5-2023

Multi-lingual multi-partite product title matching

Huan Lin TAY
Singapore Management University, huanlin.tay.2019@scis.smu.edu.sg

Wei Jie TAY
Singapore Management University, weijie.tay.2019@scis.smu.edu.sg

Hady Wirawan LAUW
Singapore Management University, hadywlauw@smu.edu.sg

Multi-Lingual Multi-Partite Product Title Matching

Huan-Lin Tay*
huanlin.tay.2019@scis.smu.edu.sg
Singapore Management University

Wei-Jie Tay*
weijie.tay.2019@scis.smu.edu.sg
Singapore Management University

Hady W. Lauw
hadywlauw@smu.edu.sg
Singapore Management University

ABSTRACT

In a globalized marketplace, one could access products or services from almost anywhere. However, resolving which product in one language corresponds to another product in a different language remains an under-explored problem. We explore this from two perspectives. First, given two products of different languages, how to assess their similarity that could signal a potential match. Second, given products from various languages, how to arrive at a multi-partite clustering that respects cardinality constraints efficiently. We describe algorithms for each perspective and integrate them into a promising solution validated on real-world datasets.

CCS CONCEPTS

• **Information systems** → *Clustering*; Multilingual and cross-lingual retrieval; **Data mining**.

KEYWORDS

Multi-Lingual Similarity, Multi-Partite Matching

ACM Reference Format:
Huan-Lin Tay, Wei-Jie Tay, and Hady W. Lauw. 2023. Multi-Lingual Multi-Partite Product Title Matching. In *Companion Proceedings of the ACM Web Conference 2023 (WWW '23 Companion), April 30-May 4, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3543873.3587322

1 INTRODUCTION

E-commerce is increasingly globalized in nature. With ever more affordable shipping rates enabled by world-wide supply chains, there is a growing trend of cross-border transactions. We investigate the problem of matching product listings (primarily based on titles) of different languages. This supports emergent applications including comprehensive product catalogues, global price comparisons, and understanding of consumer behaviour at a global scale.

We foresee two primary challenges, which correspond to the two phases of our approach as illustrated in Figure 1. One key challenge is how to determine when two product titles from different languages are similar enough that they could be referring to the same product. Another key challenge is how to efficiently arrive at the matching solutions given the combinatorial explosion that comes from the multiplicity of languages.

*Both authors contributed equally to this research.

WWW '23 Companion, April 30-May 4, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9419-2/23/04.
https://doi.org/10.1145/3543873.3587322

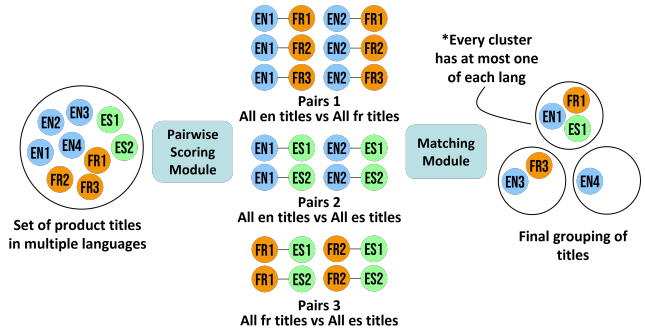

Figure 1: The 2-stage pipeline of multi-lingual pairwise similarity measurement followed by multi-partite matching.

Multi-Lingual Similarity. Product titles in different languages are not always mere translations of one another. Companies have used various strategies such as transliteration or even a new name when listing a product in a new market. Other works combined deep learning for text matching with similarity learning [8]. Existing work involves [10] or reduces into a mono-lingual problem [5].

Within product titles, the order of product attributes may be jumbled up across languages. We seek a sequence-agnostic model that could differentiate the importance of different parts of the title. We leverage Multi-Lingual BERT (mBERT) which was pre-trained on 104 languages [6]. Advantageously, it shares embeddings across languages [9], enabling the assessment of similarities among sentences of different languages. However, product attributes such as color or gender are pertinent differences, but mBERT may perceive them to be semantically similar since they are of the same brand and model. For example *"nike womens air zoom fit agility 2 bright mango white crimson blue shoe"* and *"nike women air zoom fit agility 2 purple black white shoes"* are of the same model, but different colors. To overcome this, we train our model on specific attributes to deal with hard cases brought about by attribute differences.

Multi-Partite Matching. Suppose that we are dealing with product titles from L different languages. For each language l, let T_l denote the set of product titles in that language. The goal is to cluster all the product titles in $\mathcal{T} = \cup_{l=0}^{L-1} T_l$, such that for each cluster C, the following constraints are met:

- Every product title belongs to one cluster ($\cup_i C_i = \mathcal{T}$), and only one cluster ($\forall C_i \neq C_j, C_i \cap C_j = \emptyset$).
- A cluster contains at most one product title from a particular language, i.e., $\forall (i, l), |C_i \cap T_l| \leq 1$. This assumes that individual languages have been deduplicated.

The multi-lingual similarity function $S(t_x, t_y)$ measures the similarity between any two titles from different languages. We need to generalize from the pairwise relations induced by the similarity S to the $L-$wise relations induced by the clustering C. This is a complex

problem for two reasons. For one, how to qualify a good cluster from the pairwise similarity of its members. For another, the impracticability of enumerating all the possible clusters to be scored. We thus propose efficient algorithms that build up the multi-partite clusters from effective aggregations of pairwise relations.

2 DATASETS

As we develop the techniques, we would also showcase some empirical evidence on the following real-world datasets.

Shoes & Cameras. The WDC Product Data Corpus for Large Scale Product Matching (version 2.0) was derived from the Common Crawl project [7]. Some e-commerce platforms annotated their products using schema.org vocabulary with product identifiers such as gtin8, gtin13, gtin14, mpn and sku. These identifers allow offers for the same product on different platforms to be grouped together into clusters, which we use as ground truth. The full corpus consists of 26 million offers from 79 thousand websites, grouped into 16 million clusters. As this product corpus does not come with language labels, we use the fastText-based language identification tool [3]. The offers are grouped into 25 different categories, spanning a wide array of products from clothing to electronics. For our experiments, we use two categories, namely *Shoes* and *Cameras*.

Movies. For a different source and type of entities, we make use of open-source data from WikiData and DBPedia. From DBPedia, we extract more than 100,000 movies in more than 20 languages together with movie attributes such as: Distributor, Producer, Starring and Writer. By extracting data from Wikipedia, we ensure that attributes such as writers and producers are accurate and consistent.

For each category above, we take the top 5 most common languages (the top 5 may differ among datasets), as shown in Table 1. English (en) is the largest language subset, and there is always an English title in each ground-truth cluster.

Table 1: Products from top-5 languages in the respective datasets

Dataset	en	fr	es	it	de	nl	ast
Shoes	9,734	8,578	5,417	4,577	3,031	–	–
Cameras	7,833	3,632	4,568	–	3,966	3,553	–
Movies	4,000	3,906	–	3,788	3,937	–	2,721

3 MULTI-LINGUAL SIMILARITY

We illustrate how we derive similarities between product titles of different languages. Importantly, we show our proposed way of model training to allow attributes to assist.

For each input token, mBERT produces a vector representation of 768x1 as hidden states. We add a linear layer of 128 units on top of the baseline mBERT (version: $bert-base-multilingual-uncased$ [1]) as shown by the combined model in Figure 2. This combined model then generates an embedding for the given product title, the multi-lingual similarity between a pair of product is represented as the cosine distance between the 2 embeddings.

Triplet Loss. A product matching system should be able to operate correctly for yet unrepresented products. Inspired by [8], we use triplet loss [2] where a notion of similarity between products is defined as the distance between their representations in an

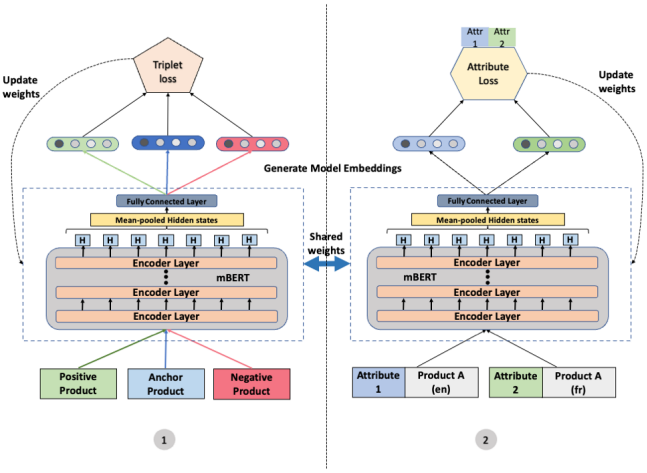

Figure 2: The pipeline is split into 2 parts using the same model: (1) train the model using triplet loss on normal product title dataset (2) generate attributes dataset and further train the model on this specific attributes dataset.

embedding space. During training with triplet Loss, information of clusters is known and triplet loss takes in 3 inputs: an anchor product (a), a positive matching product (p) to the anchor, and a negative product (n) which is a product from a different cluster from the anchor. The model parameters of the combined model in Figure 2 (mBERT and linear layer on top) are adjusted to minimise the triplet loss objective via backpropagation with the loss function:

$$L(a, p, n) = max(0, m + d(\epsilon_\theta(a), \epsilon_\theta(p)) - d(\epsilon_\theta(a), \epsilon_\theta(n))) \quad (1)$$

where m represents the hyper-parameter margin, $\epsilon_\theta(\alpha)$ the encoder and d the distance function (e.g., cosine distance). This loss function minimises the embedding distance between products in the same cluster and maximises that between those from different clusters.

The negatives are split into 3 types: (1) *hard* negatives are the ones that are nearer to the anchor than the positive product, (2) *semi-hard* negatives are further from the anchor than the positive product but still within the hyper-parameter margin m (3) *easy* negatives are more than m further to the anchor as compared to the positive product. The selection of negatives is pertinent since too distant negatives lead to zero triplet loss and too similar negatives may lead to the model being trained on noise [8].

Attribute-Assisted Similarity. To deal with hard cases due to attributes such as color and series number, we devise a training pipeline to differentiate attributes during embedding generation, as illustrated in Figure 2. The attribute-specific training is placed after the triplet loss training since it is solving challenging corner cases the first training may not already address. For each dataset, there are different attributes that impact the results, but they will use the following loss function in Equation 2 so that products with similar attributes are encouraged to have embeddings closer to each other:

$$L_{attribute} = \|\text{attribute difference} - d_{cos}(X, Y)\| \quad (2)$$

X, Y represents the model embedding of title X and Y, while the attribute difference differs across the different datasets since the attributes that each will be focusing on is different. When preparing

the attributes dataset, there are also considerations to make sure the model is able to learn the attributes in a multilingual context.

For Shoes, the attribute difference is as follows:

$$\text{attribute difference}_{color} = d_{cos}(RGB_X, RGB_Y) \tag{3}$$

RGB_X represents the 3 dimensional RGB embedding of the color present in the title X generated by matplotlib. For example, the RGB embedding of a black product will be [0, 0, 0] and a blue product will be [0, 0, 255]. This helps to tell apart colors in product title similar to an RGB embedding, an important attribute among shoes.

For Movies, the attribute difference will be using the following Equation 4 to differentiate between series numbers:

$$\text{attribute difference}_{numeracy} = 2\frac{|x-y|}{|x|+|y|} \tag{4}$$

x and y represent series number present in title X and Y respectively.

For Cameras, we re-use Equation 4 for the model number attribute where x and y represents the model number hashing of title X and Y respectively, which is derived using the following logic:

$$ModelNumberHash = \sum_{i=0}^{n} h(\alpha_i).\rho^{n-i} \tag{5}$$

α_i represents the character at index i and $h(\alpha_i)$ represents the hash of the character, where numbers 0 to 9 will be represented as it is but alphabets will be represented as value from 10 to 35. p-value (ρ) represents the weightage of each character based on its position, a value of less than one means that the characters at the back of the model number is more representative of the product and vice versa. From empirical studies of how products are named, the first few characters are more representative of a product as compared the last few characters which normally represents the version on the product instead. With this assumption, we will be setting the p-value for this research at 1.2, which is more than 1.

Experiments. For each ground-truth cluster, we rank the similarity of each product title within the cluster against all the other product titles (within and without). The recall and precision of the top-10 most similar pairs (we see similar trends for top-20, -50, -100 as well) are presented in Table 2. We further adopt the metric of Normalized Discounted Cumulative Gain or NDCG frequently used in information retrieval. Evidently, the addition of triplet loss improves upon the vanilla mBERT model significantly. Further addition of the attribute assistance produces even better performances, though the gain is relatively modest given the already high level of performance due to triplet loss.

4 MULTI-PARTITE MATCHING

We outline a baseline as well as a proposed algorithm that build on the pairwise similarity between any two products to construct multi-lingual clusters. The similarity scores between all product title pairs of different languages are computed beforehand. Although this has a computational cost $O(n^2)$ where n is the total number of titles, this is only performed once at the start of the algorithm.

Greedy Algorithm. One baseline is a greedy approach. First, we sort product title pairs (of different languages) in descending order of similarity. Beginning from the most similar pair, we accept each pair into the solution, if it does not conflict with any existing cluster. Greedy approaches are often explored when scalability is important with this approach having a minimal additional $O(n^2)$ cost to sort the pairs. However, there are significant problems with the greedy approach. If any false pairing has a higher similarity score compared to the true pairing, it is likely that the false pairing will be selected greedily since better scoring pairs are matched first.

Iterative Bipartite Matching. Attempting to solve $L-$wise multi-partite matching simultaneously is a complex problem. Our proposal is thus to "serialize" the multi-partite matching into a series of incremental bipartite matchings, as illustrated in Figure 3.

Maximum Weight Objective. We begin by constructing ${}^{L}C_2$ bipartite graphs, each involving a pair of languages. We then derive the maximum weight matching from each bipartite graph, and "accept" the strongest bipartite matching (determined in terms of the average weight of the matched edges). We then collapse the matched vertices into a pseudo-vertex. In the next iteration, we measure the similarity between each pseudo-vertex and other language product titles, resulting in ${}^{L-1}C_2$ bipartite graphs, and then repeat the process until all the languages are integrated. In the bipartite-based approach, instead of performing greedy matching, maximum weight matching is used instead to provide a more optimal solution as it considers all edges in the matching. The maximum weight matching helps to providing some robustness to the approach where it is not as easily distracted because of strong false pairings.

Ordering of Language Pairs. The ordering of the languages selected for pairing significantly affects the pairings formed. For example if the true strongest pairing is EN-IT, but EN-FR is matched first, the predicted cluster may create a false pairing between the English title with another French title. We initially experiment with two different ordering methods – largest number of titles and the strongest average pair. However, these did not consider the overall strength of the resultant matchings. We resort to a combinatorial search approach, which allows for minimal supervision regarding the ordering of bipartite matchings while allowing the algorithm to select matchings with better outcomes. As compared to a predefined ordering, the combinatorial search approach can adapt to new datasets with different sizes and languages. The combinatorial search approach also allows for more complex bipartite pairing ordering such as combining EN-FR pairing with a DE-IT pairing. It is not limited to adding one language at a time to an initial set but can combine two different sets of pairings at once.

Aggregating Similarities. Another issue is when we measure similarities across more than 2 entities, e.g., 3 titles involved in the similarity score pairing of EN-FR (Left Node) with DE (Right Node). In this case, there will be two edges between the two 'nodes' (EN-DE and FR-DE). The similarity score can be derived from these edges by taking the maximum (Max), minimum (Min) or average (Avg) of these edges. This is reminiscent of the complete, single, and average linkage concept in agglomerative hierarchical clustering.

Alternative to Blocking. We decided against implementing a blocking scheme i.e. ignoring edges below a fixed threshold value due to the multilingual nature of our problem. Languages are naturally semantically more similar to some and different to others, hence, the ideal threshold likely differs between different language pairings and determining this for every pairing will be costly. Inaccurate determination of this threshold may lead to loss of true positives in certain language pairings, negatively affecting overall performance.

Table 2: Multi-Lingual Similarity: Recall and Precision for Top-10 Ranked Product Pairs for Shoes, Camera, and Movies Datasets

Metrics	Shoes			Camera			Movies		
	mBERT	+triplet loss	+attribute assistance	mBERT	+triplet loss	+attribute assistance	mBERT	+triplet loss	+attribute assistance
Recall	0.527	0.967	0.968	0.498	0.890	0.891	0.397	0.999	0.999
Precision	0.188	0.369	0.370	0.134	0.274	0.274	0.301	0.790	0.790
NDCG	0.590	0.959	0.960	0.563	0.884	0.886	0.526	0.999	0.999

Table 3: Multi-Partite Matching: Recall of Shoes, Camera, and Movies Datasets

	Shoes			Camera			Movies		
	mBERT	+triplet loss	+attribute assistance	mBERT	+triplet loss	+attribute assistance	mBERT	+triplet loss	+attribute assistance
Greedy	0.639	0.937	0.936	0.682	0.833	0.837	0.507	0.609	0.601
Iterative Bipartite (Avg)	0.644	0.950	0.953	0.687	0.839	0.845	0.551	0.672	0.677
Iterative Bipartite (Max)	0.705	0.952	0.953	0.687	0.841	0.851	0.560	0.670	0.677
Iterative Bipartite (Min)	0.537	0.945	0.944	0.636	0.820	0.818	0.450	0.585	0.603

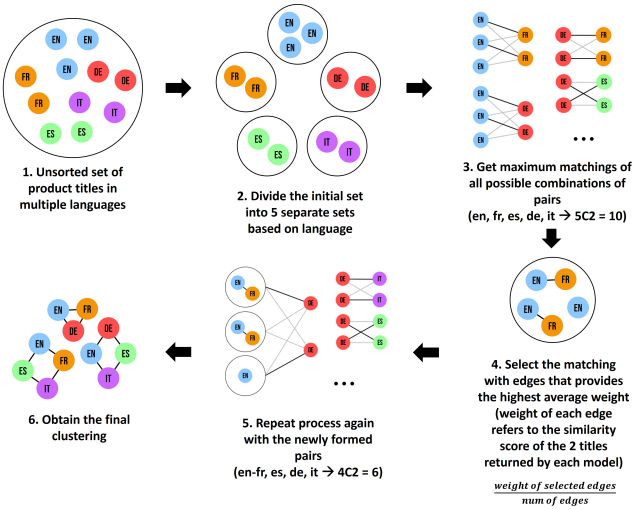

Figure 3: Illustration of iterative bipartite matching

The "serialization" of multi-partite matching through iterative bipartite matching reduces the maximum number of comparisons on each step while by performing the matching on smaller subsets first. While up to L rounds of a maximum of ${}^{L}C_2$ bipartite matchings in each round are required, each matching has complexity of $O(PQlogQ)$ [4], where P or Q is the number of titles in a particular language, which is much smaller than N total number of products.

Experiments. We measure the *Recall* of the predicted multi-partite clusters against the ground-truth clusters. The ground-truth clusters correspond to a number of product title pairs belonging to the same cluster. As the cluster size is capped and constrained (at most one title from each language in a cluster), one cannot obtain higher recall simply by enlarging the clusters. Higher recall would be due to correct clustering that recovers true pairings. The results on the three datasets of Shoes, Cameras, and Movies are shown in Table 3. Evidently, the Iterative Bipartite method generally outperforms the Greedy solution, particularly the Avg and the Max variants. In addition, the contributions of the first stage of enhancing pairwise similarity measurement using triplet loss and attribute assistance are also evident, showing improvements upon the baseline vanilla mBERT.

5 CONCLUSION

We explore matching product titles across different languages. Our contributions are in examining how to measure similarities across any pair of product titles, and how to efficiently cluster products of different languages. Triplet loss with judicious selection of negative examples improves similarity measurements with attributes assisting in hard cases. Resolving multi-partite matching via serialized iterative bipartite matching is promising as well.

ACKNOWLEDGMENTS

Hady W. Lauw gratefully acknowledges the support by the Lee Kong Chian Fellowship awarded by Singapore Management University.

REFERENCES

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805
[2] Elad Hoffer and Nir Ailon. 2015. Deep Metric Learning Using Triplet Network. In *SIMBAD*.
[3] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomás Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *CoRR* abs/1607.01759 (2016). arXiv:1607.01759
[4] Richard Karp. 1980. An algorithm to solve them M×N assignment problem in expected time O(MN logn). *Networks* 10, 2 (1980), 143–152. https://doi.org/10.1002/net.3230100205
[5] El Moatez Billah Nagoudi, Jérémy Ferrero, Didier Schwab, and Hadda Cherroun. 2017. Word Embedding-Based Approaches for Measuring Semantic Similarity of Arabic-English Sentences. In *ICALP*.
[6] Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How Multilingual is Multilingual BERT?. In *ACL*.
[7] Anna Primpeli, Ralph Peeters, and Christian Bizer. 2019. The WDC Training Dataset and Gold Standard for Large-Scale Product Matching. In *WWW Companions*. 381–386. https://doi.org/10.1145/3308560.3316609
[8] Janusz Tracz, Piotr Wójcik, Kalina Jasinska-Kobus, Riccardo Belluzzo, Robert Mroczkowski, and Ireneusz Gawlik. 2020. BERT-based similarity learning for product matching. In *ECOMNLP*.
[9] Shijie Wu and Mark Dredze. 2019. Beto, Bentz, Becas: The Surprising Cross-Lingual Effectiveness of BERT. In *EMNLP-IJCNLP*. 833–844. https://doi.org/10.18653/v1/D19-1077
[10] Tengyuan Ye and Hady W Lauw. 2015. Structural constraints for multipartite entity resolution with markov logic network. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1691–1694.