-level view of the Fine-TOM matching process.

Training Pipeline The Training Pipeline, shown in Figure 2, consists of several predefined components of the MELT [7] framework. First a *recall matcher* will create an alignment between the two ontologies O_1 and O_2, which acts as a the basis for generating training data. It usually does not feature a high precision score, but a good recall, thus, many correspondences included are not a correct match. Therefore, it marks a good starting point for generating training data. After that, a mechanism for generating negatives will create the actual training dataset, by sampling a configurable fraction f from a already known reference alignment. Internal experiments showed that 20-40% of a reference alignment has the best work to performance ratio, thus the model included in Fine-TOM has been trained with a sampling rate of 20%. These sampled correspondences mark the positive examples that a training set has to include. In order to add negatives examples to this training set, the mechanism takes the alignment generated by the Recall Matcher as an input. On the assumption that the perfect solution is of a one-to-one parity, and since for some entities the correct match is known through sampling the reference alignment, negative examples can now be picked from the alignment of the recall matcher, thus resulting in a training set that includes positive as well as negative examples. This training set is then passed on to the *transformer finetuning* component of the MELT Framework [7], which will then fine-tune the selected model and save it to the disk.

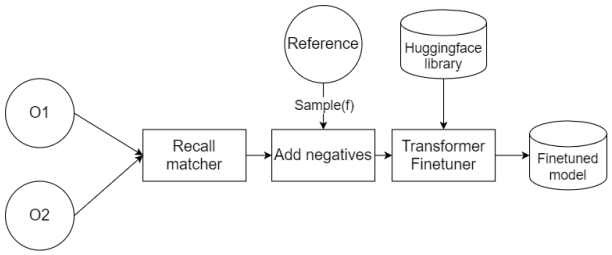

Fig. 2. High-level view of the Fine-TOM training process.

Matching Pipeline The Matching Pipeline, shown in Figure 3, also consists of several predefined components of the MELT Framework. As in the Training Pipeline, a *recall matcher* is used as a starting point, thus marking the theoretically highest recall that can be achieved with this matching system. The resulting alignment will then be processed by a "confidence splitter", which will delete all correspondences that are simple string matches and have a confidence level of 1.0, as well as their entities from the alignment returned by the *recall matcher*. These correspondences are then saved temporarily into a separate alignment, so they will not get reclassified by the transformer model. Then the cleaned up alignment is passed on to a *transformer filter*, which will load the previously fine-tuned transformer model from the disk and add another confidence score to each correspondence in the alignment. In order to make use of this newly added confidence level, and to eliminate correspondences the transformer classified as a bad match by a low confidence score, a *confidence filter* is used. It will "cut off" the alignment by a certain threshold which can be configured. Fine-TOM uses the same threshold of 0.8 as proposed by the TOM paper [9]. After all matches with a lower confidence score have been removed from the processed alignment, the previously removed correspondences with a confidence score of 1.0 are added to the alignment again. Since most OAEI datasets are typically of one-to-one arity, an efficient implementation of the Hungarian method, known as *Maximum Weight Bipartite Matching* (MWBM) [3] was used to create the final alignment and therefore the final result. All matching components are explained in more detail below.

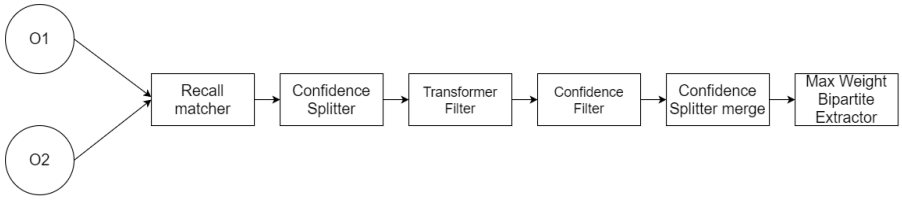

Fig. 3. High-level view of the Fine-TOM Matching process.

Recall Matcher The *recall matcher* uses a variety of string comparisons in order to generate an alignment, which marks a high recall on the expense of a rather low precision. It includes a simple string matching mechanism which compares each textual representation of an entity character by character, if a match is found, it is added to the result alignment and a confidence of 1.0 is assigned to this correspondence. Besides this mechanism it also counts how often each word of a text representation is included in the other one, if this similarity surpasses a configurable threshold, the correspondence is also added to the result alignment but only with a low confidence of 0.1.

Confidence Splitter As described earlier, the *confidence splitter* takes an alignment as input and removes every correspondence with a confidence score of 1.0, as well as every other correspondence of the entities included in the removed correspondence. This is done in order to prevent a reclassification of these rather "save" matches by another component in the pipeline. Therefore, the *confidence splitter* is also able to add the alignment, which was saved during the splitting process, to an alignment that has been passed on to it as an input.

Transformer Filter The *transformer filter* iterates over the alignment, which has been passed on to it as an input, and processes each correspondence individually by calling a separate python server which is running locally in the background. This is needed because the transformer models themselves are implemented in Python, where as the matching components and pipeline is implemented in Java. Each pair of textual representations received by the Python server is processed by the selected model, which can either be loaded from the disk or it can be sourced from the huggingface library. This transformer model will then provide a confidence level, which is send back to the *transformer filter* class and added to the actual correspondence in the alignment, therefore classifying each correspondence.

Confidence Filter The *confidence filter* will exclude every correspondence with a confidence score lower than a configurable threshold. This is needed since the *transformer filter* itself does not remove any correspondences from the alignment, it just reclassifies them. Therefore, in order to exclude matches that have been marked as a bad match by a low confidence, the *confidence filter* is needed.

Max Weight Bipartite Extractor The alignment generated by matching components can include multiple correspondences for an ontology element. However, the assumption was made earlier that the solution for the posed ontology matching problem is of a one-to-one arity. Therefore, the alignment provided as an input to the *max weight bipartite extractor* needs to be converted into an alignment with a one-to-one arity. In order to do that, an efficient implementation of the Hungarian method, known as *Maximum Weight Bipartite Matching* (MWBM) [3] was used.

2 Results

This section discusses the results of Fine-TOM during the OAEI 2021 campaign. Only the *Anatomy* [1], *Conference* [2], and *Knowledge Graph* [8,5] tracks are included, since the matching system was only designed and trained for them.

	Precision	Recall	F-Measure
StringEquiv	0.997	0.622	0.766
TOM	0.933	0.808	**0.866**
Fine-TOM	0.916	0.794	0.851

Table 1. Results on the *Anatomy* track according to the OAEI 2021 campaign

2.1 Anatomy

The results[5] of Fine-TOM on the Anatomy track are depicted in Table 1. As shown, Fine-TOM was able to outpeform the OAEI *StringEquiv* matcher in terms of recall and the f-measure, although its precision was lower. This proves that the Fine-TOM matching system is able to find matches that can not be found by checking for string equivalence. However, if compared to the TOM matching system, which is strongly related to Fine-TOM as they share a similar architecture in regards to the matching pipeline, Fine-Tom did achieve slightly lower scores (~1-1.5%) for all measures shown. That is a rather interesting result, as the transformers used in the TOM paper are not re-trained with domain specific data, nor were they pre-trained with data of an ontology matching task. Nevertheless, TOM has one advantage: it uses the Sentence-BERT transformer model *paraphrase-TinyBERT-L6-v2* [12], where as Fine-TOM uses a fine-tuned version of the *albert-base-v2* model. These Sentence-BERT models are pre-trained and designed to find semantic textual similarities between input sequences [12]. The *albert-base-v2* model on the other hand, is a variation of the *BERT* model, and was trained for masked language modelling [10], which is a completely different task compared to ontology matching. Therefore, it is remarkable that Fine-TOM was able to achieve such a similar score to TOM. This demonstrates the impact the fine-tuning process has on the performance of a matching system that includes a transformer model. Since MELT did not support Sentence-BERT transformers at the time of Fine-TOMs development, they could not be evaluated in time for Fine-TOMs OAEI 2021 submission.

2.2 Conference

	Precision	Recall	F-Measure
StringEquiv	0.76	0.41	0.53
TOM	0.69	0.48	0.57
Fine-TOM	0.64	0.53	**0.58**

Table 2. Results on the *Conference* track according to the OAEI 2021 campaign

[5] official result page: http://oaei.ontologymatching.org/2021/results/anatomy/index.html

As shown in Table[6] 2, Fine-TOM, did achieve a higher F-Measure and recall on the *Conference* track than the OAEI *StringEquiv* matcher and TOM. It was therefore, able to find more correct correspondences than both systems.

2.3 Knowledge Graph

On *Knowledge Graph*, Fine-TOM was able to achieve slightly better results as the OAEI baseline, as shown in Table 3.

	Precision	Recall	F-Measure
BaselineLabel	0.95	0.71	0.81
Fine-TOM	0.92	0.75	**0.83**

Table 3. Results on the *Conference* track according to the OAEI 2021 campaign

3 General Comments

We thank the OAEI organizers for their support and commitment.

4 Conclusion

In this paper, the Fine-TOM matching system has been presented. First a new pipeline architecture that includes a dedicated training pipeline and a matching pipeline was introduced. This training pipeline first generates a training set based on reference alignments and a highrecall matcher, which is then used to retrain a selected model. The model is then injected in a so called matching pipeline. It then performs the actual matching process by using different filters. The results showed that transformers can improve the overall performance of matching systems in terms of recall and the f-measure. Besides that, the similar results of TOM and Fine-TOM proved that fine-tuning has a great impact on the performance of transformer models, since the model used by Fine-TOM has not been pre-trained for ontology matching or to find semantic similaritys between input sequences. Therefore, the presented approach promises a lof of potential for further increases in performance in the future, by using a different model, e.g. a Sentence-BERT model, or by improving or changing different pipeline components like the highrecall matcher. In addition to that, this year's submission marks the first participation for the Fine-TOM matching system in an OAEI campaign and the results reported are promising and motivate further research in the area of transformer-based ontology and instance matching.

[6] official result page: http://oaei.ontologymatching.org/2021/results/conference/

References

1. Bodenreider, O., Hayamizu, T.F., Ringwald, M., de Coronado, S., Zhang, S.: Of mice and men: Aligning mouse and human anatomies. In: AMIA 2005, American Medical Informatics Association Annual Symposium, Washington, DC, USA, October 22-26, 2005. AMIA (2005)
2. Cheatham, M., Hitzler, P.: Conference v2.0: An uncertain version of the OAEI conference benchmark. In: The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II. Lecture Notes in Computer Science, vol. 8797, pp. 33–48. Springer (2014)
3. Cruz, I.F., Antonelli, F.P., Stroe, C.: Efficient selection of mappings and automatic quality-driven combination of matching methods. In: Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009. CEUR Workshop Proceedings, vol. 551. CEUR-WS.org (2009)
4. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018)
5. Hertling, S., Paulheim, H.: The knowledge graph track at OAEI - gold standards, baselines, and the golden hammer bias. In: The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12123, pp. 343–359. Springer (2020)
6. Hertling, S., Portisch, J., Paulheim, H.: MELT - matching evaluation toolkit. In: Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9-12, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11702, pp. 231–245. Springer (2019)
7. Hertling, S., Portisch, J., Paulheim, H.: Matching with transformers in MELT. CoRR **abs/2109.07401** (2021)
8. Hofmann, A., Perchani, S., Portisch, J., Hertling, S., Paulheim, H.: Dbkwik: Towards knowledge graph creation from thousands of wikis. In: Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017. CEUR Workshop Proceedings, vol. 1963. CEUR-WS.org (2017)
9. Kossack, D., Borg, N., Knorr, L., Portisch, J.: TOM matcher results for OAEI 2021. In: OM@ISWC 2021 (2021), to appear
10. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A lite BERT for self-supervised learning of language representations. CoRR **abs/1909.11942** (2019)
11. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2018)
12. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (11 2019)
13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. CoRR **abs/1706.03762** (2017)
14. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Brew, J.: Huggingface's transformers: State-of-the-art natural language processing. CoRR **abs/1910.03771** (2019)