KD-Boost: Boosting Real-Time Semantic Matching in E-commerce with Knowledge Distillation

Sanjay Agrawal
Amazon.com Inc., India
sanjagr@amazon.com

Vivek Sembium
Amazon.com Inc., India
viveksem@amazon.com

Ankith M S
Amazon.com Inc., India
ankiths@amazon.com

Abstract

Real-time semantic matching is vital to web and product search. Transformer-based models have shown to be highly effective at encoding queries into an embedding space where semantically similar entities (queries or results) are in close proximity. However, the computational complexity of large transformer models limits their utilization for real-time matching. In this paper, we propose KD-Boost, a novel knowledge distillation algorithm designed for real-time semantic matching. KD-Boost trains low latency accurate student models by leveraging soft labels from a teacher model as well as ground truth via pairwise query-product and query-query signal derived from direct audits, user behavior, and taxonomy-based data using custom loss functions. Experiments on internal and external e-commerce datasets demonstrate an improvement of 2-3% ROC-AUC compared to training student models directly, outperforming teacher and SOTA knowledge distillation benchmarks. Simulated online A/B tests using KD-Boost for automated Query Reformulation (QR) indicate a 6.31% increase in query-to-query matching, 2.76% increase in product coverage, and a 2.19% improvement in relevance.

1 Introduction

Accurate real-time semantic matching, i.e., identifying matching entities for a query, has become increasingly essential for web search and e-commerce product search. To address the semantic gap between the pool of queries and results(e.g., products in product search), this matching is often performed in two steps shown in Figure 1: (1) **Automated Query Reformulation (QR),** which maps a poorly formed (e.g., including code-mixed language, typos) user query to semantically similar well-formulated queries with wider coverage of results. (2) **Result/Product Sourcing (PS)** which retrieves matching results for a given user query. In

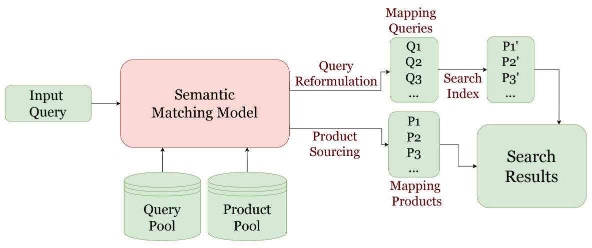

Figure 1: Semantic Matching Model: Incorporating Query Reformulation and Product Sourcing Workflow

this work, we focus on enhancing real-time representation models for both queries and results aiming to improve both QR and PS tasks. Since product search is our primary application, we refer to results as products though the matching problem has broader applicability.

Existing SOTA techniques for semantic matching are mostly based on Siamese networks (Huang et al., 2013) (Ranasinghe et al., 2019) that comprise of two identical sub-networks that generate semantic representations for the pair of entities (query-query or query-product) to be compared. Of these, transformer-based models such as BERT and DistilBERT (Devlin et al., 2018) (Sanh et al., 2019) have been shown to yield highly accurate matching performance. However, these models fail to satisfy the strict latency requirements of large scale product search in B2C e-commerce. On the other hand, smaller encoder models such as MiniLM (Wang et al., 2020) with low latency often result in poor matches. A common approach for addressing this performance gap is via knowledge distillation (KD) (Hinton et al., 2015) methods that transfer information from a larger teacher model to a smaller student model via soft labels from the teacher. The resulting models are often superior to those obtained via direct training of student models but inferior to the teacher model. Further, this approach does not permit the student models to correct for errors in the teacher model itself.

Contributions. In this work, we consider the problem of semantic matching of queries with other queries and products. We propose an efficient KD techniques that learns via imitation of soft relevance labels from a larger teacher model as well as the original ground truth. This approach allows the student model to capture the nuanced knowledge from the teacher model and also incorporate explicit guidance from the ground truth. Below we summarize our key contributions:

1. We propose KD-Boost, a novel KD algorithm designed for real-time semantic matching, which leverages soft labels from one or more teacher models as well as ground truth to learn a highly accurate and compute-efficient student model.
2. To address the dual needs of query reformulation and product sourcing, we utilize multiple sources of similarity and dissimilarity signals (e.g., query-product pairs with editorial ordinal relevance labels, user-behavioural data such as clicks and purchases, product taxonomy) with tailored loss functions to ensure that the model learns representations that can effectively capture the nuances of relevance and similarity.
3. Experimental evaluation of the proposed approach on both internal and external e-commerce datasets (Reddy et al., 2022) results in a significant boost of 2-3% ROC-AUC on the query-product relevance task compared to training student models directly and even surpasses teacher and SOTA KD benchmarks.
4. Lastly, online A/B testing of the proposed approach in real-time product search resulted in an increase of query-to-query automated query reformulation rate by 6.31% which in turn yielded in improved product coverage (+ 2.76%) and relevance (+ 2.19%.)

Note that our KD approach has wider applicability to other real-time semantic matching scenarios beyond product search. It can also be used with any choice of student and teacher encoder models.

2 Related Work

Semantic Matching: The Sentence-BERT (Reimers and Gurevych, 2019) refines the BERT algorithm by using a siamese network, thus making it suitable for semantic matching tasks with higher computational requirements, but not for real-time semantic matching tasks. To reduce the inference cost, many variants of BERT have been proposed, such as PowerBERT (Goyal et al., 2020), DistilBERT (Sanh et al., 2019). Despite these advances, these models are not suitable for real-time applications. MiniLM (Wang et al., 2020), a 3-layer transformer-based model that is less complex than BERT, is more suitable for real-time semantic matching, but suffers from a lack of performance due to its limited number of encoder layers.

In Appendix B, Figure 3 illustrates the architecture of both the teacher model, Siamese BERT (S-BERT), and the low latency model, Siamese MiniLM (S-MiniLM). We introduce S-MiniLM as a low latency alternative to the teacher model, with the key distinction being the use of the MiniLM model for calculating embedding vectors for tokens in the input text.

Knowledge Distillation (KD): Many efforts have been made in KD to improve the performance of the student models (Ankith et al., 2022) (Kim et al., 2021). (Hinton et al., 2015) proposed a knowledge distillation method, in which output from the complex network is used as a soft target for training the simple network. Since then, KD (Yim et al., 2017) has been widely adopted in many learning tasks. Distilling complex models into simple models has been shown to improve many NLP tasks to achieve impressive performance (Kim and Rush, 2016) (Mou et al., 2016). HISS (Ankith et al., 2022) proposes a KD method for real-time semantic matching to distill query-product relevance knowledge encoded in BERT to a low latency model DSSM, but its performance is still lower. In addition to single-teacher knowledge distillation, multi-teacher knowledge distillation has also been explored (Vongkulbhisal et al., 2019). In some recent studies (You et al., 2017) (Fukuda et al., 2017), teachers have been assessed equally or their importance weights have been manually adjusted.

3 Problem Statement

Our primary goal is to improve the performance of the student model on both query reformulation and product sourcing tasks via effective representations of queries and products in a shared semantic space while significantly reducing inference time. Building such a versatile model would allow us to minimize the costs associated with maintenance and production.

We now define the problem formally in terms of the three available input signals. **(i) Expert labels on product-query pairs:** Let $D_{PQ}^{label} = \{(q_i, p_i, y_i)\}_{i=1}^{n}$ denote expert annotations on product-query pairs where i is the index over the tuples, q_i and p_i represent the query and product entities respectively, and y_i represents the ground truth label belonging to one of the three classes: (i) High relevant, (ii) Low relevant, or (iii) irrelevant. **(ii) User Behavioral data:** Let $D_{PQ}^{purchase} = \{(q_i, p_i, c_i)\}_{i=1}^{m}$ denote customer purchase behavior data where q_i and p_i represent the query and product as before and c_i denotes the total number of purchases of product p_i after issuing query q_i. While this data is too noisy for direct modeling of product-query match, it can be used to arrive at highly similar queries based on the overlap in the associated product purchases. Specifically, we define the distribution over query pairs as the Gram matrix corresponding to the normalized product-query purchase counts and identify query pairs $D_{QQ+} = \{(q_i, q_i^{'})\}_i$ that show much higher occurrence relative to random chance by employing Normalized Pointwise Mutual Information (NPMI) based criteria (Section 4.1.2). **(iii) Product Browse Taxonomy:** Further, given a set of queries and classifiers that can map the queries to a product browse taxonomy, one can identify the taxonomy labels for all the queries and construct pairs $D_{QQ-} = \{(q_i, q_i^{'})\}_i$ with non-matching labels which can be viewed as hard-negatives. (Section 4.1.3).

Given all these signals, the objective is to learn an efficient model M such that for a given query q, product p and another query q', the proximity of corresponding embeddings $M(q), M(p), M(q')$ is close to that expressed in the input signals.

4 Proposed KD-Boost

Our solution approach comprises two key stages. First, we build a teacher model that accounts for the various signals mentioned in Section 4.1. Then, we learn an efficient student model that not only mimics the soft labels of the teacher model but also uses the original ground truth (Section 4.2). In Section 4.3, we discuss practical modifications that further improve model performance.

4.1 Teacher Training Objective

During the training of the teacher model, we incorporate human annotated query-product pairs D_{PQ}^{label} as well as similar and dissimilar query-query pairs from the D_{QQ+} and D_{QQ-} datasets. To establish a comprehensive framework for training the teacher model, we define custom loss functions that account for the complexity of the task at hand.

4.1.1 Ranking Loss

We construct our customized ranking loss (see eq 1) on D_{PQ}^{label} dataset to take advantage of the ordinal nature of hard labels. When considering any i^{th} training sample (q_i, p_i, y_i), the idea behind ranking loss is that cosine score should learn the actual order of relevance while training the siamese model. The relevance gradation ensures highly relevant products are prioritized over low relevant products.

$$\begin{aligned} L_{\text{PQ}} = \sum_{(q_i,p_i,y_i) \in D_{PQ}^{label}} & (1_{y_i=high}(\hat{y_i} - 1)^2 + \\ & 1_{y_i=low}((min(0, \hat{y_i} - \theta_L))^2 \\ & +(max(0, \hat{y_i} - \theta_U))^2) \\ & +1_{y_i=irrelevant}(max(\hat{y_i}, 0))^2) \end{aligned} \tag{1}$$

where θ_L and θ_U are hyper-parameters, $\hat{y_i}$ is model prediction score, and $1_{y_i=.}$ is an indicator function.

4.1.2 User Behaviour Data Loss

Generating human audited relevance data is a time-consuming and costly task. In practice, it is not feasible to generate audit data covering the entire semantic space of e-commerce. In contrast, we have copious amounts of data on customer behavior (search query followed by purchase), $D_{PQ}^{purchase}$, which implicitly contains the relevance signal. Despite customer data being abundant, it is noisy and has to be used in conjunction with relevance audit data to build a robust relevance model.

Laus (Lau et al., 2014) used Normalized Point-wise Mutual Information(NPMI) to measure topic co-occurrence, which we leverage to construct query-query relevant pairs. In this study, we analyze the possibility that two queries will co-occur, based on their individual probabilities, and compare that to the case when the two queries are completely independent. A probability distribution can be calculated by normalizing $D_{PQ}^{purchase}$'s purchase count across queries. By analyzing their common products, it is possible to measure the joint distribution of any two queries. Based on this definition, we construct the semantically similar query-query data D_{QQ+} using $D_{PQ}^{purchase}$ data with NPMI (eq. 2) scores greater than τ_{npmi}. Appendix E illustrates a few QQ positive pairs derived through this method.

$$NPMI(q_i, q_j) = \frac{log \frac{P(q_i,q_j)}{P(q_i)P(q_j)}}{-logP(q_i, q_j)} \tag{2}$$

where $P(q_i, q_j) = \sum_{k=0}^{Z} \frac{PC(q_i,p_k)}{\sum_{y=0}^{Z} PC(q_i,p_y)} \cdot \frac{PC(q_j,p_k)}{\sum_{y=0}^{Z} PC(q_j,p_y)}$ and $P(q_i) = \frac{\sum_{j=0}^{Z} PC(q_i,p_j)}{\sum_{i=0}^{Y} \sum_{j=0}^{Z} PC(q_i,p_j)}$. Y and Z represent the total number of distinct queries and ad products in $D_{PQ}^{purchase}$. $PC(q_i, p_j)$ returns the purchase count from $D_{PQ}^{purchase}$ for a given query q_i and product p_j. By leveraging the D_{QQ+} data, we define the following loss function for learning query-query semantics.

$$L_{QQ+} = \sum_{(q_i,q_i^{'}) \in D_{QQ+}} ((min(0, \hat{y}_i - \theta_L))^2 \qquad (3)$$

Unlike the loss function defined for low relevant pairs in Equation 1, the cosine score in Equation 3 does not have an upper limit. The rationale behind this loss function is that relevant query pairs in D_{QQ+} do not indicate a specific degree of relevance, such as high or low relevance.

4.1.3 Taxonomy Based Loss

E-commerce platforms employ predefined multi-level taxonomies or browse nodes to categorize their extensive product catalogs. These taxonomies encode the relevance between products and offer opportunities to derive various relationships. Query classification models have been developed by numerous e-commerce companies (Skinner and Kallumadi, 2019), which assign a distribution score to queries based on the taxonomy tree. Consequently, two queries expressing distinct intents within the taxonomy tree will receive different scores. Appendix F showcases some Q-Q hard negative examples generated using this methodology. This data allows us to effectively discern irrelevant query-query pairs in the embedding space, even if they share some common words. In our study, we define taxonomy loss as follows, where D_{QQ-} represents the query-query hard negative dataset.

$$L_{QQ-} = \sum_{(q_i,q_i^{'}) \in D_{QQ-}} (max(\hat{y}_i, 0))^2 \qquad (4)$$

4.1.4 Teacher Training

To acquire semantic understanding through the teacher model, we commence by initializing our BERT model with pre-trained weights. During the initial epochs, we employ D_{PQ}^{label} and D_{QQ+} to train the model parameters, optimizing for loss terms in equation 5. The importance of loss terms

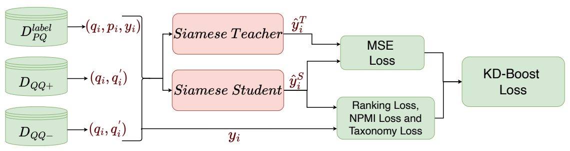

Figure 2: The training workflow of the student model adheres to the KD-Boost method.

L_{PQ} and L_{QQ+} is regulated by α_1 and α_2, respectively.

$$L_1 = \alpha_1 * L_{PQ} + \alpha_2 * L_{QQ+} \qquad (5)$$

In the subsequent epochs, we generate hard negatives using a taxonomy tree that encodes product relevance, as explained in sec 4.1.3. For each epoch, we identify query pairs that are semantically similar but do not share a common browse node. These pairs are added to the data D_{QQ-} as hard negatives, aiming to optimize the following eq.

$$L_2 = \alpha_1 * L_{PQ} + \alpha_2 * L_{QQ+} + \alpha_3 * L_{QQ-} \qquad (6)$$

The significance of the taxonomy loss is determined by the weight scalar α_3.

4.2 Student Training using KD-Boost Method

Figure 2 presents our proposed KD-Boost framework. Traditionally, KD methods force student models to mimic only teacher predictions. This is based on the idea that soft labels provide better insight than hard labels. We propose to imitate the soft label obtained by the teacher, along with learning from the hard label, to aid in recognising the areas where the teacher scores are not doing well. Our loss function for learning student model parameters is defined as follows:

$$L_{\text{KD-Boost}} = \gamma[\sum_{(q_i,p_i,y_i) \in D_{PQ}^{label}} (\hat{y}_i^T - \hat{y}_i^S)^2 + \sum_{(q_i,q_i^{'}) \in D_{QQ+} \cup D_{QQ-}} (\hat{y}_i^T - \hat{y}_i^S)^2] + (1 - \gamma)(L_2) \qquad (7)$$

In the equation mentioned above, $\hat{y}_i^T$ represents a soft label obtained from a teacher model T, while $\hat{y}_i^S$ denotes the prediction score from the Student model S. The scalar value γ (where $0 < \gamma < 1$) determines the relative significance of soft and hard labels. For further insights into the effects of altering γ from 0 to 1 on the performance of the student model, please refer to Appendix D.3.

4.3 Practical Modifications

To enhance the model performance in practice, we incorporate the following modifications:
(1) During the teacher training process outlined in Sec 4.1.4, we initially train the model using Equation 5, followed by Equation 6. By following this sequence, we effectively manage the model's stability, allowing it to learn from the data in a controlled and consistent manner.
(2) In addition, we introduce a multi-teacher KD-Boost algorithm that enables knowledge distillation from multiple teachers simultaneously. By amalgamating the knowledge from multiple teachers, the student model can gain access to a broader range of insights and information, resulting in a more comprehensive understanding of the underlying data. The multi-teacher KD-Boost algorithm involves the utilization of m soft labels, which are incorporated through m MSE loss functions.

5 Experiments and Results

We start by presenting the datasets and the setup of the experiments. Note that further details on *dataset generation* and *experimental setup* are presented in Appendix A.
Dataset Generation We collected anonymized customer behaviour $D_{PQ}^{purchase}$ data between Oct'22 and Feb'23 from India marketplace historical logs. This data is further cleaned by removing query-product pairs without sufficient purchases (<20). To generate data using taxonomy, we gather browse node associations for 400K queries randomly selected from $D_{PQ}^{purchase}$ data. D_{QQ-} is generated using browse node mapping to keep irrelevant query-query pairs apart in the embedding space. In case of D_{PQ}^{label}, we collected a sample of 4.2 Million human-annotated <query, ad title> pairs from IN marketplace which is anonymized, and is not representative of production. We generated validation and test datasets by randomly sampling 60K query-ad pairs each from IN marketplace, and removed these 120K pairs from training. As a result of our evaluation of performance, high and low relevant are considered positive classes, while irrelevant is considered a negative class.
Algorithm Baselines We compare our proposed method with the following baselines in this paper. These baselines are all trained on the same dataset to ensure a fair comparison.
(i) KD-DSSM (Nigam et al., 2019) The low latency DSSM model is trained using soft labels from the SBERT model.
(ii) S-MiniLM (Wang et al., 2020) directly train S-MiniLM model without using any KD.
(iii) KD^{Soft} (Hinton et al., 2015) S-MiniLM model is trained only using soft labels from a teacher model.
(iv) HISS (Ankith et al., 2022) The author proposes a KD method of real-time semantic matching using an additional alignment loss.
(v) Teacher Model (Devlin et al., 2018) (Sanh et al., 2019) Teacher model is directly trained using a training dataset.
(vi) Ensemble This baseline was evaluated in case of Multi-teacher KD-Boost which ensembles several teachers.
Evaluation Metric As a performance metric, we use roc-auc (Brown and Davis, 2006).

5.1 Main Results

We summarize the results of our proposed method on our internal dataset in Table 1 where it is compared to strong SOTA baseline methods. We demonstrate the effectiveness of our proposed method using two teacher models separately, S-BERT and S-DistilBERT. In addition, we use S-BERT and S-DistilBERT to prove the efficacy of the multi-teacher KD-Boost algorithm. In our experiments, KD-Boost outperforms all baseline methods by a significant margin. Furthermore, KD-Boost outperforms the respective teacher model, which has never been achieved using any previous knowledge distillation method. Regarding the *External Amazon Shopping Dataset*, summarized results are presented in Table 2. In this study, S-BERT serves as the teacher model. Among all the baselines, KD-Boost outperforms them by a significant margin, establishing its superiority in comparison to the state-of-the-art methods.

5.2 Simulated A/B Experiments

To assess the effectiveness of KD-Boost, we conducted a real-time QR A/B testing for a duration of 7 days. For more detailed information about the QR system, please refer to Appendix C. During the testing period, we utilized three key metrics to evaluate the performance of our proposed method in comparison to the current production model: i) Increase in query reformulations ii) Increase in product coverage iii) Reduction in irrelevancy. Notably, we observed a considerable 6.31% increase in query reformulations, highlighting the effectiveness of KD-Boost in generating semantically di-

Model	roc-auc/gain%	P/R @0.7
KD-DSSM	0.8732/0%	95.11/79.48
S-MiniLM	0.9280/6.27%	97.41/80.45
Teacher: S-DistilBERT, Student: S-MiniLM		
Teacher	0.9382/7.44%	97.74/82.76
KD^{Soft}	0.9324/6.78%	97.68/81.38
HISS	0.9364/7.23%	97.98/80.50
KD-Boost	**0.9441/8.12%**	**98.04/82.8**
Teacher: S-BERT, Student: S-MiniLM		
Teacher	0.9461/8.34%	98.13/83.04
KD^{Soft}	0.9394/7.58%	97.80/82.88
HISS	0.9442/8.13%	98.12/82.86
KD-Boost	**0.9473/8.48%**	**98.15/83.72**
Multi-Teachers, Student: S-MiniLM		
Ensemble	0.9471/8.46%	98.14/83.62
KD^{Soft}	0.9428/7.97%	98.02/82.54
HISS	0.9452/8.24%	98.07/82.97
KD-Boost	**0.9489/8.67%**	**98.16/84.37**

Table 1: The ROC-AUC of several models based on the query-product labelled dataset. P and R denote precision and recall at 0.7 threshold, respectively. Since KD-DSSM serves as the baseline, gain% is 0. Ensemble in the Multi-teachers section refers to the ensemble performance of several teachers.

verse queries. Furthermore, there was a noticeable expansion of 2.76% in product coverage, encompassing a broader range of products. In addition, through a comparison of relevance judgments made by human evaluators, we found a significant 2.19% improvement in reducing irrelevancy.

Latency We assessed the retrieval latency of BERT, DistilBERT and MiniLM models in online settings for embedding-based semantic matching. To achieve this, we built all models in PyTorch and converted them to ONNX (Bai et al., 2019). In an online setting, we use the Deep Java Library `https://github.com/deepjavalibrary/djl` to load ONNX models and generate embeddings for a user query. We mapped a user query to k-nearest neighbor products ((k=100) in real-time by leveraging the HNSW library (Malkov and Yashunin, 2018) (mlinks=64 and ef_construction=256). Based on our latency results, BERT/DistilBERT have a higher inference latency of 10.46ms/5.64ms on CPU cores (on m5.4xlarge) than MiniLM, 1.22ms, which means more hardware is required to reach the same TPS (transactions per second).

Model	roc-auc	gain%
KD-DSSM	0.8468	0 (baseline)
S-MiniLM	0.8723	3.01%
Teacher: S-BERT, Student: S-MiniLM		
Teacher Model	0.8868	4.72%
KD with Soft Label	0.8789	3.79%
HISS	0.8821	4.16%
KD-Boost	**0.8905**	**5.16%**

Table 2: Main Results on External Amazon shopping query dataset

Model	roc-auc	Q-Q Irrelevance
S-BERT: wo/w	0.948/0.946	22.8%/10.9%
S-MiniLM: wo/w	0.931/0.928	24.5%/12.8%
KD-Boost: wo/w	0.952/0.947	21.5%/9.6%

Table 3: An analysis of the ROC-AUC of Query-Product human audited test dataset and QQ Irrelevance of various models with (w) and without (wo) taxonomy loss.

5.3 Taxonomy Loss Effect

We gathered an anonymized dataset comprising 10K Q-Q samples, which were subsequently audited by our in-house human auditing team. In Table 3, we present the AUC values of various models on test datasets, both with (w) and without (wo) taxonomy loss, alongside the measure of Q-Q irrelevance. Our findings demonstrate that optimizing for taxonomy loss slightly reduces the AUC, but significantly decreases Q-Q irrelevance. Maintaining a low level of Q-Q irrelevance is crucial, as query reformulation (QR) relies on selecting products from semantically similar queries.

6 Conclusion

In this paper, we introduce KD-Boost, a novel knowledge distillation technique specifically designed for real-time semantic matching. KD-Boost effectively trains low latency student models by leveraging soft labels from a teacher model, along with ground truth information obtained from pairwise query-product and query-query signals sourced from diverse channels. Through the utilization of both internal and public datasets, we showcase the superior efficiency of our proposed method compared to existing SOTA KD benchmarks.

Ethics Statement

Our primary objective is to enhance the student model's performance in semantic matching tasks by creating effective representations of queries and products in a shared semantic space while achieving a significant reduction in inference time. The underlying motivation for these efforts is to decrease maintenance and production costs by constructing a multi-application model. This approach aims to broaden the accessibility of the technology to a wider community while ensuring minimal adverse environmental impact. Throughout this NLP research study, we meticulously planned and executed our methodology, adhering rigorously to ethical principles and guidelines. Prior to submission, the study underwent a thorough review and approval process by our company's research leads. Additionally, we fully complied with the guidelines established by the EMNLP conference regarding the use of language data. The researchers take complete responsibility for ensuring the ethical conduct of the study and are resolute in their dedication to upholding the utmost standards of ethical research practices in the field of NLP.

References

MS Ankith, Sourab Mangrulkar, and Vivek Sembium. 2022. Hiss: A novel hybrid inference architecture in embedding based product sourcing using knowledge distillation.

Junjie Bai, Fang Lu, Ke Zhang, et al. 2019. Onnx: Open neural network exchange. https://github.com/onnx/onnx.

Christopher D Brown and Herbert T Davis. 2006. Receiver operating characteristics curves and related decision measures: A tutorial. *Chemometrics and Intelligent Laboratory Systems*, 80(1):24–38.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Takashi Fukuda, Masayuki Suzuki, Gakuto Kurata, Samuel Thomas, Jia Cui, and Bhuvana Ramabhadran. 2017. Efficient knowledge distillation from an ensemble of teachers. In *Interspeech*, pages 3697–3701.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pages 3690–3699. PMLR.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

Taehyeon Kim, Jaehoon Oh, NakYil Kim, Sangwook Cho, and Se-Young Yun. 2021. Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation. *arXiv preprint arXiv:2105.08919*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Lili Mou, Ran Jia, Yan Xu, Ge Li, Lu Zhang, and Zhi Jin. 2016. Distilling word embeddings: An encoding approach. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1977–1980.

Priyanka Nigam, Yiwei Song, Vijai Mohan, Vihan Lakshman, Weitian Ding, Ankit Shingavi, Choon Hui Teo, Hao Gu, and Bing Yin. 2019. Semantic product search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2876–2885.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Tharindu Ranasinghe, Constantin Orăsan, and Ruslan Mitkov. 2019. Semantic textual similarity with siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1004–1011.

Chandan K. Reddy, Lluís Màrquez, Fran Valero, Nikhil Rao, Hugo Zaragoza, Sambaran Bandyopadhyay, Arnab Biswas, Anlu Xing, and Karthik Subbian. 2022. Shopping queries dataset: A large-scale ESCI benchmark for improving product search.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Michael Skinner and Surya Kallumadi. 2019. E-commerce query classification using product taxonomy mapping: A transfer learning approach. In *eCOM@ SIGIR*.

Jayakorn Vongkulbhisal, Phongtharin Vinayavekhin, and Marco Visentini-Scarzanella. 2019. Unifying heterogeneous classifiers with distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3175–3184.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4133–4141.

Shan You, Chang Xu, Chao Xu, and Dacheng Tao. 2017. Learning from multiple teacher networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1285–1294.

A Experimental Setup

A.1 Dataset Generation

External Shopping Query Dataset This dataset consists of 420K training samples and 91K test samples. To create a validation dataset, 10% of the training dataset is randomly selected and removed from the training dataset. Each query-product pair in this dataset is annotated with labels denoted as E/S/C/I, which stand for Exact, Substitute, Complement, and Irrelevant. In the context of search, the pairs labeled as Exact and Substitute are considered relevant (positive class), while the pairs labeled as Complement and Irrelevant are considered irrelevant (negative class). As such, the task can be formulated as a binary classification problem, with the goal of comparing performance in terms of roc-auc. When calculating the ranking loss, the label Exact represents the highly relevant class, Substitute represents the low relevant class, and both Complement and Irrelevant represent the irrelevant class.

A.2 Experimental Details

Teacher Training: We conducted all experiments using the PyTorch framework (Paszke et al., 2019) and the HuggingFace library (Wolf et al., 2019). Two teacher models, namely S-BERT and S-DistilBERT, were employed with identical hyperparameter settings. S-BERT utilized the bert-base-uncased EN model (Devlin et al., 2018)[1], while S-DistilBERT utilized the distilbert-base-uncased EN model (Sanh et al., 2019)[2]. During the training phase, we employed pre-trained checkpoints and trained the models for 10 epochs, incorporating early-stopping criteria. A batch size of 512 and a learning rate of $5{*}10^{-5}$ were utilized with the Adam optimizer. We set the values of θ_U and θ_L to 0.85 and 0.7, respectively. The experiments were conducted on an AWS p2.8xlarge EC2 instance with a single GPU. Throughout this study, the hyperparameters were selected empirically based on the results obtained from various experiments.

KD-Boost Architecture Training: As outlined in Section 4.2, the weights of the trained teacher models are frozen. To train the student model (S-MiniLM), we utilize a pre-trained checkpoint from sentence-transformers/paraphrase-MiniLM-L3-v2 (Wang et al., 2020)[3] as a starting point and train the model for 10 epochs, employing early-stopping criteria. The hyperparameters used for training the S-MiniLM model are identical to those used for the teacher models. This includes a batch size of 512, a learning rate of $5{*}10^{-5}$, and the Adam optimizer. Similar to the teacher models, we set the values of θ_U and θ_L to 0.85 and 0.7, respectively. In Equation 7, we set γ to 0.9. To ensure statistical significance, we repeat the experiments 5 times while altering the random seeds.

[1]https://huggingface.co/bert-base-uncased

[2]https://huggingface.co/distilbert-base-uncased

[3]https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L3-v2

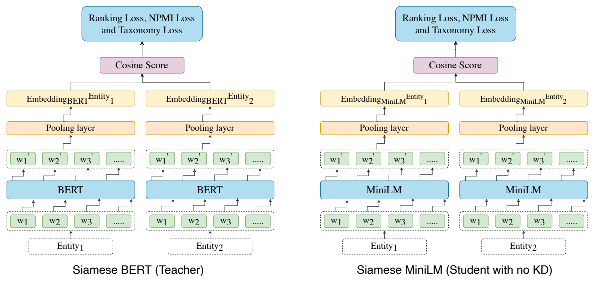
Ranking Loss, NPMI Loss and Taxonomy Loss
Cosine Score
$\text{Embedding}_{\text{BERT}}{}^{\text{Entity}_1}$
$\text{Embedding}_{\text{BERT}}{}^{\text{Entity}_2}$
Pooling layer
w1' w2' w3' ......
BERT
w1 w2 w3 ......
Entity1
Entity2
Siamese BERT (Teacher)
Ranking Loss, NPMI Loss and Taxonomy Loss
Cosine Score
$\text{Embedding}_{\text{MiniLM}}{}^{\text{Entity}_1}$
$\text{Embedding}_{\text{MiniLM}}{}^{\text{Entity}_2}$
Pooling layer
MiniLM
Siamese MiniLM (Student with no KD)

Figure 3: Teacher Model (S-BERT) and Low-Latency Encoder Model (S-MiniLM)

B Teacher and Student Models

The model architecture of both a teacher model and a low-latency student model is depicted in Figure 3.

C Realtime QR System Workflow

The QR (Query Reformulation) process enables the retrieval of relevant ads from the system based on the query $Q = q_1, q_2, ..., q_k$, where q_1,...,q_k represent different query reformulations. Our online QR system encompasses the following components:
(1) PCQC (Pre-Curated Query Cache) - The semantic representation of a pre-curated list of queries is generated by our proposed model and subsequently stored in a cache. These queries are carefully selected based on their historical performance, particularly the high number of products retrieved in the past.
(2) Query Processor - When a user requests a query, our proposed model converts it to a semantic representation in real-time.
(3) K-Nearest Neighbour (KNN) Search - Using KNN search on the semantic representation, the user query is matched to semantically similar queries, often referred to as reformulated queries, within the PCQC. The resulting reformulated queries are passed to the search index to return relevant products to customers.

D Ablation study

D.1 Robustness to misspelled user queries:

Given the lack of clarity and rarity of misspelled queries, semantic models often struggle to identify relevant products for such queries. Considering the superior performance of KD-Boost over teacher models on the overall test data, we also sought to evaluate its effectiveness specifically on data involving misspelled user queries. To accomplish this, we focused solely on samples from the test data that contained at least one misspelled word in the user query. Our experiments demonstrate roc-auc values of 0.9074 and 0.8973, respectively, when utilizing the KD-Boost and S-BERT teacher models on the misspelled query data. These results indicate that our proposed method exhibits a higher roc-auc than the teacher model, suggesting that KD-Boost is more resilient to incorrectly spelled queries. As a result, it can retrieve more relevant products even in instances where the queries contain misspellings. Table 4 provides examples of <query, product> pairs where our proposed model successfully retrieves products with misspelled queries, whereas the teacher model fails to do so.

Query	Product Title
jonk **hiyar** oil	Nature Sure™ Combo - Kalonji Tail 110ml and Jonk Tail (Leech Oil) 110ml
godex flashes	Godox Portable Lightweight Pocket Flash AD200
sheos combo packs	Ethics Perfect Combo Pack of 4 Loafer Shoes for Men

Table 4: Few <query, product> pairs involving misspelled user queries.

D.2 Query Length Level Results:

Table 5 showcases the auc results obtained by varying the number of tokens in a query. This analysis aims to assess the impact of increasing token length on the performance of our proposed method. Notably, for each token length, both KD-Boost variants, namely KD-Boost [Teacher:S-DistilBERT, Student:S-MiniLM] and KD-Boost [Teacher:S-BERT, Student:S-MiniLM], consistently outperform their respective teacher models, S-DistilBERT and S-BERT. However, it is worth noting that KD-Boost Multi Teacher surpasses all other single teacher KD-Boost variants in terms of performance.

D.3 Effect of γ (equation 7):

The KD-Boost loss, calculated by Equation 7, incorporates a weighting mechanism using γ and 1-γ for the loss functions. Our experimental findings indicate that maintaining a higher γ value, which assigns more weight to soft labels, leads to improved performance of the student model. Figure 4 presents the results of the KD-Boost method across different γ values ranging from 0 to 1. Notably, our

$Teacher_1 : S-DistilBERT, Teacher_2 : S-BERT, Student : S-MiniLM$						
#Tokens in Query	Student (No KD)	Teacher_1	KD-Boost Teacher_1	Teacher_2	KD-Boost Teacher_2	Multi-Teacher KD-Boost
1	0.9235	0.9297	0.9292	0.9368	**0.9374**	0.9370
2	0.9282	0.9344	0.9424	0.9403	0.9407	**0.9425**
3	0.9236	0.9337	0.9375	0.9427	0.9436	**0.9452**
4	0.9350	0.9434	0.9482	0.9511	0.9515	**0.9539**
5	0.9357	0.9431	0.9476	0.9498	0.9506	**0.9537**
>5	0.9461	0.9498	0.9547	0.9578	0.9574	**0.9592**

Table 5: Table comparing the ROC-AUC of different models based on the tokens in query text

findings reveal that our student model surpasses the S-BERT teacher model within a specific range of γ values, specifically between 0.8 and 0.9.

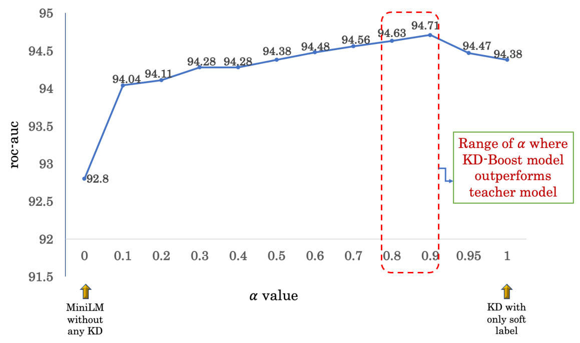

Figure 4: The figure illustrates the relationship between the increasing value of γ (as mentioned in equation 7) and the roc-auc performance. It is observed that the roc-auc steadily rises with an increasing γ value until it reaches a peak of 0.9, after which it starts to decline. These results are demonstrated using S-BERT as the teacher model. Similarly, when S-DistilBERT is employed as the teacher model, a similar trend is observed.

D.4 Is it possible to enhance efficiency by training the teacher and student models concurrently?

An alternative approach that can be considered is training the student and teacher networks simultaneously, eliminating the need for pretraining the teacher model and freezing its weights during the knowledge distillation process. However, our experimental findings reveal that co-training the teacher and student models results in a significant decline in the performance of both the teacher (SBERT AUC drop to 0.9258) and the student (AUC drop to 0.8945).

D.5 Results on tail queries:

Search engines typically classify queries into three categories: head, torso, and tail, with tail queries being less frequent. Our experimental results demonstrate that our proposed method outperforms the BERT teacher model for tail queries. Specifically, we achieve a roc-auc of 0.8621 with the KD-Boost student model and 0.8538 with the BERT teacher model for tail query human annotated samples. This indicates that our approach is more effective in addressing the unique challenges posed by tail queries.

E Query-Query positive pairs

Table 6 displays the outcomes of several positive query-query (QQ) pairs obtained through the application of Normalized Pointwise Mutual Information (NPMI) to customer purchase data. This data enables us to capture semantic relationships between entities, irrespective of whether they share any common terms.

Query1	Query2
bottle for kids	baby sipper
drawer lock	child safety lock
tv unit for living room	Low Height Television
wireless earbuds	boat airdopes
baking paper	butter paper for baking

Table 6: Examples of NPMI-identified semantically similar Q-Q pairs

F Query-Query Hard Negative Pairs

In Table 7, we showcase a collection of challenging hard negative query-query (Q-Q) pairs that were generated using taxonomy browse node information. This data enables us to effectively separate irrelevant Q-Q pairs within the embedding space, even when they share certain common words.

Query1	Query2
zoom camera	camera lens
pencil kit for girls	classmate gel pen
smart tv inch	dell 21.5 inch monitor
mens denim jeans	mens t shirt full sleeves

Table 7: Examples of hard negative Q-Q pairs generated using taxonomy browse node information