RTSM: Knowledge Distillation with Diverse Signals for Efficient Real-Time Semantic Matching in E-Commerce

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

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

Abstract

Semantic matching plays a pivotal role in e-commerce by facilitating better product discovery and driving sales within online stores. Transformer models have proven exceptionally effective in mapping queries to an embedding space, positioning semantically related entities (queries or products) in close proximity. Despite their effectiveness, the high computational demands of large transformer models pose challenges for their deployment in real-time scenarios. This paper presents **RTSM**, an advanced knowledge distillation framework designed for **R**eal-**T**ime **S**emantic **M**atching. Our approach develops accurate, low-latency student models by leveraging both soft labels from a teacher model and ground truth generated from pairwise query-product and query-query signals. These signals are sourced from direct audits, synthetic examples created by LLMs, user interaction data, and taxonomy-based datasets, with custom loss functions enhancing learning efficiency. Experimental evaluations on internal and external e-commerce datasets demonstrate a 2-2.5% increase in ROC-AUC compared to directly trained student models, outperforming both the teacher model and state-of-the-art knowledge distillation benchmarks.

1 Introduction

Precise real-time semantic matching, which involves the identification of semantic similar entities (e.g., queries or products) for a user query, has become increasingly crucial for e-commerce product search. In order to bridge the semantic gap between the user query and the semantic similar entities, this matching process typically performed in two ways, as depicted in Figure 1: **(1) Semantic Query Reformulation (SQR)**, where a user's poorly constructed query (e.g., containing code-mixed language or misspellings) is mapped to semantically similar, well-structured queries that produce a broader range of products. **(2) Semantic Product Retrieval (SPR)**, involving the retrieval of matching direct products for the given user query. In this paper, our focus lies in enhancing a real-time representation model for both query-query and query-product to enhance performance in both SQR and SPR tasks.

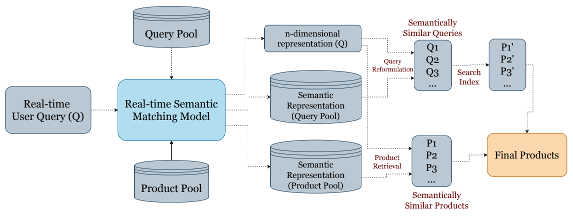
Query Pool
Real-time User Query (Q)
Real-time Semantic Matching Model
Product Pool
n-dimensional representation (Q)
Semantic Representation (Query Pool)
Semantic Representation (Product Pool)
Semantically Similar Queries
Query Reformulation
Q1
Q2
Q3
...
Search Index
P1'
P2'
P3'
...
Product Retrieval
P1
P2
P3
...
Semantically Similar Products
Final Products

Figure 1: A Semantic Matching Model that transforms a user query into an n-dimensional representation in real-time while precomputing embeddings for queries and products offline, enabling the retrieval of relevant products efficiently.

State-of-the-art (SOTA) approaches for semantic matching often utilize Siamese network architectures (Ranasinghe et al., 2019), which involve two identical sub-networks that generate semantic embeddings for query-query or query-product pairs. Transformer-based models such as BERT and DistilBERT (Devlin et al., 2018) (Sanh et al., 2019) have achieved outstanding results in this context. However, the high computational requirements of these models make them unsuitable for large-scale e-commerce applications, where latency under 5 milliseconds is paramount. On the other hand, smaller encoder models like 3 layers MiniLM (Wang et al., 2020), designed for low-latency scenarios, often underperform in terms of accuracy. A widely adopted solution to bridge this trade-off is knowledge distillation (KD) (Hinton et al., 2015), where a smaller student model learns from a larger teacher model using soft labels. While this approach enhances the performance of student models compared to direct training, the resulting models frequently fall short of the teacher model's accuracy and struggle to address its inherent errors. **Contributions.** This work focuses on tackling the challenge of real-time semantic matching in e-commerce by proposing an efficient KD framework, RTSM, which improves semantic matching for both query-query and query-product tasks. Our method leverages soft relevance labels from one or more teacher models alongside ground truth, allowing the student model to learn fine-grained insights while also correcting errors in the teacher model. Although e-commerce companies commonly utilize expert teams to annotate query-product pairs, ensuring the gradual accumulation of noise-free data, obtaining human-annotated Query-Query (Q-Q) data, crucial for SQR tasks, remains a significant challenge. To address this challenge and enhance semantic query reformulation (SQR) alongside semantic product retrieval (SPR), we leverage Large Language Models (LLMs) to generate precise Q-Q data, and also incorporating various sources of similarity and dissimilarity signals. Our key **contributions** include:

1. We propose a novel KD algorithm RTSM for real-time semantic matching that utilizes soft labels from one or more teacher models and ground truth to train an accurate student model. To meet the requirements of SQR and SPR tasks, we incorporate various similarity and dissimilarity signals, along with synthetic data generated from LLMs, and use customized loss functions to capture relevance and similarity nuances efficiently.

2. Extensive experiments on both internal and external e-commerce datasets demonstrate a 2-2.5% improvement in ROC-AUC for query-product relevance tasks over directly trained student models. The inclusion of LLM-generated query-query data significantly enhances query reformulation performance.

Note that our method can be used with any small encoder based models which support fast inferencing constraints under real-time semantic matching, and has wide applicability beyond product search.

2 Related Work

Semantic Matching: Transformer-based models, such as BERT (Devlin et al., 2018), DistilBERT (Sanh et al., 2019), have gained increasing popularity with the advancement of NLP tasks. Sentence-BERT (Reimers and Gurevych, 2019) develops upon the BERT algorithm by integrating a siamese network, typically employed for semantic matching tasks. However, this requires significant computational resources during inference, rendering it unsuitable for real-time applications. In an effort to reduce inference costs, several BERT variants have been suggested, such as PowerBERT (Goyal et al., 2020) and DistilBERT (Sanh et al., 2019). However, despite these innovations, these models are not optimal for real-time applications. MiniLM (Wang et al., 2020), a transformer-based model consisting of three layers, provides a less complex option than BERT and its variants. It is better suited for real-time applications due to its faster inference time, though its performance suffers due to the limited number of layers.

In Appendix E, Figure 3 shows the architecture of a teacher model, Siamese BERT (S-BERT), and a low latency model Siamese MiniLM (S-MiniLM).

Knowledge Distillation (KD): Several efforts have focused on knowledge distillation (KD) to enhance the efficacy of student models (Agrawal et al., 2025a), (Kim et al., 2021) (Agrawal et al., 2025b). The concept was introduced by Hinton et al. (Hinton et al., 2015), wherein the output of a complex network serves as a soft target for training a simpler network, facilitating the transfer of knowledge from complex to simple models. Consequently, KD has been widely adopted across various learning tasks (Yim et al., 2017; Chen et al., 2017). KD-Boost (Agrawal et al., 2023b) introduces a KD technique for real-time semantic sourcing, distilling BERT relevance knowledge into a low-latency MiniLM model. However, its performance on query reformulation (Agrawal et al., 2023a) task suffers due to limited query-query data. Advancements in LLMs offer potential for generating more query-query pairs. Our approach leverages these LLMs to produce effective query-query data, enhancing the real-time semantic matching model further.

3 Problem Statement

Our main objective is to enhance the performance of the student model in both semantic query reformulation (SQR) and semantic product retrieval (SPR) tasks by creating effective representations of queries and products within a shared semantic space, all while substantially reducing inference time. Achieving this versatility would enable us to minimize the expenses associated with maintenance and production.

We will now formally define the problem in terms of the four available input signals. **(i) human annotated labels on query-product pairs:** Let $D_{QP} = \{(q_i, p_i, y_i)\}_i$ denote human annotations on query-product pairs. Here, 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: (a) Strict relevant, (b) Standard relevant, or (c) Irrelevant. **(ii) Synthetic Data from LLMs:** LLMs have made synthetic data generation more accessible, significantly reducing the expertise and time required. With a user query q_i and a label y_i (relevant or irrelevant) provided through a prompt, we generate k reformulations and construct pairs $D_{QQ}^{LLMs} = (q_i, q_i^{'}, y_i)_i$ (see Section 4.1.2). **(iii) User Behavioral data:** Let $D_{QP}^{purchase} = \{(q_i, p_i, c_i)\}_i$ denote customer purchase behavior data. Here, c_i denotes the total number of purchases of product p_i after firing query q_i. While this data may be too noisy for direct modeling of query-product matches, it can be utilized to identify highly similar queries based on the overlap in associated product purchases. Specifically, we define the distribution over query pairs as the Gram matrix corresponding to the normalized query-product purchase counts and identify query pairs $D_{QQ+} = \{(q_i, q_i^{'})\}_i$ that exhibit significantly higher occurrence relative to random chance using Normalized Pointwise Mutual Information (NPMI)-based criteria (refer to Section 4.1.3). **(iv) Product Browse Taxonomy:** Given a set of queries and classifiers capable of mapping queries to a product browse taxonomy, one can determine the taxonomy labels for all queries and construct pairs $D_{QQ-} = \{(q_i, q_i^{'})\}_i$ with non-matching labels, which can be regarded as hard-negatives (refer to Section 4.1.4).

With these signals at hand, the aim is to train an effective model M so that for any user query q, product p, and another query q', the similarity of their corresponding embeddings $M(q)$, $M(p)$, $M(q')$ closely aligns with the relationships conveyed in the input signals.

4 Proposed Method

Our solution strategy involves two primary phases. Initially, we develop a teacher model considering the diverse signals outlined in Section 4.1. Subsequently, we train an effective student model, which not only replicates the soft labels of the teacher model but also integrates the original ground truth (Section 4.2). In Sec. 4.3, we elaborate on practical adjustments aimed at enhancing model efficacy.

4.1 Teacher Training Objective

During the training of the teacher model, we utilize human annotated query-product pairs D_{QP} as well as similar and dissimilar query-query pairs from the D_{QQ+}, D_{QQ-} and D_{QQ}^{LLMs} 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

In this step, we will use the data (D_{QP}) generated by human annotators who classify query-product pairs into three classes: i) Strict Relevant, ii) Standard Relevant, and iii) Irrelevant. We design our ranking loss (see eq 1) to leverage the ordinal nature of these ground truth labels. This gradation of relevance ensures that strictly relevant products are prioritized above standard relevant ones.

$$\begin{aligned} L_{\text{QP}} = \sum_{(q_i,p_i,y_i)\in D_{QP}} & (1_{y_i=strict}(\hat{y}_i - 1)^2 + \\ & 1_{y_i=standard}((min(0, \hat{y}_i - \theta_{smin}))^2 + \\ (max(0, \hat{y}_i - \theta_{smax}))^2) & + 1_{y_i=irrelevant}(max(\hat{y}_i, 0))^2) \end{aligned} \tag{1}$$

Here, θ_{smin} and θ_{smax} denote hyperparameters, $1_{y_i=.}$ is an indicator function, and $\hat{y}_i$ represents the model's prediction score.

4.1.2 Synthetic Generated data Loss

In section 4.1.1, we possess enough of noise-free human-annotated query-product pairs. However, acquiring query-query data presents a challenge in enhancing the model's performance for the semantic query reformulation task. Given the recent evolution of LLMs, which have emerged as a dominant and crucial tool for synthetic data generation, we aim to automatically reformulate user queries using LLMs, by prompting them with a carefully engineered prompt. Following this, the data (i.e., D_{QQ}^{LLMs}) is refined using a relevance model (see Appendix B.4) before being utilized in training both student and teacher models. Leveraging the D_{QQ}^{LLMs} dataset, we devise a loss function to delve into query-query semantics.

$$\begin{aligned} L_{QQ}^{LLMs} = \sum_{(q_i,q_i',y_i)\in D_{QQ}^{LLMs}} & 1_{y_i=1}\,(\min(0, \hat{y}_i - \theta_{smin}))^2 \\ & + 1_{y_i=0}\,(\max(\hat{y}_i, 0))^2 \end{aligned} \tag{2}$$

When $y_i = 1$, it denotes that the query and its reformulation is relevant, whereas $y_i = 0$ indicates that they are not relevant.

Further details on LLMs can be found in Appendix B.3, and the prompt for reformulating relevant user queries, inspired from (Yan et al., 2023), is outlined in Algorithm 1. We've adopted a few-shot learning approach, supplying a handful of query examples alongside their reformulations in the prompt.

4.1.3 User Behaviour Data Loss

Collecting human-annotated relevance data is both time-consuming and expensive. It's impractical to cover the entire semantic scope of e-commerce with audit data. Conversely, customer behavior data ($D_{QP}^{purchase}$), which includes implicit relevance signals, is abundant but noisy. To construct a robust relevance model, this data must be used alongside relevance audit data.
Lau et al. (Lau et al., 2014) utilized Normalized Point-wise Mutual Information (NPMI) to gauge topic co-occurrence, a method we employ to create semantically similar query pairs. We assess the likelihood of two queries co-occurring based on their individual probabilities and compare it to the scenario where the queries are independent. Normalizing the purchase count from $D_{QP}^{purchase}$ across queries allows us to derive a probability distribution. By examining their shared products, we can determine the joint distribution of any two queries. Utilizing this definition, we generate semantically similar query pairs, D_{QQ+}, from $D_{QP}^{purchase}$ data with NPMI scores exceeding τ_{npmi} (equation 3). Appendix D includes Table 6, provides examples of QQ positive pairs derived using this method.

$$NPMI(q_i, q_j) = \frac{log\frac{P(q_i,q_j)}{P(q_i)P(q_j)}}{-logP(q_i, q_j)} \qquad (3)$$

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 denote the total count of unique queries and products in $D_{QP}^{purchase}$. $PC(q_i, p_j)$ retrieves the purchase count from $D_{QP}^{purchase}$ for a specific query q_i and product p_j. With the utilization of D_{QQ+} data, we formulate the following loss function to acquire knowledge of query-query semantics.

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

Unlike the loss function described for standard relevant pairs in Equation 1, the cosine score in Equation 4 has no upper limit. The reasoning behind this loss function is that relevant query pairs within D_{QQ+} do not denote a particular level of relevance, whether standard or strict.

4.1.4 Taxonomy Based Loss

Most e-commerce companies structure their extensive product inventories using predefined multilevel taxonomies or browse nodes. These product taxonomies encode relationships between products and can be utilized to derive various connections. In this work, we utilize query classification models developed by various e-commerce companies, which assign a distribution score to a query based on the taxonomy tree. Consequently, two queries expressing different intents within the taxonomy tree will receive distinct scores. The appendix C contains Table 5 which provides some example cases of query-query (Q-Q) hard negative pairs that were generated using the approach. This dataset enables us to effectively distinguish irrelevant query-query pairs in the embedding space, even if they share some common words. Similar work conducted by the authors (Ankith et al., 2022) utilized the taxonomy and achieved success. We define taxonomy loss as follows, where D_{QQ-} represents the query-query hard negative dataset.

$$L_{QQ-} = \sum_{(q_i,q_i^{'})\in D_{QQ-}} (max(\hat{y}_i, 0))^2 \qquad (5)$$

4.1.5 Teacher Training

To develop semantic understanding within the teacher model, we initiate the process by initializing our BERT model with pre-trained weights. During the initial epochs, we utilize D_{QP} and D_{QQ+} to train the model parameters, optimizing the loss terms in equation 6. The relative importance of the loss terms L_{QP} and L_{QQ+} is controlled by α_1 and α_2, respectively.

$$L_1 = \alpha_1 * L_{QP} + \alpha_2 * L_{QQ+} \qquad (6)$$

In subsequent epochs, we also incorporate the other two losses, L_{QQ}^{LLMs} and L_{QQ-}, aiming to optimize in equation 7. Regarding L_{QQ-}, we generate hard negatives using a taxonomy tree encoding product relevance. For each epoch, we identify query pairs that are semantically similar but do not share a common browse node, which are then added to the dataset D_{QQ-} as hard negatives.

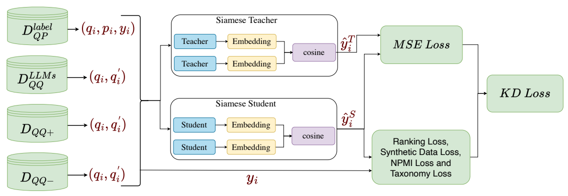

Figure 2: The training procedure for the student model adheres to the methodology outlined in our proposed approach, RTSM.

$$L_2 = \alpha_1 L_{QP} + \alpha_2 L_{QQ+} + \alpha_3 L_{QQ}^{LLMs} + \alpha_4 L_{QQ-} \quad (7)$$

Where α_3 and α_4 are the weight scalars that controls the importance of synthetic data loss and taxonomy loss.

4.2 Student Training using RTSM Method

Figure 2 showcases the framework of our proposed approach, which introduces a KD algorithm customized for real-time semantic matching. This approach leverages soft labels obtained from one or more teacher models, along with ground truth data, to enhance the accuracy of a precise student model. The formulation of our loss function for training the student model parameters is as follows:

$$L_{\text{RTSM}} = \beta \Bigg[\sum_{(q_i, p_i, y_i) \in D_{PQ}^{label}} (\hat{y}_i^T - \hat{y}_i^S)^2 + \sum_{(q_i, q_i^{'}) \in D_{QQ+} \cup D_{QQ-} \cup D_{QQ}^{LLMs}} (\hat{y}_i^T - \hat{y}_i^S)^2 \Bigg] + (1-\beta) L_2 \quad (8)$$

Where $\hat{y}_i^T$ signifies a soft label derived from the teacher model T, whereas $\hat{y}_i^S$ represents the prediction score from the student model S. The scalar β (where $0 < \beta < 1$) dictates the relative importance of soft and hard labels.

4.3 Practical Modifications

To improve the model's performance in practical applications, we implement several adjustments:
(1) Initially, during the teacher training phase outlined in Section 4.1.5, we train the model using Equation 6, followed by Equation 7. This sequential training approach ensures the stability of the model, enabling it to learn from the data consistently and effectively.
(2) Furthermore, we extend our approach to multi-teacher knowledge distillation, enabling the distillation of knowledge from multiple teachers simultaneously. This strategy, motivated by the aim to leverage diverse perspectives, enables the student model to access a wider range of insights and information. The multi-teacher RTSM algorithm integrates m soft labels through m MSE loss functions.

5 Experiments and Results

We report our findings on the benefit of our proposed method for real-time semantic matching tasks. We start by presenting the dataset details.
Datasets: 1. E-commerce datasets from regions in India for evaluating query-product relevance. All datasets used in our analysis are anonymized, aggregated, and do not represent production distribution. **2.** The publicly available ESCI dataset from Amazon for the US (English) market. More details on the generation and construction of these datasets can be found in Appendix A.
Reproducibility and Hyperparameters: For details regarding the reproducibility of our experiments and the hyperparameter configurations, please refer to Appendix B.

5.1 Algorithm Baselines

In this paper, our proposed method is compared against several baselines, all of which are trained on the same dataset to ensure equitable comparison.
(i) DSSM-KD (Nigam et al., 2019) involves training the low-latency DSSM model using soft labels derived from the SBERT model.
(ii) S-MiniLM Direct (Wang et al., 2020) involves direct training of the S-MiniLM model without employing any KD.
(iii) Soft-KD (Hinton et al., 2015) focuses on training the S-MiniLM model exclusively using soft labels obtained from a teacher model.
(iv) HISS (Ankith et al., 2022) introduces a KD method for real-time semantic matching, incorporating an additional alignment loss.
(v) Teacher-only (Devlin et al., 2018): Teacher model undergoes direct training using a training dataset.
(vi) Ensemble Baseline is evaluated within context of our proposed Multi-teacher KD method, which combines multiple teachers into an ensemble.
Evaluation Metric We use ROC-AUC (Brown and Davis, 2006) as a performance metric.

5.2 Results

We present the outcomes of our proposed technique on an **proprietary Amazon dataset** in Ta-

Model	ROC-AUC/Gain%	Precision/Recall/F1
DSSM-KD	0.8759(±0.0008) / 0%	0.9516/0.7931/0.8651
S-MiniLM Direct	0.9252(±0.0005) / 5.63%	0.9736/0.8063/0.8820
Teacher: S-DistilBERT, Student: S-MiniLM		
Teacher-only	0.9410(±0.0011) / 7.43%	0.9780/0.8265/0.8958
Soft-KD	0.9353(±0.0008) / 6.78%	0.9778/0.8120/0.8872
HISS	0.9386(±0.0013) / 7.16%	0.9801/0.8033/0.8829
RTSM	**0.9437**(±0.0006) **/ 7.74%**	**0.9805/0.8295/0.8987**
Teacher: S-BERT, Student: S-MiniLM		
Teacher-only	0.9471(±0.0005) / 8.13%	0.9816/0.8297/0.8982
Soft-KD	0.9378(±0.0009) / 7.07%	0.9782/0.8286/0.8972
HISS	0.9457(±0.0010) / 7.97%	0.9802/0.8276/0.8974
RTSM	**0.9482**(±0.0005) **/ 8.25%**	**0.9818/0.8367/0.9034**
Multi-Teachers, Student: S-MiniLM		
Ensemble	0.9483(±0.0006) / 8.27%	0.9809/0.8369/0.9031
Soft-KD	0.9420(±0.0012) / 7.55%	0.9794/0.8264/0.8964
HISS	0.9424(±0.0009) / 7.59%	0.9814/0.8313/0.9001
RTSM	**0.9502**(±0.0007) **/ 8.48%**	**0.9820/0.8427/0.9070**

Table 1: ROC-AUCs for several models on proprietary D_{QP} test dataset. Precision, Recall, and F1 scores are calculated at a threshold of 0.7. As DSSM-KD acts as the baseline, the gain% remains at 0. In the Multi-teachers section, "Ensemble" denotes the combined performance of several teachers. Mean & std. (±) error for ROC-AUCs are reported based on 5 trials runs.

Model	ROC-AUC / Gain%	Precision / Recall / F1
DSSM-KD (baseline)	0.8457(±0.0011) / 0%	0.9326 / 0.7634 / 0.8396
S-MiniLM Direct	0.8738(±0.0008) / 3.19%	0.9432 / 0.7712 / 0.8485
Teacher: S-BERT, Student: S-MiniLM		
Teacher-only	0.8881(±0.0007) / 4.93%	0.9487 / 0.7768 / 0.8542
Soft-KD	0.8778(±0.0008) / 3.71%	0.9442 / 0.7738 / 0.8505
HISS	0.8828(±0.0010) / 4.30%	0.9468 / 0.7755 / 0.8526
RTSM	**0.8922(±0.0006) / 5.00%**	**0.9536 / 0.7842 / 0.8606**

Table 2: AUC scores on Amazon Shopping Public Dataset. Precision, Recall, and F1 scores are calculated at a threshold of 0.7. Mean & std. (±) error for ROC-AUCs are reported based on 5 trials runs.

ble 1, comparing it with both the existing production model (DSSM-KD) and strong SOTA baseline methods. We demonstrate the effectiveness of our approach employing two distinct teacher models, namely S-BERT and S-DistilBERT. Furthermore, we utilize S-BERT and S-DistilBERT to verify the efficacy of multi-teacher RTSM algorithm. Our experiments reveal that our approach achieves superior performance compared to all baseline methods, notably surpassing the IN production model by a significant margin. The summarized results for the **External Amazon Shopping Dataset** are presented in Table 2, with the S-BERT model acting as the teacher model. When evaluated against all baseline approaches, our method emerges as the superior option, outperforming them by a significant margin, thereby demonstrating its dominance over the current state-of-the-art techniques. For an in-depth **latency evaluation** of our models in an online context, refer to Appendix G. Furthermore, for a detailed analysis of how the losses L_{QQ}^{LLMs}, L_{QQ+}, and L_{QQ-} affect model performance, refer to Appendix F.

5.3 Simulated Realtime A/B Experiments

To evaluate the efficacy of our proposed approach, we conducted a simulated A/B test on real-time SQR (refer to Section H for SQR system). we assessed the performance of the A/B test based on two primary metrics: **(i) Increase in product coverage:** An increase in product coverage is achieved by showing more relevant products in response to user queries. **(ii) Reduction in irrelevancy:** A sample of impressed query and product title pairs is sent for human labeling, where they are classified as strictly relevant, standard relevant, or irrelevant. Reducing the number of irrelevant classifications decreases overall irrelevancy. Our proposed approach exhibited a notable enhancement in product coverage along with a reduction in irrelevancy.

6 Conclusion

In this paper, we introduce a KD approach for real-time semantic matching, where siamese student models acquire nuanced semantic representations by emulating both (i) the soft relevance labels from the siamese teacher model and (ii) the hard relevance labels annotated by humans. To address the needs of query reformulation and product retrieval tasks, we integrate a variety of similarity and dissimilarity signals, along with synthetic data generated from LLMs, and employ tailored loss functions to efficiently capture relevance and similarity intricacies. By leveraging both internal and public datasets, we demonstrate the superior effectiveness of our proposed method compared to existing SOTA KD benchmarks.

References

Sanjay Agrawal, Faizan Ahemad, and Vivek Varadarajan Sembium. 2025a. Rationale-guided distillation for e-commerce relevance classification: Bridging large language models and lightweight cross-encoders. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 136–148.

Sanjay Agrawal, Srujana Merugu, and Vivek Sembium. 2023a. Enhancing e-commerce product search through reinforcement learning-powered query reformulation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4488–4494.

Sanjay Agrawal, Deep Nayak, and Vivek Varadarajan Sembium. 2025b. Multilingual continual learning using attention distillation. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 91–99.

Sanjay Agrawal, Vivek Sembium, and MS Ankith. 2023b. Kd-boost: Boosting real-time semantic matching in e-commerce with knowledge distillation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 131–141.

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.

Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. 2017. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems*, 30.

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*.

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).

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*.

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.

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.

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*.

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*.

Wenfeng Yan, Shaoxiang Chen, Zuxuan Wu, and Yu-Gang Jiang. 2023. Prompting large language models to reformulate queries for moment localization. *arXiv preprint arXiv:2306.03422*.

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.

A Dataset Generation

1. Proprietary Amazon Dataset: We gathered customer behavior data, denoted as $D_{QP}^{purchase}$, from historical logs of the IN marketplace spanning from January 2024 to June 2024. To ensure data quality, pairs with fewer than 15 purchases were filtered out. For data generation based on taxonomy, we utilized an internal service to acquire browse node associations for 200K randomly chosen queries from the $D_{QP}^{purchase}$ dataset. Subsequently, D_{QQ-} was generated using browse node mappings to maintain the separation of irrelevant query-query pairs within the embedding space. For D_{QQ}^{LLMs}, we compiled a dataset comprising 200k search queries. We used an Instruct LLM to generate the top 10 positive and negative reformulations for each user query. Additionally, we utilized a relevance model (see Section B.4 on relevance model details) and browse node associations to refine D_{QQ}^{LLMs} further. Regarding the D_{QP} dataset, we collected a sample of 5.6 million human-annotated <query, product title> pairs from five English-speaking marketplaces. Since our experiments focus on the Indian marketplace, we constructed validation and test datasets by randomly selecting 50K query-ad pairs each from the IN marketplace, removing these 100K pairs from training. In our performance evaluation, strict and standard relevance are treated as positive classes, while irrelevance is considered a negative class.

2. Aicrowd ESCI Amazon Public Dataset: This dataset contains 460K training samples and 91K test samples. For validation and test, 20% of the training data (10% each) is randomly selected and removed from the training set. Each query-product pair is labeled as E (Exact), S (Substitute), C (Complement), or I (Irrelevant). In the search context, pairs labeled as Exact and Substitute are considered relevant (positive class), while those labeled as Complement and Irrelevant are considered irrelevant (negative class). This can be framed as a binary classification problem, where the goal is to evaluate the performance using ROC-AUC.

B Reproducibility and Hyperparameters

In this section, we present the hyperparameters and training methodologies used in our experiments. All experiments are conducted using PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2019) frameworks. We use a consistent set of hyperparameters for both the Teacher and Student models during training, which were optimized through a series of preliminary trials and are detailed in Table 3. Further details on the training of the Teacher and Student (RTSM) models are provided in subsections B.1 and B.2, respectively, with model specifications outlined in Table 4. Additional information on using LLMs for query generation is available in subsection B.3.

Hyperparameter	**Value**
Batch Size	256
Learning Rate	1e-5
Number of Epochs	5
Weight Decay	0.0
Adam ϵ	1e-8
Adam β_1	0.9
Adam β_2	0.999
Gradient Clipping	0.1
θ_{smax}	0.75
θ_{smin}	0.6
GPU	p3.2xlarge EC2

Table 3: Hyperparameters used for training the models.

B.1 Teacher Training:

Two teacher models, namely S-BERT and S-DistilBERT, are employed, both utilizing identical hyperparameter configurations. S-BERT employs the bert-base-uncased[1] EN model, while S-DistilBERT utilizes the distilbert-base-uncased[2] EN model (Sanh et al., 2019). During the training phase, we leverage pre-trained checkpoints and train for 5 epochs with early-stopping criteria.

B.2 RTSM Architecture Training:

As outlined in Section 4.2, we have frozen the weights of the trained teacher models. To facilitate the training of a student model (S-MiniLM), we initialize with a pre-trained checkpoint from sentence-transformers/paraphrase-MiniLM-L3-v2[3] (Wang

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

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

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

Model	**Variant**	**Layers**	**Hidden Size**	**Parameters**
S-BERT	bert-base-uncased	12	768	110M
S-DistilBERT	distilbert-base-uncased	6	768	66M
S-MiniLM	paraphrase-MiniLM-L3-v2	3	384	22M

Table 4: Details of the Models Used in the Experiments

et al., 2020), and conduct training for 5 epochs, employing early-stopping criteria.

B.3 LLM-Based Query Generation Model:

To generate queries using LLMs, we utilized an Instruct model, which is available under the Apache 2.0 license. The open-source nature and permissive licensing of instruct model allow other researchers to use it in their work. For generating semantically similar queries, we applied the prompt template outlined in Algorithm 1.

B.4 Relevance Model for of LLM-Generated Query Reformulations:

The purpose of the relevance model is to evaluate the quality of a reformulated query $q_i^{'}$, generated by an LLM, based on the input query q_i. We developed a relevance model based on bert-base-uncased[4] with 12 transformer layers, pre-trained on English. It was fine-tuned on our dataset of human judgments, consisting of triplets $\{(q_i, q_i^{'}, y_i)\}_i$, where y_i represents the human judgments provided by annotators. We employed binary cross-entropy as the loss function. The scores provided by the trained relevance model is used to evaluate the quality of the generated reformulations.

C Hard Negative Q-Q Pairs from Taxonomy Browse Nodes

Table 5 showcases a set of challenging hard negative query-query (Q-Q) pairs, generated by utilizing taxonomy browse node information. This method enables the efficient distinction of irrelevant Q-Q pairs within the embedding space, despite the presence of shared common terms between the queries.

D NPMI-based Query-Query Pairs using Customer Purchase Data

Table 6 presents the results of various positive query-query (Q-Q) pairs derived by applying Normalized Pointwise Mutual Information (NPMI) on customer purchase data. This approach allows capturing semantic associations between entities, even if they do not share any common terms.

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

Query1	Query2
watch band	smart watch
laptop sleeve	long-sleeve sweater
black shoes	shoe rack
digital camera	camera lens filter
cotton bedsheet	cotton candy maker

Table 5: Instances of hard negative Q-Q pairs produced utilizing taxonomy browse node information.

Query1	Query2
travel backpack	outdoor backpack
wireless mouse	cordless computer mouse
fitness tracker	activity monitor
portable charger	mobile power bank
travel pillow	neck support cushion

Table 6: Instances of Q-Q pairs identified as semantically akin through NPMI analysis.

E Teacher and Student Model Architectures: S-BERT vs. S-MiniLM

Figure 3 illustrates the model architecture for two different models: a teacher model and a low-latency student model.

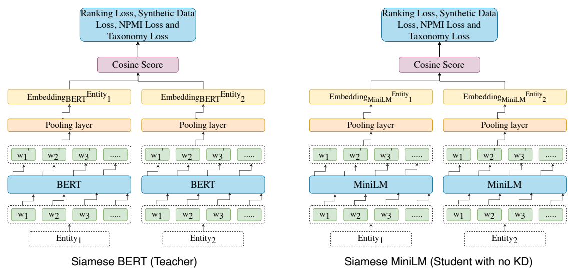
Ranking Loss, Synthetic Data Loss, NPMI Loss and Taxonomy Loss
Cosine Score
$\text{Embedding}_{\text{BERT}}{}^{\text{Entity}_1}$
$\text{Embedding}_{\text{BERT}}{}^{\text{Entity}_2}$
Pooling layer
$w_1^{'}$ $w_2^{'}$ $w_3^{'}$ .....
BERT
w_1 w_2 w_3 .....
Entity_1
Entity_2
Siamese BERT (Teacher)
Ranking Loss, Synthetic Data 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: Model Architectures

Model	roc-auc	Q-Q Irrelevance
S-BERT w/o	0.9492	21.5%
S-BERT w/	0.9471	9.9%
S-MiniLM w/o	0.9287	23.8%
S-MiniLM w/	0.9252	11.3%
RTSM w/o	0.9534	19.8%
RTSM w/	0.9482	8.2%

Table 7: ROC-AUCs and Q-Q irrelevance statistics of different models with (w/) and without (w/o) all L_{QQ}^{LLMs}, L_{QQ+} and L_{QQ-} losses.

F Combined Impact of Losses L_{QQ}^{LLMs}, L_{QQ+} and L_{QQ-}

We gathered a total of 15K Q-Q samples, which underwent auditing by our in-house human auditing team. Table 7 illustrates the AUCs of various models on test datasets (human-audited query-product pairs) with (w/) and without (w/o) L_{QQ}^{LLMs}, L_{QQ+} and L_{QQ-}, alongside Q-Q irrelevance statistics for the 15K audited Q-Q samples. Our examination indicates that optimizing for these three losses leads to a slight reduction in the AUC but significantly diminishes Q-Q irrelevance. Maintaining low Q-Q irrelevance is critical as query reformulation relies on retrieving products from other queries that are semantically similar.

G Latency Within an Online Context

We evaluated the retrieval latency of BERT, DistilBERT, and MiniLM models for embedding-based semantic matching in an online environment. To accomplish this, we developed all models using PyTorch and then converted them to ONNX format (Bai et al., 2019). In the online scenario, we utilized Java deep library to load the ONNX models and generated embeddings for user queries. Using the HNSW library (Malkov and Yashunin, 2018) with parameters mlinks=32 and ef_construction=128, we performed real-time mapping of user queries to the k-nearest neighbor products (k=200). The latency analysis was conducted by measuring the average retrieval time for 10,000 queries using only CPU cores (on m5.4xlarge instance). According to our findings, BERT and DistilBERT exhibited higher inference latencies of 10.24ms and 6.23ms, respectively, compared to MiniLM's latency of 1.17ms.

H Current SQR System Deployed in IN Marketplace

In the IN marketplace, our current real-time SQR semantic strategy relies on DSSM, trained with Knowledge Distillation utilizing Siamese BERT. Through SQR, our system surfaces relevant ads corresponding to a query $Q = q_1, q_2, ..., q_k$, where q_1,...,q_k represent query reformulations. Our online SQR system comprises:

(1) PCQC (Pre-Curated Query Cache) - Our proposed model generates semantic representations for a pre-curated list of queries and stores them in a cache. These queries are curated based on past instances where a high number of products were retrieved for them.

(2) Query Processor - Upon a user query request, our proposed model converts it into a semantic representation in real-time.

(3) K-Nearest Neighbor (KNN) Search - The user's query undergoes matching against semantically similar queries (reformulated queries) in PCQC using KNN search based on their semantic representations. The resulting reformulated queries are then utilized to retrieve relevant products from the search index for customers.

Algorithm 1 Prompt for Reformulations Generations

```
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.
### Instruction:
You are Sam, a super intelligent assistant that help users reformulate a search query of an e-commerce website.
Your reformulated query will be used by a product sourcing assistant, who sources products based on the search query. The more informative, legible, human interpretable the reformulated query is, better products will be sourced. Your task is to maximize the efficiency so that better products are sourced.
You are
- helpful and friendly
- can easily correct grammatical and language errors
- good at understanding the search query's intent and extract the core meaning hence reformulating it to a better query
- make sure that the output queries are strictly relevant to the input search query and Fenix has no difficulty in interpreting query
- strictly output only the reformulated query
You have to output 10 reformulated queries for a given search query in decreasing order of relevance to the search query. Make sure all of the reformulated queries are highly relevant to the search query.
Here are some examples:
Example 1:
query: headset below 1000
output: headphone under 2000
Example 2:
query: 3 years girls dresses modern
output: baby girls 3-4 years dress
Example 3:
query: kitchen decoration saman
output: home decor items for kitchen
Example 4:
query: dog chain+belt for large dogs
output: dog chain collar
Example 5:
query: men gift for man
output: wallet set for men gift
Example 6:
query: jewllwey set for girls simple
output: set jewellery for girls stylish
Example 7:
query: caramboard for kids avanzure pic
output: gift for girls 10 years
Example 8:
query: mala
output: laddu gopal mala
Now reformulate this query: "{User_query}"
Output 10 reformulated queries for a given search query. Strictly output only the reformulated queries in order 1 to 10. Do not include any explanation or any other stuff in your response.
### Response:
```