Motivation

In e-commerce platforms, hundreds of millions of items are being listed for sale every day, thus providing a satisfactory search and purchase experience brings many challenges. One huge challenge for e-commerce portals is introducing product-based experience. From the buyer’s perspective, this means easy search and price comparability, while merchants benefit by having access to a high-quality product catalog, speeding up the listing process and providing more complete product descriptions.

Product matching, i.e., being able to infer the product being sold for a merchant-created offer, is crucial for any e-commerce marketplace, enabling product-based navigation, price comparisons, product reviews, etc. This problem proves a challenging task, mostly due to the extent of product catalog, data heterogeneity, missing product representatives, and varying levels of data quality.

Contributions

- we apply state-of-the-art BERT-based models [1] in the similarity learning setup to solve the product matching task in the e-commerce domain.
- we compare the usefulness of modern BERT-based architectures such as BERT and DistilBERT [7] for the product matching task.
- we propose category hard batch construction strategy, which proves to increase the fraction of active training triplets and the performance of the final model.
- we adopt and evaluate different batch construction strategies in the similarity learning setup for solving product matching.

Product matching with similarity learning

Product matching aims at identifying offers of the same product across many merchants selling it in an e-commerce portal and integrating the information into a single entry in a product catalog. While offers are vendor listed items described by title, its text description, attributes, and photos, a product represents a manufacturer’s description of a good and is described similarly. Recent papers mostly focus on using only the information contained in the titles or using both titles and attributes [4]. In this work, in addition to using the title and attributes information, we also make use of the category, i.e., an identifier of a set of goods of the same type.

To solve the product matching problem with triplet loss [3], we introduce a notion of similarity between offers and products, defined as proximity of their representations in some embedding space. Each training example is defined as a triplet (o, p+, p−), denoting an offer (anchor), a matching product (positive), and a non-matching product (negative).

\[

e(o, p^+, p^-) = \max(0, m + d(q(o), q(p^+)) - d(q(o), q(p^-)))
\]

We choose the transformer [9] as the basic encoder. The transformer architecture gained a lot of attention due to achieving state-of-the-art results on Natural Language Understanding benchmarks [10, 6]. Our BERT usage as an encoder is inspired by [5].

Datasets

We perform all the experiments using proprietary datasets composed of offer-product matches originating from a real-world e-commerce application. We conduct all the experiments using the three datasets: electronics, beauty, and culture.

Available matches Products

culture 300K 800K
electronics 200K 400K
beauty 300K 200K

Baselines

We compare eComBERT, i.e., a standard BERT model with an additional layer of 768 linear units on top pretrained on domain-specific data, against the following baselines:

- a modified implementation of the StarSpace [12] BOW encoder, a commonly used neural embedding baseline for similarity learning problems,
- non-finetuned HerBERT [6], a BERT-based encoder trained on a big Polish language corpus,
- finetuned HerBERT, with an additional 768 dimension linear layer on top,
- non-finetuned eComBERT.

Since language-specific BERT models perform better than general purpose English models [6], we do not include the latter among the baselines. To make a fair comparison, we apply the same sampling strategy and objective for all the baseline experiments.

Encoder architectures

BERT pretraining is very costly and its inference time is quite substantial in comparison to simpler models. To alleviate these issues, we ran eComBERT pretraining with 4 BERT layers (small eComBERT) and we pretrained DistilBERT on our own internal data (Distil eComBERT). In Table 1 we report test accuracies for the models on all of our prepared datasets. Those models still achieve competitive results across different domains, when cutting the inference time by half and two thirds, for Distil eComBERT and small eComBERT, respectively.

Table 1. Accuracy of models with different BERT architectures trained for 5k steps with category hard sampling strategy.

Table 2. Test accuracy of models trained with different strategies for 1000 steps on electronics.

Conclusions

- BERT-based models combined with appropriately adopted similarity learning obtain high accuracy on offers with either observed or zero-shot products.
- Pre-training BERT-based models on domain specific data improves the model performance.
- Smaller BERT architectures can achieve comparable results to bigger model with significant increase in inference time.

References


ML Research