

CREST: Improving Interpretability and Effectiveness of Troubleshooting at Ericsson through Criterion-Specific Trouble Report Retrieval

Soroush Javdan

Carleton University,, soroushjavidan@cmail.carleton.ca Ottawa, Canada

Ericsson Canada Inc., soroush.javidan@ericsson.com Ottawa, Canada

Pragash Krishnamoorthy

Ericsson Canada Inc., pragash.krishnamoorthy@ericsson.com Ottawa, Canada

Olga Baysal

Carleton University, olga.baysal@carleton.ca Ottawa, Canada

Abstract

The rapid evolution of the telecommunication industry necessitates efficient troubleshooting processes to maintain network reliability, software maintainability, and service quality. Trouble Reports (TRs), which document issues in Ericsson’s production system, play a critical role in facilitating the timely resolution of software faults. However, the complexity and volume of TR data, along with the presence of diverse criteria that reflect different aspects of each fault, present challenges for retrieval systems. Building on prior work at Ericsson, which utilized a two-stage workflow, comprising Initial Retrieval (IR) and Re-Ranking (RR) stages, this study investigates different TR observation criteria and their impact on the performance of retrieval models. We propose **CREST** (Criteria-specific Retrieval via Ensemble of Specialized TR models), a criterion-driven retrieval approach that leverages specialized models for different TR fields to improve both effectiveness and interpretability, thereby enabling quicker fault resolution and supporting software maintenance. CREST utilizes specialized models trained on specific TR criteria and aggregates their outputs to capture diverse and complementary signals. This approach leads to enhanced retrieval accuracy, better calibration of predicted scores, and improved interpretability by providing relevance scores for each

criterion, helping users understand why specific TRs were retrieved. Using a subset of Ericsson’s internal TRs, this research demonstrates that criterion-specific models significantly outperform a single model approach across key evaluation metrics. This highlights the importance of all targeted criteria used in this study for optimizing the performance of retrieval systems.

Keywords:

Trouble report, software maintenance, bug reports, information retrieval, neural ranking, natural language processing, telecommunications

1. Introduction

The modern telecommunication industry is a dynamic, rapidly evolving field that relies heavily on efficient troubleshooting processes to preserve network reliability while ensuring customers receive high-quality service.

At Ericsson, a trouble report (TR) is a critical tool for tracking information regarding the detection, characteristics, and eventual resolution of problems. TRs document issues and incidents that arise during the development and maintenance phases of the software products at Ericsson.

Each TR consists of multiple sections that define its characteristics. A TR typically records a *headline* which serves as a short summary of the trouble, a *priority* tag, the *responsible team*, the *product under test*, and the *fault category* which indicate the defect’s type to aid TR triage, among others. However, the most crucial section is *the observation section*, which serves as a detailed explanation of the fault written by the creator of the TR. This section provides essential information for the designated team to promptly resolve the issue. The observation section is a free-text format, allowing the reporter to provide a comprehensive description of the problem. The **TR observation** includes detailed information, called criteria, describing the fault by providing all the necessary information for the fast resolution of the fault. Each of these criteria focuses on a different aspect of the fault. Common criteria include *general description* of the fault, *conditions* under which the fault occurred, its *impact* on the system, the *frequency* of the fault and Steps to *reproduce* the fault. These criteria are considered the most crucial based on feedback from domain experts. According to their input, these criteria most often anchor discussions between reporters and responsible teams during triage and resolution. They also appear with high frequency in recent TR templates.

To ensure the quality of TRs, they must follow a standard template and meet defined quality standards before being published for resolution. The resolution of a fault (called the answer) is only accepted after it has been validated. Previously resolved TRs (historical data) play a critical role as the information they contain can be used by quality assurance (QA) testers to improve the quality of new TRs. By quickly identifying similar past TRs and providing relevant information to designated teams, historical data enables efficient troubleshooting. Moreover, the design team can use these relevant resolved TRs during the problem analysis and correction phases, which can further reduce the time spent on debugging and testing. As a result, implementing a high-performance TR retrieval system to extract similar, previously resolved TRs can speed up software development and maintenance at Ericsson.

In recent years, there has been a notable increase in the efforts of applying machine learning (ML) and natural language processing (NLP) methods across various domains, including the telecommunication industry. Large language models (LLMs) such as bidirectional encoder representations from transformers (BERT) (Devlin et al., 2019) have shown great potential in leveraging textual data to address various tasks. Ericsson researchers proposed BERTicsson (Grimalt et al., 2022) as a BERT-based TR recommender system. It combines two stages: initial retrieval (IR) and re-ranking (RR). BERTicsson also uses the Sentence-BERT (Reimers and Gurevych, 2019) and monoBERT (Nogueira and Cho, 2019) architecture to extract and rank a subset of historical TRs based on their relevance to new TRs. Prior work by Bosch et al. (2022) uses a similar architecture to extract and identify duplicate TRs. Prior works have also combined the headline with the observation as inputs. However, the observation was treated as one unstructured block, without parsing or modeling at the level of individual criteria, which meant its internal signals were neither isolated nor weighted explicitly. Building on the foundational work (Grimalt et al., 2022; Bosch et al., 2022) that pioneered TR retrieval studies at Ericsson, we leverage an internal LLM, specifically RoBERTa, that was trained on the internal and external telecommunication data within Ericsson.

While existing models have advanced TR retrieval, the inherent complexity and diversity of TR observations suggest there is still room for enhancement. Each TR is composed of multiple sections, with the *observation section* standing out as a particularly information-rich component. This section encompasses various criteria, such as the *trouble description*, *impact on system*,

and *other criteria*, each contributing differently to the quality of retrieval. Understanding how these individual components influence retrieval outcomes is essential for identifying areas of improvement and guiding model development. Inspired by frameworks like Branch Train Merge (BTM) (Li et al., 2022) and DEMIX layers (Gururangan et al., 2022), and DORIS-MAE (Wang et al., 2023), we introduce the Criteria-specific Retrieval via Ensemble of Specialized TR models (CREST). Similar to how DORIS-MAE demonstrates the benefit of decomposing complex queries into aspects, and how BTM and DEMIX leverage expert models for modular learning, CREST trains separate models for each TR criterion. These specialized models are then combined through a weighted ensemble to improve retrieval performance.

Beyond aiming to improve retrieval performance, CREST also tackles one of the key limitations of many existing methods, which is the lack of transparency in how recommendations are generated. CREST tackles this issue by calculating separate relevance scores for each TR criterion and then combining them to determine the final ranking. This approach is designed to support interpretability¹ by making the influence of each TR component explicit, enabling users to better understand how recommendations are formed. In essence, CREST is designed with interpretability at its core, making the contribution of each criterion both clear and measurable.

To further explore CREST’s interpretability, we investigate how well its generated detailed relevance scores align with true relevance judgments by analyzing confidence calibration. Poorly calibrated scores can undermine trust in retrieval systems, even when accuracy is high. Therefore, we assess the degree to which CREST’s predicted relevance scores accurately reflect the likelihood of a correct retrieval. This calibration analysis is especially important in real-world practice, such as Ericsson, where confidence scores can guide decisions.

Finally, to ensure that improvements extend beyond offline evaluation, we complement our experiments with a pilot user study at Ericsson. The study examines whether CREST’s criterion-wise scores enhance the transparency of the rankings and whether the recommended TRs are perceived as credible and practically useful during triage, rather than merely appearing more accurate

¹In this paper, we use the terms interpretability, explainability, and transparency interchangeably to refer to the extent to which a model’s retrieval behavior can be understood and justified by human users.

in offline metrics.

In summary, our analysis shows that observation criteria contribute unevenly across different retrieval stages. Building on this, our criterion-specific ensemble (CREST) consistently outperforms single-model approaches in the two-stage workflow and remains superior even when evaluated in isolation. Moreover, CREST improves confidence calibration and provides criterion-wise relevance scores, making recommendations both more reliable and more interpretable for engineering decision-making.

Using Ericsson’s internal trouble-reporting dataset, we answer the following research questions:

1. RQ1: *What impact does each TR criterion have on retrieval model performance?*
2. RQ2: *To what extent does CREST improve the retrieval performance compared to a single model?*
3. RQ3: *How does the calibration of relevance scores produced by CREST compare to those generated by the criterion-agnostic model?*
4. RQ4: *What are the users’ perceptions and rankings of CREST in terms of transparency, usefulness, credibility, and accuracy?*

The remainder of this paper is organized as follows: Section 2 highlights background relevant to this study and summarizes related work. Section 3 presents the details of the study approach and includes an overview of retrieval systems, data related to TRs, the initial retrieval (IR) stage, the re-ranking (RR) stage, training, and inference. Section 4 presents the evaluation procedure, including evaluation strategy, metrics, and datasets. Section 5 reports the results of our empirical study. Finally, Section 7 concludes the paper by discussing possible future research directions.

2. Background and Related Work

Since we leverage Large Language Models (LLMs) in this work, this section provides an overview of LLMs, highlighting their strengths and limitations, as well as their application within the telecommunications industry. Additionally, it discusses recent research on the use of LLM-based models for text retrieval problems.

2.1. Large Language Models

Natural Language Processing (NLP) has been reshaped by the rise of large language models (LLMs), which excel at tasks such as text classification, summarization, and semantic understanding. Cutting-edge models like the GPT series (Achiam et al., 2023), LLaMA3 (Grattafiori et al., 2024), DeepSeek (Liu et al., 2024), Mistral (Jiang et al., 2023), and Qwen (Yang et al., 2024) are now increasingly integrated into retrieval pipelines, serving roles as encoders, re-rankers, or generative reasoners. In parallel, classic compact transformers including BERT (Holm, 2021), RoBERTa (Liu et al., 2019), T5 (Raffel et al., 2020), and ERNIE (Zhang et al., 2019) remain widely deployed in production due to strong efficiency–accuracy trade-offs and mature tooling (Puthenpuhussery et al., 2025).

In this study, we focus on Ericsson’s domain-trained RoBERTa (TeleRoBERTa) for two main reasons. TeleRoBERTa, trained on proprietary telecom corpora, matches the performance of much larger foundation LLMs on telecom standards QA benchmarks while using an order of magnitude fewer parameters. This makes it particularly well aligned with the distributions found in telecom-specific tasks (Karapantelakis et al., 2024). In addition, our objective is criterion-aware retrieval that is both efficient and interpretable, and that can be immediately integrated into Ericsson’s existing two-stage pipeline. Fine-tuning a compact, domain-adapted encoder not only reduces compute and latency overhead but also enables drop-in improvements within the existing production stack.

2.2. Neural Retrieval Models

Neural retrieval models (NRMs) for document search largely fall into two families: *cross-encoders* and *bi-encoders*. Cross-encoders (e.g., monoBERT (Nogueira and Cho, 2019)) score a query–document pair jointly, yielding strong relevance estimates but high computational cost; multi-stage pipelines therefore retrieve with BM25 and re-rank with monoBERT or duoBERT’s pairwise comparator (Nogueira et al., 2019). To reduce cost, bi-encoders encode queries and documents independently (e.g., TwinBERT (Lu et al., 2020) with a Siamese setup (Chicco, 2021)), enabling offline document embeddings for fast initial retrieval; late-interaction models like ColBERT (Khattab and Zaharia, 2020) bridge the two by combining separate encodings with token-level matching. Within Ericsson, BERTicsson (Grimalt et al., 2022) applied this multi-stage recipe to recommend solutions for new TRs, outperforming

BM25 (Robertson and Zaragoza, 2009). Related work on duplicate TR detection (Bosch et al., 2022) explored domain adaptation (Ruder et al., 2019), sequence learning (Ruder et al., 2019), multi-stage fine-tuning, and elastic weight consolidation (Kirkpatrick et al., 2017) to mitigate catastrophic forgetting.

All prior systems treat a TR observation as a single block of text. This monolithic view obscures criterion-specific signals and limits interpretability of relevance scores. These gaps motivate our approach which explicitly models observation criteria and aggregates their signals to improve both retrieval quality and transparency.

2.3. Transparency and Explainability in Neural Retrieval Models

While neural retrieval models have achieved impressive performance across a variety of information retrieval tasks, they often operate as black boxes, offering limited insights into how specific relevance decisions are made (Rudin, 2019). This lack of transparency poses significant challenges in domains such as software maintenance, where practitioners require justifiable and interpretable recommendations to support critical decision-making.

Broadly, approaches to explainability in neural retrieval can be grouped into two categories: post-hoc and design-time strategies (Anand et al., 2022). Post-hoc methods attempt to interpret trained models by analyzing model behaviour or internal representations after training is complete. These include techniques such as feature attribution (Wang et al., 2024; Zhang et al., 2020), attention mechanism (Lucchese et al., 2023), and representation probing (Wallat et al., 2023), each aimed at uncovering which input features or structures most influence the model’s outputs. While insightful, these approaches often operate heuristically and may not yield stable or actionable explanations across instances or domains.

On the other hand, models that explicitly integrate explainability into their architecture offer a more systematic path toward interpretability. These models often utilize modular structures or enforce architectural constraints to ensure that each part of the decision process can be traced and understood. In the context of neural retrieval, such approaches may involve segmenting the relevance estimation process into interpretable components aligned with domain-specific dimensions of the input (Yu et al., 2022; Leonhardt et al., 2023).

In the context of TR retrieval, we adopt a similar strategy through the proposed CREST model, which generates criterion-specific relevance scores

that are combined into a final ranking. This design improves both retrieval accuracy and interpretability, allowing users to trace each recommendation back to distinct TR criteria. In this way, CREST promotes explainability by design, ensuring that each component’s contribution to the final output is both meaningful and measurable.

2.4. Confidence Calibration of Neural Ranking Models

Neural rankers often output relevance scores that are not well-aligned with the actual likelihood of relevance (Penha and Hauff, 2021a). While these scores can be effective for sorting documents, the lack of calibration can limit their interpretability and downstream utility, especially in settings where scores are aggregated, thresholded, or presented to users. Calibration analysis allows us to assess how well the predicted scores reflect true probabilities, providing a clearer picture of a model’s reliability.

In retrieval systems that rely on multiple independent signals, such as those derived from different aspects or criteria, calibrated outputs are particularly valuable. When models produce confidence scores that are better aligned with observed relevance, combining their outputs becomes more meaningful and consistent. Calibrated scores also support more robust decision-making when setting thresholds or ranking candidates across criteria. Thus, even in systems that do not explicitly optimize for calibration, understanding and measuring this property is key to ensuring the reliability of the produced relevance scores.

2.5. Related Work

The text retrieval task has seen substantial advancements with the advent of large language models (LLMs). Recently, researchers have applied different techniques to improve LLM-based text retrieval systems. RocketQAv2 (Ren et al., 2021) and AR2 (Zhang et al., 2021a) use a joint training strategy for both the retriever and re-ranker parts. RocketQAv2 jointly trains the retriever and the re-ranker with a focus on reducing the difference between their relevance prediction distributions. AR2 formulates the text-ranking problem using a Generative Adversarial Network (GAN) (Aggarwal et al., 2021). It jointly optimizes a retriever as the generative model and a ranker as the discriminative model, with the retriever generating hard negatives to improve ranking performance. Poly-encoder (Humeau et al., 2020), HLATR (Zhang et al., 2022), and Uni-encoder (Song et al., 2023) aim to balance the efficiency

of bi-encoders with the rich interaction capabilities of cross-encoders. RankLLaMA (Ma et al., 2024) was introduced as a fine-tuned LLaMA model for multi-stage text retrieval, demonstrating that more advanced LLMs can outperform smaller models. DORIS-MAE (Wang et al., 2023) tackles the limitations of NIR models trained on simple queries, which often fail on complex, multifaceted inputs. It introduced a benchmark dataset with hierarchical aspect-based queries in the scientific domain and demonstrated that aspect decomposition can improve retrieval performance. Parameter-efficient fine-tuning (PEFT) techniques have also gained traction in text retrieval. Tam et al. (2023b) employed prompt tuning techniques, demonstrating that by updating only a small portion of the model parameters, performance comparable to conventional fine-tuning can be achieved. This method significantly enhances out-of-domain generalization and improves confidence calibration. Jung et al. (2022) introduced a Semi-Siamese Bi-encoder Neural Ranking Model utilizing PEFT techniques, which has shown significant improvement by updating only a small portion of the model parameters.

In the realm of transparency and explainability, several approaches have been proposed to make neural retrieval models more interpretable. Leonhardt et al. (2023) introduced the Select-and-Rank paradigm, wherein the model selects a subset of sentences from a document as an explanation and then uses this selection exclusively for prediction. This approach treats explanations as integral components of the ranking process, enhancing interpretability by design. Zhang et al. (2021b) proposed ExPred, a model employing multi-task learning during the explanation generation phase. It balances explanation and prediction losses and subsequently utilizes a separate prediction network trained solely on the extracted explanations to optimize task performance. Wallat et al. (2023) conducted an in-depth analysis of BERT’s ranking capabilities by probing its internal representations. Their study reveals insights into BERT’s effectiveness in ranking tasks and highlights areas for improvement in aligning its internal mechanisms with established information retrieval principles.

Recent studies have also emphasized the importance of calibration in neural rankers. Penha and Hauff (2021b) investigated the calibration and uncertainty of neural retrieval models in conversational search, highlighting that neural rankers often produce overconfident predictions. Tam et al. (2023a) proposed prompt tuning methods that improve generalization and calibration of dense retrievers with minimal parameter overhead. Yu et al. (2024) used LLMs to generate natural language explanations and applied

Monte Carlo sampling to achieve better scale calibration, while maintaining or improving ranking performance.

These advancements highlight the continuous evolution and optimization of text retrieval systems using LLMs and related techniques. CREST takes a different approach compared to earlier interpretable or LLM-based retrieval systems like ExPred (Zhang et al., 2021b) and RankLLaMA (Ma et al., 2024), which mainly aim to improve explanation generation or scale single end-to-end rankers. Instead, CREST introduces a modular, multi-aspect retrieval framework. Rather than depending on one model to capture all query semantics, it breaks down each query into predefined criteria and combines the outputs of specialized models trained for those aspects. This structure enables CREST to deliver stronger retrieval performance and clearer interpretability, since each result’s relevance can be directly tied to specific criteria.

3. Methodology

In this study, we conduct a comprehensive investigation to understand how different Trouble Report (TR) observation criteria affect the performance of TR retrieval models such as BERTicsson (Grimalt et al., 2022).

This study also aims to train criterion-specific models and aggregate them to optimize the retrieval process, enhancing the system’s ability to attend to information from different criteria. The TR observations are preprocessed and parsed using a standardized Ericsson TR template to extract various informative criteria. The impact of each criterion is assessed on the overall retrieval system’s performance. Moreover, this approach provides transparency in the decision-making process by providing separate relevance scores to each observation criterion, enabling users to trace retrieval outcomes back to meaningful components of the input.

3.1. Two-stage TR Retrieval Model

This methodology employs a two-stage ranking architecture similar to BERTicsson where the models used are adapted with RoBERTa, resulting in TwinRoBERTa or ColRoBERTa for initial retrieval and monoRoBERTa for re-ranking stage. The first stage utilizes a bi-encoder architecture, which is less computationally intensive and allows faster processing. This efficiency arises from the ability to compare pre-calculated document embeddings with the embeddings of new queries. The processed text from the preprocessing step serves as the input for the IR stage, which retrieves a top-K list of

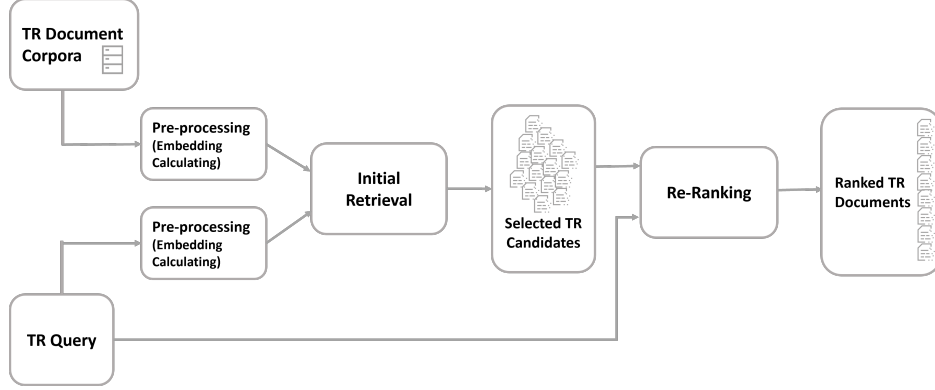


Figure 1: Overview of the utilized TR recommendation system.

candidate TRs ranked by the relevance of their accepted answers to the target TR observation.

The second stage involves re-ranking (RR) the candidates provided by the IR stage. Here, each candidate from the top-K list is paired with the query and processed through the monoRoBERTa model, a cross-encoder that provides a more detailed and computationally intensive comparison. This re-ranking stage benefits from the cross-encoder’s ability to assess finer details within the interactions between the query and document, resulting in a highly accurate, final ranked list of relevant TRs.

Figure 1 provides an overview of the TR retrieval system utilized in this study, illustrating the data flow through the multi-stage architecture. The workflow begins with the preprocessing of TRs and the calculation of embeddings. Document embeddings for the corpus are computed once and used throughout the IR step. The query embeddings are similarly generated and used to select the top-K most similar TR documents. These selected documents are then paired with the query for the re-ranking step, executed by monoRoBERTa.

This two-stage strategy efficiently integrates the speed and lower computational demands of a bi-encoder in the IR phase, with the accuracy and depth of analysis provided by a cross-encoder in the RR phase.

Criteria-specific Retrieval via Ensemble of Specialized TR models (CREST) adopts a similar two-stage approach but distinguishes itself by utilizing an aggregation of TR retrieval models, each trained with a different TR observation criterion and specializing in retrieving documents based on that specific

aspect. This ensemble setup enables the retrieval system to attend to various facets of the queries and retrieve documents relevant to each criterion. A document’s final relevance score is then computed through a weighted aggregation of its individual criterion-specific scores, ensuring that the distinct contributions of each criterion are accurately captured and utilized.

This design not only enhances the overall effectiveness of the TR retrieval system by leveraging the strengths of specialized models but also provides individual relevance scores per criterion, offering transparency and traceability in the decision-making process for end-users.

3.2. Trouble Report Data

In this study, we utilize historically resolved trouble reports (TRs) as the training data for both the IR and RR models. We primarily focus on the headline and observation sections of the TRs, which can describe the problem presented and are used as queries to retrieve relevant documents. The accepted answers, which detail the resolutions to these problems, serve as the documents in our retrieval system. This setup allows us to rank previously resolved TRs based on their relevance to new queries that are formulated from the headline and observation sections of new TRs. Notably, while our approach is centered around retrieving TRs based on the relevancy of their answers to the new TRs observation, its potential applications extend beyond this scope. For instance, our methodology can be adapted for identifying TR duplicates (Bosch et al., 2022), where the emphasis lies on finding similar TRs based on the similarity of their observations.

A typical TR at Ericsson contains the following sections:

1. *Headline*: A sentence summarizing the problem, often containing critical information about the issue.
2. *Observation*: Detailed text that describes the problem comprehensively. This free-format text includes vital details for the responsible team, such as the general description of the tester’s observation, *impact* on the system, *conditions* causing the issue, *frequency* of the problem, and *reproducibility*.
3. *Answer*: Extensive text that explains the root cause and resolution.
4. *Faulty Product Detail*: Information about the fault, the affected software product, and additional relevant details.

Parsing is applied to the TR observation to extract multiple criteria and assess their contributions to retrieval. We use a lightweight Python regex

parser that targets the standardized observation template, looking for headers explaining different criteria. Although the observation is free text, testers complete a consistent, organization-wide template. Fields are optional, so some entries can be missing or not be complete. Ericsson’s internal quality-control review takes place before TR is being published, which limits format drift and typographical variation. In practice, the parser is reliable, with only occasional edge cases such as merged or empty fields. We select the following specific criteria for this study:

1. *Trouble Description*: This is the most informative part of the observation, as it explains the problem in detail.
2. *Impact*: A short text explaining the impact of a new problem on the system.
3. *Condition*: The condition in which the problem occurred.
4. *Frequency*: The frequency of observing the problem.
5. *Steps to reproduce*: Explain if the problem is reproducible and how it can be reproduced.

Figure 2 illustrates an example of the TR observation. Although it is of free format, the observation follows a template that assists writers in structuring and positioning its various criteria. Figure 3 shows the token length distribution of observation section criteria before and after parsing. While unparsed observations often contain long, unstructured text, the parsed fields are typically more concise and structured. This reinforces the importance of evaluating each field’s individual contribution to retrieval effectiveness, offering a more targeted alternative to treating the observation as a monolithic input.

With this in mind, we generate multiple datasets that pair the TR headline with different parsed observation criteria. This setup allows us to investigate the retrieval value of each field and gain a better understanding of its role in the overall retrieval task. Moreover, these insights not only support the design of our criterion-specific modeling approach but also offer practical guidance: if certain criteria are shown to have a stronger impact on retrieval quality, TR creators can be encouraged to elaborate more on those fields, ultimately improving the effectiveness of the retrieval system.

3.3. Initial Retrieval

In the initial retrieval (IR) stage, pre-processed queries and documents extracted from TRs are utilized to generate a top-K candidate list. The ini-

1.1 Summary of the trouble

A restart in a node has been detected during a RCC test.

1.2 Observation of the impact

The restart was produced during a process related to RCC:
0x3005500

1.3 Condition

1. Run the RCC test
2. Enable feature1
3. Check metrics

1.4 Frequency

Each time the test runs

1.5 Step to reproduce

Can reproduce, install issue version then start feature1.

Figure 2: An example of the TR observation field with different criteria.

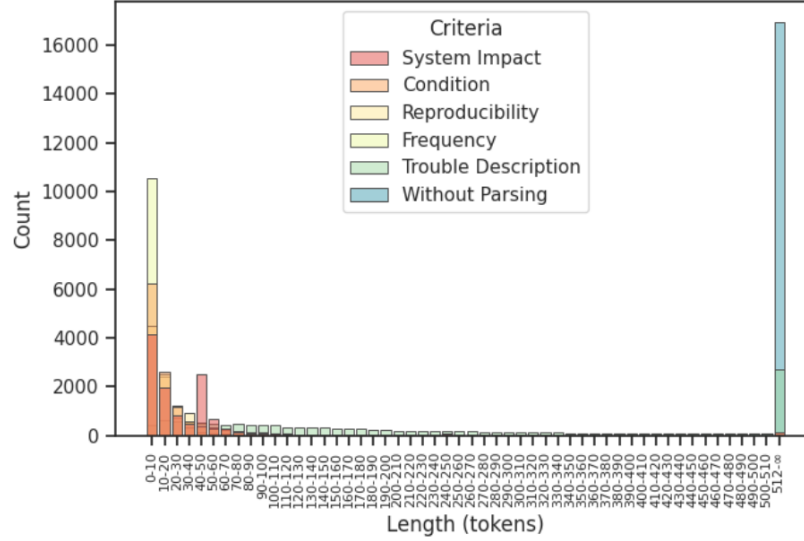


Figure 3: Distribution of TR observation criteria based on the token length.

tial retrieval of candidates within the top-K list is not essential, as a later re-ranking stage will adjust their order. However, it is crucial that the IR

stage includes as many relevant documents as possible within the top-K list to achieve a high overall system performance. For the retrieval of the top-K candidates, a bi-encoder architecture is employed, specifically the TwinRoBERTa and ColRoBERTa models. These models are frequently used in various information retrieval and similarity comparison tasks (e.g., BERTicsson (Grimalt et al., 2022)). Their main objective is to determine the similarity between two inputs. In our context, these inputs are a TR observation (including headlines and various criteria) and a TR answer. They effectively separate the query and document processing by employing distinct encoders for each, followed by a mean pooling layer to generate fixed-length representations for both inputs. To further enhance domain-specific performance, TeleRoBERTa (Holm, 2021) is incorporated, a version of the RoBERTa model further trained in telecommunications data. The domain-specific knowledge of this model can benefit the TR retrieval system performance (Nimara et al., 2024).

The similarity score between query and document embeddings is calculated using a fully connected layer. The resulting score is then used to rank and select the top-K candidates. For training, the model is exposed to relevant and irrelevant query and document pairs in order to adjust its weights, ensuring higher scores for relevant pairs and lower scores for irrelevant ones.

To minimize latency during inference, the embeddings of all documents in the corpus are pre-calculated by leveraging the decoupled nature of TwinRoBERTa and ColRoBERTa, which independently process queries and documents. This allows for only computing the query representation during the inference and comparing it against all pre-stored document embeddings. This possibility makes models that follow bi-encoder architectures significantly faster than cross-encoder architectures like monoRoBERTa, in which both query and document are processed simultaneously.

3.4. Re-Ranking Stage

In the re-ranking (RR) stage, the IR output, which is the top-K candidates, is further processed for a more precise ranking. In practice, we generate this top-K with the strongest IR configuration on our validation set, and use that retriever’s output as input to RR, so the cross-encoder operates on the best available candidate set. Following the approach used in BERTicsson (Grimalt et al., 2022), this study uses a cross-encoder architecture, specifically adopting the monoRoBERTa framework for the RR

stage. Similar to the IR stage, TeleRoBERTa is employed here to leverage its telecommunications domain knowledge.

The input of the monoRoBERTa model is a concatenated string of a query and document tokens separated by a special token as defined in large language models, e.g., “[CLS], query tokens, [SEP], document tokens, [SEP]”. Since both the query and document can contain important contextual information, we allocate tokens equally between them to preserve balanced representation.

Unlike bi-encoder models, which use a decoupled approach to represent queries and documents, the cross-encoder architecture integrates the processing of query and document representations. Since cross-encoder models compute query and document representations together, pre-computing embeddings is not feasible, resulting in higher retrieval latency. However, the increased latency remains minimal, as the model only applies to the top-K candidates identified in the IR stage, which is a subset considerably smaller than the entire TR corpus.

During training, the cross-encoder receives pairs of relevant and irrelevant queries and documents, adjusting its weights to enhance its performance. In the inference phase, the model only processes receiving query pairing with the top-K candidates extracted during the IR stage. By limiting the number of query-document pairs processing, the computational latency inherent to cross-encoder models is effectively mitigated.

Similar to the IR stage, applying CREST in the RR stage can improve performance. However, unlike the IR stage, using CREST in the RR stage will introduce additional latency, as the top-K candidates must be processed by each criterion-specific model. This increases latency by a factor equal to the total number of criterion-specific models. This added latency is acceptable, as the ensemble size is small, with CREST utilizing only four criterion-specific models.

3.5. *Criteria-specific Retrieval via Ensemble of Specialized TR models (CREST)*

In this study, we introduce the Criteria-specific Retrieval via Ensemble of Specialized TR models (CREST), an ensemble-based framework designed not only to enhance retrieval performance but also to improve transparency (aka interpretability) in the retrieval process. By leveraging the structured nature of TR observations, each model in the ensemble is trained using a distinct criterion extracted from the observation section, enabling it to specialize in handling a specific type of information. This setup allows CREST to

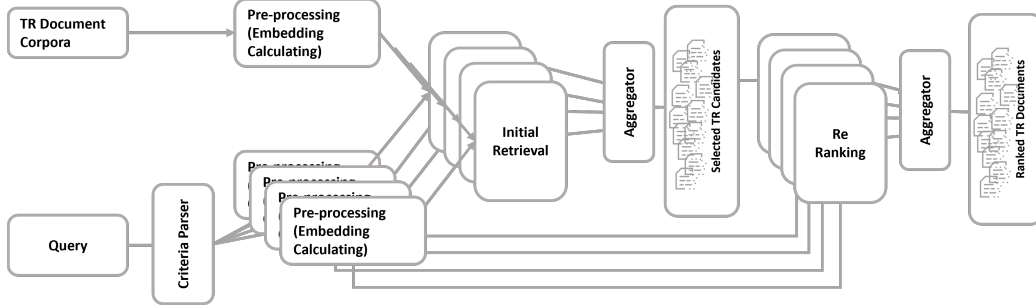


Figure 4: Overview of CREST in a two-stage pipeline: bi-encoders (Twin/ColRoBERTa) retrieve top-K candidates and a cross-encoder (monoRoBERTa) re-ranks them. Unlike the baseline criteria-agnostic two-stage workflow shown in Figure 1, CREST adds criterion-specific models whose per-criterion scores are aggregated into the final relevance score.

capture diverse aspects of the TR content and aggregate them to form a more comprehensive and interpretable relevance signal.

CREST supports both bi-encoder (TwinRoBERTa, ColRoBERTa) and cross-encoder (monoRoBERTa) architectures. For each criterion, a separate query is generated, and its corresponding model is used to evaluate document relevance. These individual scores are then combined through a learned, criterion-specific weighted aggregation, implemented as a linear combination of criterion-specific scores with non-negative weights in range of $[0,1]$ to produce the final relevance score for each TR document. These aggregation weights are optimized in a separate training stage for each cross-encoder and bi-encoder models, while the models parameters are kept frozen. The optimization uses a hinge loss applied over the aggregated scores, and the final aggregation model is selected based on the Mean Reciprocal Rank (MRR) achieved on the validation set. This mechanism ensures that the specific contribution of each criterion is reflected in the final ranking, offering greater transparency and traceability in the retrieval process. Figure 4 provides an overview of the CREST setup.

Figure 5 illustrates what CREST may look like in practice. Users can interactively select which criteria to activate, such as system impact, condition, frequency, or how to reproduce, depending on the context or diagnostic goal. Once a new trouble description is entered, CREST generates criterion-specific queries, scores each candidate TR using its corresponding model, and visualizes both the aggregated and disaggregated scores. The coloured bar indicators enable users to assess the relative contribution of each crite-



Figure 5: Mockup of the CREST interface showing selectable criteria and both disaggregated (per-criterion) and aggregated relevance scores, enabling configurable focus and clearer rationale for retrieved results.

tion to the final score, providing transparency and control over the retrieval process. The framework is adaptive to missing inputs; if certain criteria are unavailable during inference, only the relevant specialized models are triggered. Likewise, users can configure the system to focus solely on a particular criterion when diagnosing a specific issue or activate all available criteria to maximize overall performance. This flexibility makes CREST suitable for a range of retrieval scenarios and user needs.

CREST is not positioned as a replacement for existing retrieval models but rather as a modular enhancement that can be integrated into various TR retrieval pipelines. Its criterion-specific decomposition and aggregation strategy make it a flexible solution that can extend to systems based on other LLMs.

3.6. Training

The training process of the TR retrieval system involves training both the bi-encoder and cross-encoder models in a supervised manner. For training, queries, positive documents, and negative documents are created from the

extracted TRs. The query is the description of the issue mentioned in the TR (e.g., headline, and various criteria). The positive document is the answer section paired with the same TR, while the negative document is the same query from the TR, paired with the answer from a different TR. The result is two pairs: $\langle \text{query}, \text{positive document} \rangle$ (called a relevant pair) and $\langle \text{query}, \text{negative document} \rangle$ (called an irrelevant pair). The final training datasets maintain a 1:1 ratio for positive and negative pairs, meaning that for each collected TR, there is one positive pair and one negative pair.

The goal of this study is to explore the effect of various TR observation criteria on retrieval performance and train criterion-specific models. To support this, we construct separate datasets for each observation criterion, allowing us to both train specialized retrieval models and assess the individual contribution of each field to the overall system. This setup enables a detailed evaluation of criterion-level impact while also serving as the foundation for the proposed ensemble framework, where models trained on different criteria are later aggregated.

In total, two types of models were trained in this study: TwinRoBERTa, as a bi-encoder, and monoRoBERTa, as a cross-encoder. All models use the same triplet hinge loss function (Jung et al., 2022), which leverages the relevance scores calculated for both positive and negative pairs to optimize the ranking performance. We used a batch size of 64 and the Adam optimizer with a learning rate of 10^{-5} for all encoders.

3.7. Inference

During the inference phase, the TR retrieval system begins the ranking process once the tester initiates a new TR with both headline and observation details. The query is formed by combining the headline and available criteria from the observation section.

The IR stage starts by computing the query that is then used to calculate relevance scores for different documents, extracting the top-K candidates. During the RR stage, the system re-orders the extracted candidates from the IR stage. Each candidate is paired with the query, and the pair is processed through a cross-encoder model to compute a new relevance score, which is used to refine the ranking of top-K candidates.

Adjusting the value of K affects both performance and latency. A higher K increases the likelihood of capturing relevant documents but also increases the input for the RR stage, thus increasing latency.

Stage	Single Model	CREST
IR and RR(Training)	Single training	4 separate trainings
IR(Inference)	1 LLM pass	4 LLM passes
RR(Inference)	k LLM passes	$4 * k$ LLM passes

Table 1: Comparison of computation and latency costs between the single model and the CREST model, with k representing the number of candidates returned in the IR stage.

For both IR and RR stages, the number of criterion-specific models used in CREST is determined by the availability of the observation criteria after parsing the observation entered by the tester. If the TR observation contains all criteria, then all models in CREST are active. This helps to ensure that the system uses all available information to optimize retrieval performance.

As the document embeddings are pre-calculated for the IR stage, only query embeddings need to be generated at runtime. In CREST, this involves four query embeddings instead of one, but they are computed in parallel so the latency remains close to that of a single model. At a large scale, this keeps first-stage retrieval feasible on very large TR corpora, since each query’s four embeddings are matched against pre-computed indexes to select a small candidate set for reranking (RR).

The RR stage is more expensive because CREST scores every candidate with all criterion-specific models. We manage this by keeping the candidate set relatively small and by batching candidates so each model processes them in one pass. Running models across two GPUs further reduces wall-clock time and keeps end-to-end latency within acceptable service windows for production. The number of criteria can also be reduced for tasks that require very low latency, which creates a trade-off between latency and performance. Table 1 presents training and inference costs.

As presented in Table 1, the training latency for CREST is four times higher than that of a single model when using a single GPU, due to the need to train four separate criterion-specific models. However, we leveraged four GPUs to run these trainings concurrently, which kept the overall training time comparable to that of a single model, while increasing GPU resource usage accordingly.

4. Evaluation

In this study, we create distinct datasets incorporating diverse information for both training and evaluation. We also determine the impact of various TR observation criteria on retrieval system performance and the enhancement that the CREST can bring to the retrieval system.

4.1. Datasets

To construct datasets, trouble reports were organized into groups based on the presence of specific criteria within their observation sections, as illustrated in Figure 2. Due to the heterogeneous nature of TRs, not all contain identical observation fields, which affects the dataset’s composition. We consistently include the headline section and trouble description criteria in all datasets to preserve essential information. Approximately 60% of the parsed TRs used for this study contain all listed criteria, and this coverage is significantly higher among more recent TRs than among older reports partially included in this study.

To prepare data for the criterion-specific models and analyze the significance of each of the individual criteria, the following approach was applied for each criterion:

1. Filtering TRs to isolate those containing the criterion under study.
2. Forming queries and documents from this refined subset to produce datasets focused on that specific criterion. Documents are pre-processed versions of TR answers.
3. Combining the “headline” and “trouble description” with the criteria under evaluation to formulate queries, ensuring each dataset is tailored to our research focus.

A baseline dataset is created for each experimental set, consisting of TRs with queries derived solely from the “headline” and “trouble description”. This baseline allows for direct performance comparison between criterion-specific datasets and their corresponding baseline, measuring the impact of each criterion.

From a subset of the TR corpus that includes all criteria shown in Figure 2, we randomly extract two non-overlapping sets of 1,000 TRs. These sets form the basis for the validation and test datasets used in each experiment. It is important to note that the TRs in the validation and test datasets remain constant across all experiments to ensure consistent evaluation. Table 2 reports the different datasets along with their metrics.

Dataset	Included Fields	Number of TRs for Training
HTI	H + T + Impact	8,641
HTF	H + T + Frequency	11,864
HTC	H + T + Condition	10,175
HTR	H + T + Reproducibility	8,097
Single Model	-	14,504

Table 2: Criterion-specific training datasets and sample sizes (number of TRs used for training). Included fields indicate query construction. Abbreviations: H = *headline*, T = *trouble description*, I = *impact*, F = *frequency*, C = *condition*, R = *reproducibility*.

4.2. Evaluation Strategy

To assess the effectiveness of each experiment, we undertake a comprehensive comparison between the performance of each criterion-specific model and its corresponding baseline. This comparative approach enables us to measure the impact of the examined TR observation criteria on the overall performance of the retrieval system, which is calculated for each criterion (impact on system, condition, and others) as follows:

$$I_C = P_C - P_{C_{baseline}} \quad (1)$$

Where P_C represents the performance of a criterion-specific model for the criterion under study. $P_{C_{baseline}}$ denotes the performance of the corresponding baseline model, which includes headline and trouble description, but not the criterion under study. I_C is the impact score, indicating the performance difference caused by introducing the specific criterion being evaluated.

The impact score for each criterion is calculated for both bi-encoder and cross-encoder architectures within the two-stage workflow, as shown in Figure 1. In addition to this, we compare the overall retrieval performance across criterion-specific models, the CREST ensemble, and the single model baseline. This comparison is conducted under both the full two-stage setup and the cross-encoder (monoTeleRoBERTa) in isolation. The goal of this evaluation is to determine the effectiveness of incorporating criterion-specific signals into the retrieval process and to assess whether the ensemble strategy in CREST leads to consistent performance improvements over individual models and baselines across different configurations.

To assess confidence calibration, we adopt the methodology proposed by Penha and Hauff (2021b), which transforms the ranking task into a multi-

class classification problem. For each query, we select the top five documents based on their retrieval scores and apply a softmax function to normalize the scores into probabilities. This reformulation enables the computation of calibration metrics, allowing us to evaluate how well the predicted relevance scores reflect actual relevance likelihoods. We follow the same evaluation setting presented in their study to ensure consistency and comparability in our calibration analysis.

4.3. Evaluation Metrics

In our study, we employ three key metrics to quantify the performance of models on datasets introduced in Section 4.1: Mean Reciprocal Rank (MRR), Recall@K, and nDCG.

Mean Reciprocal Rank (MRR): For each query, the Reciprocal Rank is the inverse of the rank of the first relevant document. For instance, if the first relevant document appears at position 2 in the retrieval system output, the RR is $\frac{1}{2}$. *MRR* is calculated by obtaining the mean over the RR of all queries.

Recall@K: Evaluates the model by its effectiveness in retrieving relevant documents within the top-K results without considering their actual rank.

Normalized Discounted Cumulative Gain (nDCG): Measures the ranking quality of retrieval results, giving a higher score for relevant documents ranked higher in the list.

Expected Calibration Error (ECE): To evaluate how well the predicted relevance scores reflect the true likelihood of relevance, we employ the Expected Calibration Error. This metric quantifies the difference between predicted confidence and empirical accuracy. The predictions are grouped into M equally spaced bins based on their confidence scores. For each bin B_m , the absolute difference is computed between the average predicted confidence $\hat{p}_i$ and the observed accuracy (i.e., the fraction of correct predictions). The final ECE is the weighted average of these differences across all bins, defined as:

$$\text{ECE} = \sum_m \frac{|B_m|}{n} \left| \frac{1}{|B_m|} \sum_{i \in B_m} [\mathbb{I}(\hat{y}_i = y_i) - \hat{p}_i] \right| \quad (2)$$

This metric was originally proposed by Naeini et al. (2015) and has been adapted for neural ranking settings in recent works (Penha and Hauff, 2021b;

Tam et al., 2023a). A lower ECE indicates better calibration, meaning that the model’s predicted relevance scores are more trustworthy.

Calibration Diagrams: These plots visualize the relationship between confidence and accuracy. A perfectly calibrated model aligns with the diagonal line, where confidence matches accuracy. Deviations from this line indicate miscalibration, with underconfidence or overconfidence depending on the direction of the shift.

5. Results

In this section, we present the evaluation results of the CREST compared to a single criterion-agnostic model TR retrieval model. We also discuss the impact score of each criterion on retrieval performance. The performance is assessed by several key metrics: Recall@K ($R@5$, $R@10$, $R@15$), Mean Reciprocal Rank (MRR), and $nDCG@15$. We present the evaluation results for all bi-encoder and cross-encoder models, including their performance within a two-stage workflow and the cross-encoder in isolation.

5.1. Impact of Each Criterion ($RQ1$)

Table 3 and Table 4 present the criterion-specific models alongside their baselines, using TwinRoBERTa and ColRoBERTa for the IR stage, respectively. Table 5 presents the criterion-specific models with their baselines using the monoRoBERTa model for the RR stage. The findings from the evaluation demonstrate that criterion-specific models enhance retrieval performance compared to their respective baselines. In the IR stage using TwinRoBERTa model, HTI, HTF, and HTC consistently outperform their corresponding baselines across all metrics. Compared to the baseline single model that uses all available information, these models also show improved performance, with the exception of HTI in $R@5$, where it slightly underperforms. On the other hand, the HTR model performs worse than both its baseline and the single model across all metrics, indicating that this criterion may not contribute as effectively to improving retrieval quality in the IR setting.

In the IR stage with ColRoBERTa, the criterion-specific models once again outperform their baselines for HTI and HTC. The same holds for HTF, with the exception of the $R@15$ metric, where the baseline performed slightly better. Similar to the TwinRoBERTa results, these models consistently surpass the single baseline model that uses all available information, except in the case of HTF at $R@5$. HTR follows a comparable trend to TwinRoBERTa

Initial Retrieval (Bi-Encoder)	TwinRoBERTa-base									
	HTI	HTI baseline	HTF	HTF baseline	HTC	HTC baseline	HTR	HTR baseline	Single Model	BM25
$R@5$	49.85%	45.95%	52.95%	49.25%	51.35%	51.05%	50.15%	51.45%	49.95%	46.85%
$R@10$	58.56%	53.55%	61.36%	57.56%	59.79%	57.96%	57.66%	59.46%	57.96%	51.55%
$R@15$	64.36%	59.16%	65.77%	62.26%	65.17%	62.96%	61.96%	64.96%	62.66%	54.95%
MRR	42.19%	38.59%	43.89%	40.75%	43.14%	41.71%	41.92%	42.95%	42.04%	30.58%
$nDCG@15$	52.87%	49.68%	48.39%	45.08%	47.58%	46.00%	45.95%	47.40%	46.19%	42.73%

Table 3: Performance of criterion-specific models, their baselines, and a single criterion-agnostic model for TwinRoBERTa-base encoder in the IR stage.

Initial Retrieval (Bi-Encoder)	ColRoBERTa-base									
	HTI	HTI baseline	HTF	HTF baseline	HTC	HTC baseline	HTR	HTR baseline	Single Model	BM25
$R@5$	58.16%	56.96%	55.06%	54.95%	56.86%	53.85%	52.95%	53.75%	55.56%	46.85%
$R@10$	65.76%	63.66%	63.86%	63.36%	65.77%	62.26%	61.86%	61.27%	62.66%	54.95%
$R@15$	69.37%	68.57%	67.97%	68.27%	70.97%	66.97%	67.17%	66.27%	67.17%	51.55%
MRR	49.07%	47.42%	47.22%	46.93%	49.13%	47.09%	45.61%	46.79%	45.98%	30.58%
$nDCG@15$	53.25%	52%	51.45%	51.28%	53.63%	51.06%	50.01%	50.70%	50.29%	42.73%

Table 4: Performance of criterion-specific models, their baselines, and a single criterion-agnostic model for ColRoBERTa-base encoder in the IR stage.

by performing worse than both its respective baseline and the single baseline model. However, in the ColRoBERTa setting, HTR performs considerably better, and its negative effect is relatively minor. Overall, these outcomes confirm the recurring pattern that HTR can reduce performance when used alone in IR stage, while the other criteria contribute positively in most cases.

RR results are reported using the top-K candidates produced by the *ColRoBERTa* IR configuration, which demonstrated consistently superior performance and thus provides the most reliable candidate pool for re-ranking. In the RR stage, all criterion-specific models outperform their baselines and the single model across most metrics. A notable reversal occurs with HTR, which underperforms its baseline in IR under both TwinRoBERTa and ColRoBERTa, yet in RR achieves the highest recall at all cutoffs and ties for the top $nDCG@15$, with only MRR slightly higher for HTF. This suggests that the richer contextual information available during re-ranking enables the model to make better use of HTR-specific signals that were less effective in the retrieval-only setting. HTI and HTC remain strong in RR, but their advantage is smaller than in IR. This pattern suggests that criterion-specific modeling continues to add value, with impact and condition cues driving retrieval performance, while the other criteria help refine the final ranking.

Table 6 presents the performance of criterion-specific models with their baselines using only the isolated monoRoBERTa cross-encoder, without benefiting from the IR stage. The results reveal that the single model, which utilizes all available criteria, consistently achieves better performance across

Re-Ranking (Cross-Encoder)	monoRoBERTa-base								
	HTI	HTI baseline	HTF	HTF baseline	HTC	HTC baseline	HTR	HTR baseline	Single Model
$R@5$	65.87%	60.16%	65.47%	62.06%	66.27%	61.56%	66.67%	60.36%	63.66%
$R@10$	74.17%	68.77%	74.17%	70.97%	74.67%	70.17%	75.88%	68.77%	70.27%
$R@15$	77.78%	72.37%	78.48%	73.67%	79.28%	73.87%	79.58%	72.27%	74.17%
MRR	52.02%	49.52%	52.88%	51.79%	50.58%	51.19%	52.42%	48.16%	51.77%
$nDCG@15$	57.89%	54.72%	58.66%	56.78%	57.17%	56.40%	58.66%	53.72%	56.93%

Table 5: Performance of criterion-specific models, their baselines, and a single model for the monoTeleBERTa-base encoder in the RR stage.

Isolated (Cross-Encoder)	monoRoBERTa-base								
	HTI	HTI baseline	HTF	HTF baseline	HTC	HTC baseline	HTR	HTR baseline	Single Model
$R@5$	56.06%	55.36%	55.66%	55.86%	54.35%	54.75%	58.46%	50.65%	60.96%
$R@10$	67.47%	65.27%	65.57%	65.47%	66.47%	65.47%	67.07%	61.56%	68.57%
$R@15$	73.67%	70.97%	70.57%	70.07%	72.37%	70.57%	72.97%	67.87%	73.27%
MRR	43.76%	42.53%	43.14%	42.69%	40.40%	42.21%	44.68%	37.55%	47.24%
$nDCG@15$	50.15%	48.60%	48.98%	48.51%	47.29%	48.28%	50.78%	43.94%	52.90%

Table 6: Performance of criterion-specific models, their baselines, and a single model for the isolated monoTeleBERTa-base encoder.

nearly all metrics and settings. This stands in contrast to the RR stage results, where criterion-specific models surpassed both their baselines and the single model. The lack of improvement here suggests that the re-ranking benefits observed earlier rely heavily on the synergy between the IR and RR stages rather than on the cross-encoder alone. Without the support of an IR stage to pre-select relevant candidates, the cross-encoder operates over a broader, noisier set of inputs, which can dilute the effectiveness of specialized representations.

Overall, these results emphasize that criterion specialization is most effective when integrated into a pipeline where the IR stage helps isolate more relevant candidates, creating a setting where the RR model can more effectively leverage targeted representations.

Table 7 presents the impact scores of each criterion across both the IR and RR stages. With TwinRoBERTa in IR, “system impact” (HTI) shows the strongest positive influence across all metrics, with “frequency” (HTF) and “condition” (HTC) following, and “reproducibility” (HTR) reducing performance. With ColRoBERTa in IR, “condition” emerges as the leading signal while “system impact” remains beneficial, “frequency” offers smaller gains, and “reproducibility” is mixed, hurting early precision but providing slight improvements at deeper recall. In the RR stage using monoRoBERTa over ColRoBERTa candidates, “reproducibility” delivers the largest gains on all metrics, while impact and “frequency” remain positive and “condition” contributes less to MRR.

Two Stage Retrieval	IR stage - TwinRoBERTa-base				IR stage - ColRoBERTa IR				RR Stage - monoRoBERTa-base			
	HTI	HTF	HTC	HTR	HTI	HTF	HTC	HTR	HTI	HTF	HTC	HTR
$R@5$	3.9	3.7	0.3	-1.3	1.2	0.1	3.01	-0.8	5.71	3.41	4.71	6.31
$R@10$	5.01	3.8	1.83	-1.8	2.1	0.50	3.51	0.59	5.40	3.20	4.50	7.10
$R@15$	5.2	3.51	2.21	-3.0	0.8	-0.3	4.0	0.9	5.41	4.81	5.41	7.31
MRR	3.6	3.14	1.43	-1.03	1.65	0.29	2.04	-1.18	2.50	1.09	-0.61	4.26
$nDCG@15$	3.19	3.31	1.58	-1.45	1.25	0.17	2.57	-0.69	3.17	1.88	0.77	4.94

Table 7: Impact of each criterion on the retrieval model performance. RR-stage result is computed using monoRoBERTa with *ColRoBERTa* as the preceding IR stage.

Model	Variant	MRR	R@5	R@10	R@15	nDCG@15
IR - ColRoBERTa	CREST (All)	52.50%	61.36%	69.27%	75.18%	57.19%
	CREST w/o I	52.20% ($\downarrow$ 0.30)	60.62% ($\downarrow$ 0.74)	67.94% ($\downarrow$ 1.33)	72.14% ($\downarrow$ 3.04)	56.26% ($\downarrow$ 0.93)
	CREST w/o F	52.46% ($\downarrow$ 0.04)	61.66% ($\uparrow$ 0.30)	68.87% ($\downarrow$ 0.40)	74.47% ($\downarrow$ 0.71)	57.00% ($\downarrow$ 0.19)
	CREST w/o C	51.21% ($\downarrow$ 1.29)	59.96% ($\downarrow$ 1.40)	69.27% (0.00)	73.27% ($\downarrow$ 1.91)	55.77% ($\downarrow$ 1.42)
	CREST w/o R	51.87% ($\downarrow$ 0.63)	60.92% ($\downarrow$ 0.44)	68.54% ($\downarrow$ 0.73)	74.05% ($\downarrow$ 1.13)	56.46% ($\downarrow$ 0.73)
IR - TwinRoBERTa	CREST (All)	50.64%	60.02%	67.74%	72.04%	55.07%
	CREST w/o I	49.53% ($\downarrow$ 1.11)	59.02% ($\downarrow$ 1.00)	66.63% ($\downarrow$ 1.11)	71.14% ($\downarrow$ 0.90)	53.99% ($\downarrow$ 1.08)
	CREST w/o F	49.25% ($\downarrow$ 1.39)	58.02% ($\downarrow$ 2.00)	64.93% ($\downarrow$ 2.81)	71.24% ($\downarrow$ 0.80)	53.75% ($\downarrow$ 1.32)
	CREST w/o C	49.63% ($\downarrow$ 1.01)	58.86% ($\downarrow$ 1.16)	66.47% ($\downarrow$ 1.27)	71.47% ($\downarrow$ 0.57)	54.16% ($\downarrow$ 0.91)
	CREST w/o R	50.14% ($\downarrow$ 0.50)	58.92% ($\downarrow$ 1.10)	67.23% ($\downarrow$ 0.51)	71.44% ($\downarrow$ 0.60)	54.54% ($\downarrow$ 0.53)
RR - monoRoBERTa	CREST (All)	57.69%	70.07%	77.73%	81%	63.08%
	CREST w/o I	56.62% ($\downarrow$ 1.06)	69.68% ($\downarrow$ 0.39)	77.62% ($\downarrow$ 0.11)	80.87% ($\downarrow$ 0.13)	62.26% ($\downarrow$ 0.82)
	CREST w/o F	56.28% ($\downarrow$ 1.41)	69.99% ($\downarrow$ 0.08)	77.92% ($\uparrow$ 0.19)	80.87% ($\downarrow$ 0.13)	62% ($\downarrow$ 1.08)
	CREST w/o C	57.24% ($\downarrow$ 0.45)	69.55% ($\downarrow$ 0.52)	77.49% ($\downarrow$ 0.24)	80.75% ($\downarrow$ 0.25)	62.68% ($\downarrow$ 0.4)
	CREST w/o R	55.91% ($\downarrow$ 1.78)	68.43% ($\downarrow$ 1.64)	76.88% ($\downarrow$ 0.85)	80.86% ($\downarrow$ 0.14)	61.66% ($\downarrow$ 1.42)

Table 8: Ablation of criterion-specific models in the CREST ensemble (percent). Differences in parentheses indicate percentage-point change vs. the model’s *CREST (All)*. I = Impact (HTI), R = Reproducibility (HTR), F = Frequency (HTF), C = Condition (HTC). “w/o X” excludes criterion X from the ensemble. Differences are percentage points relative to *CREST (All)* for the same backbone.

One possible reason why HTR performs worse in the IR stage is that the bi-encoder compares short query embeddings with pre-computed document embeddings, which tends to favour concise descriptors. HTR text often includes procedural steps, boilerplate phrases, and local details such as paths or versions, which may dilute the main semantic signal. Because answers do not always mirror these stepwise details, the similarity match is often weaker than for HTI, HTF, or HTC, which contain clearer fault descriptors and affected components. In contrast, the RR stage may benefit more from HTR since the cross-encoder can capture word interactions, ordering, and negation, making it easier to align procedural steps with the answer section resolution explanations.

To further analyze the impact of each criterion on ensemble performance,

Table 8 presents results in which CREST is compared with all criteria active and with one removed. The comparison shows that across all three backbones, *CREST (All)* consistently delivers the most reliable performance. Removing any single criterion degrades every evaluation metric, highlighting their strong complementarity. A notable case arises with reproducibility (R), which on its own has a negative effect in IR tasks as shown in Table 7. Yet Table 8 reveals that excluding R from the ensemble reduces effectiveness, indicating that CREST is able to exploit its value in combination with other criteria. This suggests that R supplies a complementary signal that is not fully captured elsewhere. For both IR models the effect of removing R is less severe than removing other criteria, whereas for RR it has a greater negative impact. These patterns suggest that the ensemble leverages R as a high-precision discriminator in RR while down-weighting its noisier effect in IR, which explains why its exclusion harms final performance.

Overall, the ablation study demonstrates that each criterion contributes positively when aggregated. By providing distinct forms of evidence, they enhance ranking quality once richer interactions are integrated, supporting the use of the complete criterion set in CREST. The results also suggest that the contribution of each criterion is not uniform and varies across the IR and RR stages. HTI, HTF, and HTC consistently show a positive impact, highlighting their critical role in both stages of retrieval. Interestingly, HTR, which negatively affects performance in the IR stage, shows notable improvements in the RR stage. This indicates that certain criteria, like HTR, may require a richer interaction or additional context to become useful, which the RR stage is better able to provide. Moreover, it underscores the need to revisit the current TR template structure to ensure that the most influential information is surfacing early, making it easier for readers to quickly identify the most important information.

5.2. Performance improvement by CREST (RQ2)

Table 9 demonstrates the performance improvement of CREST over the single criterion-agnostic retrieval model in the two-stage workflow. This comparison includes results for the IR stage (TwinRoBERTa and ColRoBERTa), followed by the RR stage (monoRoBERTa) over ColRoBERTa candidates.

The performance evaluation of CREST demonstrates significant gains across all metrics, compared to a single criterion-agnostic model approach in both the IR and RR stages. In the IR stage, by aggregating relevance scores from all criterion-specific models (HTI, HTC, HTF, HTR) through weighted

	IR (TwinRoBERTa)		IR (ColRoBERTa)		RR (monoRoBERTa)	
	CREST	Single criterion-agnostic retrieval model	CREST	Single criterion-agnostic retrieval model	CREST	Single criterion-agnostic retrieval model
$R@5$	60.16%	49.95%	61.36%	55.56%	70.07%	63.66%
$R@10$	67.67%	57.96%	69.27%	62.66%	77.73%	70.27%
$R@15$	72.67%	62.66%	75.18%	67.17%	81%	74.17%
MRR	50.21%	42.04%	52.5%	45.98%	57.69%	51.77%
$nDCG@15$	54.87%	46.19%	57.19%	50.29%	63.08%	56.93%

Table 9: CREST performance improvement over the single criterion-agnostic model in the two-stage (IR-RR) workflow. RR-stage result is computed using monoRoBERTa with *ColRoBERTa* as the preceding IR stag

ensembling, CREST achieves over 10.2% improvement for $R@5$ compared to the single model. This indicates a substantial improvement in retrieving relevant documents within the top 5 ranked results. Similarly, CREST achieves 9.7% and 10.0% improvement for $R@10$ and $R@15$, respectively, compared to the single model. CREST also shows 8.1% improvement in MRR and 8.6% improvement in $nDCG@15$. Using ColRoBERTa for IR, CREST remains superior to the single model with gains of 5.8% for $R@5$, 6.6% for $R@10$, 8.0% for $R@15$, 6.5% for MRR , and 6.9% for $nDCG@15$.

In the RR stage, CREST continues to outperform the single model across all metrics with a 6.4% improvement for $R@5$, 7.5% for $R@10$, and 6.8% for $R@15$. Compared to the criterion-agnostic single model, MRR and $nDCG@15$ also improve with CREST by 5.9% and 6.1%, respectively. These results show that the benefits of CREST persist across both ranking stages, consistently improving retrieval effectiveness.

Table 10 presents the performance of CREST compared to a single criterion-agnostic model, focusing solely on the cross-encoder (monoRoBERTa) without relying on the IR stage. While Table 6 showed limited gains for criterion-specific models in isolation, the ensemble approach in CREST yields clear improvements across all metrics. Notably, $R@5$ improves by 5.47%, $R@10$ by 7.08%, $R@15$ by 8.29%, MRR by 5.44%, and $nDCG@15$ by 6.09%. This suggests that breaking down the input based on TR criteria and assigning it to expert models is more effective than applying a single model to process the entire TR. Furthermore, it shows that results demonstrate that CREST effectively leverages the strengths of individual criterion-specific models and highlights the benefit of integrating multiple perspectives, even without the aid of the IR stage.

These findings indicate that CREST consistently outperforms the single model approach across all metrics in both the IR and RR stages, underscoring

Isolated monoRoBERTa		
	CREST	Single criterion-agnostic retrieval model
$R@5$	66.43%	60.96%
$R@10$	75.65%	68.57%
$R@15$	81.56%	73.27%
MRR	52.68%	47.24%
$nDCG@15$	58.99%	52.90%

Table 10: Performance of CREST and a single retrieval model.

the effectiveness of leveraging criterion-specific models for TR retrieval. By modeling each criterion independently and then aggregating their outputs, CREST can capture diverse and complementary signals that a single model might overlook. This leads to more accurate retrieval of relevant TRs, which can directly benefit the troubleshooting workflow.

A more accurate TR retrieval system helps engineers find previously resolved TRs that are more closely aligned with the current issue, improving the relevance and reliability of the suggested solutions. The improved performance of CREST is directly reflected in the utility of the CREST tool. Alongside retrieving more relevant TRs, the tool provides a breakdown of relevance scores across criteria, offering insight into why each result was selected. This makes the retrieval process more interpretable and actionable. For instance, if a match is primarily driven by “system impact” and “condition”, users can quickly assess its relevance to the new issue. This integration not only supports faster resolution but also helps teams focus on the most critical aspects of a problem, reducing manual effort and improving workflow efficiency.

In practice, this means that CREST doesn’t just retrieve better matches, it also supports decision-making by highlighting why those matches were selected. This capability helps engineers respond more efficiently, reduces trial-and-error in solution discovery, and enables more consistent handling of recurring issues.

That said, CREST does not always outperform a single model. Failures typically occur when the IR candidate set excludes the true match, a situation more likely with short or ambiguous queries or with sparse observations. In such cases, a single model may be preferable, as it processes the TR observation holistically, increasing query length and potentially reducing ambiguity by incorporating information overlooked by criterion-specific mod-

	Initial Retrieval	Re-Ranking
HTI	0.0287	0.0186
HTF	0.0308	0.0197
HTC	0.0304	0.0213
HTR	0.0344	0.0199
CREST	0.0249	0.0175
Single retrieval model	0.0345	0.0254

Table 11: Expected calibration error (ECE) for all models.

els. Despite quality assurance measures prior to TR publication, instances still arise where TR creators comply with all requirements yet provide descriptions that lack sufficient detail or contain ambiguity, negatively affecting both CREST and the single model. Another scenario in which a single model is advantageous arises when only a single criterion (for example, “system impact”) is present for a TR. Criterion-specific models become vulnerable to noise from that single criterion, with no others available to mitigate it. In such situations, a reasonable fallback is to adopt a criterion-agnostic single model.

5.3. Relevance Score Calibration Analysis (RQ3)

To better understand the quality of the predicted relevance scores and their alignment with actual outcomes, we analyze the calibration of each model using the Expected Calibration Error (ECE). A lower ECE indicates that the predicted probabilities better reflect the true likelihood of relevance, contributing to more trustworthy and reliable relevance scores for each criterion. This is particularly important in retrieval systems where confidence estimates play a role in guiding downstream decisions. Table 11 reports the ECE values for all models across both the initial retrieval and re-ranking stages.

As shown in Table 11, the criterion-specialized models consistently achieve lower ECE scores compared to the single retrieval model, with the difference being more notable in the re-ranking stage. This suggests that modeling relevance per criterion results in more reliable confidence estimation, without the need for additional calibration methods. Even though calibration was not explicitly targeted during training, the specialized models yield better alignment between predicted and actual relevance probabilities.

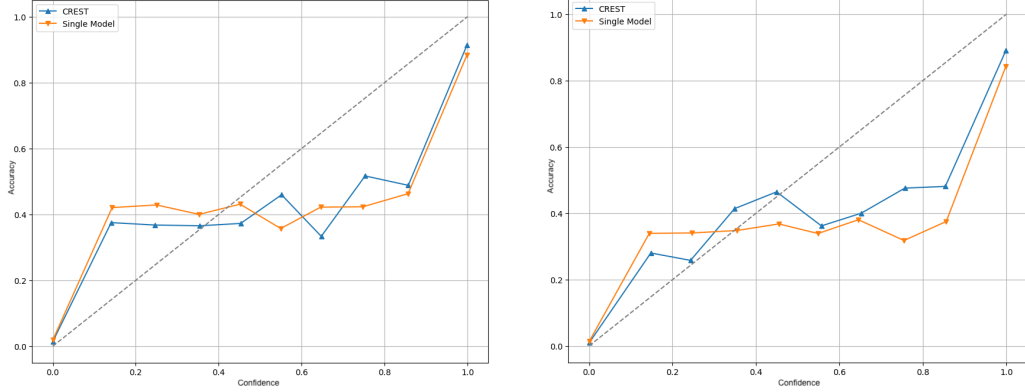


Figure 6: Calibration diagrams for initial retrieval (left) and re-ranking (right). The left plot shows the calibration performance of the initial retrieval stage, while the right plot illustrates the calibration diagram obtained after re-ranking.

Figure 6 shows the calibration diagrams for both stages, comparing the CREST model with the single criterion-agnostic retrieval model. In both cases, the ensemble model tracks more closely to the ideal calibration line, while the single model shows larger deviations. These trends are consistent with the ECE results and indicate that combining criterion-specific predictions leads to a better confidence calibration of relevance scores.

This makes the confidence scores easier to interpret, since they can be read as reliable probabilities rather than opaque values. Engineers can then set risk-aware thresholds, decide when to auto-suggest or defer to manual review, and identify which criterion drives a match with dependable certainty. It also lowers the risk of over-confidence and reduces the chance of irrelevant evidence being passed to downstream components such as re-rankers or RAG-style assistants.

Together, these findings highlight that leveraging criterion-specific models improves not only retrieval performance but also the calibration and reliability of the predicted scores. This is particularly important when transparency in the decision-making process is a key requirement.

5.4. Pilot User Study (RQ4)

We complemented the offline evaluation with a small pilot to check whether the gains we observe translate into practice. The goal was to understand if the criterion-wise scores make the ranking more transparent, whether the

Participants	TR1			TR2			TR3			TR4			TR5		
	P1	P2	P3	P1	P2	P3	P1	P2	P3	P1	P2	P3	P1	P2	P3
Explainability of Scoring	Good	Good	Good	Good	Fair	Fair	Good	Fair	Very Poor	Good	Fair	Fair	Good	Fair	Good
Helpfulness of Ranking	Good	Good	Very Poor	Good	Fair	Fair	Good	Fair	Good	Good	Good	Good	Excellent	Fair	Good
Trustworthiness of Ranking	Good	Fair	Very Poor	Good	Fair	Fair	Good	Good	Fair	Very Poor	Good	Good	Good	Fair	Good
Accuracy of Ranking	Excellent	Fair	Very Poor	Excellent	Good	Good	Excellent	Good	Fair	Very Poor	Good	Good	Good	Fair	Good

Table 12: Participants’ (P1–P3) ratings for five TRs (TR1–TR5).

recommendations are useful during triage, and how credible and accurate the top results appear to possible end users.

This study was conducted at Ericsson with three practitioners and five real-life TRs. All participants had QA testing backgrounds: two were actively involved in QA testing at the time of the study, and one was not currently testing but had prior QA experience. One participant had less than five years of experience at Ericsson, and the other two had more than five years. For each TR, participants first read its description and then reviewed the top five candidate TRs retrieved by CREST. Each candidate was presented with both the aggregated CREST score and the detailed criterion-wise scores, allowing participants to examine how individual criteria contributed to the final relevance score. They then rated (i) explainability of scoring, (ii) helpfulness for triage tasks (such as root-cause clues, mitigation, symptom matching, and TR authoring), (iii) trustworthiness, and (iv) perceived accuracy, using a six-point scale ranging from “Not at all” to “Excellent”. Participants were given one week to complete the evaluation at their own pace and on average spent about 15 minutes evaluating each TR. At the end of the evaluation, they were also asked to share their overall impressions of CREST. Given the small number of participants and TRs, this pilot should be viewed as exploratory and intended to provide initial qualitative insights.

Table 12 reports the results of the pilot user study. With 15 ratings per evaluation metric, *explainability of scoring* was ranked as mostly *Good* (8/15, 53%) or *Fair* (6/15, 40%), indicating that criterion-wise scores improved transparency, yet leaving room for clearer presentation. *Helpfulness of ranking* was *Good/Excellent* in 10/15 (67%), suggesting the provided TR lists often surfaced actionable cues. *Trustworthiness* was perceived as *Good* in 8/15 (53%) and *Fair* in 5/15 (33%). Lastly, perceived *accuracy* was *Good/Excellent* in 10/15 (67%), with a minority of *Very Poor* judgments (2/15). We did not observe any “Not at all” ratings for any metric.

Differences across TRs were mainly explained by input quality, with sparse or unclear criteria affecting CREST’s performance. Findings consistent with both performance and calibration gains observed in offline ex-

periments. Moreover, explainability benefited from the criterion-wise scores, which participants used to interpret why a given TR surfaced and to justify keeping or discarding specific candidates.

Participants feedback on negative cases reinforces our earlier findings, highlighting the strength of CREST when the criteria are informative and its weaker performance when they are brief or ambiguous. Moreover, participant P3 noted that the scoring explanations allowed them to justify selecting the most relevant TR from the candidate list provided, illustrating how transparency in the rankings can support more informed decision-making.

6. Discussion

We now discuss the implications of our work (Section 6.1) and address the threats to validity and how we mitigated them (Section 6.2).

6.1. Implications

The proposed CREST model has practical implications for improving the efficiency and reliability of quality assurance (QA) and design teams at Ericsson. By offering a more accurate and interpretable retrieval of relevant trouble reports (TRs), CREST can assist engineers in identifying related past issues more effectively, thereby accelerating the fault resolution process. This is particularly valuable in complex systems where understanding the context and history of software faults plays a crucial role in diagnosing and resolving new incidents.

One of the key benefits of CREST is its ability to surface criterion-specific relevance scores, helping users understand which aspects of a TR (e.g., functional area, fault type, or impacted component) contributed most to the retrieval outcome. This transparency can guide engineers in validating the retrieved TRs, increase their trust in the system, and potentially uncover new resolution strategies based on previously overlooked criteria.

From a practical standpoint, CREST can be integrated into existing TR retrieval workflows within Ericsson with minimal disruption. Depending on the requirements of the task, CREST can be used in the Initial Retrieval stage, especially for latency-sensitive applications, or extended in both stages of a two-stage pipeline to fully leverage its benefits. In addition to its role in TR retrieval, CREST can also serve as a retrieval engine for Retrieval-Augmented Generation (RAG) systems tailored to TR-related tasks. In this

context, CREST’s interpretable retrieval helps explain why specific information sources were selected and how they contributed to the final output, enhancing both the transparency and reliability of the generated responses.

Overall, CREST offers a practical and effective enhancement to existing TR retrieval systems, enabling Ericsson teams to reduce resolution times, improve traceability, and support more informed decision-making during software maintenance and troubleshooting.

6.2. Threats to Validity

Several factors may threaten the validity of our findings. First, the internal data from Ericsson is proprietary and cannot be publicly shared due to non-disclosure obligations. This data may not generalize to other telecommunication environments, limiting external validity due to different TR characteristics and structures.

Second, the quality of TRs and parsing tools is critical and can impact model performance, especially during deployment. Inconsistencies in these tools or data can influence the effectiveness of the TR retrieval system.

Third, criterion-specific datasets may not contain TRs of similar quality, resulting in performance variations. We introduced a baseline to isolate the effect of each criterion, but differences may still affect results.

Fourth, CREST is designed to enhance explainability by exposing per-criterion relevance scores. To assess its practical value, a pilot user study was conducted. The results suggest that per-criterion scores support sense-making and traceability of rankings, highlighting CREST’s potential for interpretability and transparency. However, the study was limited in scale and scope, and a larger, more comprehensive evaluation involving broader user groups is needed and left as a possible future direction.

Moreover, due to Ericsson’s policy and resource limitation challenge, we have only evaluated our approach with an internally trained RoBERTa model, and as a result, the impact of the CREST approach may vary across different large language models (LLMs). Additionally, the effectiveness of the CREST approach may not be consistent across various TR retrieval applications, which limits the generalizability of our findings.

While we acknowledge these limitations, we hope to address them in future research to validate and generalize our findings.

7. Conclusion

This study investigates the impact of various trouble report criteria on the performance of the Initial Retrieval (IR) and Re-Ranking (RR) stages within the TR retrieval system. By utilizing a bi-encoder in the IR stage and a cross-encoder in the RR stage, we were able to evaluate each criterion’s influence in a comprehensive two-stage workflow. Notably, criteria such as “system impact” (HTI) significantly improved recall and ranking metrics during the IR stage, whereas “reproducibility” (HTR) negatively influenced the IR stage but showed positive effects in the RR stage, highlighting that different stages benefit from different types of information. This illustrates the importance of selectively parsing and utilizing specific information to enhance retrieval performance, as opposed to leveraging all available data indiscriminately. The standalone evaluation of each criterion’s impact also highlighted the benefits of criterion-specific modeling, especially in re-ranking, where the detailed reasoning capabilities of cross-encoders are better suited to exploit criterion-specific signals.

The proposed Criteria-specific Retrieval via Ensemble of Specialized TR models (CREST) demonstrates a significant advancement in TR retrieval approaches. By training each model within the ensemble to focus on a unique criterion, we cultivated specialization in handling specific types of information. This specialization allowed CREST to effectively combine the diverse strengths of each model, leading to improved overall performance. The findings of this study demonstrate that CREST consistently outperforms single-model approaches across key metrics in both IR and RR stages when applied in a two-stage workflow. In addition, even when evaluated in isolation (i.e., without the IR stage), CREST continued to outperform the single-model approach across all metrics, reinforcing the strength of criterion-specific modeling in constrained input settings. Moreover, CREST improves the calibration of predicted confidence scores, resulting in outputs that are not only more precise but also more reliable, which is an important factor when transparency and interpretability are critical in decision-making processes. Finally, the pilot user study demonstrates that CREST’s criterion-wise scores can improve perceived transparency and that the recommendations were often judged useful, credible, and accurate for triage, with some responses also suggesting areas for further refinement.

In summary, CREST substantially enhances the accuracy and efficiency of TR retrieval systems on our industrial dataset. Beyond improved retrieval

accuracy, the CREST tool can also support practical decision-making by providing criterion-wise relevance scores for each retrieved TR, making the retrieval process interpretable and traceable. This capability enables engineers to better understand the match rationale, prioritize investigation based on critical factors, and ultimately accelerate issue resolution. Future research should aim to refine criteria aggregation methods and explore the integration of non-TR sources and non-textual information to further elevate TR retrieval system performance. These advancements hold the potential to significantly improve the capabilities and effectiveness of TR retrieval systems in industrial applications.

References

- Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al., 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774 .
- Aggarwal, A., Mittal, M., Battineni, G., 2021. Generative adversarial network: An overview of theory and applications. *International Journal of Information Management Data Insights* 1, 100004. doi:10.1016/j.jjime.2020.100004.
- Anand, A., Lyu, L., Idahl, M., Wang, Y., Wallat, J., Zhang, Z., 2022. Explainable information retrieval: A survey. arXiv preprint arXiv:2211.02405 .
- Bosch, N., Shalmashi, S., Yaghoubi, F., Holm, H., Gaim, F., Payberah, A.H., 2022. Fine-tuning bert-based language models for duplicate trouble report retrieval, in: *2022 IEEE International Conference on Big Data (Big Data)*, pp. 4737–4745.
- Chicco, D., 2021. Siamese Neural Networks: An Overview. US, New York, NY. doi:10.1007/978-1-0716-0826-5_3.
- Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding, in: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies*, volume 1 (long and short papers), pp. 4171–4186.

- Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al., 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783 .
- Grimalt, N.M.I., Shalmashi, S., Yaghoubi, F., Jonsson, L., Payberah, A.H., 2022. Berticsson: A recommender system for troubleshooting. SDU@AAAI .
- Gururangan, S., Lewis, M., Holtzman, A., Smith, N.A., Zettlemoyer, L., 2022. Demix layers: Disentangling domains for modular language modeling, in: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5557–5576.
- Holm, H., 2021. Bidirectional encoder representations from transformers (bert) for question answering in the telecom domain. Master’s thesis. KTH, School of Electrical Engineering and Computer Science (EECS).
- Humeau, S., Shuster, K., Lachaux, M.A., Weston, J., 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring, in: Proceeding of 8th International Conference on Learning Representations, 2020.
- Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M.A., Stock, P., Scao, T.L., Lavril, T., Wang, T., Lacroix, T., Sayed, W.E., 2023. Mistral 7b. URL: <https://arxiv.org/abs/2310.06825>, arXiv:2310.06825.
- Jung, E., Choi, J., Rhee, W., 2022. Semi-siamese bi-encoder neural ranking model using lightweight fine-tuning, in: Proceedings of the ACM Web Conference 2022, pp. 502–511. doi:10.1145/3485447.3511978.
- Karapantelakis, A., Thakur, M., Nikou, A., Moradi, F., Olrog, C., Gaim, F., Holm, H., Nimara, D.D., Huang, V., 2024. Using large language models to understand telecom standards, in: 2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN), IEEE. pp. 440–446.
- Khattab, O., Zaharia, M., 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert, in: Proceedings of the

- 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 39–48. doi:10.1145/3397271.3401075.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al., 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences* 114, 3521–3526.
- Leonhardt, J., Rudra, K., Anand, A., 2023. Extractive explanations for interpretable text ranking. *ACM Transactions on Information Systems* 41, 1–31.
- Li, M., Gururangan, S., Dettmers, T., Lewis, M., Althoff, T., Smith, N.A., Zettlemoyer, L., 2022. Branch-train-merge: Embarrassingly parallel training of expert language models, in: *First Workshop on Interpolation Regularizers and Beyond at NeurIPS 2022*.
- Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., et al., 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V., 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv:1907.11692*. *arXiv preprint*.
- Lu, W., Jiao, J., Zhang, R., 2020. Twinbert: Distilling knowledge to twin-structured compressed bert models for large-scale retrieval, in: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 2645–2652.
- Lucchese, C., Minello, G., Nardini, F.M., Orlando, S., Perego, R., Veneri, A., 2023. Can embeddings analysis explain large language model ranking?, in: *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 4150–4154.
- Ma, X., Wang, L., Yang, N., Wei, F., Lin, J., 2024. Fine-tuning llama for multi-stage text retrieval, in: *Proceedings Of The 47th International ACM SIGIR Conference On Research And Development In Information Retrieval*, pp. 2421–2425. doi:10.1145/3626772.3657951.

- Naeini, M.P., Cooper, G., Hauskrecht, M., 2015. Obtaining well calibrated probabilities using bayesian binning, in: Proceedings of the AAAI conference on artificial intelligence.
- Nimara, D.D., Gebre, F.G., Huang, V., 2024. Entity recognition in telecommunications using domain-adapted language models, in: 2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN), IEEE. pp. 240–245.
- Nogueira, R., Cho, K., 2019. Passage re-ranking with bert. `arXiv:1901.04085`. arXiv preprint.
- Nogueira, R., Yang, W., Cho, K., Lin, J., 2019. Multi-stage document ranking with bert. `arXiv:1910.14424`. arXiv preprint.
- Penha, G., Hauff, C., 2021a. On the calibration and uncertainty of neural learning to rank models for conversational search, in: Merlo, P., Tiedemann, J., Tsarfaty, R. (Eds.), Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Association for Computational Linguistics, Online. pp. 160–170. URL: <https://aclanthology.org/2021.eacl-main.12/>, doi:10.18653/v1/2021.eacl-main.12.
- Penha, G., Hauff, C., 2021b. On the calibration and uncertainty of neural learning to rank models for conversational search, in: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pp. 160–170.
- Puthenputhussery, A., Kang, C., Magnani, A., Zhang, T., Shang, H., Yadav, N., Chandran, P., Madhani, B., Fu, Y.T., Wang, H., et al., 2025. Large scale deployment of bert based cross encoder model for re-ranking in walmart search engine, in: Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 4365–4369.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* 21, 1–67.

- Reimers, N., Gurevych, I., 2019. Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3982–3992. doi:10.18653/v1/D19-1410.
- Ren, R., Qu, Y., Liu, J., Zhao, W.X., She, Q., Wu, H., Wang, H., Wen, J.R., 2021. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 2825–2835. doi:10.18653/v1/2021.emnlp-main.224.
- Robertson, S., Zaragoza, H., 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.* 3, 333–389. doi:10.1561/15000000019.
- Ruder, S., Peters, M.E., Swayamdipta, S., Wolf, T., 2019. Transfer learning in natural language processing, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials, pp. 15–18. doi:10.18653/v1/N19-5004.
- Rudin, C., 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence* 1, 206–215.
- Song, C., He, H., Yu, H., Fang, P., Cui, L., Lan, Z., 2023. Uni-encoder: A fast and accurate response selection paradigm for generation-based dialogue systems. *Findings of the Association for Computational Linguistics: ACL 2023*, 6231–6244. doi:10.18653/v1/2023.findings-acl.388.
- Tam, W., Liu, X., Ji, K., Xue, L., Liu, J., Li, T., Dong, Y., Tang, J., 2023a. Parameter-efficient prompt tuning makes generalized and calibrated neural text retrievers, in: *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 13117–13130.
- Tam, W.L., Liu, X., Ji, K., Xue, L., Zhang, X., Dong, Y., Liu, J., Hu, M., Tang, J., 2023b. Parameter-efficient prompt tuning makes generalized and calibrated neural text retrievers. *Findings of the Association for Computational Linguistics: EMNLP 2023*, 13117–13130. doi:10.18653/v1/2023.findings-emnlp.87.

- Wallat, J., Beringer, F., Anand, A., Anand, A., 2023. Probing bert for ranking abilities, in: European Conference on Information Retrieval, Springer. pp. 255–273.
- Wang, J.A., Wang, K., Wang, X., Naidu, P., Bergen, L., Paturi, R., 2023. Scientific document retrieval using multi-level aspect-based queries. *Advances in Neural Information Processing Systems* 36, 38404–38419.
- Wang, Y., Chen, X., Verberne, S., 2024. Quids: Query intent generation via dual space modeling. *arXiv preprint arXiv:2410.12400* .
- Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., Qiu, Z., 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115* .
- Yu, P., Cohen, D., Lamba, H., Tetreault, J., Jaimes, A., 2024. Explain then rank: Scale calibration of neural rankers using natural language explanations from large language models. *arXiv e-prints* , arXiv–2402.
- Yu, W., Sun, Z., Xu, J., Dong, Z., Chen, X., Xu, H., Wen, J.R., 2022. Explainable legal case matching via inverse optimal transport-based rationale extraction, in: Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval, pp. 657–668.
- Zhang, H., Gong, Y., Shen, Y., Lv, J., Duan, N., Chen, W., 2021a. Adversarial retriever-ranker for dense text retrieval. *arXiv:vol. abs/2110.03611*.
- Zhang, R., Guo, J., Fan, Y., Lan, Y., Cheng, X., 2020. Query understanding via intent description generation, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 1823–1832.
- Zhang, Y., Long, D., Xu, G., Xie, P., 2022. Hlatr: enhance multi-stage text retrieval with hybrid list aware transformer reranking. *arXiv:2205.10569*. *arXiv preprint*.

- Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., Liu, Q., 2019. Ernie: Enhanced language representation with informative entities, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Jul. 2019, pp. 1441–1451. doi:10.18653/v1/P19-1139.
- Zhang, Z., Rudra, K., Anand, A., 2021b. Explain and predict, and then predict again, in: Proceedings of the 14th ACM international conference on web search and data mining, pp. 418–426.