Auto-Tagging of Text Using Large Language Models (LLM)

Aleksandr Vashchenko
Email: a.vashchenko@innopolis.university

Grigoriy Nesterov
Email: g.nesterov@innopolis.university

1. Introduction

The rapid expansion of data science and machine learning has significantly increased the volume and complexity of datasets available for analysis. As the demand for structured and accessible datasets continues to grow, the ability to efficiently tag and categorize these datasets becomes crucial. This project, titled "Auto-Tagging of Text Using Large Language Models (LLM)", addresses this challenge by leveraging advanced natural language processing (NLP) techniques to automate the tagging process of dataset descriptions, taking Kaggle as a reference.

Kaggle, a prominent platform for data science competitions and collaboration, hosts a vast repository of datasets contributed by users worldwide. A key feature of Kaggle's dataset repository is the ability to tag datasets with relevant keywords, enhancing their discoverability and usability. Traditionally, this tagging process has been manual, which can be time-consuming and prone to inconsistencies. Users must select from a set of over 500 tags organized into 6 groups, with a maximum of only 5 tags per dataset.

The primary objective of this project is to develop a system that automatically recommends relevant tags for dataset descriptions. By utilizing Large Language Models (LLMs)[8], we aim to improve the efficiency and accuracy of the tagging process, thereby streamlining the workflow for data scientists and researchers.

Automated tagging of dataset descriptions offers several advantages:

- Enhanced Discoverability: Accurate tagging improves the searchability and organization of datasets, making it easier for users to find relevant datasets.
- Time Efficiency: Automating the tagging process reduces the time and effort required to manually assign tags, allowing users to focus more on data analysis and experimentation.
- Consistency and Quality: Automated systems can maintain a high level of consistency in tagging, reducing the variability and potential errors associated with manual tagging.

Our approach involves the application of state-of-the-art Large Language Models, specifically fine-tuned versions of models like BERT[1] and LLaMA[7], to understand and generate relevant tags for dataset descriptions.

Through extensive experimentation and evaluation, we aim to demonstrate the effectiveness of our system in providing accurate and contextually relevant tags, thereby contributing to the broader goal of enhancing dataset management and utilization on our platform.

In this system, the model acts as a recommendation engine, proposing relevant tags based on the dataset's description, title, and subtitle. The final decision on tag selection rests with the user, who can choose to accept, modify, or add additional tags. This ensures that the user retains control over the tagging process while benefiting from the model's suggestions.

2. Motivation

The increasing volume and complexity of datasets in the data science and machine learning fields have made efficient tagging and categorization of datasets a crucial task.

2.1. Problem Statement

However, the current manual tagging process presents several significant challenges:

2.1.1. Overwhelming Number of Tags. Kaggle offers a repository of over 516 tags organized into 6 groups for dataset categorization. The sheer number of available tags makes it challenging for users to manually select the most appropriate tags for their datasets. This often results in the selection of irrelevant or suboptimal tags, which affects the overall quality of the tagging process.

2.1.2. User Fatigue and Inconsistency. Manually choosing correct tags can be a tedious and time-consuming task. Users might experience fatigue or indifference, leading to inconsistent or incorrect tag selection. This inconsistency compromises the discoverability and accessibility of datasets, as improperly tagged datasets become difficult to locate through search functions.

2.1.3. Impact on Dataset Discoverability. Incorrect or suboptimal tagging directly impacts the ease with which datasets can be found. Accurate tagging is essential for efficient dataset retrieval, enabling users to quickly identify and access relevant datasets for their research and analysis. Poor tagging practices hinder this process, making it harder for data scientists and researchers to find the datasets they need.

2.1.4. Consistency and Quality Control. An automated tagging system ensures a higher level of consistency in tag application. This consistency improves the overall quality of the dataset repository, making it more reliable and user-friendly. Automated systems can apply tags based on a thorough understanding of dataset descriptions, ensuring that tags are relevant and accurate.

2.1.5. Scalability. As the number of datasets continues to grow, manual tagging becomes increasingly unsustainable. An automated system can scale effortlessly to accommodate the expanding dataset repository, maintaining high standards of tagging accuracy and relevance without additional manual effort.

2.2. Proposed Solution: Leveraging Large Language Models

By leveraging Large Language Models (LLMs) to automate the tagging process, this project aims to address these challenges. LLMs, such as BERT and LLaMA, are capable of understanding and generating relevant tags based on dataset descriptions. This approach promises to enhance the efficiency, accuracy, and consistency of the tagging process, ultimately improving the discoverability and usability of datasets on our platform.

3. Related Work

3.1. Text Vectorization and Feature Extraction

Text vectorization is a fundamental technique in natural language processing (NLP) used to convert textual data into numerical representations suitable for machine learning models. Among traditional methods, Term Frequency-Inverse Document Frequency (TF-IDF) is widely recognized. TF-IDF measures the importance of a word in a document relative to its frequency across a collection of documents. This technique has proven effective and simple for various text classification tasks [6].

3.2. Embeddings and Tokenization

Embeddings are a critical aspect of modern NLP, mapping words or phrases to dense vectors that capture their semantic meaning. Word2Vec and GloVe are early techniques that laid the groundwork for generating word embeddings used in NLP tasks [3, 4]. More recent models, such as BERT and LLaMA, employ advanced embedding techniques to enhance their understanding of language and context.

Tokenization is another crucial preprocessing step, involving the conversion of text into tokens. Modern tokenizers, including those used in BERT and LLaMA, handle subword units to manage vocabulary more effectively and improve model performance across diverse text inputs.

3.3. Existing Datasets and Challenges

Several datasets have been developed to address the challenges associated with multi-label text classification. The Amazon Review dataset offers a substantial collection of product reviews categorized by sentiment and attributes. Similarly, the 20 Newsgroups dataset comprises around 20,000 newsgroup documents, organized into 20 categories, and is frequently used to benchmark text classification algorithms [2].

Nevertheless, many datasets face limitations in the number and diversity of labels. The creation of datasets specifically designed for multi-label classification with a wide range of tags remains an ongoing research challenge. For example, the Kaggle dataset repository features a broad spectrum of datasets but faces issues with tag consistency and relevance due to its extensive tag system with over 500 tags organized into six groups.

3.4. Text Classification with Large Number of Labels

Classifying texts into a large number of labels presents unique challenges, such as managing class imbalance and ensuring effective representation of each class. Zhang and Zhou (2014) address these challenges in their work on "Multi-Label Text Classification with Deep Neural Networks," which explores various neural network architectures and emphasizes the need for scalable and efficient models [10].

3.5. Use of BERT for Text Classification

BERT (Bidirectional Encoder Representations from Transformers), introduced by Devlin et al. (2019), has significantly advanced NLP by providing pre-trained language representations that capture deep contextual relationships between words [1]. BERT has been effectively applied to diverse text classification tasks, including sentiment analysis and topic categorization, owing to its ability to understand context and semantics, making it suitable for fine-tuning on specific classification tasks.

3.6. Advances with LLaMA

LLaMA (Large Language Model Meta AI), developed by Touvron et al. (2023), represents a significant advancement in large language models designed to be both efficient and scalable [7]. LLaMA builds upon the successes of previous models like BERT and GPT [5], offering improvements in performance and efficiency. Its capability to handle complex language tasks makes it a promising tool for text classification and tagging applications.

4. Methodology

4.1. Data Collection

The data collection phase involved scraping dataset tags and metadata from Kaggle, a prominent platform for data science and machine learning competitions. Kaggle's repository includes a comprehensive tagging system with 516 tags divided into six categories: data type, geography and location, subject, technology, task, and language. Metadata for each dataset was collected uniformly across all tags to maintain balanced representation and avoid data imbalance issues.

The collected metadata includes the following attributes for each dataset:

- **User-selected tags:** Up to five tags chosen by users.
- **Dataset title:** The main title of the dataset.
- **Subtitle:** An additional descriptive title.
- **Description:** A detailed description of the dataset.
- **Dataset ID:** A unique identifier consisting of the username and the dataset's internal name.
- **Usability rating:** A rating indicating the dataset's ease of use.
- **View count:** The number of times the dataset has been viewed.
- **Rating count:** The number of ratings the dataset has received.
- **Download count:** The number of times the dataset has been downloaded.
- **License:** The license under which the dataset is released.
- **Boolean indicators:** Flags indicating the presence of specific text variables.
- **Privacy status:** Whether the dataset is private.
- **Collaborator list:** The list of collaborators involved in the dataset.

A total of 4203 metadata entries were downloaded, ensuring a comprehensive and balanced dataset for training.

4.2. Data Preprocessing

Data preprocessing involved several key steps to prepare the data for model training:

- **Translation:** Tags were translated from English to Russian to cater to the Russian-speaking audience, as the final system is intended for use by a diverse audience, including Russian speakers.
- **Feature Selection:** Only the dataset title, subtitle, and description were retained as features for model input. These features were concatenated into a single text input and processed according to the requirements of the chosen models, either BERT or LLaMA.
- **Tokenization for BERT:** The BERT tokenizer was used to convert the concatenated features into token embeddings suitable for the model.
- **Prompt Creation for LLaMA:** A prompt was created incorporating the concatenated text features to align with the LLaMA model's requirements.

4.3. Model Training

Large Language Models (LLMs) were employed to automate the tagging process. The primary models used in this project are:

- **BERT (Bidirectional Encoder Representations from Transformers):** Developed by Devlin et al., BERT is fine-tuned to understand and generate contextually relevant tags based on dataset descriptions. It captures deep contextual relationships between words, making it effective for text classification tasks.
- **LLaMA (Large Language Model Meta AI):** Developed by Touvron et al., LLaMA enhances efficiency and scalability in handling large-scale text data. Its advanced capabilities in understanding complex language patterns improve the tagging process.

The training process includes the following steps:

4.3.1. Model Selection. Appropriate pre-trained models were selected and initialized for fine-tuning. Used the torch library, the models were fine-tuned with specific architectures.

- **For BERT**, the BertModel from the transformers library and its corresponding tokenizer were chosen. The architecture included the bert_model, a dropout layer, two linear layers, and a Leaky ReLU function.
- **For LLaMA**, the pre-trained model IlyaGusev/saiga_LLaMA3_8b was selected, suitable for processing Russian text. A standard architecture with bfloat16 data type was used.

4.3.2. Hyperparameter Tuning. The following hyperparameters were considered to optimize the performance of the model:

- **For BERT:** Number of tokens, model architecture, after bert layer, layer coefficients, loss function.
- **For LLaMA:** Prompt type, prompt configuration, quantization and other techniques to reduce the required temporary memory, optimizer, learning rate of the optimizer, number of tokens, various LLaMA model coefficients.

4.3.3. Fine-Tuning.

- **For BERT**, the model was fine-tuned on the prepared balanced dataset from Kaggle.
- **For LLaMA**, the model was fine-tuned on the prepared balanced dataset from Kaggle. Additionally, outlines for label classes were generated to provide structured information about each class, thereby enhancing the model's ability to comprehend and generate relevant tags through improved context and consistency.

4.4. Model Evaluation

Given the complexity of the task, qualitative evaluation through manual review was chosen:

Quality Assessment: A manual review of the tags proposed by the model was conducted to ensure their contextual relevance and usefulness. The model's suggested tags were compared with the original tags set by users during dataset creation to evaluate consistency and accuracy.

4.5. Implementation of the Recommendation System

The final model is integrated into a recommendation mechanism, encompassing the following process:

- **Tag Suggestion:** The model generates a list of relevant tags based on the dataset's description, title, and subtitle.
- **User Interface:** In the user interface, users can view the suggested tags, make modifications, and finalize the tagging. This approach ensures that users have control over the final tags applied to their datasets, enhancing user satisfaction and tag accuracy.

5. Experiments and Evaluation

5.1. Experimental Setup

The experiments aimed to evaluate the performance of the BERT and LLaMA models for automated dataset tagging. Various configurations and hyperparameters were tested to optimize the models for multi-label classification tasks. The experiments were conducted with the following details:

5.1.1. BERT Model Experiments. Several configurations were tested for the BERT model to determine the optimal settings for tagging performance:

- **Target Labels:** The number of target labels was varied to assess its impact on model performance.
- **Linear Layers:** Different dimensions and numbers of linear layers were experimented with to find the optimal network architecture.
- **Epochs:** The number of training epochs was adjusted to evaluate its effect on the model's convergence and performance.
- **Loss Functions:** Two loss functions were used:
  - **Cross-Entropy Loss:** A commonly used loss function for classification tasks.
  - **Asymmetric Loss for Multi-Label Classification:** An advanced loss function designed to handle class imbalance and improve performance in multi-label scenarios [9].

5.1.2. Hyperparameter Tuning. The following hyperparameters were selected to optimize model performance:

- **Dropout rate:** 0.3
- **First linear layer dimensions:** 1024 to 2048
- **Second linear layer dimensions:** 2048 to 516 (number of classes)
- **Maximum token count**: 512
- **Loss function**: Assymmetric Loss for Multi-Label Classification

The experiments revealed that changes in the number of linear layers and epochs significantly affected model performance, while the choice of loss function provided varying results based on the class distribution and label imbalance. But in the end, the model often gave out often only a fraction of the classes and some selected tags did not directly correspond to the expected result. For example, the dataset for regression was defined as a dataset for classification.

5.1.3. LLaMA Model Experiments. The LLaMA model was evaluated under different experimental settings to identify the most effective configuration:

- **Optimizer:** The optimal optimizer and learning rate for it were selected.
- **LORA:** the lora method was applied.
- **Hyperparameters of LLaMA model:** The optimal parameters of the model were selected.
- **Prompt Variations:** Different prompts were tested to optimize the model's ability to generate relevant tags.
- **Outlines:** Structured outlines for label classes were incorporated to improve the model's understanding and generation of contextually relevant tags.

The experiments indicated that LLaMA demonstrated superior performance when using specific outlines for label classes, which provided structured information and enhanced the model's contextual understanding. The optimal configuration for LLaMA included the following hyperparameters:

- **Prompt:** Zero-Shot Learning prompt
- **LORA Parameters:** alpha of 32
- **Dropout Rate:** 0.1
- **train_batch_size:** 1
- **eval_batch_size:** 8
- **r:** 8
- **seed:** 42
- **Learning Rate:** 3e-4
- **Optimizer:** AdamW
- **epsilon:** 1e-8
- **betas:** (0.9, 0.999)
- **Fine-Tuning Epochs:** 1
- **Maximum Sequence Length:** 1024
- **Maximum New Tokens:** 512
- **Top_p:** 0.9
- **Temperature:** 0.5
- **Repetition Penalty:** 1.1

5.2. Results and Discussion

The analysis of the experiments demonstrated that LLaMA, with the optimal configuration, outperformed BERT in terms of accuracy and relevance of generated tags. The use of structured contours for label classes did not show positive results, so it was decided not to use it in the final solution. LLaMA's superior performance can be attributed to its advanced model architecture and the parameters of fine-tuning, which improved its ability to handle complex language patterns and multi-label classification tasks in current domain.

BERT, while effective, showed limitations in handling large numbers of labels and class imbalances. The best-performing settings for BERT included a balanced approach to linear layer dimensions and epochs, as well as the use of asymmetric loss for multi-label classification.

6. Analysis and Observation

6.1. Performance Comparison

The comparison of BERT and LLaMA models revealed significant differences in their performance for automated dataset tagging. LLaMA consistently outperformed BERT in generating accurate and relevant tags. This superiority can be attributed to LLaMA's advanced model architecture, optimized prompt and the effective use of parameters of fine-tuning, which provided enhanced contextual understanding in current domain. Changes in hyperparameters had notable effects: for BERT, variations in the number of linear layers and epochs impacted its performance. For LLaMA, the choice of prompts and fine-tuning epochs played a crucial role in optimizing performance.

6.2. Model Strengths and Limitations

BERT demonstrated strengths in understanding context and performing well with a moderate number of labels. LLaMA, on the other hand, excelled in managing complex language patterns and fine-tuning, which improved its performance in multi-label classification tasks. Despite its advantages, LLaMA faced challenges, such as incorrect operation after applying outlines.

7. Conclusion

This study aimed to enhance automated dataset tagging by evaluating the performance of BERT and LLaMA models. The experiments showed that LLaMA, with optimized fine-tuning and prompting, outperformed BERT in generating accurate and contextually relevant tags. LLaMA's advanced understanding of complex language patterns, prompting, fine-tuning and structured information provided a significant advantage in multi-label classification tasks, resulting in superior tagging performance.

While BERT was effective in understanding context, it struggled with large numbers of labels. The experiments underscored the importance of selecting appropriate hyperparameters, such as the number of linear layers and suitable loss functions, to improve BERT's performance. However, even with these adjustments, BERT's performance was less consistent than LLaMA's.

Repo for development: https://github.com/abobafett-dev/IU-LLM-project-2024/tree/main.

Repo for deployment part with docker: https://gitlab.pg.innopolis.university/g.nesterov/tab-with-datasets-aes-ml-deploy

8. References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2019. URL https://arxiv.org/abs/1810.04805.

[2] Ken Lang. Newsweeder: Learning to filter netnews. *Proceedings of the Twelfth International Conference on Machine Learning*, 1995. URL https://dl.acm.org/doi/10.5555/645528.655813.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26, 2013. URL https://arxiv.org/abs/1310.4546.

[4] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. *Empirical Methods in Natural Language Processing (EMNLP)*, 2014. URL https://aclanthology.org/D14-1162.

[5] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Blog*, 2018. URL https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

[6] Gerard Salton and Michael J. McGill. Introduction to modern information retrieval. *McGraw-Hill*, 1988.

[7] Hugo Touvron, Matthieu Cord, Lancelot Weissenborn, Pierre-Antoine Manzagol, Paul Sermanet, and Hervé Jégou. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. URL https://arxiv.org/abs/2302.13971.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. URL https://arxiv.org/abs/1706.03762.

[9] J. Wang and X. Zhang. Asymmetric loss for multi-label classification. *arXiv preprint*, 2020. URL https://arxiv.org/abs/2009.14119.

[10] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014. URL https://ieeexplore.ieee.org/document/6868444.