E-commerce products
Project Report for NLP Course, Winter 2022

Paweł Golik, Mateusz Jastrzebiowski and Aleksandra Muszkowska
Warsaw University of Technology
pawel.golik.stud@pw.edu.pl,
mateusz.jastrzebiowski.stud@pw.edu.pl,
aleksandra.muszkowska.stud@pw.edu.pl

supervisor: Anna Wróblewska
Warsaw University of Technology
anna.wroblewska1@pw.edu.pl

Abstract

In this project, we will explore modern methods used in the product matching problem, which is a generalization of the entity matching problem. These tools are mainly based on deep neural networks that encode individual offerings into vectors called embeddings representing specific knowledge. We will analyze the embedded space and conduct an attempt to develop probing tasks aimed at investigating the properties of embeddings in this domain. In addition, we will check whether encoded vectors of offers for the same products will show high similarity and, on the contrary, offers of unrelated products will not be similar. We believe our work will help understand black-box knowledge representations (embeddings) and shed light on the similarity properties of embedded vectors.

1 Introduction

The e-commerce sector has seen much growth in recent years, compounded by the global coronavirus pandemic. Customers were previously limited to the offerings of local sellers. However, the growth of the Internet and delivery services can open up new sources of goods provided by e-commerce sites such as Allegro.pl and Amazon. The vast number of offers published daily by vendors leads to new challenges in efficiently finding offers of potential interest to customers. Unfortunately, many offers are presented in different formats and with different variations of product names which makes it challenging to build automated tools for matching offers of the same product. Having a tool capable of comparing two different offers with each other would allow for solving many problems, such as offers matching, but also suggesting similar offers in the absence of offers related to the selected product or detecting offers misclassified by vendors as a given product based on too little similarity to other, valid offers.

Relying on the traditional comparison of strings representing offers is not reliable because descriptions and titles of offers can be formed in many different ways (different order of a product number, brand, and name) or contain many additional words that do not help to distinguish the product (e.g., 'best-seller,' 'offer,' etc.). For this reason, many modern tools are based on transformer models, which encode input information as vector embeddings. These solutions are characterized by high efficiency, but they are so-called black boxes from which it is not easy to deduce which features are essential for making predictions. Constructing tools to examine the coded forms of offers would allow a better understanding of these methods. For this purpose, we propose probing tasks examined in the work described in the article.

2 Related Work

Modern state-of-the-art product matching methods rely on deep learning techniques using Transformer models, which allow for creating embeddings from input representing specific knowledge about the encoded entities (Możdżonek et al., 2022), (Tracz et al., 2020). The embeddings map words to real-valued vectors, which reveal semantic aspects, for example, if words are related in meaning or belong to the same topic. Creating such an embedding means enriching as well as filtering out information. As far as we know, most of the research in product matching focuses on building classifiers on top of the extracted embedding representation (Możdżonek et al., 2022). In the training phase, the encoder learns to transform the input into the embedding space, which serves well for the classification task. A slightly different approach is presented in (Tracz et al., 2020), where the embeddings are directly compared and passed to the Loss function assessing their similarity. The Loss function is then minimized for the embedded inputs, which causes the weights of the encoder to change accordingly. This approach may probably ensure more suitable embedding space for future examininig of the similarity of the embeddings as it already imposes a similarity constraint during the training phase.

2.1 Encoder architectures

For e-commerce product matching one can use a Cross-encoders (Wolf et al., 2019) architecture e.g. solution proposed in (Możdżonek et al., 2022). This architecture allows to get high accuracy of matching at the expense of the computation time. Cross-encoders based architectures also require a lot of data to fine-tune and it is impossible to extract embeddings for each sentence provided as an input as the architecture uses only one Bert model to calculate one aggregated embedding for the input.

To solve these problems a Bi-encoders architecture was proposed. Unlike Cross-encoders it uses two separate Bert models and pooling for embeddings calculation. Such an approach enables the user to calculate and easily extract separate embeddings (which are integral part of the architecture) for each sentence provided as an input. To asses the similarity of offers, Bi-encoders use similarity measure e.g. cosine-similarity measure. Overall, Bi-encoders are faster and require less data for fine-tuning than Cross-encoders but obtain lower accuracies.

2.2 Probing tasks

As far as we know, we would be the first to describe probing tasks for embeddings in the product-matching domain. Probing has been extensively described in (Şahin et al. , 2020), but it focuses mainly on probing word embeddings. In our case, we need to probe the embeddings of the offers created from texts consisting of many words. Lindstrom et al. (2020) propose novel probing tasks for the visual-semantic case (pairing images and text), defining three classification tasks relating to the images and text from which the embeddings were created:

- **ObjectCategories** - which of the 80 MS-COCO object categories are present in a given image,
- **NumObjects** - to estimate the number of objects in an image,
- **SemanticCongruence** - whether a caption has been modified (2020).

We took inspiration from this approach and created similar probing tasks corresponding to the product-matching domain.

3 Dataset and EDA

We focus on Web Data Commons - Training Dataset and Gold Standard for Large-Scale Product Matching dataset (WDC for short) prepared by the staff of the University of Mannheim (Primpeli et al., 2019). The dataset contains offers in four categories - Cameras, Computers, Watches, and Shoes. Additionally, each offer is linked to a specific product (cluster_id) and contains textual attributes such as title, description etc. Each observation is a pair of such offers and a label indicating whether these two offers are for the same product (a positive pair) or not (a negative pair). Even in the case of a negative pair, both offers belong to the same category (but different clusters/products).

The training datasets are available in different sizes, varying from small to extra large. In every dataset, the ratio between positive and negative pairs is 1:3. Table 1 presents the exact sizes for each dataset. The proportions between the different collections within one category are as follows: 1 – small, 3 – medium, 15 – large, and 50 – extra-large (xlarge). We do not consider the extra-large datasets as our computational and time resources are limited.

The Gold Standard is verified manually and should be used for testing purposes. Each product contains highly similar negative pairs (complex cases) and less similar negative pairs (easy cases). Table 2 depicts the statistics for the Golden Standard dataset per each category.

The WDC dataset has already been used for product-matching tasks (Możdżonek et al., 2022) and the results of this research is depicted in Table 3.

4 Approach and research methodology

4.1 Obtaining embeddings

To begin with, it was necessary to create an embedded space, the properties of which we probe.

Table 1: WDC datasets sizes. Source: (Peeters et al., 2022)

Category	Size	Positive	Negative	Total
Cameras	Small	486	1,400	1,886
	Medium	1,108	4,147	5,255
	Large	3,843	16,193	20,036
	xLarge	7,178	35,099	42,277
Computers	Small	722	2,112	2,834
	Medium	1,762	6,332	8,094
	Large	6,146	27,213	33,359
	xLarge	9,690	58,771	68,461
Watches	Small	580	1,675	2,255
	Medium	1,418	4,995	6,413
	Large	5,163	21,864	27,027
	xLarge	9,264	52,305	61,569
Shoes	Small	530	1,533	2,063
	Medium	1,214	4,591	5,805
	Large	3,482	19,507	22,989
	xLarge	4,141	38,288	42,429

To do this, we wanted to replicate work from (Możdżonek et al., 2022), which used a cross-encoder architecture with a Bert-like model to distinguish pairs of offers of the same products (positive pairs) from pairs of offers of different products (negative pairs).

The cross-encoder architecture gives better results but requires more data and extended training. In addition, its significant drawback is the lack of explicit use of embeddings for single sentences, which would require additional pooling from embeddings of single tokens.

Therefore, to solve the task of distinguishing between positive and negative pairs of offers, we decided to use a bi-encoder architecture that uses separate Bert models for each sentence and produces embeddings for each sentence. The embeddings are then compared using cosine distance, which is compared with the target (a positive or negative pair). In this way, the embedded space is constructed to consider the similarity of the offers, and the retrieving of embeddings is straightforward.

4.1.1 Fine-tuning of encoders

We took the pretrained Bert model ('bert-base-multilingual-uncased') from the HugginFace Transformers library, then fine-tuned the model on the WDC dataset. Due to resource constraints (using the free version of Google Colab) and the priority of constructing a good-quality embedded space, we decided to use the 'medium' size set and only the 'cameras' category. As an input sentence representing an offer, we take only its 'title' feature to reduce the computational cost. Fine-tuning continued for 200 epochs, with a batch size of 16 and a cosine similarity function to evaluate embeddings, using the SentenceTransformer library that handled the bi-encoder architecture for us. The model's test accuracy at the best epoch was 85% - we used the model from this epoch to create embedded space for probing tasks.

After obtaining the final encoder, we calculated the embeddings (724-dimensional vectors) for each sentence (offer) and then visualized the embedded space using projector.tensorflow.org.

4.2 Probing tasks

The main task of this project is product matching. However, the models proposed are so-called black boxes from which it is difficult to deduce why such decisions were made. The probing aims to reveal what information an embedding encodes (Lindström et al. , 2020).

The general outline of probing (well described in (Belinkov , 2021)) is to take a model trained on some task, product matching in our case. Then generate representations using the model and train another classifier that takes the representations and predicts some properties. From the probing classifier's performance, it should be possible to conclude the probed embedding; if the classifier succeeds, it indicates that the semantic embedding captures interpretable information regarding the aspect under consideration. Unfortunately, the converse is invalid: if classifiers perform poorly, the reason may be that the embedding simply does not capture the property or that the chosen classifier was unsuitable for the task (Lindström et al. , 2020).

In our case, the probing classifier's input will be the offers' embeddings. We will examine whether they have learned a relationship with certain properties and check whether they contain information unrelated to the task (high accuracy of probing classifiers).

4.2.1 Proposed probing tasks

- **Common words**: The goal of the task is to build a classifier that, based on the embedding, will predict whether the listing from

Table 2: Gold Standard Statistics per category. Source: (Primpeli et al., 2019)

Category	#positive	#negative	#combined	title	s:description	spec Table
Computers	150	400	550	100%	88%	21%
Cameras	150	400	550	100%	79%	5%
Watches	150	400	550	100%	77%	5%
Shoes	150	400	550	100%	88%	3%

Table 3: Current state-of-the-art F1 scores in product matching task for models trained on English WDC datasets. Mean value and standardized error (confidence level 95%) for each dataset were calculated from 4 samples. Source: (Możdżonek et al., 2022)

Category	Size	mBERT [x]	XLM-RoBERTa [y]	Ditto [z2]	WDC-Deepmatcher [z]
Cameras	Small	82.13(±4.70)	81.96(±7.75)	80.89	68.59
	Medium	87.86(±2.04)	88.11(±4.22)	88.09	76.53
	Large	90.88(±2.28)	92.36(±0.76)	91.23	87.19
	xLarge	-	-	93.78	89.21
Computers	Small	86.43(±3.69)	81.10(±13.40)	80.76	70.55
	Medium	90.13(±1.89)	88.69(±2.19)	88.62	77.82
	Large	92.48(±2.33)	93.71(±0.77)	91.70	89.55
	xLarge	-	-	95.45	90.80
Watches	Small	79.20(±7.89)	74.98(±13.36)	75.89	73.86
	Medium	84.11(±3.40)	81.30(±8.21)	82.66	79.48
	Large	90.28(±2.36)	91.26(±2.09)	88.07	90.39
	xLarge	-	-	90.10	92.61
Shoes	Small	87.31(±1.64)	83.78(±4.38)	85.12	66.32
	Medium	91.17(±4.21)	89.50(±3.69)	91.12	79.31
	Large	93.52(±2.63)	93.62(±0.67)	95.69	91.28
	xLarge	-	-	96.53	93.45

which the embedding was calculated contained at least one of the common words ('camera,' 'digital,' 'lens'). Such words can be added to any offer sentence without changing their meaning. The training included 1522 offers without common words and 1260 offers with at least one occurrence of any of the common words. The probing task is model-agnostic and dataset-agnostic under the assumption that one would choose a different set of common words.

- **Brand name**: The task of classifying embeddings received from bid sentences into two groups - offers that contain the brand name or those without it. We extracted the brand names from the dataset (they often appeared as a separate 'brand_name' feature), and then for some offers, we removed the brand name to obtain a more balanced dataset for probing (1521 offers without and 1261 with a brand name).

- **The Levenshtein distance**: Instead of operating on single offers (embeddings) in this probing task, we return to the concept of offer pairs from the WDC dataset. For each offer A and offer B from the pair, we calculate the Levenshtein distance (Farouk, M, 2019) $lev\_dist(sentenceA, sentenceB)$, obtaining a new target variable for probing, which denotes the similarity of the sentences (strings) in the pair. The variable is further discretized into five classes ('Similar,' 'Quite similar,' 'Neutral,' 'Hardly similar,' and 'Not similar'). The embedding of offer A and the embedding of offer B are passed as input to the probing classifier. The goal is to predict the similarity of sentences calculated using the Levenshtein distance - the smallest number of edit operations (insertion, deletion, substitution, and transposition) required to transform one string into another.

- **Length of sentences**: To investigate whether the embeddings encode information that allows us to distinguish between the original sentence lengths representing the offers, we constructed a classifier that will try to predict sentence lengths based on the offer embedding. The values of sentence length were discretized into five categories to enable classification ([0, 10), [10, 15), [15, 20), [20, 100)). Choosing correctly sized bins is essential to ensure a balanced dataset.

5 Experiment results and conclusions

We used the entire training set of 2782 different offers to train classifiers for probing tasks. This set was divided using the *train_test_split* function from *Sklearn* library into a training set and a probing set in a 1:4 ratio. In the three probing tasks (Common words, Brand name, Length of sentences), in which the observations were individual bids, the set consisted of 2782 offers. In contrast, the probing task: The Levenshtein distance, which had pairs of offers as input, consisted of 5255 observations.

In all probing tasks, we tested various classifiers: Random Forest, XGBoost, and Logistic Regression. We trained the Logistic Regression model with the Lasso penalty, which is suitable for many features. In our case, there were 768 features for the tasks: Common words, Brand name, Length of sentences, and 1536 (two offers) for the task: The Levenshtein distance. Thanks to Lasso regularization, the classifier selects features that affect prediction, reducing the number of features considered.

The results of the experiments can be seen in Table 4. As we can see, we obtained the best results using a logistic regression model.

5.1 Results interpretation

The two probing tasks with high accuracy scores were Common Words and Brand Name, 82.3% and 76.1 %, respectively. We can interpret such high scores as success in the probing process. Our goal was to find out whether the embeddings of the offers(titles) contain information/are related to common words (first task) such as: 'Camera', 'Len', 'Digit', or product brand (second task). By obtaining high classification results, we can conclude that these two pieces of information are included in the embeddings so that we can get information from them about the occurrence of these words in the titles of the offers.

For the task: The Levenshtein Metric, the results were mediocre. If the results are good, we know that the embeddings are related to the information being tested. Unfortunately, the converse is not true: if classifiers perform poorly, the reason may be that the textual part of the embedding does not capture tested information or that the chosen classifier was unsuitable for the task. In this case, the classifier's poor performance may be due to two things. First, the Levenshtein metric may not accurately replicate the similarity between bids. Second, the classifier may be too simple for such a large feature space.

The last probing task, measuring the relationship of bid title length to bid embeddings, received average results of 65%. In this case, the goal was to get a not-so-high score, as this would have shown a correlation of embeddings with title length, which was not desirable. The result obtained is, on the one hand, satisfactory for us because 65% with a division into four groups is not a high result. However, as in the previous task, we cannot be sure that this result is a consequence of the lack of correlation of embeddings with the length of the listings. This result may be related to the poor choice of classifier for the probing task.

In summary, the results we obtained are satisfactory to us. We are very satisfied that the classification results were high for the first two tasks and that the classification results were average for the last task.

5.2 Discussion on results

In order to carry out innovative probing tasks, we first trained a bi-encoder model, which, compared to the cross-encoder model, gives the possibility to obtain the embeddings we tested more accurately. The results of our model are comparable to those of the cross-encoder model described in (Możdżonek et al., 2022), so we focused most of our attention on training and developing probing tasks, the results described above.

6 Future plans

In Project #2, we plan to continue working on probing as we have received promising results for

Table 4: Accuracy scores[in %] - probing tasks.

Probing task/Classifier	Logistic Regression	Random Forest	XGB
Common words	**82.3**	77.0	79.7
Brand name	**76.1**	72.0	75.8
The Levenshtein distance	33.5	38.0	**40.0**
Length of sentences	**67.0**	64.6	65.3

the Common words and Brand names tasks. We will pay special attention to proposing new probing tasks, e.g., using other metrics comparing offer sentences.

We will check the results on the new WDC dataset category and test another Bert-like model - XLM-RoBERTa.

In addition, we will conduct probing of two embedded spaces - for embeddings created from an encoder without fine-tuning and after fine-tuning, which will shed light on the fine-tuning phase importance.

7 Work division

Name	Work
Paweł Golik	Research, Model fine-tuning, Contributing to writing report
Mateusz Jastrzebiowski	Research, Probing tasks, Contributing to writing report
Aleksandra Muszkowska	Research, Probing tasks, Contributing to writing report

References

[Belinkov 2021] Yonatan Belinkov. 2021. *Probing Classifiers: Promises, Shortcomings, and Advances*. Computational Linguistics, 48(1):207–219.

[Lindström et al. 2020] Lindström, Adam & Björklund, Johanna & Bensch, Suna & Drewes, Frank. 2021. *Probing Multimodal Embeddings for Linguistic Properties: the Visual-Semantic Case*. *In Proceedings of the 28th International Conference on Computational Linguistics*, pages 730–744, Barcelona, Spain (Online). International Committee on Computational Linguistics.

[Şahin et al. 2020] Şahin, Gözde Gül and Vania, Clara and Kuznetsov, Ilia and Gurevych, Iryna 2020. LINSPECTOR: Multilingual Probing Tasks for Word Representations. *Computational Linguistics*. **46**, 335-385 (2020,6), https://doi.org/10.1162/coli

[Możdżonek et al.2022] Możdżonek, Michał & Wróblewska, Anna & Tkachuk, Sergiy & Łukasik, Szymon 2022. *Multilingual Transformers for Product Matching – Experiments and a New Benchmark in Polish.*. 1-8. 10.1109/FUZZ-IEEE55066.2022.9882843.

[Duffner et al.2021] Duffner, Stefan & Garcia, Christophe & Idrissi, Khalid & Baskurt Atilla 2021. *Similarity Metric Learning. Multi-faceted Deep Learning - Models and Data*

[Tracz et al.2020] Tracz, Janusz & Wójcik, Piotr Iwo & Jasinska-Kobus, Kalina & Belluzzo, Riccardo & Mroczkowski, Robert & and Gawlik, Ireneusz 2020. *BERT-based similarity learning for product matching. In Proceedings of Workshop on Natural Language Processing in E-Commerce, pages 66–75, Barcelona, Spain. Association for Computational Linguistics.*

[Primpeli et al.2019] Primpeli, A., Peeters, R., & Bizer, C. 2019. *The WDC Training Dataset and Gold Standard for Large-Scale Product Matching. Companion Proceedings of The 2019 World Wide Web Conference.*

[Peeters et al.2022] Peeters, Ralph & Bizer, Christian 2022. *Cross-language learning for product matching. WWW Companion, 2022a*

[Farouk, M2019] Farouk, M. 2019. *Measuring sentences similarity: a survey. arXiv preprint arXiv:1910.03940*

[Wolf et al.2019] Wolf, Thomas & Sanh, Victor & Chaumond, Julien & Delangue, Clement. 2019. *Transfertransfo: A transfer learning approach for neural network-based conversational agents.*

[Mazare et al.2018] Mazare, Pierre-Emmanuel & Humeau, Samuel & Raison, Martin & Bordes, Antoine. 2018. *Training millions of personalized dialogue agents.*