Using Small Language Models For Local Email Categorization Into Variable Number Of User-Defined Labels

1st Jakub Matłacz
Faculty Of Electrical Engineering
Warsaw University Of Technology
Warsaw, Poland
01153072@pw.edu.pl

***Abstract*—This paper presents a novel approach, Analyze-Select-Match (ASM), for local email categorization using small language models. The objective is to enable users to organize emails on their local machines by categorizing them into user-defined labels with flexibility in both quantity and quality. A bespoke dataset of 100 email samples was curated, and five models, sized for widespread graphics card compatibility, were tested using various prompt engineering techniques. The highest achieved accuracy was 97%, with a focus on maintaining efficiency measured by the accuracy-to-average time ratio for classifying a single message. The proposed ASM methodology, emphasizing Analyze, Select, and Match stages, proved efficient without the need for retraining, making it suitable for quick transitions between tasks. The models' performance and efficiency were visualized, with the openhermes mistral 2.5 7b model and cot_1 prompt combination achieving the highest accuracy. The developed model fulfills expectations of high accuracy, efficient processing time, effective performance, agent capability, compact model size, high availability, small contextual footprint, and full customization. This research contributes a valuable methodology and insights for practical and customizable local email organization.**

***Index Terms*—Analyze-Select-Match (ASM), email categorization, small language models**

I. Introduction

The objective of the project was to develop a system enabling the private organization of emails on a user's local machine. The organizational process involved categorizing messages into user-defined categories, with flexibility in terms of quantity and quality of categories. A dedicated dataset comprising 100 email samples was curated, encompassing legitimate classes and randomly assigning potential classes for the model to categorize messages accurately. Five different models were tested, sized to accommodate most users' graphics cards in VRAM [1]. Additionally, eight different approaches were explored, involving various prompt engineering techniques and five different prompts, including proprietary formulations. The highest achieved accuracy reached 97 percent. However, performance was equally crucial, measured as the accuracy-to-average time ratio for classifying a single message. Ultimately, an efficient method suitable for local use by the majority of hardware consumers was identified. The model functions as an agent, requiring no retraining, providing a significant advantage for quick transitions between different tasks and agents. As a result, a new methodology was introduced, denoted as Analyze-Select-Match, abbreviated as ASM (Figure 1).

II. Methods

In light of the absence of datasets closely aligned with the project's specifications on the Internet, the decision was undertaken to construct a bespoke dataset. A dataset was created featuring messages, a genuine category, and a list of categories, among which only one was authentic. The number of categories within the list was variable and randomly selected from those extant in the broader dataset. The assumed email landscape for each user comprised approximately ten categories, with ten subcategories designated for each primary category. Subsequently, an illustrative message was crafted for each subcategory, culminating in the development of 100 messages characterized by unique categories. Drawing from data accessible on the Steam platform, the VRAM distribution (Figure 2) among its user base was ascertained. Subsequently, only language models of suitably modest sizes were considered, facilitating expeditious and localized deployment, thereby aligning with the core requisites of the project.

From a pool of adequately sized models, those renowned and validated within the community were discerningly chosen, encompassing five models subjectively curated by the author. Each model underwent testing with an identical set of prompt engineering techniques and prompts, with due consideration to both accuracy and temporal metrics. Default parameters of the models, meticulously calibrated by their respective authors, were adhered to, with occasional adjustments limited to context size, temperature, and response length. Tools such as Python, the difflib library [2], and LM Studio [3] were judiciously employed troughout the project. Typically, the classification process unfolded in several stages. Initially, the model scrutinized the message, generating supplementary thoughts and knowledge associated with it. Subsequently, it responded with a singular word, selecting a category from those provided by the user. Following this, the Gestalt Pattern Matching algorithm [4] was employed to choose the class that best

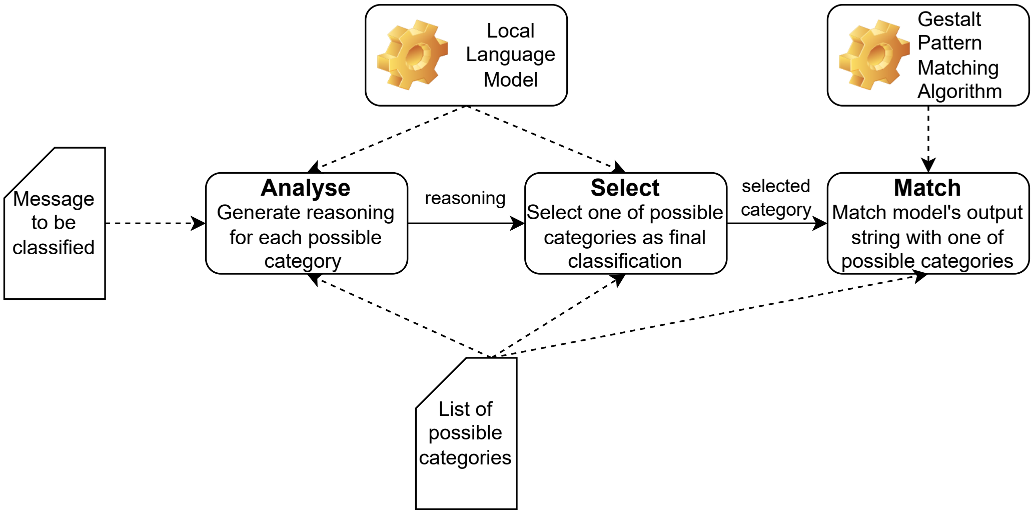
Local Language Model
Gestalt Pattern Matching Algorithm
Message to be classified
Analyse
Generate reasoning for each possible category
reasoning
Select
Select one of possible categories as final classification
selected category
Match
Match model's output string with one of possible categories
List of possible categories

Fig. 1. Graph of ASM method's data processing.

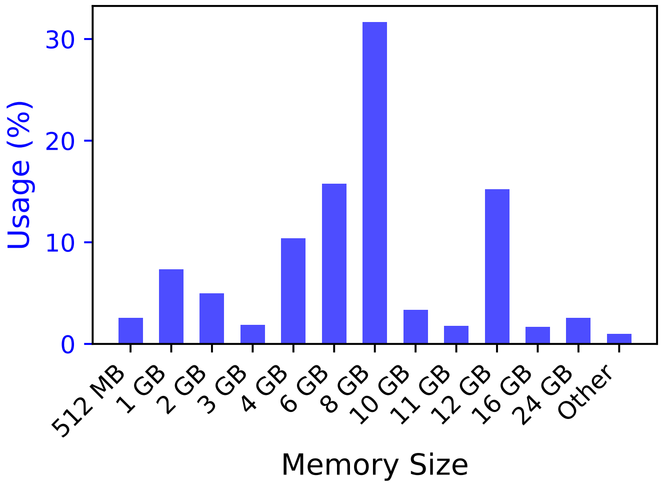

Fig. 2. Distribution of VRAM amount on Steam users' computers [1].

Model Name	Quantization	Context Length	Parameters
ehartford_dolphin-2.0 -mistral-7b [5]	Q6_K	32768	7B
teknium_openhermes -2.5-mistral-7b [6]	Q6_K	32768	7B
Phi2 [7]	Q6_K	2048	3B
mistralai_mistral-7b -instruct-v0.2 [8]	Q6_K	32768	7B
open-orca_mistral-7b -openorca [9]	Q6_K	32768	7B

TABLE I
TECHNICAL DETAILS OF USED LANGUAGE MODELS.

matched one of the possibilities. This approach significantly augmented results, given that models often respond concisely or slightly modify original class names. This methodology was denoted as Analyze-Select-Match, abbreviated as ASM (Picture 1).

Inference Parameters	Value
Temperature (temp)	0.8
Tokens to Generate (n_predict)	-1
Top K Sampling (top_k)	40
Repeat Penalty (repeat_penalty)	1.1
Min P Sampling (min_p)	0.05
Top P Sampling (top_p)	0.95

TABLE II
DEFAULT INFERENCE PARAMETERS THAT WERE USED IN EXPERIMENTS WITH ALL LANGUAGE MODELS.

III. RESULTS

The investigation into local email categorization involved the testing of five models, each meticulously chosen for compatibility with most users' graphics cards in VRAM (Figure 2). Full results are visible on Figures no 3, 4, 5. The smallest model, Phi 2 3b [10], was found to be highly inefficient for practical applications, emphasizing the importance of model size for real-world usability. The evaluation of prompt engineering techniques revealed that Input-Output (IO) techniques [11], while efficient, exhibited lower accuracy. The Chain-of-Thought (COT) [12] approach with prompt no 2 demonstrated lower efficiency compared to others, highlighting the nuanced trade-offs in model performance. Notably, the openhermes mistral 2.5 7b model, paired with the cot_1 prompt, achieved the highest accuracy at 97%, showcasing the significance of model-prompt combinations. Conversely, the phi 2 3b model, employing the 3-shot cot_3 [13] approach (inspired by CARP technique [14]), yielded the lowest accuracy at 12%. Among various prompt engineering techniques, cot_2, io_1, and io_1 w/ cs-5 [15] exhibited the most variability when changing models. In the context of local email categorization, the presented data provides valuable insights into the effectiveness of different techniques and models. Techniques like ”COT_1” and ”3-SHOT COT_1” consistently yielded high correctness values across various models, showcasing their robust performance, particularly in conjunction with

”OPENHERMES MISTRAL 2.5 7B.” Conversely, the ”PHI 2 3B” model demonstrated comparatively lower correctness across techniques, emphasizing the influence of the chosen model. The efficiency analysis highlighted the superiority of the ”IO_1” technique, making it a compelling choice across models, especially when paired with ”MISTRAL INSTRUCT 0.2 7B”. A nuanced approach is recommended, considering the balance between correctness and elapsed time. Optimal choices include ”COT_1” or ”3-SHOT COT_1” techniques with ”OPENHERMES MISTRAL 2.5 7B” for balanced performance, while the ”IO_” technique with various models, particularly ” MISTRAL INSTRUCT 0.2 7B” proves advantageous for efficiency-focused applications. Tailoring choices to specific use case requirements is crucial for informed decision-making in implementing these categorization techniques and models. Mistral model and its derivatives presented strong results [16].

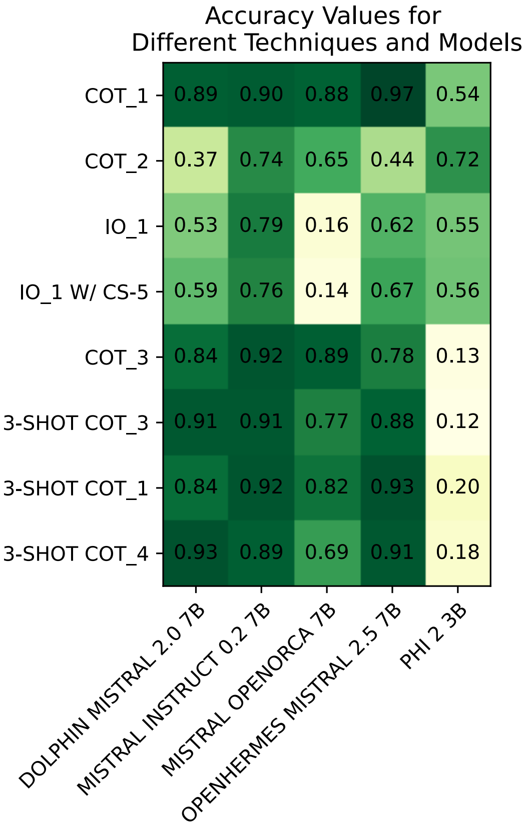

Fig. 3. Experimental accuracies in different approaches.

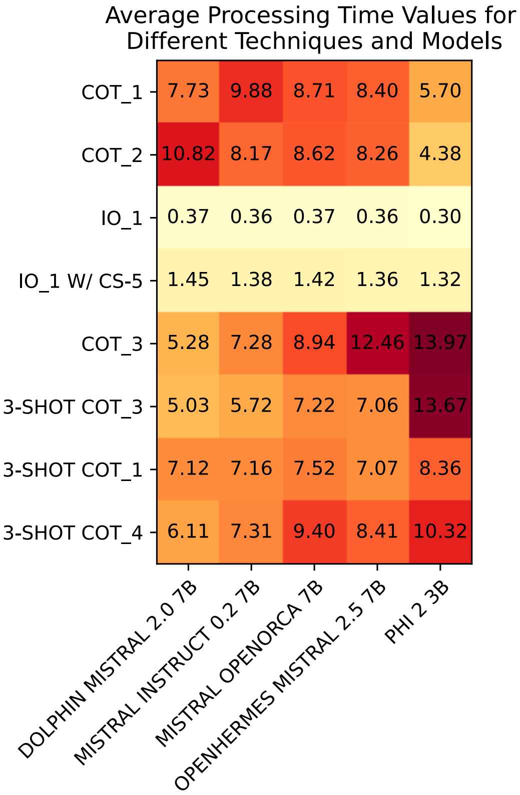

Fig. 4. Experimental average processing times of single message in different approaches.

IV. Conclutions

The successful development of the model is characterized by its alignment with a predefined set of expectations in the domain of local email categorization. The model consistently exhibits commendable accuracy, showcasing its proficiency in executing classification tasks and effectively discerning diverse email categories. In addition to its high accuracy, the model boasts efficient processing times, ensuring prompt and responsive performance and thereby substantiating its operational efficiency. Noteworthy is the model’s overall effectiveness, demonstrating robust performance that effectively meets predefined criteria and showcases proficiency in the intricate task of email organization. An outstanding feature of this model is its autonomy from the need for continuous retraining, highlighting its stability and reliability over extended usage periods.

Beyond its classification prowess, the model functions adeptly as an agent, facilitating seamless transitions between various tasks and agents, thereby enhancing overall user experience and workflow efficiency. The model’s compact size contributes to its practical utility, allowing for convenient deployment and usage. Ensuring high availability, the model promotes consistent and reliable access, contributing to a seamless and uninterrupted user experience. Moreover, the model maintains a small contextual footprint, optimizing resource utilization and reflecting an efficient use of computational resources. A noteworthy attribute is the system’s provision of extensive customization options for email organization. This feature empowers users with flexibility in tailoring their experience, enhancing the adaptability of the model to diverse user preferences and requirements. In summary, the developed model stands as a testament to its multifaceted capabilities, embodying accuracy, efficiency, reliability, and user-centric

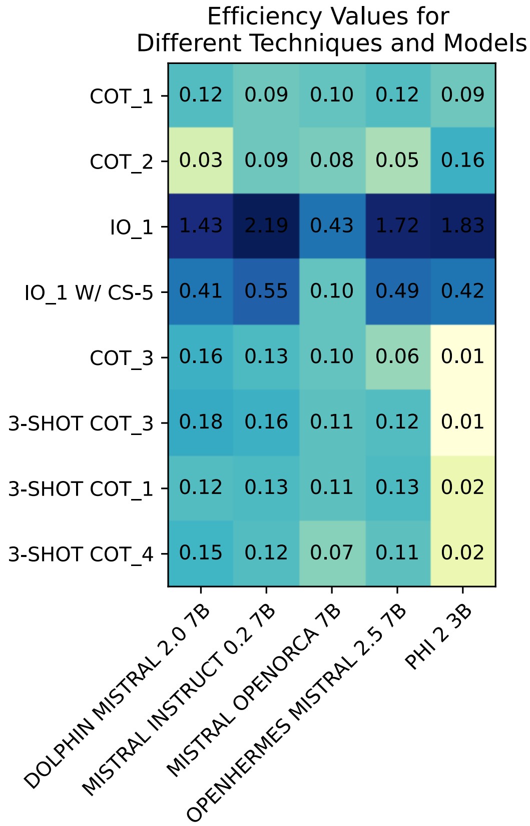

Fig. 5. Experimental efficiencies (accuracy/time) in different approaches.

customization within the context of local email categorization.

V. Future work

In the pursuit of advancing the current work, several promising avenues for future research and development emerge. Firstly, the exploration of more intricate and nuanced prompt engineering techniques holds the potential to refine the model's performance further. Investigating novel formulations and combinations of prompts may contribute to enhanced accuracy and efficiency, particularly in scenarios where customization requirements vary. Additionally, the scalability of the model across larger datasets could be a focal point for future endeavors, ensuring its applicability and robustness in real-world, diverse email landscapes. The integration of advanced natural language processing techniques, such as transfer learning and domain adaptation, could also fortify the model's adaptability to evolving language patterns and user preferences. Furthermore, an in-depth analysis of user interactions with the system and iterative feedback loops may provide valuable insights for continuous improvement, facilitating a user-centered approach to model refinement. Collaboration with cybersecurity experts to fortify the model against potential adversarial attacks and the exploration of energy-efficient deployment strategies are also promising areas for further exploration. Overall, the envisioned future directions encompass a comprehensive refinement of the model's capabilities, addressing both technical intricacies and user-centric considerations to propel the efficacy of local email categorization systems.

References

[1] Valve Corporation. (2024) Steam Hardware & Software Survey. Accessed on January 11, 2024. [Online]. Available: https://store.steampowered.com/hwsurvey/Steam-Hardware-Software-Survey-Welcome-to-Steam

[2] Python Software Foundation, *difflib – Helpers for computing deltas*, 2023, https://docs.python.org/3/library/difflib.html.

[3] LM Studio, "LM Studio GitHub Repository," https://github.com/lmstudio-ai, 2023, accessed on January 11, 2024.

[4] Wikipedia contributors. (2023) Gestalt pattern matching. [Online]. Available: https://en.wikipedia.org/wiki/Gestalt_pattern_matching

[5] TheBloke. (2023) TheBloke/dolphin-2.0-mistral-7B-GGUF. Accessed on January 11, 2024. [Online]. Available: https://huggingface.co/TheBloke/dolphin-2.0-mistral-7B-GGUF

[6] ——. (2023) TheBloke/OpenHermes-2.5-Mistral-7B-GGUF. Accessed on January 11, 2024. [Online]. Available: https://huggingface.co/TheBloke/OpenHermes-2.5-Mistral-7B-GGUF

[7] ——. (2023) TheBloke/phi-2-GGUF. Accessed on January 11, 2024. [Online]. Available: https://huggingface.co/TheBloke/phi-2-GGUF

[8] ——. (2023) TheBloke/Mistral-7B-Instruct-v0.2-GGUF. Accessed on January 11, 2024. [Online]. Available: https://huggingface.co/TheBloke/Mistral-7B-Instruct-v0.2-GGUF

[9] ——. (2023) TheBloke/Mistral-7B-OpenOrca-GGUF. Accessed on January 11, 2024. [Online]. Available: https://huggingface.co/TheBloke/Mistral-7B-OpenOrca-GGUF

[10] S. Gunasekar, Y. Zhang, J. Aneja, C. C. T. Mendes, A. D. Giorno, S. Gopi, M. Javaheripi, P. Kauffmann, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, H. S. Behl, X. Wang, S. Bubeck, R. Eldan, A. T. Kalai, Y. T. Lee, and Y. Li, "Textbooks are all you need," 2023.

[11] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Finetuned language models are zero-shot learners," 2022.

[12] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023.

[13] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[14] X. Sun, X. Li, J. Li, F. Wu, S. Guo, T. Zhang, and G. Wang, "Text classification via large language models," 2023.

[15] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," 2023.

[16] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7b," 2023.

VI. Prompts and other details

Prompts and other details can be found here: https://github.com/matlaczj/ASM-Method-for-Local-Email-Categorization.

Prompts can be found here: https://github.com/matlaczj/ASM-Method-for-Local-Email-Categorization/blob/main/docs/prompts.pdf.