Leveraging LLM for Enhanced Incident Management in Wireless Networks

Md. Shamim Towhid*, Nasik Sami Khan*, Nashid Shahriar*, Massimo Tornatore†, Raouf Boutaba‡,

Aladdin Saleh§

*Department of Computer Science, University of Regina, {mty754, nku618, nashid.shahriar}@uregina.ca

†Politecnico di Milano, {massimo.tornatore}@polimi.it

‡David R. Cheriton School of Computer Science, University of Waterloo, {rboutaba}@uwaterloo.ca

§Rogers Communications Canada Inc., {aladdin.saleh}@rci.rogers.com

***Abstract*—Incident management in telecommunications networks generates large volumes of incident management tickets (IMTs), each containing heterogeneous and often unstructured text describing service outages, performance degradations, or security issues. Accurately categorizing these IMTs into multiple impact and cause labels is essential for rapid diagnosis and resolution. However, existing rule-based and standard language-model-based approaches struggle with noisy data, overlapping categories, and limited contextual understanding. To address these challenges, we propose two complementary solutions for automated multi-label classification of IMTs. To mitigate the effects of noisy data and overlapping categories, the first solution employs an encoder-based language model (i.e., Bidirectional Encoder Representations from Transformers (BERT)) with a relevance-guided feature selection strategy that focuses on semantically meaningful attributes. To improve contextual understanding and label consistency, the second solution leverages a decoder-based large language model (i.e., Phi-3.5) enhanced with retrieval-augmented generation (RAG) and a novel probabilistic re-ranking mechanism to refine label predictions. Experimental results show that our encoder-only model achieves an F1 score of 79.20%, while our RAG-enhanced decoder model achieves 94.98%, outperforming traditional machine learning models and BERT baselines by 23.59% and 29% on average, respectively. These findings demonstrate that combining fine-tuned language models with intelligent retrieval and re-ranking significantly improves classification accuracy in incident management systems.**

***Index Terms*—incident management, large language model, imbalanced data, root cause, deep learning, automation**

I. Introduction

Incident management tickets (IMTs) are essential for maintaining operational efficiency and service reliability in large-scale telecommunication networks. Telecom operators depend on IMTs to record incidents, assess their severity, identify root causes, and implement timely resolutions to minimize network disruptions and maintain quality of service [1]. Each IMT typically contains attributes such as a short textual description, network and service impact indicators, identified root cause, and the final resolution. However, the process of accurately identifying and classifying these attributes, particularly root causes and appropriate resolutions, is highly challenging due to the large volume of tickets, inconsistent textual formats, and varying levels of detail. Automated IMT generation processes further exacerbate the issue by introducing false positives and redundant entries, which strain operational resources and delay incident resolution.

A key operational requirement of incident management in large-scale telecommunication systems is the ability to predict multiple IMT attributes, such as network impact, service impact, and root cause, immediately upon ticket creation using only the initial textual description. From a communication systems perspective, these attributes directly correspond to network-layer failures, service-level agreement (SLA) violations, and cross-domain fault propagation across access, core, and transport networks. Early and accurate prediction of such attributes enables proactive network control actions, faster fault isolation, and improved resilience of carrier-grade infrastructures. Such predictive capability can accelerate incident triage, improve prioritization of critical issues, and guide faster root-cause analysis. However, achieving accurate predictions is non-trivial because historical IMT data exhibits several challenges: (i) significant class imbalance, where high-impact incidents are rare compared to routine cases, (ii) the presence of domain-specific and often abbreviated technical language, and (iii) high contextual variability between IMTs even within the same category.

Conventional supervised learning approaches, including Naïve Bayes, k-Nearest Neighbors, and Support Vector Machines [2]–[4], require extensive manual feature engineering and struggle to capture the semantic dependencies between network components, services, and failure modes described in IMT text [5]. Similarly, traditional keyword-based or rule-based methods used in many operational support systems (OSS) [6]–[8] fail to generalize beyond predefined fault patterns and cannot adapt to evolving network architectures, emerging technologies (e.g., 5G/6G), or previously unseen incident types.

Recent advances in natural language processing (NLP) through large language models (LLMs) offer a promising path to overcome these limitations. Unlike conventional models, LLMs can leverage contextual embeddings to understand nuanced domain language, enabling them to generalize from limited examples and infer relationships between incident attributes. Their ability to perform transfer learning and in-

context reasoning makes them particularly suitable for handling unstructured and noisy telecom data, where explicit annotations are often sparse [9], [10]. Furthermore, open-source LLMs provide an accessible and customizable alternative to proprietary telecom-specific solutions, allowing operators to deploy advanced NLP capabilities without relying on closed systems or extensive labeled datasets.

In this paper, we investigate the potential of open-source LLMs to understand telecom domain language and perform multi-label classification of IMTs. We call it multi-label classification because we are aiming to predict multiple attributes of IMT at the same time. Specifically, we explore both encoder-only (e.g., BERT, RoBERTa) and decoder-only (e.g., Phi-3.5, Falcon, MPT) architectures. Transformer-based models can be broadly categorized by their use of an encoder, which produces contextualized representations of the input sequence, and/or a decoder, which generates outputs autoregressively. Accordingly, we explore both encoder-only (e.g., BERT) and decoder-only (e.g., Phi-3.5, Falcon, MPT) architectures. The encoder-only approach leverages contextual embeddings from transformer layers for classification via a task-specific prediction head. Although decoder-only models are primarily trained for text generation, they can also be adapted for classification. We evaluate decoder-only architectures using two strategies: direct text generation of IMT attribute predictions, and feature-based classification using hidden representations extracted from the decoder's transformer layers.

To enhance prediction accuracy and mitigate the adverse effects of class imbalance inherent in telecom IMT datasets, we integrate our previously proposed Bayesian relevance-guided feature selection strategy [11] with both encoder-only and decoder-only architectures. This method prioritizes the tokens which are most indicative of specific IMT attributes, improving performance without extensive fine-tuning. Our study leverages a comprehensive dataset of real IMTs collected from the wireless platform of a major telecommunication operator in Canada. Our empirical evaluation compares the performance of encoder-only and decoder-only LLM architectures, analyzing their strengths and limitations in modeling domain-specific telecom language and predicting multiple IMT attributes. The findings provide new insights into the applicability of open-source LLMs for large-scale, real-world telecom incident management.

In summary, the key contributions of this paper are as follows:

- We present a comprehensive empirical analysis of open-source encoder-only and decoder-only LLMs for the multi-label classification of telecom IMTs. This study is among the first to benchmark both architectures with a real-world telecom dataset, highlighting their relative strengths for domain-specific language understanding.
- We integrate a Bayesian posterior probability-based feature selection strategy into the encoder-based approach to select the most relevant tokens for classification and into the decoder-based approach to re-rank retrieved tickets, thereby mitigating class imbalance and enhancing classification accuracy. This relevance-guided strategy improves the interpretability and robustness of both encoder and decoder models without requiring extensive domain-specific fine-tuning.
- We propose and evaluate two complementary uses of decoder-only architectures, (i) a *generative text-based* prediction mode and (ii) a *transformer feature-based* classification mode, providing new insights into how generative and discriminative paradigms perform in telecom IMT prediction tasks.
- Using a large dataset of real IMTs from a major Canadian telecom operator, we conduct extensive experiments demonstrating that our LLM-based methods substantially outperform traditional baselines in terms of F1 score and overall predictive reliability.
- We apply the Local Interpretable Model-agnostic Explanations (LIME) [12] technique to explain how LLMs capture telecom-specific terminology and relationships in the IMT classification task, offering valuable interpretability for operational deployment.

The remainder of this paper is structured as follows. Section II provides a review of relevant literature. Our proposed methodologies are detailed in Section IV. Section V presents experimental results and performance analysis. We discuss findings, limitations, and future research directions in Section VI. Finally, Section VII concludes with a summary of the work.

II. Related Works

In this section, we discuss state-of-the-art work focusing on problems that are similar to ours, such as extracting relevant features in text classification, adaptation of custom-domain knowledge, approaches to tackle multi-label classification problems, and the implementation of large language models and generative AI in the networking domain and text classification task. The objective is to identify how existing methods address the same problems and to highlight the remaining gaps our proposed approach fills.

A. Feature Extraction

In a text classification problem, selecting the best features from the words in the embedding space is a vital task that impacts the overall result of the classification model. Authors in [6] present a method for selecting the first token using the BERT model, which is used to better capture the linguistic nuances and increase performance on a wide range of NLP tasks. The study in [7] employs a combined strategy of numerical augmentation and the BERT model to address the problem of data imbalance.

Another common way to concentrate on data imbalance issues is discussed in [13] by employing a weighed loss mechanism, which we combined with BERT in this paper. Authors in [14] introduce a three-way model that divides the description space into confirmatory, disconfirmatory, and neutral regions for evaluating confirmation in classifications. These regions are used to establish classification rules for acceptance, rejection, and non-commitment. To enhance accuracy, the proposed framework utilizes Bayesian confirmation theory, where the neutral region can be refined through a

sequential model that employs attributes or attribute-value pairs. This method offers a practical solution for evaluating hypotheses based on evidence, potentially leading to improved and dependable classification results by splitting characteristics or attribute-value pairs into useful trisections based on their utility. The token prioritization technique in our research is inspired by this study.

B. Domain Adaptation

Incident tickets often consist of domain-specific words that are unknown or irrelevant to general English vocabulary. This makes it difficult for pre-trained language models to learn the pattern and classify accurately.

To classify domain-specific log messages, the study in [15] enhances the word embedding-based neural network's adaptability by focusing on domain-specific vocabulary, word-level and character-level information. The approach divides log messages into templates based on contextual similarity and uses volatile tokens to generalize similar words with minor differences by masking the keywords in place of volatile tokens. This helps analyze and effectively categorize log messages by reducing feature dimensionality. Another method for fault localization that employs word embeddings to convey semantic relationships in IT infrastructure event data is discussed in [16]. The method uses transfer learning to cluster extracted vectors based on semantic similarity, enhancing online fault detection by gradually adding domain-specific token sequences to generic word embeddings. Authors of [17] propose a multimodal deep-learning framework for the classification of short texts into multiple classes using an imbalanced and extremely small dataset. Domain adaptation strategies in the above-mentioned studies lack scalability for continuously evolving telecom terminology. Our approach to fine-tune both the tokenizer and transformer layers on the IMT dataset expands the domain vocabulary and enables better generalization to unseen incidents.

C. Approaches of Multi-label Text Classification

Multi-label text classification is of particular interest in this paper because IMT descriptions in the telecom domain often convey multiple, interdependent attributes within a single text instance, making independent single-label predictions insufficient. Accurately capturing these co-occurring attributes is essential for downstream analysis and automated understanding of IMTs. The performance of traditional approaches for multi-label text classification varies according to the characteristics of the dataset, such as label density, label correlation, class imbalance, and the dimensionality of the feature space. The authors in [18] propose Binary Relevance, which converts the task into multiple binary classifications. Label Powerset, instead, treats each unique label combination as a single class for multi-class classification. Classifier Chain links binary classifiers sequentially to model label dependencies but is order-sensitive and computationally costly. Neural models like CNN-RNN and Seq2Seq integrate convolutional and recurrent networks to capture semantic features [18].

The study in [19] presents ML-Reasoner, which predicts all labels simultaneously, avoiding order sensitivity and improving multi-label performance. In [20], 26 techniques were tested on 42 datasets, showing Random Forest of Predictive Clustering Trees (RFPCT), Binary Relevance with Random Forest of Decision Trees (RFDTBR), and Ensemble of Classifier Chains as the best performers. The authors in [21] propose XML-CNN, a deep model for extreme multi-label tasks. Although our dataset is not extreme, it faces class imbalance, requiring a new accuracy-focused approach. In [22], deep learning effectively classifies multi-label incidents from social media using CNN, Contextual Long Short-Term Memory (CLSTM), and Region-based CNN (RCNN). The study in [23] proposes a Hierarchical Attention-based RNN (HARNN) for hierarchical multi-label text classification, combining attention and top-down mechanisms to capture dependencies between hierarchy levels. The authors in [24] present X-BERT, which enhances BERT through semantic label indexing, neural matching, and ensemble ranking to cluster labels semantically and leverage language representations. Similarly, [25] introduces XR-Transformer, a transformer-based model designed to handle challenges in extreme multi-label classification. Existing multi-label text classification models are not optimized for highly imbalanced, domain-specific telecom data, where the number of labels is moderate but the class distribution is skewed. Our paper addresses this gap by combining multi-label modeling with relevance-guided feature selection and fine-tuned LLM architectures.

D. Applications of Large Language Models

LLMs have been applied across diverse domains, demonstrating impressive versatility and reasoning capabilities [26]. In multi-label classification tasks, LLMs and generative AI improve prediction accuracy by enabling the simultaneous prediction of multiple categories from a single input. They exhibit strong zero-shot and few-shot reasoning abilities, performing well on varied tasks without extensive task-specific data or fine-tuning [8]. However, despite their generalization power, LLMs face challenges when applied to domain-specific problems such as telecom incident management. Their limitations include difficulty in handling domain-specific terminology, data imbalance, and context fragmentation within lengthy or noisy incident descriptions. Furthermore, without targeted adaptation, LLMs often produce hallucinated or inconsistent classifications, especially when multiple overlapping labels must be predicted. In the following, we discuss how general-purpose LLMs are customized to address different problems in the networking domain.

The study in [27] proposes a Generative Pre-trained Transformer framework, which enhances text classification using adaptive boosting and recurrent ensembling of LLMs. Researchers have also explored domain-specific adaptation such as, TelecomGPT [28], which customizes general-purpose LLMs for telecom applications through a three-stage domain adaptation process. While such approaches improve contextual relevance, they typically rely on proprietary data and closed-source architectures, limiting reproducibility and accessibility.

The authors in [29] and [30] highlight the role of LLMs and multi-agent systems in managing complex network slicing and incident management in large-scale cloud environments. Their work analyzes common root causes, detection failures, and mitigation strategies to find the gaps within the existing incident management framework. Similarly, [31] and [32] show how LLMs can support hypothesis generation and mitigation planning, with RCACopilot automating root cause analysis. However, these systems focus on automation and reasoning, not classification of incident data. In contrast, our work targets the challenge of classifying real-world telecom incidents, where ambiguity, overlapping categories, and unbalanced label distributions persist.

Recent studies have also explored LLM applications in the telecom domain for standards interpretation, customer support, and retrieval-augmented comprehension [33]. For instance, the authors in [34] and [35] introduce a RAG system fine-tuned for 3GPP queries, while other works emphasize multimodal data integration [36] and [37]. Although these studies demonstrate domain adaptation potential, they do not address multi-label incident classification or the limitations of general-purpose, open-source LLMs in such settings.

In this paper, we bridge these gaps by explicitly addressing the identified limitations of LLMs in telecom incident management. Our contributions include: (i) mitigating domain misalignment through probabilistic feature selection tailored to telecom-specific vocabulary, (ii) reducing classification inconsistency by adapting tokenization and retrieval-augmented prompting to maintain contextual relevance, and (iii) enhancing interpretability and reproducibility by leveraging open-source LLMs within an accessible retrieval-augmented classification framework. The proposed approach implicitly mitigates data imbalance by assigning higher probabilistic relevance to minority-class instances during retrieval. It further alleviates contextual inconsistencies through evidence-based prompting that aligns responses with retrieved domain content. Finally, hallucination is reduced by constraining the generation process to information with high probabilistic relevance, ensuring that output remains consistent with verified, domain-specific knowledge rather than the LLM's pre-trained parameters. Together, these contributions allow general-purpose LLMs to perform reliable, explainable, and domain-aware multi-label classification of real-world incident data.

III. Preliminary

In this section, we introduce the dataset used in our experiments and all the mathematical notations used to describe our proposed methods in the next section. This section also discusses the probabilistic formula used to calculate the relevance score in our proposed method.

A. IMT Dataset

We leverage a repository of a large number of IMTs from a major telecommunication operator in Canada. All IMTs are collected from the wireless network platform over four months. There are 7,447 IMTs in the dataset. We select four crucial attributes of IMT ("Network Impact", "Service Impact", "Root Cause", "Resolution") to be predicted using encoder-only and decoder-only language models. Each attribute has different values in it. Table I shows the possible values for each selected attribute of IMT. The number of samples in each category is shown in Table I. From Table I, it is clear that the dataset is imbalanced.

Attribute	Value	Number of Samples
Network Impact	Outage	2969
	Degraded	3457
	Threatened	1089
Service Impact	Outage	3296
	Degraded	3743
Root Cause	Hardware failure	2475
	Commercial power failure	1122
	Cause not identified	1249
	Mother nature	529
	Software failure	450
	No fault found	586
	Facilities - Environment	337
	Change management activity	210
	Fiber	191
	Third party	146
	Automation	418
Resolution	Network/Service validation	3398
	Hydro restored	863
	Repaired	851
	Replaced	750
	Reboot	691
	Re-set	387
	Reconfigured	294
	Temp powered	156
	Linked ticket	137
	Re-Set Up / Re-alignment	110
	Alarm cleared	73
	Spliced	55

TABLE I: Class distribution of the IMT dataset

Each IMT contains a field called "Description", which provides a textual summary of the incident. We use this description as the input to our models. Based on this input, the model classifies the ticket into appropriate categories for four key attributes: "Network Impact", "Service Impact", "Root Cause", and "Resolution". For example, a sample IMT description such as "Customers are experiencing slow internet connectivity in the area covering $< siteIDs >$" may be classified with the following labels: Network Impact: Degraded, Service Impact: Degraded, Root Cause: Hardware Fault, and Resolution: Replaced.

As part of the data cleaning process, we first remove the case sensitivity from the data by converting all the text to lowercase. As our dataset is collected from a real operator's IMT repository, there are several challenges in the text-cleaning process, such as occurrences of NULL and duplicate values. We remove the tickets with NULL values and duplicate tickets. Afterwards, we use several regular expressions to remove repeated punctuation and characters that are not meaningful for classification, i.e., new line, HTML tag. Furthermore, we replace sensitive but meaningful information, such as IP addresses and locations, with a volatile token [38] to ensure data privacy.

B. Probabilistic relevance score calculation

We leverage a probabilistic relevance score that considers how important a word is for the classification of a specific class

Variables / Nomenclature	Description
$C_{i,a}$	i^{th} class for an attribute (a) of IMT
T_j	j^{th} token in a sample data
$R_{j,a}$	Relevance score for j^{th} token for attribute a
L	Total number of classes for an attribute
H	Average cross-entropy loss
N	Number of selected tokens based on $R_{j,a}$ in the encoder-only approach
K	Number of re-ranked tickets from the vector database based on $R_{j,a}$
SC-BERT	BERT model with single classifier to predict all IMT attributes
MC-BERT	BERT model with multiple classifiers, each for one attribute of IMT
FT-BERT	Fine-tuned BERT model without any feature selection
RS-FT-BERT	Fine-tuned BERT model with relevance score as feature selection
SC-RoBERTa	RoBERTa model with single classifier to predict all IMT attributes
MC-RoBERTa	RoBERTa model with multiple classifiers, each for one attribute of IMT
RS-FT-RoBERTa	Fine-tuned RoBERTa model with relevance score as feature selection
TG-(LLM)	Text generation mode of an LLM
CL(LLM)	Classification mode of an LLM. A classifier is used for final classification.
CL-FT(LLM)	Classification mode of a fine-tuned LLM

TABLE II: Variables and nomenclature used in the paper

in the dataset. We adopt this relevance score from our previous work [39] because it shows better performance in a similar task. We apply the Bayesian confirmation theory mentioned in [40] to calculate the relevance score. According to the Bayes theorem [41], the posterior probability combines our initial belief, called the prior probability, with the likelihood of observing the evidence if the event is true. In our case, we want to measure the likelihood of classifying a ticket to a class given a particular token or word. Here, a token or word is the observed evidence. The posterior probability can be calculated as follows:

$$P(C_{i,a}|T_j) = \frac{P(C_{i,a}) \times P(T_j|C_{i,a})}{P(T_j)} \tag{1}$$

Here, $C_{i,a}$ is the i^{th} class for an attribute (a) of IMT, and T_j is the j^{th} token in a sample data. For the definitions of all variables and nomenclature used in this paper, refer to Table II. Using the Bayes theorem, we can calculate the posterior and prior probabilities. Now, we can use the Bayesian confirmation theory [40] to either confirm or disconfirm whether a token is relevant for classifying in a particular class or not. The confirmation theory is given below:

$$\begin{cases} T_j \textit{ confirms } C_{i,a}, & \textit{iff } P(C_{i,a}|T_j) > P(C_{i,a}) \\ T_j \textit{ is irrelevant to } C_{i,a}, & \textit{iff } P(C_{i,a}|T_j) = P(C_{i,a}) \\ T_j \textit{ disconfirms } C_{i,a}, & \textit{iff } P(C_{i,a}|T_j) < P(C_{i,a}) \end{cases}$$

See Table II for the definition of the variables. According to the confirmation theory, if the difference between posterior and prior is positive, then the token confirms the classification to a specific class. If the difference is 0, then the token is irrelevant for classification. Finally, if the difference is negative, then the token is not relevant for classification in that particular class. Since we have four attributes of IMT and each attribute has multiple classes, we calculate both posterior and prior probability for each token given an attribute of IMT and a specific class of that attribute. We use the following formula to combine all these probability scores and get a single relevance score for each token in the input.

$$R_{j,a} = \frac{1}{L} \times \sum_{i=0}^{L-1} |P(C_{i,a}|T_j) - P(C_{i,a})| \tag{2}$$

Here, $R_{j,a}$ is the relevance score for j^{th} token for attribute a, and L is the total number of classes for that attribute. Using the above formula, we get a single value for each token in the input. For example, consider the token "GSM" while classifying the network impact attribute of an IMT. There are three possible classes for this attribute as shown in Table I): *outage*, *degraded*, and *threatened*. The prior probabilities $P(C_{outage}) = 0.4$, $P(C_{degraded}) = 0.35$, and $P(C_{threatened}) = 0.25$ are estimated from the training set, and the likelihood $P(T_{GSM}|C_i)$ are also computed from the same training data. Using equation 1, the posterior probabilities given the token "GSM" are calculated as $P(C_{outage}|T_{GSM}) = 0.7$, $P(C_{degraded}|T_{GSM}) = 0.2$, and $P(C_{threatened}|T_{GSM}) = 0.1$. The relevance score is then obtained using equation 2 as:

$$\begin{aligned} R_{GSM,network} = \frac{1}{3}\big(&|0.7 - 0.4| + |0.2 - 0.35| \\ &+ |0.1 - 0.25|\big) = 0.1667 \end{aligned} \tag{3}$$

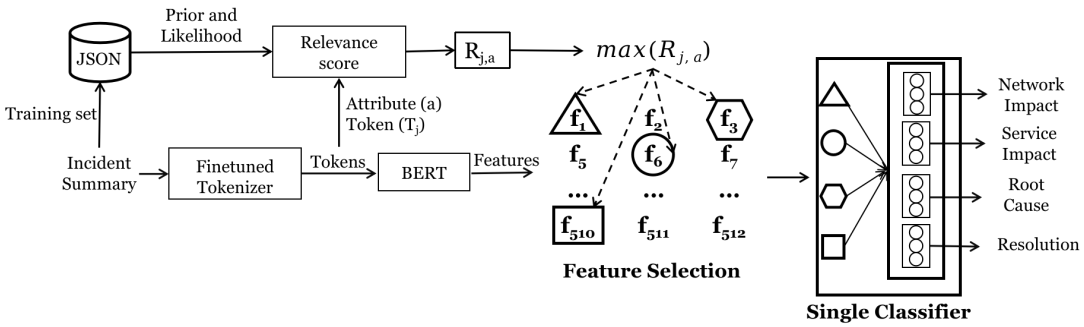
After the calculation of the relevance score, the token with the highest relevance score for a given attribute of IMT is selected to classify the ticket based on that selected token's feature vector. In our encoder-only approach, the top N tokens are selected based on their relevance scores, whereas in our decoder-only approach, the top K retrieved IMT tickets are ranked by the highest relevance score of their most relevant token.

IV. Methodology

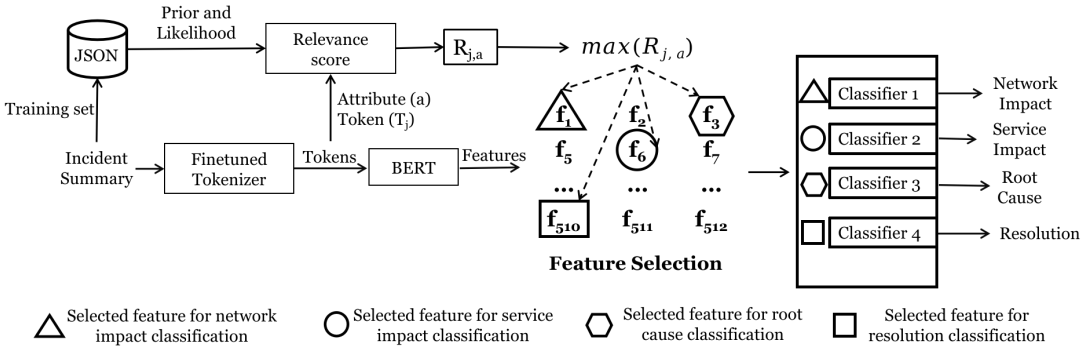
In this section, we discuss our proposed approaches in detail. First, we present our approach for multi-label classification using an encoder-only architecture. We then describe the probabilistic token-prioritization strategy employed in both encoder-only and decoder-only architectures. The decoder-only approach is discussed later in this section. We divide each of our approaches into multiple sub-sections for better understanding.

A. Encoder-only architecture for IMT classification

In the first stage, we use BERT [40], an encoder-only model architecture that uses a stack of encoder layers from the well-known transformer [42] architecture for IMT classification. BERT has twelve layers of encoder transformers stacked on top of one another. The main purpose of using BERT is to extract features from input text. The extracted features are



(a) BERT with single classifier (SC-BERT)



(b) BERT with multiple classifiers (MC-BERT)

Fig. 1: Probability-based encoder-only approach

used in the downstream task which is multi-label classification. For multi-label classification, we can take two approaches: a separate classifier for each of the labels, and one classifier that outputs multiple labels at once. The BERT model with single classifier and multiple classifiers are denoted as SC-BERT and MC-BERT, respectively in the rest of the paper. The components of SC-BERT and MC-BERT are shown in Figure 1. The main difference between these two approaches is in the classification stage. The other components such as tokenization, fine-tuning, and feature selection are the same in both approaches. In MC-BERT, each attribute of the IMT is classified using a separate classifier or linear layer, whereas SC-BERT uses a single linear layer whose specific neurons are trained for a specific attribute of IMT. For example, the first three neurons in SC-BERT predict the network impact, the next two neurons predict the service impact, and so on. The components of SC-BERT and MC-BERT are discussed in the next few sections.

To adapt to the domain-specific words in the dataset, we follow two strategies in our proposed approach. First, we fine-tune the tokenizer on our dataset. Second, we fine-tune the pre-trained transformer layers of BERT on our dataset.

Finetune the tokenizer: The tokenization process is a simple way of converting textual input to a numeric representation. In this process, we maintain a vocabulary list. After splitting the sentence into words, we represent each word with the index position of that word in the vocabulary list. The BERT model uses a specific tokenization called "WordPiece" tokenization, which breaks down words into sub-words or characters if it is not present in the vocabulary list. For example, if the word "cutover" is not present in the vocabulary list, but the separate words "cut" and "over" are present, the BERT tokenizer will split the word ("cutover") into these two separate words. Then the indices of these two words are used in the numeric representations.

Since a lot of domain-specific words are present in our dataset, we fine-tune the BERT tokenizer on our dataset to accommodate these domain-specific words in the vocabulary list. During the fine-tuning process, the BERT tokenizer looks through the training dataset, and based on the frequency of a word in the dataset, words are added to the new vocabulary list. In total, the BERT tokenizer has 30,522 words in its vocabulary list. After this fine-tuning process, we create a new vocabulary list based on our dataset.

Finetune the transformer layers: The common approach of text classification is to load a pre-trained model like BERT, available online, and then fine-tune the layers on a specific dataset. This approach works well when the dataset is similar to the pre-trained dataset of BERT. The specific dataset used for pre-training BERT is known as the "BookCorpus" and "English Wikipedia" [40]. Both of these datasets contain regular English sentences and words. In our case, we have domain-specific words such as IPRAN and QAM. These words are rare in regular English sentences. Therefore, we fine-tune the transformer layers on our dataset at first. This fine-tuning is done on our dataset in the same way the pre-trained BERT layers are trained. The BERT model uses a technique called "Masked Language Modeling" (MLM) to train its layers in an unsupervised manner [40]. In MLM, the task of the model is to predict a randomly masked word from the dataset. We

do not use actual class labels for this training. To evaluate the performance of the BERT model in this fine-tuning, we use an evaluation metric called "perplexity score" [43]. It measures how well a language model predicts the next word in a sequence of words. A lower perplexity score indicates that the language model is better at predicting the next word in a given context. It means that the model has learned the patterns and structures of the training data more effectively. The formula for calculating the perplexity score in the context of language models is as follows:

$$Perplexity = 2^H$$

Here, H represents the average cross-entropy loss. The cross-entropy loss measures how well the model predicts the next word compared to the ground truth. The cross-entropy loss is calculated for each word in the test dataset, and then the average is taken. This average is represented by H in the above formula. For this fine-tuning, we train for 200 epochs on our dataset with a batch size of 64 and learning rate 5×10^{-5}. We use the Adam [44] optimizer for the fine-tuning. The fine-tuned BERT model is denoted as FT-BERT in the result section of this paper.

Feature selection: Figure 1 shows that we obtain a set of feature vectors (f1, f2, f3, ..., f512) after the feature extractor component in the classification pipeline. For each token in the input text, we obtain a vector of size 768. The input length in our experiment is 512, which means the input can have at most 512 words in it. If the input has less than 512 words, then it is padded with a special token called "pad" by the BERT tokenizer. On the other hand, if the input has more than 512 words, the rest of the words are truncated. Now, for each of the 512 words, the features extractor gives us a vector of size 768. We need to decide which vector should be taken as input to the classifier.

We propose to select feature vectors based on a probability score. The probability score gives us the relevance of a token to classify the text in a specific class. Since we are dealing with multi-label classification, separate tokens are selected for each attribute of IMT. In Figure 1, the triangle, circle, hexagon, and rectangle shapes denote these separate selected tokens for each attribute of IMT. For SC-BERT the selected feature vectors go through the single classifier one after another, whereas MC-BERT takes each selected token to its corresponding classifier. The classifier outputs one class per attribute of IMT. We use weighted cross-entropy loss during training time to address the class-imbalance present in our dataset.

Relevance score: We calculate the prior probability and likelihood before starting the training for each attribute and save them in JSON files. During the training, we use this information to calculate the posterior probability for each class and then calculate the final relevance score. Afterwards, based on the relevance score, we select the feature vectors of the top N tokens from the input for each attribute. When we select multiple tokens for one attribute classification, we concatenate the selected feature vectors and change the number of neurons of the classifier accordingly. This proposed approach that combines relevance scores and BERT model is denoted as RS-BERT in the rest of this paper.

B. Decoder-only architecture for IMT classification

Conventional strategies for handling class imbalance, such as over-sampling, under-sampling, or weighted loss functions, operate primarily during model training to compensate for minority classes. In contrast, our approach addresses imbalance implicitly within the RAG pipeline. Specifically, during similarity matching and re-ranking, the probabilistic relevance score assigns higher importance to retrieved samples and tokens that are semantically aligned with minority classes. This mechanism enriches the contextual input provided to the decoder model, ensuring that underrepresented patterns are more prominently reflected in the generated responses. While the original training data remains imbalanced, the retrieved context is biased toward semantically similar instances rather than raw class frequency. This induces an adaptive reweighting effect where minority-class samples, when relevant, appear with higher probability in the conditioning context. Consequently, the decoder receives a more balanced signal at inference time without explicit resampling or loss reweighting, mitigating the well-known trade-off between minority-class recall and majority-class accuracy. Our encoder-based model (RS-FT-BERT) with relevance score outperforms the weighted loss-based approach (FT-BERT) by 10.86%, while our best decoder-based model (CL-FT-Phi-3.5) with relevance score powered RAG achieves a 26.64% improvement over FT-BERT. The proposed approach improves recall on rare classes without distorting the data distribution or degrading accuracy on frequent cases.

We explore the use of open-source decoder-only LLMs for multi-label classification of IMTs from wireless networks. Specifically, we utilize three LLMs in our experiments: Phi-3.5 from Microsoft [45], MPT-7B from MosaicML [46], and Falcon-7B from the Technology Innovation Institute [47]. These LLMs are selected due to their ability to perform inference on a single NVIDIA RTX A5000 GPU, which provides 24 GB of CUDA memory and is paired with 64 GB of system RAM in our infrastructure. Moreover, these models differ in their pre-training regimes, which contributes to their complementary capabilities. Phi-3.5 is instruction-tuned, making it particularly effective in following explicit instructions during inference, whereas MPT-7B and Falcon-7B are pre-trained in an auto-regressive manner for next-token prediction.

We explore two task formulations using the decoder-only models: (1) Classification, where the transformer layers of the LLM are used as a feature extractor, and four separate dense layers are trained on top to predict each IMT attribute as a class label; and (2) Text Generation, where the LLM is prompted to generate attribute values in an auto-regressive manner. The main advantage of using the generation mode lies in the open-ended output capability of LLMs. The classification mode relies on a fixed set of predefined categories, which makes it difficult to incorporate new classes without retraining the classifiers. Instead, in generation mode, new classes can

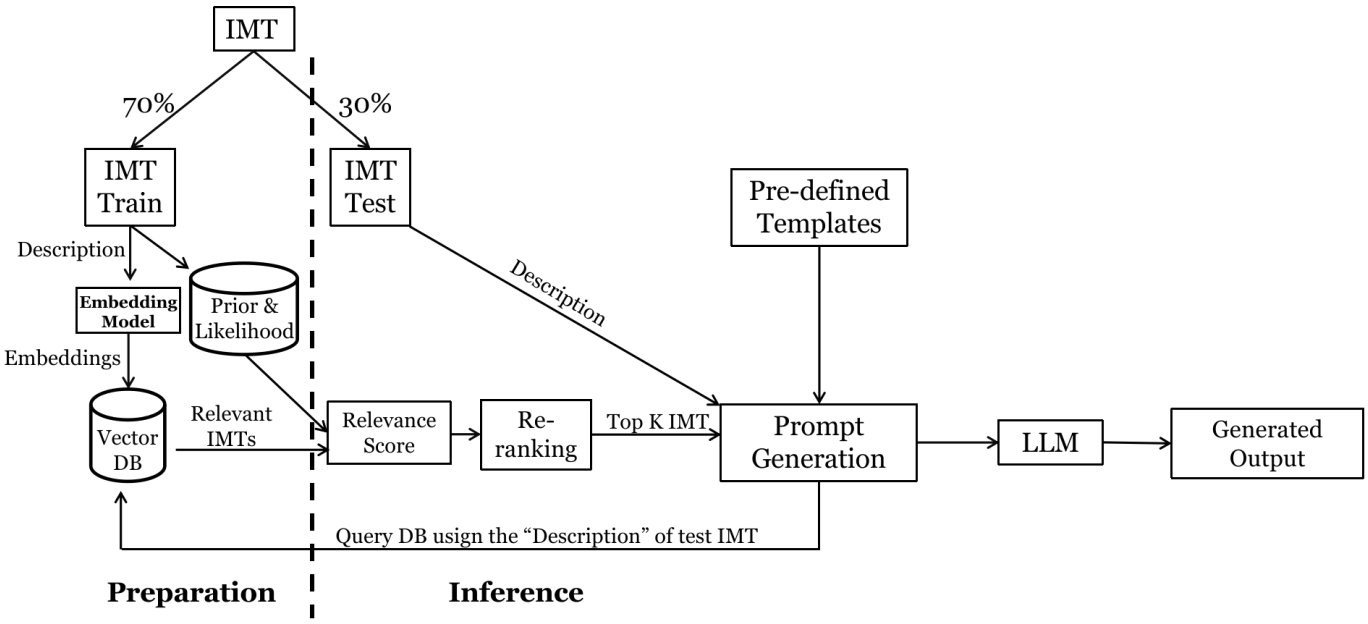


Fig. 2: Overview of the RAG pipeline that uses probabilistic relevance score and re-ranking

Prompt
You are a helpful assistant who classifies telecom incident tickets. Below are some similar historical tickets from our knowledge base that may help you: {retrieved_examples} Now, classify this new incident: {summary} We have four attributes to determine: 1) Network Impact (valid values: {network_impact_values}) 2) Service Impact (valid values: {service_impact_values}) 3) Resolution (valid values: {resolution_values}) 4) Root Cause (valid values: {root_cause_values}) Respond in valid JSON with exactly these keys: "network_impact", "service_impact", "resolution", "root_cause"

TABLE III: Pre-defined prompt for the RAG pipeline

be introduced simply by updating the prompt and adding corresponding examples to the vector database. This flexibility allows the model to generate outputs for the newly added classes without requiring any retraining. In the generation mode, the model outputs predictions for all four attributes in a JSON format. We perform post-processing on the generated output to extract the predicted class values for each IMT attribute.

To evaluate the performance of these LLMs in the text generation setup, we explore a diverse set of prompting techniques [48]: zero-shot prompting, few-shot prompting, Chain-of-Thought (CoT), and RAG. Prompts are generated using OpenAI's prompt generation tools and adapted to the specific context of our classification problem. The best performing prompt for the RAG approach is shown in Table III. In Table III, everything inside the curly brackets is variable. These variables are replaced with appropriate values before going as input to the LLM models.

RAG combines the benefits of retrieval-based and generation-based paradigms. Our proposed RAG pipeline, illustrated in Figure 2, integrates a re-ranking strategy that leverages the probabilistic relevance score discussed in Section IV-A. For each target IMT attribute, we compute a relevance score for the retrieved tickets based on their semantic alignment with class-specific tokens. These scores are used to re-rank the retrieved tickets, enhancing the quality of contextual input to the LLM. The RAG pipeline consists of two phases: a preparation phase and an inference phase. During the preparation phase, we split the IMT dataset into training (70%) and testing (30%) subsets. For each ticket in the training set, we generate embeddings from the "multi-qa-mpnet-base-dot-v1" model from Hugging Face [49] and store them in a vector database. These embeddings are 768-dimensional and enable semantic similarity search.

During inference, for each test ticket summary, we retrieve 20 similar IMTs from the vector database using cosine similarity. The retrieved tickets are ranked using our probabilistic relevance score, and the top K are injected into a prompt alongside the new ticket summary. The final prompt, constrained by the input token limit of the LLM (e.g., 128K tokens for Phi-3.5), is then passed to the decoder-only model to generate predictions. Our approach allows flexibility in handling new IMT attribute classes without retraining the model, highlighting the advantage of generative LLMs over conventional classifiers. Moreover, by re-ranking the retrieved documents based on contextual relevance, we improve the alignment of the input prompt with the classification objective, leading to better performance.

V. Evaluation

We discuss the experiments and results of our proposed approaches in this section. First, we introduce our compared

approaches in both the encoder-only architecture and the decoder-only architecture. Next, we discuss the results of our proposed approaches in comparison with the compared approaches. Finally, we show the explainability of the best-performing model using the LIME explainer [12].

A. Evaluation Setting

We evaluate several traditional machine learning (ML) models on the dataset to establish baseline performance. We then compare these baselines with the proposed encoder-only and decoder-only architectures under multiple architectural variations. In some variants, we apply the proposed feature selection or re-ranking strategy, while in others we rely on conventional approaches for selecting input features or contextual information for the models. Table IV shows variations of the encoder-only architecture. We experiment with two encoder-only architectures, BERT and RoBERTa [50]. Both models share the same underlying architecture. However, RoBERTa incorporates several training-time optimizations, such as dynamic masking instead of static masking and the removal of the next-sentence prediction task, which yield only marginal performance gains. We compare BERT and RoBERTa models that use a probabilistic relevance score for feature selection against three variations of the same model to demonstrate the effectiveness of relevance-score–based feature selection. Finally, we compare the performance of encoder-only architectures with that of multiple decoder-only LLMs using different prompting methods.

The first compared approach in Table IV, SC-BERT is the pre-trained BERT model proposed in [40] with a single classifier to classify all four attributes of an IMT. As recommended in [40], we always select the feature of the first token for this approach. The second approach in Table IV replaces the single classifier with multiple classifiers. For each attribute of IMT, a separate linear layer is used as classifier in this approach. Comparison between this single classifier and multiple classifiers is necessary to understand whether the prediction of one attribute from the model influences the other. In the single classifier approach, all attributes are classified at once by the same linear layer, whereas in multi-classifier approach a dedicated classifier is applied for each attribute classification. Since the decision is taken by a separate classifier in MC-BERT, one prediction does not influence the other. The third compared approach, FT-BERT uses multiple classifiers too, but the tokenizer and transformers layers of the BERT model are fine-tuned as described in Section IV-A. Finally, the last approach mentioned in Table IV is our proposed encoder-only approach, denoted as RS-FT-BERT. The main difference of RS-FT-BERT from other compared approaches is the feature selection strategy. In RS-FT-BERT, the feature is selected according to the probabilistic relevance score. Similar to the variations of BERT, we have the three variations of the RoBERTa model shown in Table IV, SC-RoBERTa, MC-RoBERTa, and RS-FT-RoBERTa.

Table V shows variations of the decoder-only architectures. As mentioned in Section IV, we evaluate the LLMs in two tasks: Text Generation (TG) and Classification (CL). The first three rows of Table V show LLMs for text generation task, and the last four rows show the classification task. For the text generation task, we use four types of prompting methods with all three LLMs in Table V. Based on the results of the text generation task, we adopt RAG as the prompting strategy and employ the Phi-3.5 and Falcon-7B models for the classification task. We also experimented with and without fine-tuning the LLMs on our dataset. The last four rows of Table V show with and without fine-tuning LLM models.

Since there is class imbalance in our dataset, the F1 score is the most important evaluation metric in our experiments. When dealing with imbalanced datasets, accuracy alone can be misleading and may not provide a comprehensive understanding of the model's performance. The F1 score is the harmonic mean of precision and recall. Hence, it combines precision and recall into a single metric, providing a more reliable measure of a model's ability to classify samples from both the majority and minority classes correctly. To get an overall idea about the performance of the model on each class, we calculate the weighted average of the precision, recall, and F1 score. In weighted average, we calculate class-wise evaluation metric at first and then we take the average over all the classes and multiply it with the percentage of samples from each class. The average score for each IMT attribute is reported in the following subsections. The experiments are conducted on a machine equipped with one NVIDIA RTX A5000 GPU, having 24GB of memory, and a main memory of 64GB.

TABLE IV: Variations of the encoder-only language model

Approaches	**Classifier**	**Feature Selection**	**Domain adapt.**	**Loss**
SC-BERT	Single	First token	✗	CE
MC-BERT	Multiple	First token	✗	CE
FT-BERT	Multiple	First token	✓	Weighted CE
RS-FT-BERT	Multiple	Relv. score	✓	Weighted CE
SC-RoBERTa	Single	First token	✗	CE
MC-RoBERTa	Multiple	First token	✗	CE
RS-FT-RoBERTa	Multiple	Relv. score	✓	Weighted CE

Relv. = Relevance, CE = Cross Entropy,
Adapt. = Adaptation

TABLE V: Variations of the decoder-only language model

Approaches	**Task**	**Prompt**	**Domain adapt.**
TG-Phi-3.5	Generation	*	✗
TG-Falcon	Generation	*	✗
TG-MPT	Generation	*	✗
CL-Phi-3.5	Classification	RAG	✗
CL-Falcon	Classification	RAG	✗
CL-FT-Phi-3.5	Classification	RAG-RS	✓
CL-FT-Falcon	Classification	RAG-RS	✓

* All four prompting methods discussed in Section IV-B
RAG-RS: RAG pipeline with re-ranking using probabilistic relevance score

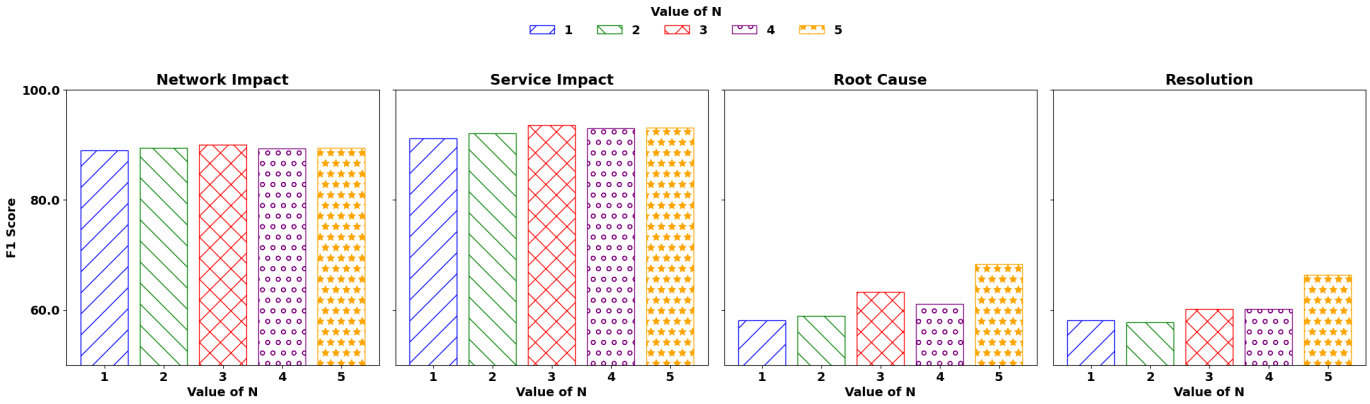


Fig. 3: F1 score of RS-FT-BERT for different attributes classification by varying N

TABLE VI: Evaluation metrics of traditional machine learning models

Approach	Precision (%)	Recall (%)	F1
Xgboost	77.00	70.76	72.53
Random forest	79.95	78.60	78.16
Naive bayes	73.37	54.65	56.81
SVM	84.11	78.18	**80.32**
Logistic regression	80.38	66.03	69.15

B. Evaluation Results

1) Results of the traditional machine learning models: At first, we establish baseline performance on our dataset by training a set of traditional ML models. The textual descriptions of incidents contained in the tickets are converted into numerical representations using the Term Frequency–Inverse Document Frequency (TF-IDF) vectorizer implemented in the scikit-learn library. All ML models presented in Table VI are trained using the default parameter configurations provided by scikit-learn. Similarly, the XGBoost model is trained using the default settings from the XGBoost library. Among the traditional ML models, the Support Vector Machine (SVM) demonstrates the best F1 score. Then, we compare the performance of our proposed methods against these traditional ML baselines, as summarized in Table VI.

2) Results of the encoder-only architecture: The average score of all the evaluation metrics mentioned in the previous section is shown in Table VII. Table VII shows that the RS-FT-BERT outperforms all the compared approaches in terms of all the evaluation metrics. Although RS-FT-RoBERTa outperforms the other two RoBERTa variants in Table VII, demonstrating the effectiveness of our probability-based feature selection strategy, RS-FT-BERT achieves the best overall performance. While RoBERTa is generally stronger on large-scale benchmarks [50], it tends to overfit on smaller datasets such as ours, which may explain its comparatively weaker performance in this setting. RS-FT-BERT outperforms all traditional machine learning approaches mentioned in Table VI except SVM. The performance of SVM is slightly better (1.12%) than RS-FT-BERT. This slight advantage of SVM can be attributed to its ability to generalize well on limited and high-dimensional textual data, where deep learning models like RS-FT-BERT may require larger training samples to fully leverage their representation learning capabilities. This result suggests that the probabilistic feature selection approach improves the performance of the BERT model. There is a hyperparameter in RS-FT-BERT approach that decides how many tokens need to be selected per attribute based on the relevance score. We denote this hyperparameter as N. The best value of N depends on the dataset. In our experiments, we vary the value of N from 1 to 5. We found the best value to be 5 in this case. The result mentioned in Table VII for RS-BERT is obtained when $N = 5$. It is crucial to understand the effect of N on the performance of the model. Figure 3 shows the weighted F1 score for each attribute when we vary the value of N for the best-performing approach, RS-FT-BERT. We see a little improvement when the value of N increases. However, the complexity of the model also increases as we consider more tokens.

TABLE VII: Evaluation metrics of encoder-only language models

Approach	Precision (%)	Recall (%)	F1
SC-BERT	70.08	64.15	64.54
MC-BERT	69.68	69.42	65.06
FT-BERT	74.96	71.68	68.34
RS-FT-BERT	**80.33**	**78.83**	**79.20**
SC-RoBERTa	72.94	68.35	70.57
MC-RoBERTa	73.08	70.19	71.61
RS-FT-RoBERTa	**75.46**	**73.29**	**74.36**

3) Results of the decoder-only architecture: In this subsection, we discuss the results of decoder-only LLMs mentioned in Table V. Since the output of the LLM is open-ended for the text generation task, evaluation is difficult using metrics such as F1 score. A slight change in the generated output by the LLM is considered as a complete separate class when evaluating with the F1 score. Therefore, we need to make sure that the LLM outputs in the same manner as the true labels in our dataset. We write a Python script to post-process the LLM outputs and extract the expected output. By observing the LLM output we use regular expressions to extract the output in a specific format. In our experiment, we found that the effect of the prompt is significant on the LLM output. As discussed in Section IV-B, we utilize four prompting methods

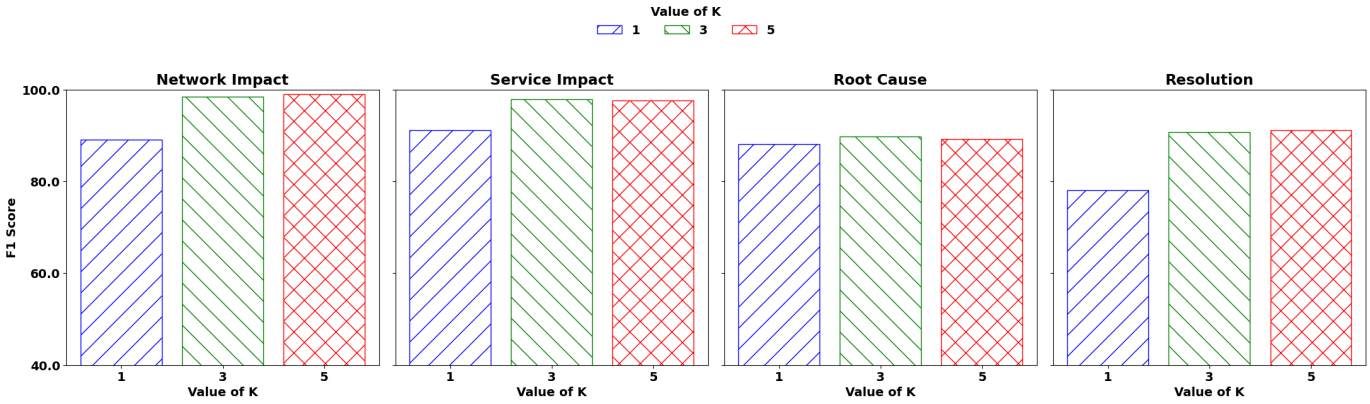


Fig. 4: F1-score of CL-FT-Phi-3.5 for different attributes classification by varying K

in our experiments. Table VIII shows the performance of each prompt for the three selected LLMs.

TABLE VIII: Evaluation metrics of the decoder-only language models for the text generation task

Prompt	Approach	Pr (%)	Rec (%)	F1
Zero-shot	TG-Phi-3.5	6.40	14.50	6.60
	TG-MPT	20.50	21.00	15.10
	TG-Falcon	13.60	13.60	11.00
Few-shot	TG-Phi-3.5	13.20	14.50	7.50
	TG-MPT	20.00	22.90	18.80
	TG-Falcon	16.80	18.40	14.90
CoT	TG-Phi-3.5	17.70	33.00	19.30
	TG-MPT	27.50	32.80	24.30
	TG-Falcon	42.80	38.50	30.40
RAG(k=3)	TG-Phi-3.5	**68.20**	**59.10**	**59.59**
	TG-MPT	**78.28**	**75.68**	**75.62**
	TG-Falcon	**79.44**	**77.20**	**77.57**

Table VIII shows the performance of the four selected prompting methods with pre-trained LLMs. The zero-shot and few-shot prompting method achieves a low F1 score in all three LLMs, indicating that pre-trained LLMs struggle to understand telecom domain language when there is limited context in the prompt. The CoT prompt achieves a moderate performance compared to zero-shot and few-shot, which shows the benefit of encouraging step-wise reasoning in the prompt. However, the RAG pipeline outperforms all three prompt engineering methods by a large margin in terms of F1 score. This result shows the benefit of providing appropriate context in the prompt. We use $k = 3$ for the RAG pipeline, meaning that the top three similar tickets from the IMT train dataset are given as context in the prompt. We can conclude from Table VIII that the RAG pipeline performs best in our dataset compared to the other three prompting methods. Therefore, for the classification task, we use variations of the RAG pipeline. We select Phi-3.5 and Falcon-7B in the classification task since the MPT performs similarly to Falcon in Table VIII.

When we use RAG and LLM as classifiers, we first augment the IMT tickets using the RAG pipeline and then classify the augmented tickets using the LLMs. Table IX shows that the RAG pipeline with re-ranking and fine-tuned LLM outperforms the baselines in Table VI, encoder-only variants, and all the other decoder-only approaches in the classification task. The significant improvement is due to the fine-tuning of LLM, as we can see in the second row of Table IX. Among the two tasks for LLMs, classification outperforms text generation models. As discussed earlier, text generation has open-ended output, and as a result output of LLMs do not always follow the expected format. In addition, the slight improvement when the re-ranking strategy is used with the RAG pipeline shows that the order of the relevant IMTs given as context also impacts the LLM output.

TABLE IX: Evaluation metrics of the decoder-only language models for the classification task

Prompt	Approach	Pr (%)	Rec (%)	F1
RAG(k=3)	CL-Phi-3.5	69.43	56.26	58.01
	CL-Falcon	81.91	73.34	76.71
RAG(k=3)	CL-FT-Phi-3.5	90.13	98.47	94.12
	CL-FT-Falcon	91.82	88.65	90.21
RAG-RS(k=3)	CL-FT-Phi-3.5	**93.09**	**98.37**	**94.98**
	CL-FT-Falcon	**92.21**	**89.19**	**91.10**

Another crucial parameter in the RAG approach is "K" that determines how many similar tickets should be fetched from vector database. We vary this number to see the effect on the result. We found that increasing the value of K increases the performance. Figure 4 shows the performance of RAG with re-ranking and fine-tuned Phi-3.5. For each attribute classification, we see a slight improvement in the F1 score with larger values of K. As mentioned earlier, a Python script is written that uses regular expression to extract final output from the RAG pipeline. We notice that for some inputs the LLM output is blank, and this number varies based on the prompt and LLM model. There is no blank output for fine-tuned Phi-3.5. TG-Falcon and TG-MPT generate eighteen and three blank outputs, respectively. We replace the blank output with the "Unknown" keyword during the evaluation.

Table IX shows that RAG with Re-ranking (RAG-RS in Table V) using instruction-tuned decoder-only models in classification mode outperforms all encoder and decoder-based approaches. Specifically, CL-FT-Phi-3.5 with re-ranking achieves an F1 score of 94.98, while CL-FT-Falcon follows closely with 91.10. Compared to the best encoder-only model (RS-FT-BERT, 79.20 F1), RAG-RS approach improves the F1 score by 19.88% with Phi-3.5 and 15.00% with Falcon. This demonstrates that combining instruction-tuned decoder-only models

with classification heads and retrieval-based context selection, enhanced through a probabilistic re-ranking strategy, provides a significant boost in performance over both traditional fine-tuned encoder models and naive generative LLM prompting approaches.

C. Model explainability

We leverage a well-known LIME explainer [12] to explain the output of the used models. LIME helps answer to this question: which parts of the input text are most responsible for the classification decision? LIME is designed to explain the predictions of any black-box model, including complex pipelines like RAG used for text classification. In the context of RAG, where a model first retrieves relevant documents and then performs classification based on both the input and retrieved knowledge, understanding why a particular label was predicted can be difficult. LIME addresses this by approximating the model's behavior locally around a specific input, helping us interpret which parts of the input text were most responsible for the classification decision.

The process begins by taking the original input text (IMT summary in our case) and creating many perturbed versions of the input. These perturbations are generated by systematically removing or masking certain words or phrases in the input. Each perturbed input is then passed through the full RAG pipeline, which performs retrieval and classification, producing output probabilities for different labels. This allows LIME to observe how slight changes in the input affect the final predictions.

Next, LIME constructs a simplified, interpretable surrogate model (such as a linear regression) using the perturbed inputs and the corresponding model outputs. This surrogate model is trained to mimic the behavior of the RAG model but only in the neighborhood of the original input. Each word in the input becomes a feature, and the surrogate model learns how the presence or absence of these features influences the prediction.

From this surrogate model, LIME extracts importance scores, which reflect how much each word contributed to the final classification. The strength of LIME in an RAG pipeline lies in its model-agnostic nature. It does not require access to the internals of the retrieval or classification components. Instead, it treats the entire RAG system as a black box and infers interpretability solely from input-output behavior.

Figure 5 shows the visualization created by the LIME explainer for an IMT ticket. The alarm and device IDs are intentionally changed to protect data privacy in Figure 5 (shown in red color). Positive values in Figure 5 indicate that the presence of a word increases the model's confidence in the predicted class. Similarly, negative values indicate that the presence of a word reduces the model's confidence in the predicted class. From Figure 5, we conclude that device IDs and alarm IDs are important information for these attributes classification. In the given example, the RAG pipeline correctly classifies network and service impact and misclassified the root cause and resolution. Therefore, Figure 5 shows that the model focuses on uninformative words such as "by", "is" "a" and so on for these two classifications.

VI. Discussion and Future Works

The experimental results demonstrate that decoder-only language models, when combined with RAG and a probabilistic re-ranking strategy, can significantly outperform traditional encoder-only models in multi-label classification of IMTs. While encoder-only models such as RS-FT-BERT achieved good performance due to task-specific fine-tuning and feature selection, they lack flexibility in adapting to evolving classification schemas or newly introduced IMT attributes without retraining. Decoder-only models, in contrast, offer greater flexibility by leveraging open-ended generation capabilities and contextual reasoning through prompts.

Our investigation reveals that naive prompting strategies (e.g., zero-shot and few-shot) result in poor F1 scores due to the complex and domain-specific nature of IMT classification. Chain-of-Thought prompting showed moderate improvements by encouraging step-wise reasoning, but still fell short of encoder baselines. However, when these models are equipped with relevant contextual information via our RAG framework, performance improves significantly.

This improvement is largely attributed to three design choices: (1) the use of instruction-tuned models like Phi-3.5 that better follow instructions given in the prompt; (2) the injection of relevant context using RAG; and (3) the re-ranking of retrieved tickets based on semantic alignment with class-specific tokens. These elements allowed the decoder-only models to generalize better without additional fine-tuning.

Despite these promising results, several limitations remain. First, inference time for decoder-only models, especially in RAG-based pipelines, is significantly higher (11.34s per ticket) than encoder-based classifiers (0.87s per ticket). Although incident ticket analysis is an offline task and processing time on the order of a few seconds does not impact operational efficiency, we investigate the cause of the high inference time observed in the RAG-based approach. We identify the sequential "re-ranking" of the retrieved tickets for multiple attributes of IMT as the primary bottleneck. This is due to the fact that re-ranking operation involves input/output (I/O) processes, as the prior and likelihood probabilities are stored in a JSON file (Fig. 1). To validate our finding, the retrieved tickets are re-ranked separately for each of the four attributes before inference is performed using the LLM. Table X reports the inference times when retrieved tickets are re-ranked separately for each IMT attribute using our best-performing RAG pipeline and LLM (CL-FT-Phi-3.5). The results clearly indicate that the inference time decreases significantly when the prediction is performed for a single IMT attribute at a time. While our infrastructure (NVIDIA RTX A5000 with 24GB VRAM) supports these experiments, scaling to production environments with strict latency constraints may require model optimization.

As future work, we plan to explore (i) integrating lightweight and quantized LLMs to reduce inference overhead, (ii) expanding the classification to include additional ticket metadata, and (iii) investigating continual learning methods for adapting to evolving ticket taxonomies without retraining from scratch. In our study, we explored two prediction modes:

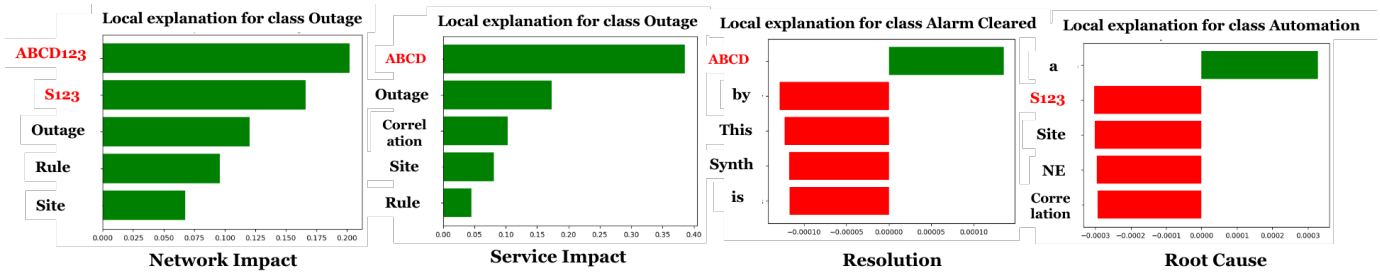


Fig. 5: LIME visualization for an example IMT ticket

IMT attribute	Inference time (s)
Network Impact	2.54
Service Impact	2.13
Root cause	3.91
Resolution	3.89

TABLE X: Inference time for different IMT attributes by CL-FT-Phi-3.5

classification and text generation. While the classification mode relies on attribute-specific classifiers, the text generation mode allows the model to predict multiple attributes simultaneously in an open-ended manner. With appropriately designed prompts and context, this approach can potentially generalize to different taxonomies beyond the current telecom domain. However, as our experiments were constrained by the available dataset, we were unable to evaluate performance across different languages. Further experiments can be done in the future to investigate the generalization of the proposed method on different languages. For generalized applications we speculate that a multi-lingual pre-trained LLM could provide gain in terms of cross-lingual understanding, improved retrieval accuracy for non-English tickets, and reduced need for language-specific fine-tuning. This could enable the system to handle incident tickets from diverse regions more effectively while maintaining consistent performance across languages.

VII. Conclusion

Incident ticket classification plays a vital role in the automation of incident management systems in large-scale telecommunication networks. Accurate classification enables efficient prioritization, faster incident resolution, and more informed decision-making by relevant teams. Classifying attributes such as Network Impact, Service Impact, Root Cause, and Resolution early in the incident lifecycle significantly reduces response time and operational overhead.

In this paper, we addressed the multi-label classification challenge for IMT tickets using a dataset collected from a major Canadian telecom operator. We investigated two complementary approaches: (1) encoder-based language models that use a token prioritization strategy to enhance feature representation, and (2) a decoder-based language modeling pipeline that integrates RAG and a probabilistic re-ranking mechanism. While RS-FT-BERT outperformed other encoder-only baselines by incorporating relevance-guided fine-tuning, the decoder-based RAG-RS approach demonstrated even higher effectiveness. An existing explainability method is leveraged to show the model's explainability.

The text classification model, CL-FT-Phi-3.5 with RAG and re-ranking, achieved an F1 score of 94.98%, outperforming the traditional machine learning model (SVM, 80.32%) and encoder-only baseline (RS-FT-BERT, 79.20%) by 14.66% and 19.88%, respectively. This result highlights the potential of combining instruction-tuned decoder-only LLMs with semantically-aware document retrieval and relevance-guided prompt construction. Furthermore, our RAG pipeline enables scalable classification without retraining, making it adaptable to evolving taxonomies in real-world telecom environments.

Our findings suggest that LLM-based classification, when paired with intelligent retrieval and prompt engineering, provides a robust solution for dynamic and high-dimensional classification tasks. We believe this methodology can generalize to other domains where labeled data is sparse and class distributions are skewed.

Acknowledgement

This work was supported in part by Rogers Communications Canada Inc. and in part by a Mitacs Accelerate Grant and an NSERC Discovery Grant.

References

[1] M. W. Asres, M. A. Mengistu, P. Castrogiovanni, L. Bottaccioli, E. Macii, E. Patti, and A. Acquaviva, "Supporting telecommunication alarm management system with trouble ticket prediction," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1459–1469, 2021.

[2] M. E. Maron, "Automatic indexing: An experimental inquiry," *J. ACM*, vol. 8, no. 3, p. 404–417, Jul. 1961.

[3] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[4] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Proceedings of the 10th European Conference on Machine Learning*, ser. ECML'98. Berlin, Heidelberg: Springer-Verlag, 1998, p. 137–142.

[5] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu, and L. He, "A survey on text classification: From traditional to deep learning," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 2, Apr. 2022.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[7] J. David, J. Cui, and F. Rahimi, "Classification of imbalanced dataset using bert embeddings," *Dalhousie Univ., Halifax, Canada, Jan*, 2020.

[8] Y. Zhang, M. Wang, C. Ren, Q. Li, P. Tiwari, B. Wang, and J. Qin, "Pushing the limit of llm capacity for text classification," 2024. [Online]. Available: https://arxiv.org/abs/2402.07470

[9] W. X. Zhao *et al.*, "A survey of large language models," 2026. [Online]. Available: https://arxiv.org/abs/2303.18223

[10] W. Fan, Y. Ding, L. Ning, S. Wang, H. Li, D. Yin, T.-S. Chua, and Q. Li, "A survey on RAG Meeting LLMs: Towards retrieval-augmented large language models," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2024, p. 6491–6501.

[11] M. Hu, "Three-way bayesian confirmation in classifications," *Cognitive Computation*, vol. 14, pp. 2020 – 2039, 2021.

[12] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2016, p. 1135–1144.

[13] I. Shrivastava, "Handling class imbalance by introducing sample weighting in the loss function," December 2020, gumGum Tech Blog, Medium.

[14] M. Hu, "Three-way bayesian confirmation in classifications," *Cognitive Computation*, vol. 14, 09 2021.

[15] Y. Shehu and R. Harper, "Enhancements to language modeling techniques for adaptable log message classification," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4662–4675, 2022.

[16] Y. Shehu and R. Harper, "Improved fault localization using transfer learning and language modeling," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*. IEEE, 2020, pp. 1–6.

[17] J. Tong, Z. Wang, and X. Rui, "A multimodel-based deep learning framework for short text multiclass classification with the imbalanced and extremely small data set," *Computational Intelligence and Neuroscience*, vol. 2022, 2022.

[18] R. Wang, R. Ridley, X. Su, W. Qu, and X. Dai, "A novel reasoning mechanism for multi-label text classification," *Information Processing & Management*, vol. 58, no. 2, p. 102441, 2021.

[19] F. Herrera, F. Charte, A. J. Rivera, and M. J. del Jesus, *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Cham: Springer, 2016.

[20] J. Bogatinovski, L. Todorovski, S. Džeroski, and D. Kocev, "Comprehensive comparative study of multi-label classification methods," *Expert Systems with Applications*, vol. 203, p. 117215, 2022.

[21] J. Liu, W.-C. Chang, Y. Wu, and Y. Yang, "Deep learning for extreme multi-label text classification," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2017, p. 115–124.

[22] S. R. Anggraeni, N. A. Ranggianto, I. Ghozali, C. Fatichah, and D. Purwitasari, "Deep learning approaches for multi-label incidents classification from twitter textual information," *Journal of Information Systems Engineering and Business Intelligence*, vol. 8, no. 1, pp. 31–41, Apr 2022.

[23] W. Huang, E. Chen, Q. Liu, Y. Chen, Z. Huang, Y. Liu, Z. Zhao, D. Zhang, and S. Wang, "Hierarchical multi-label text classification: An attention-based recurrent network approach," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, New York, NY, USA, 2019, p. 1051–1060.

[24] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. S. Dhillon, "X-bert: extreme multi-label text classification using bidirectional encoder representations from transformers," *arXiv: Learning*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:208617329

[25] M. U. Hadi *et al.*, "A survey on large language models: Applications, challenges, limitations, and practical usage." [Online]. Available: https://api.semanticscholar.org/CorpusID:272333785

[26] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2022.

[27] H. Zou, Q. Zhao, Y. Tian, L. Bariah, F. Bader, T. Lestable, and M. Debbah, "Telecomgpt: A framework to build telecom-specific large language models," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 3, pp. 948–975, 2025.

[28] A. Dandoush, V. Kumarskandpriya, M. Uddin, and U. Khalil, "Large language models meet network slicing management and orchestration," 2024. [Online]. Available: https://arxiv.org/abs/2403.13721

[29] S. Ghosh, M. Shetty, C. Bansal, and S. Nath, "How to fight production incidents? an empirical study on a large-scale cloud service," in *Proceedings of the 13th ACM Symposium on Cloud Computing*, New York, NY, USA, 2022, p. 126–141.

[30] P. Hamadanian *et al.*, "A holistic view of ai-driven network incident management," in *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks*, ser. HotNets '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 180–188.

[31] Y. Chen *et al.*, "Automatic root cause analysis via large language models for cloud incidents," in *Proceedings of the Nineteenth European Conference on Computer Systems*, ser. EuroSys '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 674–688.

[32] B. Chen, Z. Zhang, N. Langrené, and S. Zhu, "Unleashing the potential of prompt engineering for large language models," *Patterns*, vol. 6, no. 6, p. 101260, 2025.

[33] H. Zhou *et al.*, "Large language model (LLM) for telecommunications: A comprehensive survey on principles, key techniques, and opportunities," *IEEE Communications Surveys & Tutorials*, vol. 27, no. 3, pp. 1955–2005, 2025.

[34] A.-L. Bornea, F. Ayed, A. Domenico, N. Piovesan, and A. Maatouk, "Telco-rag: Navigating the challenges of retrieval augmented language models for telecommunications," pp. 2359–2364, 12 2024.

[35] N. S. Khan, M. M. Hasan, M. S. Towhid, S. Basnet, and N. Shahriar, "Enhancing large language models for telecom networks using retrieval-augmented generation," in *2024 IEEE Globecom Workshops (GC Wkshps)*, 2024, pp. 1–7.

[36] A. Maatouk, N. Piovesan, F. Ayed, A. D. Domenico, and M. Debbah, "Large language models for telecom: Forthcoming impact on the industry," *IEEE Communications Magazine*, vol. 63, pp. 62–68, 2023.

[37] T. Zanouda, M. Masoudi, F. G. Gebre, and M. Dohler, "Telecom foundation models: Applications, challenges, and future trends," 2024. [Online]. Available: https://arxiv.org/abs/2408.03964

[38] C. Shorten, T. Khoshgoftaar, and B. Furht, "Text data augmentation for deep learning," *Journal of Big Data*, vol. 8, 07 2021.

[39] M. S. Towhid, N. S. Khan, N. Shahriar, M. Tornatore, R. Boutaba, and A. Saleh, "A token-prioritization strategy for handling data imbalance in network-change ticket classification," in *2023 19th International Conference on Network and Service Management (CNSM)*, 2023, pp. 1–7.

[40] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019.

[41] J. M. Joyce, "Bayes' theorem," 2021, first published 2003; archived Fall 2021 edition. [Online]. Available: https://plato.stanford.edu/archives/fall2021/entries/bayes-theorem/

[42] A. Vaswani *et al.*, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[43] L. Mao, "Entropy, perplexity and its applications," https://leimao.github.io/blog/Entropy-Perplexity/, 2019, updated: 2023. [Online]. Available: https://leimao.github.io/blog/Entropy-Perplexity/

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[45] M. Abdin *et al.*, "Phi-3 technical report: A highly capable language model locally on your phone," 2024. [Online]. Available: https://arxiv.org/abs/2404.14219

[46] Databricks, "Introducing mpt-7b: A new standard for open-source, commercially usable llms," May 2023, databricks Blog. [Online]. Available: https://www.databricks.com/blog/mpt-7b

[47] E. Almazrouei *et al.*, "The falcon series of open language models," 2023. [Online]. Available: https://arxiv.org/abs/2311.16867

[48] S. Schulhoff *et al.*, "The prompt report: A systematic survey of prompt engineering techniques," 2025. [Online]. Available: https://arxiv.org/abs/2406.06608

[49] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Q. Liu and D. Schlangen, Eds. Association for Computational Linguistics, Oct. 2020, pp. 38–45.

[50] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 07 2019.

Md. Shamim Towhid is a Ph.D. student at the Department of Computer Science, University of Regina, Canada. He received M.Sc. and B.Sc. degrees in Computer Science from the University of Regina and Ahsanullah University of Science and Technology in 2023 and 2017, respectively. He is a recipient of the Saskatchewan Innovation and Excellence Graduate Scholarship 2022, 2023, and 2024 and the PST 2025 Best Paper Award. His research interests include network fault management, 5G network security, and reliability.

Nasik Sami Khan is an AI Software Developer at BDM Healthcare Inc., located in Saskatchewan, Canada. He received his M.Sc. and B.Sc. degrees in Computer Science from the University of Regina and International Islamic University Malaysia in 2025 and 2021, respectively. He is a recipient of the Saskatchewan Innovation and Excellence Graduate Scholarship (2023). His research interests include the application of large language models in healthcare and telecommunications, as well as building and optimizing machine learning and forecasting models.

Nashid Shahriar (M'20) is an Associate Professor in the Department of Computer Science at the University of Regina, Canada. He received his PhD from the School of Computer Science, University of Waterloo in 2020. Dr. Shahriar's research has been recognized with multiple best paper awards, including the PST 2025 Best Paper Award, NOMS 2022 Best Student Paper Award, IM 2021 Best PhD Dissertation Award, CNSM 2019 Best Paper Award, NetSoft 2019 Best Student Paper Award, and the CNSM 2017 Best Paper Award. He is the recipient of 2023 Young Professional Award from the IEEE Communications Society Technical Committee on Network Operation and Management (CNOM). He also received the 2020 Alumni Gold Medal for outstanding academic performance in a doctoral program and the Mathematics Doctoral prize from the faculty of Mathematics at the University of Waterloo. His research interests include network management, network slicing and virtualization, and network security and reliability.

Massimo Tornatore is a Professor at Politecnico di Milano, Italy. He has also held appointments as Adjunct Professor at University of California, Davis, USA and as visiting professor at University of Waterloo, Canada. His research interests include performance evaluation and design of communication networks (with an emphasis on optical networking), and machine learning application for network management. He co-authored more than 500 conference and journal papers (with 23 best-paper awards) and of the recent Springer "Handbook of Optical Networks". He is member of the Editorial Board of IEEE Communication Surveys and Tutorials, IEEE Transactions on Network and Service Management, IEEE Transactions on Networking, and Elsevier Optical Switching and Networking. He is a fellow of the IEEE.

Raouf Boutaba (F'12) received the M.Sc. and Ph.D. degrees in computer science from Sorbonne University in 1990 and 1994, respectively. He is currently a University Chair Professor and the Director of the David R. Cheriton School of Computer science at the University of Waterloo, Canada. His research interests fall in the areas of computer networking and distributed systems. Dr. Boutaba served as the founding Editor-in-Chief of the IEEE Transactions on Network and Service Management (2007-2010) and the Editor-in-Chief of the IEEE Journal on Selected Areas in Communications (2018-2021). He is a fellow of the IEEE, the Engineering Institute of Canada, the Canadian Academy of Engineering, and the Royal Society of Canada.

Aladdin Saleh received the Ph.D. degree in electrical and electronic engineering and the M.B.A. degree in international management from the University of London, U.K. He is currently an Adjunct Professor with the Cheriton School of Computer Science, University of Waterloo. He is currently priming research and innovation activities with Rogers Communications, among them the joint research partnership with the University of Waterloo on 5G and emerging technologies.