Open-World Text Classification by Combining Weak Models and Large Language Models

Daniel P. Zitei[1], **Kenzo M. Sakiyama**[1], **Ricardo M. Marcacini**[1]

[1]Laboratório de Inteligência Computacional
Instituto de Ciências Matemática e de Computação
Universidade de São Paulo (ICMC-USP)

Abstract. *Open-world classification presents significant challenges in text classification. Large Language Models (LLMs) have made advances in addressing these challenges by leveraging their contextual understanding to improve classification accuracy without requiring knowledge of the entire label space. However, current LLM-based approaches still encounter limitations, such as context size constraints and computational scalability. To overcome these issues, we draw inspiration from strategies like Retrieval-Augmented Generation (RAG) to adapt LLMs more effectively to open-world classification problems. Our proposed approach combines a Weak Classifier (WM) Model with LLMs. In this case, we use the WM to filter and identify the top-k most probable classes, and then use a LLM to make the final classification decision.*

1. Introduction

Text classification with thousands of classes and open-world classification presents a complex challenge in Natural Language Processing (NLP). In the first problem, a classifier must understand the meaning of each class, each of which can have a very specific definition. In open-world text classification, the total number of classes is not defined. The classifier deals with both known classes (seen during training) and unknown classes (new ones that appear later). The classifier needs to identify instances from known classes, but also detect new classes it hasn't seen before [Parmar et al. 2023].

Even the latest LLMs have limitations when directly handling a large number of classes in few-shot scenarios. These challenges include context size, computational scalability, class imbalance, and generalization across a wide variety of categories [Chang et al. 2024]. Additionally, a challenge arises from the prompt length limit or context size, which makes the direct use of Large Language Models (LLMs) infeasible for problems with thousands of classes, as prompts may exceed their input capacity. LLMs that allow for a larger context also have limitations in efficiently pre-processing and retrieving information [Krishnamurthy et al. 2024]. In this challenging context, combining LLMs with content filtering strategies, such as Retrieval-Augmented Generation (RAG) [Fan et al. 2024], emerges as a promising approach to overcome these difficulties. However, RAGs are more suitable for improving information retrieval tasks and are not ideal for classification tasks. Thus, we raise the following question: *how can we determine the most probable classes in a problem with thousands of classes so that the LLM can make the final decision in a similar manner to a RAG*?

Large Language Models (LLMs) offer a promising alternative. They can use brief descriptions from users to identify relevant classes, even without labels or ground truth,

by relying on a small amount of context. Some methods use LLMs to iteratively discover and refine the label space after adjusting prompts [Li et al. 2024]. While this approach is interesting, it is relatively costly due to multiple inference steps with the LLM. We propose a more straightforward strategy. In open-world problems, a classifier like 1-NN rarely performs well. However, the literature shows that by selecting a reasonable number of top-k nearest neighbors, there is a high likelihood that the true class is among these top-k predictions [Petersen et al. 2022]. Alternatively, when none of the top-k neighbors is a good match, a minimum distance threshold can be used to determine whether the instance belongs to an unknown class. Our approach uses this idea to offer an alternative to open-world classification. In this case, we use the LLM more efficiently in the final step, leveraging its semantic capabilities to determine the final class from the top-k classes.

We introduce an alternative strategy to address the challenge of text classification with thousands of classes or open-world classification tasks with pre-defined classes. Our approach involves using a Weak Model (WM) based on K-Nearest Neighbours to perform an initial classification, identifying the top-k most probable classes for each text. We argue that evaluating the performance of the WM within this set of possible top-k classes, instead of seeking the final classification, yields significant results [Petersen et al. 2022]. This approach enables a preliminary filtering, significantly reducing the search space for the LLM. The LLM utilises its language generation capability to weigh the contextual and semantic information provided by the prompt with top-k classes, resulting in a more precise and refined final classification. This combination of WM and LLM allows us to explore the strengths of both models while mitigating their limitations, providing an efficient approach to text classification with a large number of classes.

Furthermore, we also explore Open LLMs in this work. Our study focuses on using LLama, Qwen2, Mistral, Phi3, anda Gemma2 models. These models were developed following best practices in data collection [Zhang et al. 2022]. They aim to promote responsible and reproducible large-scale research, as well as bring greater diversity of perspectives to the study of the impact of these LLMs. Thus, we explore alternatives that prioritise accessibility, transparency, and inclusion in the evaluation and research of these LLM models. In summary, the main contributions of this paper are two-fold:

- **A new approach for open-world text classification:** We combine a Weak Model based on K-Nearest Neighbours to perform an initial classification, identifying the top-k most probable classes. We demonstrate that this preliminary filtering allows for the generation of prompts for an LLM to obtain the final classification.
- **Evaluation and encouragement for the use of Open Large Language Models (LLMs):** We show that open models achieve competitive results, with a focus on comparing several state-of-the-art LLMs as part of our proposal.

We carried out experimental results that show LLMs can effectively identify the correct class among the top-15 most probable classes provided by KNN. This demonstrates the approach's value in open-world classification, leveraging the LLM's context to refine classification outcomes. The LLM+KNN combination achieved above 10% increase in F1-score over the baseline methods.

The remainder of this paper is organized as follows: In Section 2, we briefly discuss traditional methods in open-world classification and detail the advancements that LLMs are bringing to this field. In Section 3, we provide a detailed presentation of the

proposed approach, covering text preprocessing, the selection of the top-k most probable classes, and the generation of prompts for the LLM's final class decision. Section 4 presents the evaluation procedure. Experimental results are discussed in Section 5. Finally, Section 6 presents the conclusions and directions for future work.

2. Background and Related Work

In open-world classification, we deal with the problem of classifying instances into categories without having access to a complete list of all possible classes during training or testing [Parmar et al. 2023]. In other words, the model operates in an environment where new, previously unseen classes may emerge, and it must adapt to identify these new classes as they appear, while managing the potentially vast and unknown number of classes.

Open-world classification can be addressed as a hierarchical problem when the classes have a natural structure that can be organized into distinct levels of abstraction [Lin et al. 2024]. In these cases, classes can be grouped into broader categories and then further subdivided into more specific ones. If there is a clear hierarchical relationship among classes (e.g., general categories that are subdivided into subcategories), hierarchical organization can help the model navigate classification tasks. The model can first determine the broader category and then refine the decision to a more specific subcategory. For instance, consider a scenario wherein news articles are classified into journalistic categories. A hierarchical structure can be devised, encompassing broader categories such as 'Politics', 'Sports', and 'Entertainment', each cascading into finer-grained subcategories. 'Politics', for instance, might branch into subcategories like National Politics' and 'International Politics', further subdividing the taxonomy.

The functionality of these methods can be encapsulated into two stages:

1. Firstly, for the root layer (the broadest category), the model computes $P(y_1|x)$ for all classes y_1 within L_1, where x represents a document and $P(y_1|x)$ signifies the probability of the document's class being associated with any class in level L_1.
2. Building upon the classification of the root layer, the process is iteratively extended to subsequent layers, relying on the prior layer's classification and the classes within the present layer.

To estimate $P(y_l|x)$ in each layer, a diverse range of classification models can be employed, encompassing neural networks, decision trees, SVMs, contingent on the data characteristics. However, this strategy harbours limitations that hinder its application across many real-world scenarios. A principal limitation of this approach is the error propagation as the classification process descends the hierarchy. Erroneous classification at one level permeates subsequent classifications, leading to cascading inaccuracies.

In this sense, KNN classification has some advantages for handling open-world classification [Zhang et al. 2023]. Let C represent the set of all possible classes, where $|C| = N$, with N being large (potentially thousands or millions). In open-world classification, we define $C_{seen} \subset C$ as the set of known classes seen during training, and $C_{unknown} \subset C$ as the set of classes that are not seen during training but may appear later. Given a dataset D with n instances and m features, let x be a new instance to classify. The KNN algorithm assigns a class label based on the majority vote from the k nearest

neighbors. KNN can be adapted to recognize both known and unknown classes by using a decision rule, which can be formalized in Equation 1:

$$\begin{aligned} &\text{if } d(x, C_{seen}) < \epsilon \text{ then class}(x) \in C_{seen} \\ &\text{else class}(x) \in C_{unknown} \end{aligned} \tag{1}$$

where $d(x, C_{seen})$ is the distance from the instance x to the nearest class centroid in C_{seen}, and ϵ is a predefined threshold for classification confidence. This adaptation allows KNN to classify an instance based on its proximity to known class boundaries while also being capable of identifying instances that do not fit within these boundaries, defined as proxy unknowns [Zhang et al. 2023].

Two other popular approaches for open-world classification are (1) Incremental Learning and (2) Open Set Recognition. In Incremental Learning [Chen and Liu 2018], the model is continuously trained with new classes as they emerge. The model is periodically updated to include new information and classes, allowing it to adapt to new data and evolve over time. This adjustment can be done in real-time or through periodic retraining to incorporate new classes. On the other hand, Open Set Recognition [Geng et al. 2020] aims to identify when an instance does not belong to any of the known classes, allowing the model to determine if the instance is from a new or unknown class. This can be achieved using techniques such as anomaly detection or methods that assign a "reject class" to instances that do not fit well into any of the known classes.

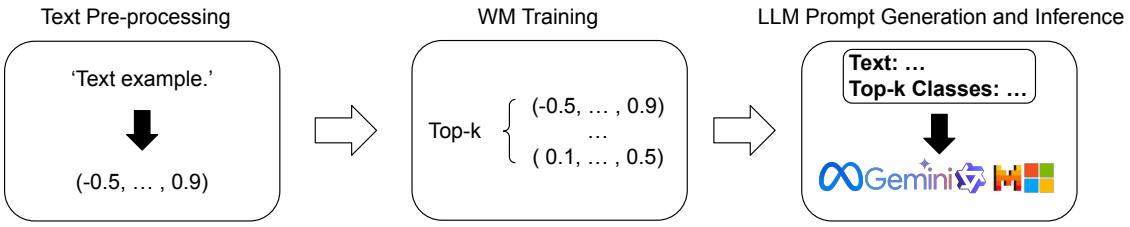
The zero-shot capabilities of Large Language Models (LLMs) make them a promising alternative for open-world classification [Kejriwal et al. 2024]. Unlike traditional models that need extensive training on predefined classes, LLMs can generalize to new, unseen classes without explicit examples [Li et al. 2024]. However, in zero-shot mode, LLMs can sometimes produce inaccurate results, as they are not specifically designed for classification tasks. To address this, it is helpful to condition the text generation by incorporating potential classes into the prompt [Zhang et al. 2024]. Yet, in open-world problems, the number of possible classes can be extremely large, potentially exceeding the context memory or including classes that are not well-known.

In this context, we reconsider the challenge of open-world classification by transforming the problem into a two-step process. The initial step involves a Weak Classifier based on K-Nearest Neighbours, which produces an initial classification. Its primary aim is to generate a list of the k most probable classes. Subsequently, we explore Large Language Models, wherein prompt engineering is employed to determine the ultimate class from the k most probable options.

3. Methodology

As previously mentioned, the goal of this paper is to propose a new approach for open-world text classification, combining WM and LLMs. Figure 1 presents an overview of the sequential steps present in our proposal. We start by pre-processing the text examples for classification, by converting them to meaningful dense embeddings. Next, we explore the WM in order to filter examples to the next step. At last, we use the filtered examples to create prompts that will be evaluated using the LLMs. The three steps will be further discussed in the following Sections.

Figure 1. Overview of the proposed methodology for open-world text classification, combining a Weak Model and a LLM.



3.1. Text Pre-processing

The text pre-processing step consists in converting raw text in to dense features (or embeddings). Given a dataset consisting of text samples $\{x_1, x_2, ..., x_n\}$ and their corresponding class labels $\{y_1, y_2, ..., y_n\}$, we aim to use an embedding function $f : x \to \mathbb{R}^d$ that maps each text x_i to a d-dimensional vector, capturing its semantic information. Mathematically, the embedding function can be defined as Equation 2,

$$e_i = f(x_i), \quad \text{for } i = 1, 2, ..., n \tag{2}$$

where $e_i \in \mathbb{R}^d$ represents the embedding vector for text x_i. The embedding function f is an encoder that utilizes the power of the LLM to generate embeddings that capture the semantic information of the texts. To obtain the embeddings, the Python package sentence-transformers [Reimers and Gurevych 2019] was utilised, employing the multilingual model paraphrase-multilingual-MiniLM-L12-v2 to create embeddings with 384 dimensions.

3.2. Weak Model Training

After converting text in to embeddings, the next step in our proposal is the definition of the Weak Model. We used a WM based on the K-Nearest Neighbours (KNN) algorithm in our experiments.

To build the KNN model, we utilize the embeddings as features. Given a new text sample x, we calculate its embedding e_x and find the k-nearest neighbours in the embedding space. This can be represented in Equation 3,

$$\text{Top-k neighbors of } x = \text{argmin}_{x_i}\, d(e_x, e_i) \tag{3}$$

where $d(\cdot, \cdot)$ represents a distance metric, such as Euclidean or cosine distance, used to measure the similarity between embeddings.

During preliminary experiments, we optimise the parameters of the k-nearest neighbours model to enhance its performance in identifying the top-k most probable classes for each text. This can be done by maximising the likelihood of the correct class labels within the top-k neighbours. Mathematically, we aim to find the optimal parameters θ by solving the following optimisation problem define in Equation 4,

$$\hat{\theta} = \text{argmax}_\theta \sum_{i=1}^{n} \mathbb{I}(y_i \in \text{Top-k neighbors of } x_i) \tag{4}$$

where $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if the condition is true and 0 otherwise. This objective ensures that the WM focuses on identifying the most probable classes within the top-k neighbors.

For example, when making a prediction, the K-Nearest Neighbours (KNN) algorithm identifies the 15 most similar instances (k=15) within the training set using the cosine distance metric, and assigns the new data point to the class that is most prevalent among these 15 nearest neighbours. This top-k classification acts as an initial filtering step, effectively narrowing the search space for subsequent stages and enhancing computational efficiency.

For this work, we used a KNN classifier with a pre-defined set of known classes. However, as mentioned in Section 2, we described a simple method for extending KNN classification to open-world scenarios by incorporating the concept of proxy unknowns. This technique allows the model to handle instances that do not fit into any known class.

3.3. Large Language Model Prompt Generation and Inference

In this step, we leverage the output of the WM to generate a specific prompt for each text, which will serve as input for the Large Language Model (LLM) for the final classification. The prompt is carefully constructed to incorporate relevant information from the top-k classes identified by the WM. This can include class labels, textual descriptions, or other contextually informative features that aid the LLM in making accurate predictions.

To generate the prompt, we employ a few-shot prompt approach. This involves providing the LLM a limited amount of labelled data in the prompt, specifically the text samples associated with the top-k classes from the WM's output. We can represent the few-shot prompt learning as follows. Let D_{train} be the training dataset consisting of labeled text samples and their associated class labels. Given the WM's output of top-k classes for a text x, denoted as $C_{\text{top-k}}$, we extract the corresponding labeled samples from D_{train}, as defined in Equation 5.

$$D_{\text{few-shot}} = \{(x_i, y_i) \,|\, (x_i, y_i) \in D_{\text{train}}, \text{ and } y_i \in C_{\text{top-k}}\} \tag{5}$$

We then use $D_{\text{few-shot}}$ to compose the LLM prompts, enabling it to generate more accurate predictions based on the given prompt. Mathematically, let's denote the prompt as P and the target class label as y. We aim to generate a prompt that maximizes the conditional probability of the target class label given the prompt, denoted as $P(y|P)$.

The LLM's task is to generate the next word conditioned on the prompt, as defined in Equation 6,

$$P(y|P) = P(y|\text{Prompt}) = \prod_{t=1}^{T} P(y_t|y_{t-1}, \text{Prompt}), \tag{6}$$

where y_t represents the predicted word at time step t, and Prompt represents the prompt provided to the LLM. The probability of each word y_t is conditioned on the previous words y_{t-1} and the prompt. For example, let's consider an instance where the WM identifies three top-k classes for a given text:

Input text: "***A devastating earthquake struck a small town, causing widespread damage and loss of life***". Using the WM, we obtain the top-k most probable classes for this text. In this case, let's assume k=10. The top-k classes based on the WM's output are:

1. Earthquakes
2. Disasters and Accidents
3. Emergency Planning
4. Natural Disasters
5. Seismic Activity
6. Geology
7. Crisis and Emergency Management
8. Earth Science
9. Environmental Accidents and Disasters
10. Emergency Incidents

Now, let's construct the prompt incorporating this information. Figure 2 shows an example of prompt containing the filtered top-k classes.

Figure 2. Example of prompt containing the top-k labels obtained from the WM.

Classify the following text related to a recent natural disaster into one of the following categories: Earthquakes, Disasters and Accidents, Emergency Planning, Natural Disasters, Seismic Activity, Geology, Crisis and Emergency Management, Earth Science, Environmental Accidents and Disasters, Emergency Incidents.
Text: "A devastating earthquake struck a small town, causing widespread damage and loss of life."
Answer:

In this example, the prompt clearly indicates the task of classifying the given text related to a recent natural disaster into one of the provided categories, enabling the LLM to focus only on relevant classes of the classification task. We omitted the few-shot examples for visual clarity. It is worth noting that when none of these classes are a good match (as defined in Equation 1), we can change the prompt to request a new (potential) class based on the top-k most similar classes. Alternatively, instead of creating a new class, the model can use the concept of ***proxy unknowns*** to handle instances that do not fit any known category [Zhang et al. 2023].

4. Experimental Evaluation

This Section describes the evaluation procedure used to validate our proposal.

4.1. Evaluated LLMs

For this study, five state-of-the-art language models (see Table 1) representing major industry players were selected: LLaMA 3.1 [Dubey et al. 2024] (Meta), Gemini 2 [Team et al. 2024] (Google), Mistral-NeMo [MistralAI and NVIDIA] (NVIDIA),

Qwen-2 [Yang et al. 2024] (Alibaba), and Phi-3 [Abdin et al. 2024] (Microsoft). We chose to evaluate open and free-to-use models, due to the financial budget of this work. In addition, open models enable the usage of private and sensible data in local hardware, instead of online third-party APIs. This is specially important in countries with data protection laws, such as Brazil's LGPD ('Lei Geral de Proteção de Dados Pessoais').

To evaluate the performance of models with varying capacities, we opted to use the largest available variant for each model within the 7 billion to 14 billion parameter range. Given that the models utilised do not necessarily possess prior knowledge of the labels presented in the dataset, we are conducting experiments with large language models (LLMs) by providing contextual information derived from a nearest neighbors model.

This approach allows us to evaluate the performance of LLMs when supplemented with additional context, ensuring a more effective handling of domain-specific labels that may not be inherently recognised by the models.

Table 1. Used LLMs and their respective sizes in terms of parameters.

Model	Size
Llama (LLaMA 3.1)	8B
Gemma2	9B
Mistral (nemo)	12B
Qwen2	7B
Phi3	14B

4.2. Dataset

Our proposal aims to investigate an approach for text classification with large amounts of labels. For this purpose, we used the DBPEDIA dataset[1]. This dataset includes 342,782 Wikipedia articles, each classified into taxonomic categories with three hierarchical levels: 9 top-level classes, 70 mid-level classes, and 219 detailed classes. This dataset is widely used as a benchmark for NLP and text classification tasks. Moreover, it is particularly challenging when using the lower-level categories as targets, making it a strong benchmark to simulate open-world classification taslks.

In our experiments, we use use 220 most frequent classes for different categories of texts, capturing the richness and diversity of real-world entities and concepts. Among the categories represented in this dataset are, but are not limited to, the following:

- Entertainment Categories: Such as 'Adult Actor', 'Album', 'Manga', and 'Hollywood Animated Film'.
- Sports Categories: Such as 'American Football Player', 'Basketball Team', 'Cycling Athlete', and 'Mixed Martial Arts Event'.
- Political and Social Categories: Such as 'President', 'Philosopher', 'Poet', 'Political Party', and 'Prime Minister'.
- Geographic and Natural Categories: Including 'Volcano', 'Mountain', 'River', 'Planet', 'Village', and 'Galaxy'.
- Cultural Categories: Such as 'Classical Music Artist', 'Classical Music Composer', 'Theatre', and 'Brewery'.

[1] https://www.kaggle.com/datasets/danofer/dbpedia-classes

4.3. Evaluation Details

To evaluate our proposal we used a standard holdout procedure. We stratified the dataset prior to splitting it into a 75% training set and a 25% validation set. This stratification process ensured that the class proportions in each subset were similar to those in the original dataset. Then, we evaluated all methods using the same data splits for training and evaluation.

In order to evaluate the effectiveness of our proposal, we evaluated our methodology in two scenarios:

- **Standalone LLMs**: we evaluate the LLMs without using the WM to filter the top-k most probable labels. Hence, the LLM receive as input, a prompt with the example to be classified and the 220 DBPEDIA labels to choose from. The standalone LLMs and WM act as our baselines for the experiments.
- **LLMs+WM**: we use the LLM+WM combination (our proposal) in the classification procedure, and compare the results with the previous experiment.

5. Results and Discussions

Table 2. Classification metrics for the evaluation of standalone LLMs. Phi3 did not finish the execution in our execution time limit (3 days). Precision, Recall and F1 scores correspond to the macro average.

Model	Accuracy	Precision	Recall	F1
KNN (top-1)	0.7549	0.7656	0.7590	0.7457
Qwen2	0.6895	0.7733	0.6881	0.6865
Llama	0.6774	0.8067	0.6734	0.6949
Mistral	0.7585	0.8512	0.7529	0.7680
Gemma	0.7788	0.8488	0.7758	0.7742
Phi3*	-	-	-	-

Table 2 present the results of our experiments with the standalone LLMs, in addition to the metrics obtained by the WM. The underlined values correspond to the simpler model KNN (using most probable class). The LLMs show larger gains in Precision, in relation to the WM, indicating a lower number of false positives. However, the gains in other metrics doesn't reflect the greater complexity of LLMs. In fact, the LLMs had lower performance in Accuracy and F1 than the WM in some cases (Qwen and Llama). These results reinforce the limitations of LLMs when dealing with a large amount of labels in few-shot scenarios.

Next, Table 3 shows the results of the LLMs+WM. In bold, we highlight the LLM+KNN (Gemma+KNN) combination that yielded the best results. It is evident that the combination of the LLM+KNN significantly outperformed WM and standalone LLMs (see Table 2) in all metrics. In relation to the WM, we achieved a 15.62 percentage point increase in F1. In relation to the best standalone LLM (Gemma) we observed a 12.78 pecentage point increase in F1. This result shows that our proposed method was effective in improving both the performance of the WM and LLMs.

We observed that, although Gemma had the best results, all the LLMs had similar results when combined with the WM. Comparing the worst (Llama) with the best

Table 3. Same metrics shown in Table 2, but for for each combination of WM and LLM.

Model	Accuracy	Precision	Recall	F1
KNN (top-1)	0.7549	0.7656	0.7590	0.7457
+ Llama	0.8714	0.8955	0.8733	0.8734
+ Qwen2	0.8754	0.8902	0.8775	0.8740
+ Mistral	0.8773	0.8944	0.8779	0.8769
+ Phi3	0.8864	0.9032	0.8888	0.8880
+ Gemma2	**0.9034**	**0.9091**	**0.9050**	**0.9020**

(Gemma) LLMs, the difference in between all metrics doesn't reach three percentage points. This result is relevant because can help the LLM choice. If the goal is to maximize a metric, the Gemma LLM would be prefered. However, if the available computation resources are more restrict, the Qwen2 model (with less parameters) could be used instead with a minor performance degradation. Other interesting result is that Qwen2+KNN outperforms all the standalone LLMs, including models that are larger than Qwen2.

The key result is that LLMs can accurately identify the correct class among the top-15 most probable classes provided by KNN. This approach is valuable for open-world classification, as it allows the model to use additional context from the LLM to refine and improve classification outcomes. Notably, the approach does not require the LLM to know all possible classes but focuses on the relevant ones for each instance, enabling the creation of specific prompts for new instances. Moreover, new documents and classes inserted into training will be naturally introduced to the LLM if they are among the most probable given the neighborhood space. Also, as shown by the results, the combination can be used to make smaller models outperform larger LLMs.

Finishing the discussions, we will discuss qualitatively some miss-classifications of the models. Analyzing the standalone Gemma errors we observed that the model exhibited a consistent bias towards some specific classes. To illustrate, we discuss two examples of errors found in the **Diocese** and **WomensTennisAssociationTournament** (WTAT) class below.

In the case of the first text, *"The Vicariate Apostolic of Inírida (Latin: Apostolicus Vicariatus Iniridanus) in the Catholic Church is located in the town of Inírida, Guainía in Colombia,"* the actual class is **Diocese**, yet the LLM without WM classified it as **Religious**. Although the LLM correctly identifies the religious context of the text, the model fails to assign the specific classification of "Diocese", which refers to a distinct organizational unit within the Catholic Church. This highlights a limitation in the LLM's ability to distinguish between more nuanced and hierarchical terms within the religious domain.

For the second text, *"The Nuremberg Cup (currently sponsored by Nürnberger Versicherung) is a tennis tournament for women on the WTA Tour held in Nuremberg, Germany,"* the actual class is **WomensTennisAssociationTournament**, and while the KNN model accurately classifies the text, the LLM categorises it more broadly as **TennisTournament**. This result indicates that while the LLM can recognize the general context of the text, it lacks the specificity provided by the KNN model, which correctly identifies the association with the WTAT. The LLM+WM, in this instance, generalizes too much,

missing out on critical details such as the gender and specific organization related to the event.

6. Concluding Remarks

We presented a new open-world text classification approach, which integrates K-Nearest Neighbors (KNN) with Large Language Models (LLMs). This strategy combines two well-established techniques: the KNN algorithm for initial classification and prompt-based LLM classification for refined prediction. The experiments demonstrated that this combined approach effectively allows LLMs to accurately identify the correct class among the top-k classes filtered by kNN in most cases, outperforming both the KNN and the standalone LLMs.

The combination of a KNN with a pre-trained off the shelf LLM (Gemma), without further fine-tunning, yielded gains above 10 percentage points in traditional classification metrics in the DPBEDIA dataset. The other evaluated LLM also presented competitive performance, and we highlighted the Qwen2 (lowest parameter count) performance. We also investigated the miss-classifications of our pipeline, and discussed potential causes.

As future works, we plan to extend the experimental evaluation of the presented methodology, evaluating both more datasets and baselines. Evaluating prompts and the performance of LLMs in suggesting new classes beyond those seen in training, or in handling proxy unknowns, is a promising area for future research. We also intend to evaluate more WMs, aiming to improve the example filtering for the LLM classification. Additionally, we plan to include paid LLMs to our baselines (such as GPT-4). Finally, we will explore the integration of feedback mechanisms and continuous learning techniques for future LLM prompt adjustments.

References

Abdin, M., Jacobs, S. A., Awan, A. A., Aneja, J., Awadallah, A., Awadalla, H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H., et al. (2024). Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., et al. (2024). A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Chen, Z. and Liu, B. (2018). Open-world learning. In *Lifelong Machine Learning*, pages 77–89. Springer.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. (2024). The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., and Li, Q. (2024). 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*, pages 6491–6501.

Geng, C., Huang, S.-j., and Chen, S. (2020). Recent advances in open set recognition: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3614–3631.

Kejriwal, M., Kildebeck, E., Steininger, R., and Shrivastava, A. (2024). Challenges, evaluation and opportunities for open-world learning. *Nature Machine Intelligence*, pages 1–9.

Krishnamurthy, A., Harris, K., Foster, D. J., Zhang, C., and Slivkins, A. (2024). Can large language models explore in-context? *arXiv preprint arXiv:2403.15371*.

Li, X., Jiang, J., Dharmani, R., Srinivasa, J., Liu, G., and Shang, J. (2024). Open-world multi-label text classification with extremely weak supervision. *arXiv preprint arXiv:2407.05609*.

Lin, Z., Yang, W., Wang, H., Chi, H., Lan, L., and Wang, J. (2024). Scaling few-shot learning for the open world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13846–13854.

MistralAI and NVIDIA. Mistral nemo - technical report. Available at `https://mistral.ai/news/mistral-nemo/` (2024/08/22).

Parmar, J., Chouhan, S., Raychoudhury, V., and Rathore, S. (2023). Open-world machine learning: applications, challenges, and opportunities. *ACM Computing Surveys*, 55(10):1–37.

Petersen, F., Kuehne, H., Borgelt, C., and Deussen, O. (2022). Differentiable top-k classification learning. In *International Conference on Machine Learning*, pages 17656–17668. PMLR.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Team, G., Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., et al. (2024). Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., et al. (2024). Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Zhang, Q., Shi, Z., Zhang, X., Chen, X., Fournier-Viger, P., and Pan, S. (2023). G2pxy: generative open-set node classification on graphs with proxy unknowns. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 4576–4583.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. (2022). Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhang, Y., Wang, M., Ren, C., Li, Q., Tiwari, P., Wang, B., and Qin, J. (2024). Pushing the limit of llm capacity for text classification. *arXiv preprint arXiv:2402.07470*.