Best RoLLM feature extractor

Ana-Maria Roșu, Alexandru Miclea, Liviu Fircă

Faculty of Mathematics and Computer Science, University of Bucharest

Objectives

This study investigates which Romanian LLM (RoLLM) is the most effective feature extractor for downstream tasks, as well as which pooling strategy yields the best results. We also include a comparison between Romanian-specific models and general English LLMs to assess the value of language specialization. Our methodology involves extracting hidden states from open-source LLMs and using them as fixed embeddings for downstream classification.
We evaluate performance on two Romanian-language benchmarks:

- LaRoSeDa (Sentiment Classification)
- RoARC (Commonsense Reasoning)

Introduction

Large Language Models (LLMs) are increasingly used as general-purpose computation engines in artificial intelligence. They can often solve tasks with minimal or even zero labeled data by simply prompting them. However, this zero-shot approach is inherently non-deterministic and can yield unreliable results. Moreover, most LLMs are not explicitly optimized for classification tasks, which may limit their performance in such settings. An alternative is to use transfer learning — a well-established technique in machine learning. In our approach, we extract hidden state vectors from the final transformer block (just before the unembedding layer), leveraging the richer representations found in higher layers. These vectors are then used as features for training a lightweight multi-layer perceptron (MLP) classifier on the downstream task.

Embedding Extraction

To extract embeddings, we used a rented NVIDIA A100 GPU via Google Colab Pro to load models hosted on Hugging Face. We constructed prompts according to the strategies described in the Embedding Methods and Strategies section. After tokenizing the prompts, we extracted the hidden states corresponding only to the target input and saved the resulting embedding vectors in .pt and .npy format for efficient use in downstream MLP classification tasks.

Model Architecture

We use a three-layer multilayer perceptron (MLP) defined as:

$$f(x) = \sigma(M_3 \cdot ReLu(M_2 \cdot ReLu(M_1 \cdot \vec{v})) \quad (1)$$

$$\textbf{where } M_3 \in \mathbb{M}_{1\times 64} \text{ , } M_2 \in \mathbb{M}_{64\times 128} \text{ , } M_1 \in \mathbb{M}_{128\times 2048} \quad (2)$$

We found that increasing the MLP's capacity did not improve performance, suggesting that the relevant features were already effectively captured by the LLM embeddings.

Embedding Methods and Strategies

- **Last Token**: Use the last token's hidden state.
- **Mean Pooling**: Compute the average of all token embeddings from the final hidden layer.
- **Echo Embedding**: Instead of feeding a prompt of type "Write this sentence: {text}" we feed a prompt of type "Please repeat this sentence:{text}, repeated sentence: {text}". Mean pooling is then applied only over the second occurrence of the sentence in the prompt.
- **Summary**: Prompt the model with: "Please summarize this sentence:{text}. Answer in one word:" to generate a few output tokens, get the first word, then re-feed the generated summary into the model to extract its hidden states as the final embedding.

Results

Below we present the F1 scores obtained on the test split, based on the pooling method and strategy used:

Used Model	Sentiment Classification (LaRoSeDa)			
	Last	**Mean**	**Echo**	**Summ**
LLama3.1-8b	92.18	92.89	91.84	*95.90*
RoLlama3.1-8b	**92.94**	**94.78**	**93.95**	***96.45***
mGPT1.3B	*89.26*	86.94	88.20	89.23
LLmic 3B	*78.57*	66.67	58.89	22.22

Table 1:Bold text indicates best in strategy, italic text indicates best in model.Reported results represent the average performance across embeddings extracted from three input configurations: title, content, and title+content. Notably, the best performance is consistently achieved using title alone.

Used Model	Commonsense Reasoning (RoARC)					
	CLast	**CMean**	**ELast**	**EMean**	**SLast**	**SMean**
LLama3.1-8b	*38.39*	23.74	28.11	25.02	32.13	23.31
RoLlama3.1-8b	**64.61**	**26.56**	***68.72***	**56.64**	**64.87**	24.34
mGPT1.3B	25.79	23.39	27.25	23.39	*27.59*	22.96
LLmic 3B	*26.56*	26.14	*26.56*	23.91	*26.56*	**25.96**

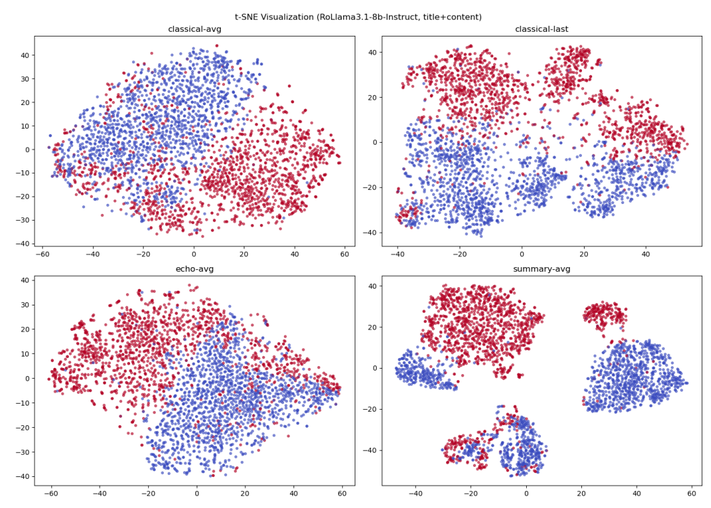
Table 2:Bold text indicates best in strategy, italic text indicates best in model

Important Results

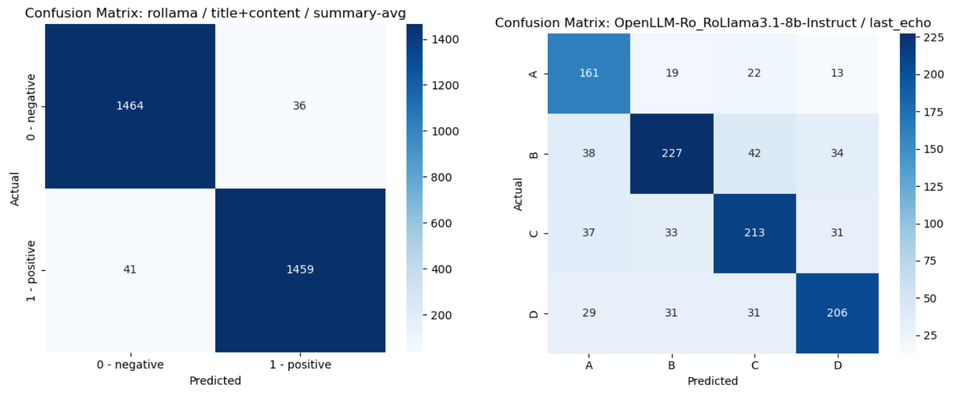
We observe that LLMic underperforms compared to mGPT-1.3B, despite having more parameters. This highlights the importance of effective compute and model optimization over simply increasing model size. Additionally, LLaMA and RoLLaMA perform similarly on LaRoSeDa (sentiment classification), but diverge significantly on RoARC (commonsense reasoning), emphasizing the value of task-specific fine-tuning in large language models.

Conclusion

Based on the results presented, we concluded that the best model across both tasks is RoLlama. Therefore, if you are faced with a task in Romanian, it might be worth it to consider a fine tuned LLM for the language.
Below we present the TSNE embeddings for the best LaRoSeDa model (based on F1 score):



And the confusion matrices for the two tasks:



LaRoSeDa RoARC

Acknowledgements

We would like to thank the Bitdefender ML team for the guidance provided.