DNB-AI-Project at SemEval-2025 Task 5: An LLM-Ensemble Approach for Automated Subject Indexing

Lisa Kluge
Deutsche Nationalbibliothek
Frankfurt am Main, Germany
l.kluge@dnb.de

Maximilian Kähler
Deutsche Nationalbibliothek
Leipzig, Germany
m.kaehler@dnb.de

Abstract

This paper presents our system developed for the SemEval-2025 Task 5: LLMs4Subjects: LLM-based Automated Subject Tagging for a National Technical Library's Open-Access Catalog. Our system relies on prompting a selection of LLMs with varying examples of intellectually annotated records and asking the LLMs to similarly suggest keywords for new records. This few-shot prompting technique is combined with a series of post-processing steps that map the generated keywords to the target vocabulary, aggregate the resulting subject terms to an ensemble vote and, finally, rank them as to their relevance to the record. Our system is fourth in the quantitative ranking in the all-subjects track, but achieves the best result in the qualitative ranking conducted by subject indexing experts.

1 Introduction

The LLMs4Subject task (D'Souza et al., 2025) aims at utilising large language models (LLMs) for the task of automated subject indexing on a dataset of open-access publications. Automated subject indexing is a task that helps enabling access to user-relevant publications by identifying and recording their most important themes and topics in the tagged subject terms. The ever-growing number especially of digital publications requires reliable automated systems for this task, which has become infeasible to achieve manually. In our previous work on automated subject indexing on a similar dataset (Kluge and Kähler, 2024), we found that the performance of LLMs, while succesfully applied to a range of other tasks (Zhao et al., 2023; Yang et al., 2024; Patil and Gudivada, 2024), was not yet on par with classical supervised machine learning methods. Therefore, it is important to do further research on the capabilities of LLMs in this context.

Rather than fine-tuning models ourselves, the main strategy of our system is to leverage the existing capabilities of off-the-shelf foundational or instruction-tuned open-weight LLMs. In contrast to our previous work, the key contribution of this system is that it does not rely on only one LLM, but a combination of different language models along with varying prompts to generate the subject terms. We found this ensemble approach to dramatically improve the performance of our system. To handle the challenge of the controlled vocabulary unknown to the LLMs, we first generate free keywords with generative LLMs and then map these onto the vocabulary with a smaller embedding model.

The official quantitative results put us in fourth place, the qualitative results even in first place. We think that our approach provides valuable insights into the chances and bounds of the few-shot prompting approach, showing that competitive results are possible without fine-tuning and large training corpora, simply by combining several LLMs into an ensemble.

Our code is publicly available.[1]

2 Background

Outlining the field of automated subject indexing, Golub (2021) presented important fundamentals, approaches and best practices for the task. Referring to it as index term assignment, Erbs et al. (2013) compared and combined two strategies to perform this task: multi-label classification (MLC) and keyword extraction. Detecting separate strengths, their results aligned with Toepfer and Seifert (2020), who also found the combination of approaches to be beneficial.

Regarding frameworks for automated subject indexing, the Annif system (Suominen, 2019) is an important contribution. Annif has established

[1] https://github.com/deutsche-nationalbibliothek/semeval25_llmensemble

Proceedings of the The 19th International Workshop on Semantic Evaluation (SemEval-2025), pages 1118–1128
July 31 - August 1, 2025 ©2025 Association for Computational Linguistics

methods built in, like Omikuji[2], which is based on partitioned-label-tree-method Bonsai (Khandagale et al., 2020), or MLLM[3], a lexical approach building on Medelyan (2009)'s Maui.

In earlier work (Kluge and Kähler, 2024), we presented experiments with a closed-source LLM on automated subject indexing, but the two baseline methods implemented in Annif mentioned above were found to be as good as or even outperform our LLM-based method.

LLMs have also been utilised for MLC (Peskine et al., 2023; D'Oosterlinck et al., 2024; Zhu and Zamani, 2024) and keyword extraction (Maragheh et al., 2023; Lee et al., 2023).

Recently, building ensembles or fusioning (the results of) LLMs has been addressed as a promising research direction. There are different works sharing the idea of exploiting the individual strengths and diminishing the weaknesses in different LLMs (Jiang et al., 2023b; Lu et al., 2023; Wang et al., 2023; Fang et al., 2024; Wan et al., 2024). Exploring the goal of building ensembles, Tekin et al. (2024) aimed at maximising diversity and efficiency, whereas Chen et al. (2023) targeted the reduction of inference cost. Not only LLMs have been combined, but also prompts (Pitis et al., 2023; Hou et al., 2023). Combining both prompts and models on the task of phishing detection, Trad and Chehab (2024) contrasted prompt-based ensembles (with one prompt and several LLMs), model-based ensembles and an ensemble consisting of a mixture of prompts and models.

3 System Overview

Our system is an enhancement from our previous LLM-based subject indexing approach, described in Kluge and Kähler (2024). In total, it consists of 5 stages, *complete*, *map*, *summarise*, *rank* and *combine*, as depicted in the overview in Figure 1. At its core, the system approaches the subject indexing task as a keyword generation problem which is solved by a few-shot prompting LLM procedure. As these generated keywords are a priori not restricted to the target vocabulary, a mapping stage with a smaller word embedding model is needed as a supplementary step. In comparison to our previous approach, we have extended the system by combining multiple LLMs and prompts to an ensemble and by introducing an LLM-powered ranking step as in D'Oosterlinck et al. (2024).

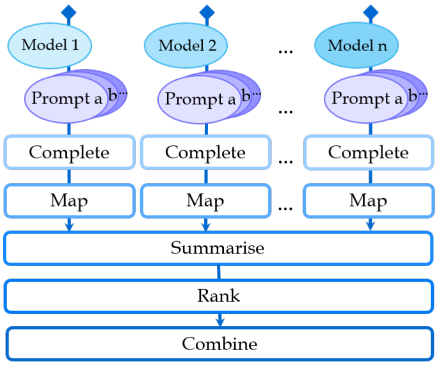

Figure 1: Illustration of our LLM-ensemble approach.

3.1 Complete

The first step in our subject indexing system, *complete*, is the generation of keywords following the few-shot paradigm, similar to the procedure in Lee et al. (2023). The *complete*-step is repeated over a range of diverse off-the-shelf open-weight LLMs and prompts with varying few-shot examples. Our plan and intention of employing a broad variety of models and prompts are twofold. In comparison to a single-model single-prompt setting, we aim to:

- Improve recall with an overall greater set of generated subject terms in the ensemble.
- Improve precision by utilising the overlap of various model×prompt combinations.

Each prompt consists of an instruction and a set of 8-12 examples illustrating how to perform the subject indexing task with example texts and their gold-standard subject terms.

Details for LLM selection and the composition of the few-shot prompts will be discussed in Sections 4.3 and 4.4.

3.2 Map

Keywords generated in the first stage are *mapped* to controlled subject terms in the target vocabulary using a word embedding model as in Zhu and Zamani (2024).

For our *map*-stage, we used Chen et al. (2024)'s BGE-M3-embeddings. Both generated keywords and target vocabulary are embedded with the same model.[4] To perform nearest neighbour search, we uploaded the embeddings to a Weaviate[5] vector

[2] https://github.com/tomtung/omikuji

[3] https://github.com/NatLibFi/Annif/wiki/Backend:-MLLM

[4] We embedded the keywords and vocabulary entries without integrating them into a context sentence (which we did in our previous approach).

[5] https://weaviate.io/

storage enabling efficient HNSW-Search (Malkov and Yashunin, 2016) in $\mathcal{O}(\log L)$ complexity, where L is the vocabulary size. One feature of the vector storage is that it may be used in a hybrid search mode (Cardenas, 2025), combining vector search and traditional BM25 search (Robertson and Zaragoza, 2009). Thus, each suggested keyword is mapped to the most similar subject term and the similarity score is stored for later use in the *summarise*-step. Matches with a low similarity score can be discarded at this stage with a tunable threshold, eliminating keywords not represented in the vocabulary.

3.3 Summarise

Each prompt and model outputs its own set of predicted subject terms per document after *complete* and *map*. In the subsequent *summarise*-step, the subject terms are aggregated over all model×prompt combinations by summing the similarities obtained in the *map*-stage (3.2). This score is normalised to a value between 0 and 1 by dividing it by the overall number of model×prompt combinations. Hence, we obtain an ensemble score s_{ens} for each suggested subject term. This ensemble score acts as a confidence measure of the individual suggestions and will be included in the final ranking score in the later *combine*-stage (3.5).

3.4 Rank

In the *rank*-stage, that D'Oosterlinck et al. (2024) also incorporated in their approach on MLC, another LLM is employed to rank the subject terms by their relevance. For each predicted subject term, we ask the model to assess its relevance to the test record at hand on a scale from 0 (not relevant) to 10 (extremely relevant). Normalised to a value between 0 and 1, we obtain a relevance score s_{rel} for each suggested subject term. Including this additional *rank*-step has two reasons: Firstly, the relevance score may improve the ensemble score that is, by now, purely based on frequency and mapping similarity. Asking an LLM to rate the suggestions also takes into account the context of the text and can thus determine the *relevance* of the suggestions. Secondly, this step can be an additional control step for the *map*-stage.

3.5 Combine

In the *combine*-stage, a final ranking score for each suggested subject term is obtained as a weighted average from the ensemble and relevance scores.

$$s_{\text{fin}} = \alpha \times s_{\text{ens}} + (1 - \alpha) \times s_{\text{rel}} \qquad (1)$$

In our experiments, we learned setting $\alpha = 0.3$ in equation 1 resulted in the best ranking (refer to Appendix A.4 for more details). In other words, the ordering of the subject terms was best when relying more on the *ranking* than on the *summarisation*.

4 Experimental Setup

4.1 Data Handling

We used two randomly sampled disjoint subsets ($n = 1000$) taken from the union of the development sets given in the all-subjects and the tib-core collection for optimisation and results analysis. On the first subset, *dev-opt*, we tuned parameters like the model×prompt selection (see Section 4.5) and *combine*-parameter α. The second one, *dev-test*, comprises the data on which we conducted our own evaluation. In both subsets, we included both English and German texts, as well as all five text types (Article, Book, Conference, Report, Thesis), while keeping the proportions of the overall development set through stratified sampling.

For both input texts and prompts, we used the concatenation of title and abstract as text representation.

4.2 Vocabulary Adaptation

When inspecting early results of our system, we found that the provided vocabulary, GND-Subjects-all, was insufficient to represent the free keywords resulting from the *complete*-stage. One particular issue was the absence of named entities, that do appear in the full GND but not in this collection. Plausible keyword candidates, such as country names, are missing and therefore falsely mapped to unrelated subject terms. Choosing a threshold for minimum similarity between keyword and subject terms was not enough to prevent this kind of error. Thus, we extended the vocabulary to also include named entities. As the full GND would comprise over 1.3 million concepts, we chose to only include named entities that are actually used in the catalogue of the DNB. In total, our extended vocabulary includes 309,417 distinct concepts (including 200,035 subject terms from the all subjects collection as well as 109,382 named entities from the DNB-catalogue). We found that our system produces fewer false positives if we map the named entities cleanly to the extended GND vocabulary

HF user	Model Name
meta-llama	Llama-3.2-3B-Instruct
	Llama-3.1-70B-Instruct
mistralai	Mistral-7B-v0.1
	Mistral-7B-Instruct-v0.3
	Mixtral-8x7B-Instruct-v0.1
teknium	OpenHermes-2.5-Mistral-7B
openGPT-X	Teuken-7B-instruct-research-v0.4

Table 1: LLMs used for the completion on the test set.

and exclude subject terms not belonging to the targeted GND-Subjects-all collection afterwards. Note that this is also why we only work with the broader GND-Subjects-all vocabulary and not with the tib-core subset.

4.3 Language Models

We experimented with a range of different models for the *complete*-step. We used Llama 3 in 3B-Instruct and 70B-Instruct variants (Grattafiori et al., 2024), a few versions of Mistral 7B (Jiang et al., 2023a), Mixtral of Experts (Jiang et al., 2024) and Teuken-7B-Instruct (Ali et al., 2024). The overview of models in our final selection is presented in Table 1. We used Llama-3.1-8B-Instruct (Grattafiori et al., 2024) as the ranking model.

For the *complete*-stage, the number of keywords generated by the LLMs was controlled by setting the minimum number of tokens to 24 and the maximum tokens to 100. Find the rest of the hyperparameters affecting the LLMs on our Github[6].

4.4 Prompts

We sampled different sets of prompt examples from the train splits of the all-subjects and tib-core datasets. To account for the multilinguality of the data, we assembled prompts with only German, only English and mixed-language texts. However, the gold-standard subject terms that we show to the LLMs are always in German. Additionally, we also created prompts with a restricted number of subject terms and lemma overlap. Lemma overlap is a measure for similarity between the example text and its subject terms, which we also used in Kluge and Kähler (2024).

Note that we leave the handling of multilinguality completely to the LLMs. Analysing the keywords resulting from the *complete*-stage on dev-test, it wasn't the case that the models tended to generate English terms, but instead they followed the few-shot demonstrations and output German keywords.

You can view an overview of the prompt example sampling in Appendix A.1. You can also see the instructions for the *complete*- and *rank*-stages there. The list of examples for each prompt is available on our system's Github[7]. The templates we used to build the final prompt are also on our system's Github[8].

4.5 Ensemble Optimisation

On the dev-opt subset we ran experiments with 9 models $\times$ 15 prompts, resulting in 135 sets of subject term suggestions. However, one cannot expect ever increasing the number of models and prompts to unlimitedly lead to better performance. Naturally, there is a tipping-point where ensemble performance deteriorates when adding more models or prompts. Also, there is a trade-off between the number of models and prompts and the computing effort at inference time involed in the *complete*-step. Therefore, we conducted an additional optimisation step to find the best subset of models and prompts.

Our optimisation strategy was twofold: In a first Monte-Carlo-like approach, we repeatedly sampled model$\times$prompt combinations and tested their joint performance as an ensemble, yielding a subset of 50 out of 135 combinations that achieve the best precision-recall (PR) balance in terms of area under the precision-recall curve (PR-AUC) on the dev-opt set. In a second step, we used a chain strategy, where we iteratively removed model$\times$prompt combinations that did not contribute to the overall performance, narrowing down the selection to 20 combinations. See Appendix A.2 for further results comparing our ensemble strategy with other strategies as in Trad and Chehab (2024). Also, see the impact of α on the results on the dev-test set in Appendix A.4.

4.6 Implementation Details

We used vLLM (Kwon et al., 2023) to serve the LLMs in the *complete*- and *rank*-stages. Embeddings for the keywords and the vocabulary were generated using HuggingFace's Text Embeddings

[6] https://github.com/deutsche-nationalbibliothek/semeval25_llmensemble/blob/main/params.yaml

[7] https://github.com/deutsche-nationalbibliothek/semeval25_llmensemble/tree/main/assets/prompts

[8] https://github.com/deutsche-nationalbibliothek/semeval25_llmensemble/tree/main/assets/templates

Team	**P_5**	**R_5**	**$F1_5$**	**R_{50}**	**R_{avg}**
Annif	**0.263**	**0.494**	**0.343**	**0.681**	**0.630**
DUTIR831	0.256	0.484	0.335	0.640	0.605
RUC	0.230	0.438	0.302	0.642	0.586
icip	0.198	0.387	0.262	0.596	0.530
Ours	0.246	0.471	0.323	0.579	0.563

Table 2: Official quantitative results for top five teams on the all-subjects task.

Team	**P_5**	**R_5**	**$F1_5$**	**R_{20}**	**R_{avg}**
DUTIR831	0.488	0.316	0.384	0.611	0.485
RUC Team	0.481	0.287	0.359	**0.618**	0.465
Annif	0.457	0.301	0.363	0.577	0.448
jim	0.404	0.287	0.335	0.545	0.426
Ours	**0.526**	**0.339**	**0.412**	0.615	**0.509**

Table 3: Official qualitative ranking of the top five teams (`case 2`).

Inference[9]. As previously stated, we used Weaviate[10] as vector storage. To create our pipeline and to manage our experiments, we used DVC (Kuprieiev et al., 2025).

5 Results

5.1 Quantitative Findings

Table 2 shows the quantitative results on the all-subjects data of our system and the highest-ranking other teams sorted by averaged recall (R_{avg}). In this metric, we are in fourth position. Note that our approach, in contrast to supervised MLC algorithms, does not estimate a probability for each subject term in the entire vocab, but rather positively suggests a set of subject terms for each document. Modifying the hyperparameters affecting the number of output tokens can slightly increase the number of different keywords, but our approach doesn't produce result lists of arbitrary length. For recall@k values with high k, the average length of our submitted label lists of 18 makes these scores less adequate to properly estimate our system's performance. Therefore, we also included the scores precision@5 (P_5), recall@5 (R_5) and F1@5 ($F1_5$) in the table, as we find these metrics to be more insightful to our system's performance. Figure 4 in the Appendix demonstrates how our system drops off early in recall, while showing competitive results for lower values of k.

Looking at the more detailed results for our system (depicted in Appendix 7), we learned that, language-wise, one can observe better performance on the German than on the English documents (F1@5=0.332/F1@5=0.307). This could be attributed to the facts that we use a German instruction and that the vocabulary is presented in German. Potentially, using an English instruction and translating the vocabulary to English - both for the few-shot examples and the mapping stage - would help decrease this gap. Record-type-wise, Articles are by far the worst category for our system with F1@5=0.157. One reason for this could be the absence of articles in most of our prompts. All other text types achieve an F1@5 of at least 0.318. Interestingly, Articles are the best record type for the other leading teams, F1@5-wise.

5.2 Qualitative Findings

Table 3 shows the overall results for the top five teams in the qualitative ranking. Here, we see our system in the top position. In particular, the evaluation scenario (case 2) that eliminates those keywords that are technically correct but irrelevant puts a margin of 2.8% between our system and the second best team w.r.t. F1@5. It is unsurprising that the qualitative results are better than the quantitative ones, as our approach does not involve fine-tuning to the gold-standard. Subject terms may be helpful and specific in describing the text content, but at the same time not follow the formal rules applied by TIB's subject specialists when annotating the gold-standard.

Table 8 in the Appendix shows the F1@5 scores for different subject categories. Our system was rated particularly high in architecture, computer science and economics. Worst performance was in history, traffic engineering and mathematics.

5.3 Error Analysis

To get an understanding of the struggles our system faces, we put a small subset of the dev-test set under manual inspection and compared our system's suggested subject terms to the gold-standard. We also analysed the content of title and abstract for these documents. The questions we had in mind while making this analysis were:

- Are there groups of gold-standard subject terms we completely miss?
- Are there gold-standard subject terms that are difficult to infer from the given text content?

Upon this manual inspection, we noticed that our system benefits from two factors: specificity of a term and its presence in the concatenated content

[9] `https://huggingface.co/docs/text-embeddings-inference/index`

[10] `https://weaviate.io/`

of title and abstract. Specific subject terms that are either directly present in the text or are paraphrased in it seem to have the best chance of being correctly predicted. Generic subject terms are often not found or falsely assigned (e.g. gold-standard: law, found: international law, European law; gold-standard: agricultural policy, found: agriculture). Still, in the list of the most frequent subjects assigned to the dev-test set, there are a lot of general terms, such as history, politics and culture. Especially when analysing results for the Article text type, which our system performs worst on, we noticed many gold-standard subject terms we suspect to be difficult to directly infer from the given text alone. For example, see the text and its assigned keywords in Appendix A.8. In this record, a lot of words related to the keywords are mentioned in the text (e.g. economic development/growth, agriculture). The exact concepts are not in the text and are also not predicted by our LLM-ensemble. Our system relying only on the prompt examples and the concatenation of title and abstract struggles with these types of more complex/abstract relationships between text and subject terms. This is where supervised learning approaches might have an advantage, as they can learn these relationships from the training data.

Refer to Appendix A.8 for more details regarding this error analysis.

6 Conclusion

To sum up, we have demonstrated that our ensemble appoach is a promising way to combine the strengths of different models and prompts, achieving competitive results in the LLMs4Subjects task. As we have covered a wide range of prompts and LLMs, we expect our system to provide a good estimate of the results possible by prompting LLMs even without fine-tuning. While our system comes with no extra training costs, a significant drawback is the high cost involved in prompting multiple LLMs at inference time. Appendix A.9 demonstrates the costs of processing the documents with each of the LLMs used in our ensemble. Particularly larger models use up an enourmous amount of GPU-ressources that may be infeasible in productive settings. In future work, we would like to further investigate techniques for automated prompt optimisation, such as DSPy (Khattab et al., 2023), or methods belonging to the family of Parameter-Efficient-Fine-tuning (PEFT). Also we would like to investigate more sophisticated methods for the ensemble combination.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback.

This work is a result of a research project at the German National Library (DNB)[11]. The project is funded by the German Minister of State for Culture and the Media as part of the national AI strategy. With the AI strategy, the German federal government is supporting the research, development and application of innovative technologies.

References

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Jan Ebert, Alexander Arno Weber, Richard Rutmann, Charvi Jain, Max Lübbering, Daniel Steinigen, Johannes Leveling, Katrin Klug, Jasper Schulze Buschhoff, Lena Jurkschat, Hammam Abdelwahab, Benny Jörg Stein, Karl-Heinz Sylla, Pavel Denisov, Nicolo' Brandizzi, Qasid Saleem, Anirban Bhowmick, Lennard Helmer, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Alex Jude, Lalith Manjunath, Samuel Weinbach, Carolin Penke, Oleg Filatov, Shima Asaadi, Fabio Barth, Rafet Sifa, Fabian Küch, Andreas Herten, René Jäkel, Georg Rehm, Stefan Kesselheim, Joachim Köhler, and Nicolas Flores-Herr. 2024. Teuken-7b-base & teuken-7b-instruct: Towards european llms.

Erika Cardenas. Hybrid search explained [online]. 2025.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv e-prints*, page arXiv:2402.03216.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv e-prints*, page arXiv:2305.05176.

Karel D'Oosterlinck, Omar Khattab, François Remy, Thomas Demeester, Chris Develder, and Christopher Potts. 2024. In-context learning for extreme multi-label classification. *arXiv preprint arXiv:2401.12178*.

Jennifer D'Souza, Sameer Sadruddin, Holger Israel, Mathias Begoin, and Diana Slawig. 2025. Semeval-2025 task 5: Llms4subjects - llm-based automated subject tagging for a national technical library's open-access catalog. In *Proceedings of the 19th International Workshop on Semantic Evaluation (SemEval-2025)*, pages 1082–1095, Vienna, Austria. Association for Computational Linguistics.

[11] https://www.dnb.de/ki-projekt

Nicolai Erbs, Iryna Gurevych, and Marc Rittberger. 2013. Bringing order to digital libraries: From keyphrase extraction to index term assignment. *D-Lib Magazine*, 19(9/10):1–16.

Chenhao Fang, Xiaohan Li, Zezhong Fan, Jianpeng Xu, Kaushiki Nag, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2024. Llm-ensemble: Optimal large language model ensemble method for e-commerce product attribute value extraction. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2910–2914, New York, NY, USA. Association for Computing Machinery.

Koraljka Golub. 2021. Automated subject indexing: An overview. *Cataloging & Classification Quarterly*, 59(8):702–719.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models.

Bairu Hou, Joe O'Connor, Jacob Andreas, Shiyu Chang, and Yang Zhang. 2023. Promptboosting: Black-box text classification with ten forward passes. In *International Conference on Machine Learning*, pages 13309–13324. PMLR.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023b. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Sujay Khandagale, Han Xiao, and Rohit Babbar. 2020. Bonsai: diverse and shallow trees for extreme multi-label classification. *Machine Learning*, 109(11):2099–2119.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2023. Dspy: Compiling declarative language model calls into self-improving pipelines. *arXiv e-prints*, page arXiv:2310.03714.

Lisa Kluge and Maximilian Kähler. 2024. Few-shot prompting for subject indexing of German medical book titles. In *Proceedings of the 20th Conference on Natural Language Processing (KONVENS 2024)*, pages 141–148, Vienna, Austria. Association for Computational Linguistics.

Ruslan Kuprieiev, skshetry, Peter Rowlands, Dmitry Petrov, Paweł Redzyński, Casper da Costa-Luis, David de la Iglesia Castro, Alexander Schepanovski, Ivan Shcheklein, Gao, Batuhan Taskaya, Jorge Orpinel, Fábio Santos, Daniele, Ronan Lamy, Aman Sharma, Zhanibek Kaimuldenov, Dani Hodovic, Nikita Kodenko, Andrew Grigorev, Earl, Nabanita Dash, George Vyshnya, Dave Berenbaum, maykulkarni, Max Hora, Vera, and Sanidhya Mangal. 2025. Dvc: Data version control - git for data & models.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Wanhae Lee, Minki Chun, Hyeonhak Jeong, and Hyunggu Jung. 2023. Toward keyword generation through large language models. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces*, IUI '23 Companion, page 37–40, New York, NY, USA. Association for Computing Machinery.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. Routing to the expert: Efficient reward-guided ensemble of large language models.

Yu A. Malkov and D. A. Yashunin. 2016. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:824–836.

Reza Yousefi Maragheh, Chenhao Fang, Charan Chand Irugu, Parth Parikh, Jason Cho, Jianpeng Xu, Saranyan Sukumar, Malay Patel, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2023. Llm-take: Theme-aware keyword extraction using large language models. In *2023 IEEE International Conference on Big Data (BigData)*, pages 4318–4324.

Olena Medelyan. 2009. *Human-competitive automatic topic indexing*. Ph.D. thesis, The University of Waikato, New Zealand.

Rajvardhan Patil and Venkat Gudivada. 2024. A review of current trends, techniques, and challenges in large language models (llms). *Applied Sciences*, 14(5).

Youri Peskine, Damir Korenčić, Ivan Grubisic, Paolo Papotti, Raphael Troncy, and Paolo Rosso. 2023. Definitions matter: Guiding gpt for multi-label classification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4054–4063, Singapore. Association for Computational Linguistics.

Silviu Pitis, Michael R. Zhang, Andrew Wang, and Jimmy Ba. 2023. Boosted prompt ensembles for large language models. *arXiv e-prints*, page arXiv:2304.05970.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Osma Suominen. 2019. Annif: Diy automated subject indexing using multiple algorithms. *LIBER Quarterly: The Journal of the Association of European Research Libraries*, 29(1):1–25.

Selim Furkan Tekin, Fatih Ilhan, Tiansheng Huang, Sihao Hu, and Ling Liu. 2024. Llm-topla: Efficient llm ensemble by maximising diversity.

Martin Toepfer and Christin Seifert. 2020. Fusion architectures for automatic subject indexing under concept drift: Analysis and empirical results on short texts. *International Journal on Digital Libraries*, 21(2):169–189.

Fouad Trad and Ali Chehab. 2024. To ensemble or not: Assessing majority voting strategies for phishing detection with large language models. *arXiv e-prints*, page arXiv:2412.00166.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. Knowledge fusion of large language models. *arXiv e-prints*, page arXiv:2401.10491.

Hongyi Wang, Felipe Maia Polo, Yuekai Sun, Souvik Kundu, Eric Xing, and Mikhail Yurochkin. 2023. Fusing models with complementary expertise. *arXiv e-prints*, page arXiv:2310.01542.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2024. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Trans. Knowl. Discov. Data*, 18(6).

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *arXiv preprint arXiv: 2303.18223*.

Yaxin Zhu and Hamed Zamani. 2024. Icxml: An in-context learning framework for zero-shot extreme multi-label classification. *arXiv preprint arXiv:2311.09649*.

A Appendix

A.1 Prompt Examples and Instructions

A.1.1 Prompt Examples Sampling

	Language	N_{examples}	N_{labels}	$\text{Sim}_{\text{lemma}}$
1	German	8	random	random
2	German	8	random	random
3	German	8	random	random
4	German	8	random	random
5	German	8	random	random
6	English	8	random	random
7	English	8	random	random
8	English	12	random	random
9	Mixed	8	random	random
10	Mixed	8	random	random
11	Mixed	12	random	random
12	German	8	1-2	0.7-1
13	German	8	1-2	0-0.3
14	German	8	5-10	0.7-1
15	German	8	5-10	0-0.3

Table 4: Prompt sampling overview.

A.1.2 Instruction for *complete*

The instruction we used for the *complete*-stage:

> Dies ist eine Unterhaltung zwischen einem intelligenten, hilfsbereitem KI-Assistenten und einem Nutzer. Der Assistent antwortet mit Schlagwörtern auf den Text des Nutzers.
>
> *This is a conversation between an intelligent, helpful AI-assistant and a user. The assistant replies with keywords to the text entered by the user.*

A.1.3 Instruction for *rank*

This is the instruction we used for the *rank*-stage:

> Du erhälst einen Text und ein Schlagwort. Bewerte auf einer Skala von 1 bis 10, wie gut das Schlagwort zu dem Text passt. Nenne keine Begründungen. Gib nur die Zahl zwischen 1 und 10 zurück.
>
> *You receive a text and a keyword. On a scale from 1 to 10, estimate how well the keyword fits to the text. Do not give reasons. Only reply with the number between 1 and 10.*

A.2 Ablation Study Ensemble Strategy

An interesting insight into our system is to evaluate the additional value of our ensembling approach. As in Trad and Chehab (2024), we complemented the top-20-set of models×prompt combinations with other strategies:

- `top-20-ensemble`: with varying models and prompts that generate candidates in the complete stage.
- `one-model-all-prompts`: All prompts are used with a single model.
- `one-prompt-all-models`: All models are used with a single prompt.
- `one-prompt-one-model`: A best performing single model-prompt combination is used.

All strategies include the rank step and the final combination step as in our overall system description. Figure 2 shows the PR-curves for the different strategies on the dev-test set. Table 5 shows the values of recall, precision and F1-score that could be obtained with an (F1-)optimal calibration of the system, also marked with a cross in Figure 2. Note, unlike the PR-curves in Figure 4, the curves in Figure 2 are not built only on the rank of the suggested subject terms, but also their confidence scores s_{fin}, as in Equation 1. Therefore, the curves achieve higher precision values in comparision to the curves that are built on rank only.

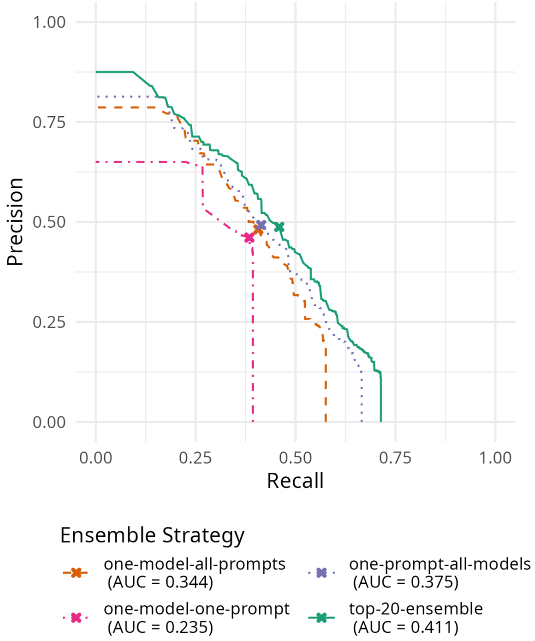

Figure 2: Precision-Recall curves for different model and prompt combinations, evaluated as doc-averages over our dev-test set.

Ensemble Strategy	Precision	Recall	F1
`top-20-ensemble`	0.488	0.459	0.420
`one-model-all-prompts`	0.481	0.407	0.393
`one-prompt-all-models`	0.492	0.414	0.407
`one-prompt-one-model`	0.461	0.385	0.380

Table 5: Precision, recall, F1-score for F1-optimal calibration of the system w.r.t. thresholding on confidence scores and limiting on rank, computed on dev-test set.

Comparing the precision-recall curves in Figure 2 , we can see that the `top-20-ensemble` is well above the other strategies in the high precision as well as the high recall domain. However, in the part of the curve where the F1-score is optimal, the ensembling strategies are quite close so that the added value of the ensemble is not as pronounced compared to the other strategies. This may indicate that the selection of models and prompts is good (yielding high precision and high recall in the extreme), but the weighting mechanism of the model-prompt combinations might be improved. Furthermore, we may conclude that varying the LLMs adds more value to the ensemble in contrast to varying the prompts.

A.3 Ablation Study: Single Model Performances

Another interesting insight into our system is how each different LLM combined with various prompts performs on its own. To illustrate the spread of precision and recall for the different model×prompt combinations, see Figure 3. These results are computed on the bare candidate sets suggested by the llm and mapped to the vocabulary. In this figure, no ranking stage has been applied.

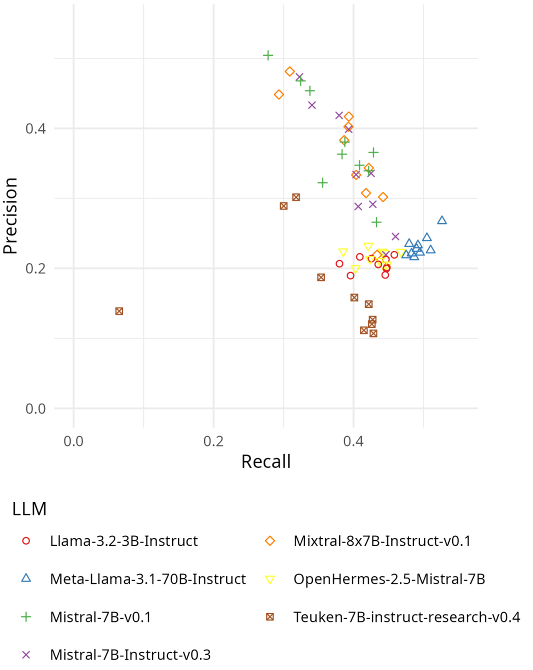

Figure 3: Precision-Recall Balance of single prompt-model combinations on the dev-test sample.

We can see that the most resource-intense model Llama-3.1-70B achieves highest recall with all prompts. However, precision is not as high as with the results stemming from the Mistral family. The Teuken model performs generally worst. Note, however, that even though a model-prompt combination may have low performance individually, it may still add value to an ensemble, as it may provide a different suggestion set than other models. Indeed, for overall ensemble performance we still found the Teuken model useful, probably due to its unique tokenizer.

A.4 Influence of α on PR-AUC

α	**1M-1P**	**1M-AP**	**1P-AM**	**top20**
0	**0.239**	0.285	0.297	0.301
0.1	0.235	0.344	0.373	0.402
0.2	0.235	**0.345**	**0.377**	**0.411**
0.3	0.235	0.344	0.375	**0.411**
0.4	0.234	0.340	0.369	0.408
0.5	0.232	0.335	0.366	0.405
0.6	0.232	0.333	0.363	0.402
0.7	0.231	0.330	0.359	0.397
0.8	0.230	0.327	0.355	0.394
0.9	0.229	0.324	0.350	0.391
1.0	0.170	0.312	0.328	0.384

Table 6: PR-AUC scores on the dev-test set for different values of α, which determines if the final ranking relies more on the relevance score (α<0.5) or the ensemble score (α>0.5). The ensembles are abbreviated: one-model-one-prompt (1M-1P), one-prompt-all-models (1M-AP), one-prompt-all-models (1P-AM) and top-20-ensemble (top20).

A.5 Comparing Precision-Recall Balance among Top Five Teams

Figure 4 shows the PR curves for the top five teams on the all-subjects task, plotting the values of precision@k and recall@k along the increasing values of k as reported in the shared task's leaderboard.

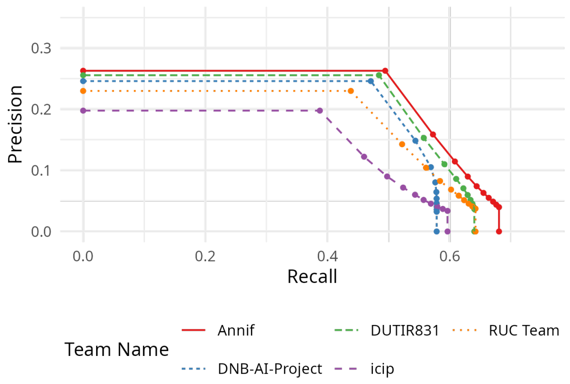

Figure 4: Precision-Recall curves for the top five teams on the all-subjects task.

A.6 Results by Language and Record Type

Record Type	P_5	R_5	$F1_5$
Article	0.1108	0.2685	0.1569
Book	0.2396	0.4898	0.3218
Conference	0.2603	0.4561	0.3314
Report	0.2385	0.4784	0.3183
Thesis	0.2912	0.3932	0.3346
Language	P_5	R_5	$F1_5$
de	0.2545	0.4787	0.3323
en	0.2307	0.4566	0.3065

Table 7: Metrics precision@5, recall@5 and F1@5 on the test set grouped by record type and language.

A.7 Qualitative Ratings by Subject Category

Table 8 shows the F1@5-score in the qualitative rating for each individual subject category.

Subject Category	F1@5
Architecture	0.502
Chemistry	0.428
Electrical Engineering	0.389
Material Science	0.435
History	0.322
Computer Science	0.531
Linguistics	0.421
Literature Studies	0.356
Mathematics	0.343
Economics	0.486
Physics	0.357
Social Sciences	0.409
Engineering	0.352
Traffic Engineering	0.343

Table 8: F1@5 scores in the qualitative ranking for different subject categories.

A.8 Ablation Study: Error Analysis

In addition to the quantitative results, we had a look at $n = 50$ random items from the dev-test split. The results are in Table 9.

		Not found		
Gold	Found	Close	Distant	Difficult
140	86 (61.4%)	20 (14.3%)	34 (24.3%)	44 (31.4%)
26	10 (38.5%)	6 (23.1%)	10 (38.5%)	17 (64.4%)

Table 9: Overview of how many of the gold subject terms in the ablation set are *found*, *not found* but have one or more *close* suggestions, *not found* with only *distant* suggestions found and *difficult*. Bottom row is `Article`-only.

Sample text[12] to illustrate difficulties of our system with the text type *Article*:

> Chapter 29 Agriculture and economic development "This chapter takes an analytical look at the potential role of agriculture in contributing to economic growth, and develops a framework for understanding and quantifying this contribution. The framework points to the key areas where positive linkages, not necessarily well-mediated by markets, might exist, and it highlights the empirical difficulties in establishing their quantitative magnitude and direction of impact. Evidence on the impact of investments in rural education and of nutrition on economic growth is reviewed. The policy discussion focuses especially on the role of agricultural growth in poverty alleviation and the nature of the market environment that will stimulate that growth.
> **Keywords**: Landwirtschaftliche Betriebslehre (*Agricultural economics*), Agrarpolitik (*Agricultural policy*), Landwirtschaft (*Agriculture*), Wirtschaftstheorie (*Economic theory*)

A.9 Hardware and Ressources

All our computations were run on our internal hardware consisting of `2 x Intel(R) Xeon(R) Gold 6338T CPU @ 2.10GHz` processors with two `NVIDIA A100` GPUs (each 80GB RAM) attached. Table 10 shows GPU-hours consumed by generating suggestions for the all-subjects test set of 27.987 documents.

Model Name	Size	GPUh	it/s
`Llama-3.2-3B-Instruct`	3B	2× 02:44 h	2.84
`Llama-3.1-70B-Instruct`	70B	2× 17:36 h	0.44
`Mistral-7B-v0.1`	7B	2× 04:16 h	1.82
`Mistral-7B-Instruct-v0.3`	7B	2× 03:50 h	2.03
`Mixtral-8x7B-Instruct-v0.1`	56B	2× 06:28 h	1.20
`OpenHermes-2.5-Mistral-7B`	7B	2× 03:36 h	2.16
`Teuken-7B-instruct-research-v0.4`	7B	2× 02:59 h	2.61

Table 10: Number of model parameters, GPU hours and iterations per second for different models generating keywords in the *complete* stage. Times measured for generating suggestions for all-subjects test set.

[12] Source: `https://github.com/jd-coderepos/llms4subjects/blob/main/shared-task-datasets/TIBKAT/all-subjects/data/dev/Article/en/3A1831638150.jsonld`