# Final Report: Multi-Agent System (MAS) for USOS Courses Recommendations

Łojek Grzegorz (g.lojek@student.uw.edu.pl),
Wilczyński Jakub (j.wilczynski2@student.uw.edu.pl),
Supervisor: Szczuka Marcin

July 1, 2025

**Abstract**

Each year, students struggle with choice of general university courses. With over 500 options available, manually reviewing them is time-consuming and exhausting. While it is relatively straightforward to filter courses using fixed criteria such as ECTS points or availability, matching students' contextual preferences (often embedded in course descriptions) remains a complex problem.

In this project, we propose the approach that leverages the capabilities of Large Language Models (LLMs) to comprehend and process rich textual data. We explore and evaluate the components of a Multi-Agent System (MAS) designed to assist students in making informed course selections through personalized recommendations related to their individual interests or preferences.

We identify two key components of the system: a Course Filtrating Agent and a Recommendation-Making Agent. Together, these agents enable the MAS to efficiently narrow down options and provide accurate, preference-aligned suggestions.

For evaluation, we constructed an annotated validation dataset based on data web-scraped from the USOS system. Each component was assessed using a variety of performance metrics.

Our findings demonstrate the potential of LLM-driven systems in enhancing the course recommendation process and offer promising directions for future development within the USOS ecosystem.

Code avaiable at: https://github.com/molocchus/NLP_MAS.git (Łojek & Wilczyński, 2025)

## 1 Introduction

Information about courses in the USOS system is partially structured and easy to extract—for example, fields such as ECTS points, course name, or current registration availability status are typically well-defined. However, more nuanced details such as course topics, assessment criteria, and learning outcomes are often embedded within long, inconsistent, and unstructured textual descriptions. Additionally, some information may be incomplete or missing entirely—for instance, the mode of course delivery might not be explicitly stated in a designated section and must be inferred from other sections.

These challenges highlight the need for a recommendation system that is not only flexible, but also intelligent enough to comprehend and extract relevant information from heterogeneous and partially structured data sources. A system capable of understanding context, handling ambiguity, and making inferences is essential for providing accurate and personalized course recommendations.

### 1.1 Goals

This project aims to address the challenges of course selection by developing an intelligent recommendation system based on a Multi-Agent System (MAS) architecture (Aziz, 2010). By assigning specialized tasks—such as filtering and ranking courses or interpreting user preferences—to individual agents, the system can deliver more precise and personalized recommendations. Additionally, the project explores and

compares different Large Language Models (LLMs) to identify the most effective combination for capturing and matching students' interests.

Our project took a practical, engineering-oriented approach, leveraging LLMs to address the recommendation problem. The main objectives were:

- To explore various approaches for building a MAS related to the courses recommendations.
- To identify and design key components that could improve both recommendation accuracy and system efficiency.
- To implement each component and thoroughly evaluate its performance.
- To gain hands-on experience with deploying LLMs in real-world scenarios.

### 1.2 Our Contributions

- We collected and processed course data through web scraping and prepared an annotated evaluation dataset.
- We proposed two MAS components based on LLMs: a **Course Filtrating Agent** and a **Recommendation-Making Agent**.
- We implemented both components using three different LLMs and evaluated each configuration.
- We obtained valuable insights into system performance under realistic data and resource constraints.

Due to time and computational limitations, we were not able to integrate and evaluate a complete MAS. However, the experiments we conducted provided meaningful insights into the effectiveness of such a system in the context of university course recommendations.

## 2 Related Work

In the landscape of intelligent recommendation systems, two prominent paradigms have emerged for leveraging language models and external knowledge: Multi-Agent Systems (MAS) and Retrieval-Augmented Generation (RAG) (Lewis et al., 2020). Our project focuses on the MAS approach, which offers some distinct advantages compared to RAG-based systems.

### 2.1 MAS vs. RAG: Conceptual Overview

Retrieval-Augmented Generation (RAG) combines a retrieval mechanism with a generative language model by first retrieving relevant documents or knowledge snippets from an external corpus, then conditioning the generation on these retrieved contexts. This approach excels at grounding responses in factual information, improving accuracy and up-to-date knowledge integration (Lewis et al., 2021). However, RAG requires direct access to retriever embeddings and vector indices, which may be closed-source or proprietary. While the traditional RAG pipeline may limit flexibility in modular reasoning or complex task decomposition, this limitation is addressed by recent improvements such as ComposeRAG (Wu et al., 2025), which introduces a modular and composable architecture for retrieval-augmented generation.

In contrast, Multi-Agent Systems (MAS) utilize multiple specialized agents—each potentially backed by independent LLMs or tool modules—that collaborate and communicate to solve complex tasks (Aziz, 2010). MAS architectures enable division of labor into interpreters, filters, rankers, and other functional units. This modularity offers several advantages:

- **Flexibility in model choice:** MAS can leverage closed-source LLMs accessible only via API, without needing access to internal vector representations or retriever infrastructure.
- **Broader conceptual coverage:** Separate agents can focus on different aspects of the problem, enabling more nuanced reasoning and the capture of diverse user preferences and course features. (Han et al., 2025; Ling et al., 2023)

- **Easier incorporation of domain-specific logic:** Agents can be customized and extended with task-specific heuristics or data integration without retraining a monolithic model.
- **Improved interpretability and control:** The agent interactions can be monitored and adjusted, offering clearer insight into recommendation rationale.

However, MAS approaches also come with challenges such as increased architectural complexity, potential communication overhead, and often slower inference due to multiple sequential agent calls.

### 2.2 Zero-shot Question Answering

Zero-shot question answering (QA) refers to the capability of language models to respond accurately to queries without requiring any task-specific fine-tuning or prior examples. Instead, the model leverages its pre-trained knowledge and contextual understanding to infer relevant answers directly from the prompt. This approach is particularly valuable in scenarios where labeled training data is scarce or unavailable, enabling flexible and scalable deployment across diverse domains.

In the context of course recommendation systems, zero-shot QA can be employed to interpret and match user queries or preferences against complex course descriptions. Rather than relying on handcrafted rules or supervised classifiers, a zero-shot QA agent can dynamically understand natural language requests—such as filtering courses by topics, delivery modes, or assessment types—and return relevant outputs accordingly.

This capability aligns well with the modular design of our MAS, where the **Course Filtrating Agent** can leverage zero-shot QA techniques to efficiently reduce the candidate pool based on loosely defined or evolving user preferences. The subsequent **Recommendation-Making Agent** can then perform a more fine-grained evaluation with access to complete course metadata.

Zero-shot QA approaches thus offer a flexible, data-efficient alternative to traditional supervised methods, facilitating rapid adaptation to new tasks and domains—a crucial advantage in dynamic educational environments such as USOS.

## 3 Model

We propose two distinct modules within a Multi-Agent System (MAS) architecture related to the recommendation of USOS courses:

- **Course Filtrating Agent** — This module is responsible for quickly filtering the pool of available courses based on their titles and the user's preferred topics. Due to the limited input scope, it operates efficiently and conserves computational resources for the next stage.
- **Recommendation-Making Agent** — This module provides detailed recommendations by evaluating each course against the full spectrum of user preferences. It considers all available information for a given course, including the name, description, assessment criteria, mode of delivery, and more. It processes one course at a time, comparing its features to the user's profile to generate a recommendation score.

Both modules can be easily parallelized to improve processing speed.

## 4 Experimental setup

We evaluated each MAS component independently using three different Large Language Models (LLMs):

1. **Bielik-1.5B-v3.0-Instruct** (Ociepa et al., 2025) — A lightweight model optimized for Polish language understanding. It was run locally on the *Entropy* server at MIMUW.
2. **GPT-4.1-nano** (OpenAI, 2025) — A general-purpose model developed by OpenAI, designed to balance performance and computational efficiency.

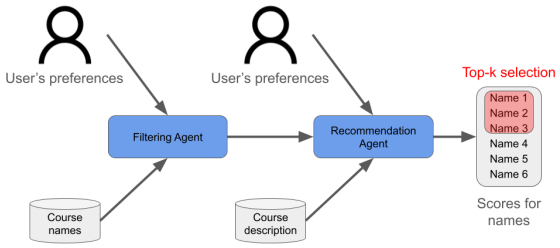



Figure 1: Workflow diagram of the proposed MAS. The system outputs a ranked list of courses based on their recommendation scores.

3. **GPT-4o-mini** (OpenAI, 2024) — A lightweight reasoning-oriented model from OpenAI, offering improved contextual understanding with a smaller computational footprint.

Each model was integrated into an agent-based architecture using the **Swarms** (Gomez, 2025) framework. We opted to use Swarms instead of the more commonly known **CrewAI** (CrewAI, 2025), in order to explore alternative multi-agent orchestration frameworks and evaluate their practical utility.

## 4.1 Dataset

The dataset was collected from the USOS web portal (Uniwersytet Warszawski & Międzyuniwersyteckie Centrum Informatyzacji, 2025) (with a permission from the administrators to perform web scraping). In total, we successfully gathered information for 585 courses.

From this dataset, we constructed a validation set consisting of 111 carefully selected samples. These samples were chosen based on the completeness of their descriptions and the ease with which they could be annotated. Each validation sample includes the following features:

- **Name** — The title of the course.
- **ECTS** — The number of European Credit Transfer and Accumulation System points.
- **Course Conduct Mode** — The format in which the course is delivered (e.g., in-person, online, hybrid).
- **Passing Criteria** — A description of how students are assessed (e.g., exams, assignments, attendance).
- **Course Topics** — A list or description of the primary topics covered by the course.

## 4.2 Evaluation of the Course Filtrating Agent

The Course Filtrating Agent was configured to return responses in the following structured format:

```
odpowiedź:{
    'nazwa przedmiotu': {nazwa przedmiotu},
    'prawidlowość przedmiotu': {tak/nie w zależności od tego czy pasuje czy nie},
    'zgodność tematyki zajęć': {liczba 0-10 odzwierciedlająca dopasowanie
        Opisu kursu do Preferencje studenta}
}
```

This unified response format allowed us to extract both categorical (lingual) and numerical similarity scores, enabling a comparison between different evaluation strategies. Ideally, each format would be tested using a separate prompt to avoid confounding effects; however, due to limited access to computational resources, we opted to evaluate both from a shared output structure.

Categorical (lingual) responses enabled the computation of standard classification metrics such as precision, recall, and accuracy. Numerical responses, on the other hand, allowed us to compute ranking-based metrics such as precision@k and recall@k (KS & Shajan, 2024) for various values of $k \in \{5, 10, 50\}$.

For each model output, we collected both the predicted and ground truth labels. Using this, we constructed confusion matrices for the categorical responses, and subsequently calculated accuracy, precision, and recall.

For the numerical scores, we ranked all courses by their predicted similarity and evaluated precision@k and recall@k for $k = 5, 10, 50$.

Additionally, we computed the mean score of numerical response (number from 0-10) conditioned on categorical response ("tak" (yes) and "nie" (no))

### 4.3 Evaluation of the Recommendation-Making Agent

The course ranker agent was designed to return responses in the following structured format:

```
odpowiedź: {
    'nazwa przedmiotu': {nazwa przedmiotu},
    'zgodność tematyki zajęć': {liczba 0-10 odzwierciedlająca dopasowanie
        opisu kursu do preferencji studenta},
    'zgodność trybu prowadzenia zajęć': {liczba 0-10 odzwierciedlająca dopasowanie
        opisu kursu do preferencji studenta},
    'zgodność rodzaju zaliczenia': {liczba 0-10 odzwierciedlająca dopasowanie
        opisu kursu do preferencji studenta}
}
```

This structured numerical output made it straightforward to compare results and compute ranking metrics. Categorical (lingual) responses were not used in this case, as they would complicate the process of producing ranked recommendations for a fixed number of samples.

With these numerical outputs, we were able to compute metrics such as **precision@k** and **recall@k**.

To fairly evaluate the course ranker agents, we designed a contrastive setup. Each agent was tasked with scoring courses under controlled conditions—comparing courses that matched specific student preferences against those that did not.

We defined four preference scenarios:

- **All categories combined** – the student provided preferences for three categories (topic, course mode, and passing criteria). The final score was computed as the sum of valid scores for each of these categories.
- **Single-category preferences** – only one preference category was provided (e.g., topic), while others were set to "no preference". Only the score corresponding to the selected preference category was used in evaluation.

For each preference category, we selected a sample of 10 courses that matched the user preferences and 10 that did not. In total, 240 responses were collected across the different categories.

**Accuracy** was computed by comparing the scores assigned to matching vs. non-matching courses. For the "All categories combined" scenario, all three individual category scores needed to be higher for the matching course to be counted as correct.

To compute **precision@k** and **recall@k**, we ranked all courses by their final score and calculated how many of the top-$k$ courses belonged to the group of matching samples.

# 5 Results and Discussion

## 5.1 Results

Our experiments revealed that the **Bielik** model produced a significant number of erratic responses (appendix: Figure 2, Figure 6). This behavior is likely due to its relatively small size—1.5 billion parameters—making it more prone to hallucinations and misunderstandings (Kaplan et al., 2020). Consequently, Bielik's overall performance was considerably poorer compared to the GPT-based models. Based on differences of means of scores (appendix: Figure 5, Figure 8), there is also a possibility that the Bielik model did not fully comprehend both tasks, especially the recommendation task, which could further explain its lower effectiveness.

The GPT models demonstrated consistent and reliable scoring across both the filtering and recommendation tasks. Notably, the reasoning-focused GPT-4o-mini model outperformed others in the recommendation task (appendix: Figure 7), suggesting its stronger capability in nuanced decision-making. In the filtrating task, we observed a trade-off: the reasoning model (GPT-4o-mini) achieved higher recall, while the more generalist GPT-4.1-nano model showed higher precision (appendix: Figure 3). This insight indicates that model selection within a MAS should be guided by the specific properties and goals of the system, balancing between precise selection and comprehensive coverage.

### 5.1.1 Time Execution

For each model configuration, we measured the total execution time of the filtrating task to assess the practical efficiency of deploying such a system. The results indicate that, in its current form, the proposed architecture is not suitable for personal use due to excessive inference time—particularly for locally hosted models. The **Bielik** model, for instance, was deployed on the *Entropy* cluster using an **NVIDIA RTX 2080 Ti** GPU, yet still exhibited significantly slower performance compared to cloud-based alternatives.

Table 1: Execution Time of the Recommendation-Making Agent Experiments

| **Model** | **Total Time** | **Runs** | **Time per Output** |
|---|---|---|---|
| Bielik | 2h 04m 36.3s | 412 | 18.14s |
| GPT-4.1-nano | 5m 37.7s | 412 | 0.82s |
| GPT-4o-mini | 13m 28.9s | 412 | 1.96s |

Table 2: Execution Time of Course Filtrating Agent Experiments

| **Model** | **Total Time** | **Runs** | **Time per Output** |
|---|---|---|---|
| Bielik | 1h 26m 44.9s | 240 | 21.68s |
| GPT-4.1-nano | 6m 14.2s | 240 | 1.55s |
| GPT-4o-mini | 20m 43.3s | 240 | 5.17s |

## 5.2 Discussion

Our evaluation of the two MAS components—Course Filtrating Agent and the Recommendation-Making Agent—demonstrated promising capabilities in leveraging LLMs for personalized course recommendations within the USOS system. The GPT-based models consistently outperformed the locally hosted Bielik model in terms of accuracy, precision, and recall across all tasks and evaluation metrics. Notably, the reasoning-focused GPT-4o-mini model showed better accuracy in the recommendation task based on complete course descriptions and achieved higher recall in the filtrating task. This characteristic may enhance the system's overall efficiency; however, the high precision of GPT-4.1-nano in the filtrating task suggests it might be a preferable choice depending on the application's priorities.

Our experiments revealed significant challenges regarding the practical deployment of such systems. The inference times for GPT models, although substantially faster than Bielik, remain too long for real-time or

large-scale production use. Bielik's slow performance and higher rate of erratic or invalid responses further underscore the need for careful model selection and optimization when designing MAS architectures.

# 6 Conclusions

Overall, our results indicate that while LLM-powered MAS architectures hold potential for intelligent course recommendation, significant engineering and optimization work remains necessary to bridge the gap between research prototypes and practical, scalable solutions.

We also gained practical insight into a novel framework designed specifically for building Multi-Agent Systems—**Swarms**. While Swarms provided a flexible base for implementing and orchestrating our agents, we found several limitations that impacted development. The documentation was sparse and incomplete, which made understanding key features and debugging challenging. For example, agents occasionally accumulated tokens excessively, degrading performance, but due to the lack of detailed explanations in the documentation, we were unable to fully diagnose or resolve this issue. Given these challenges, for future projects, more mature and better-supported frameworks such as **CrewAI** may offer a smoother development experience and improved stability.

Through this project, we gained valuable practical experience working with LLM-based agents and tackling real-world challenges related to data collection, system design, and evaluation. Future work should focus on improving inference efficiency, robustness across diverse preference combinations, and full MAS integration to realize a scalable and user-friendly course recommendation system.

# References


Aziz, H. (2010). Multiagent systems: Algorithmic, game-theoretic, and logical foundations by y. shoham and k. leyton-brown cambridge university press, 2008. *ACM Sigact News*, *41*(1), 34–37.

CrewAI, I. (2025). The leading multi-agent platform [Accessed: 2025-07-01]. https://www.crewai.com/

Gomez, K. (2025, June). Swarms docs [Accessed: 2025-07-01]. https://docs.swarms.world/en/latest/

Han, Z., Wang, J., Yan, X., Jiang, Z., Zhang, Y., Liu, S., Gong, Q., & Song, C. (2025). Coreaagents: A collaboration and reasoning framework based on llm-powered agents for complex reasoning tasks. *Applied Sciences*, *15*(10), 5663.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. https://arxiv.org/abs/2001.08361

KS, S., & Shajan, R. (2024). Evaluating similarity measures in collaborative filtering: Insights into accuracy, precision, and computational performance. *Georgian Education Mine (GEM)*.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2021). Retrieval-augmented generation for knowledge-intensive nlp tasks. https://arxiv.org/abs/2005.11401

Ling, C., Zhao, X., Lu, J., Deng, C., Zheng, C., Wang, J., Chowdhury, T., Li, Y., Cui, H., Zhang, X., et al. (2023). Domain specialization as the key to make large language models disruptive: A comprehensive survey. *arXiv preprint arXiv:2305.18703*.

Łojek, G., & Wilczyński, J. (2025, July). Github repository: Group project for nlp. https://github.com/molocchus/NLP_MAS/

Ociepa, K., Flis, Ł., Kinas, R., Wróbel, K., & Gwoździej, A. (2025). Bielik v3 small: Technical report. *arXiv preprint arXiv:2505.02550*.

OpenAI. (2024, July). Gpt-4o mini: Advancing cost-efficient intelligence [Accessed: 2025-07-01]. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

OpenAI. (2025, April). Introducing gpt-4.1 in the api [Accessed: 2025-07-01]. https://openai.com/index/gpt-4-1/

Uniwersytet Warszawski & Międzyuniwersyteckie Centrum Informatyzacji. (2025, June). Uniwersytecki system obsługi studiów usosweb [Accessed: 2025-07-01]. https://usosweb.uw.edu.pl/

Wu, R., Lee, Y., Shu, F., Xu, D., Hwang, S.-w., Yao, Z., He, Y., & Yan, F. (2025). Composerag: A modular and composable rag for corpus-grounded multi-hop question answering. https://arxiv.org/abs/2506.00232

# Appendices

In the appendices, all plots and metrics are shown.

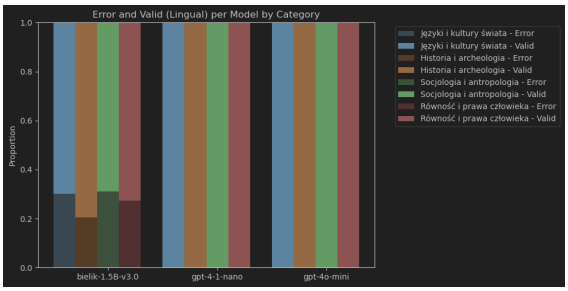
## A Metrics and plots for the Course Filtrating Agent

On the Course Filtrating Agent's evaluation plots, the following categories correspond to the specific user preferences provided in the prompts:

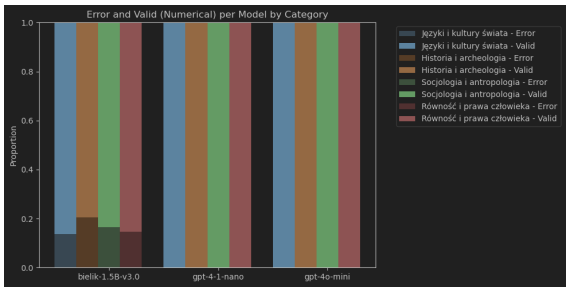
- **Języki i kultury świata** (Languages and Cultures of the World)
- **Historia i archeologia** (History and Archaeology)
- **Socjologia i antropologia** (Sociology and Anthropology)
- **Równość i prawa człowieka** (Equality and Human Rights)

These categories directly reflect the thematic preferences selected in the prompts for course filtering. They serve as the basis for assessing how accurately the Course Filtrating Agent can identify and retain courses matching the user's chosen topics.

### A.1 Course Filtrating Agent, Models erratic responses



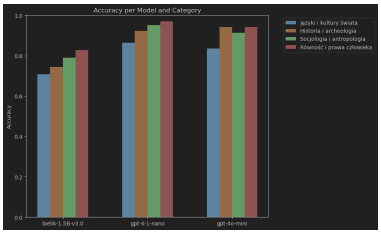

(a) Errors of lingual responses.



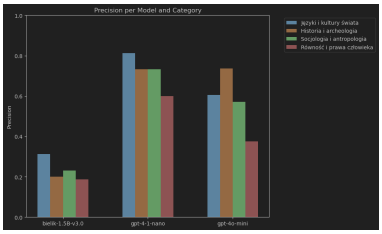

(b) Errors of numerical respones.

Figure 2: Errors in responses were observed exclusively in the case of the **Bielik** model. These errors occurred more frequently in the categorical (lingual) outputs. Typical issues included repeating the prompt in the response, using English instead of Polish, or returning values in an incorrect format—such as raw numerical or boolean answers instead of the expected categorical form ("tak"/"nie"). .

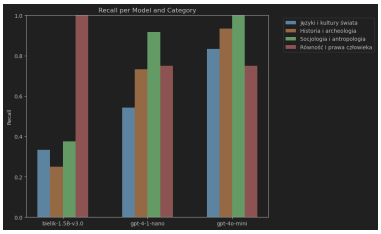
## A.2 Course Filtrating Agent, Metrics for lingual data




(a) Accuracy of filtering with use of lingual data



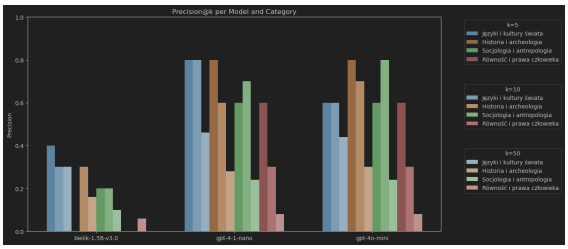

(b) Precision of filtering with use of lingual data



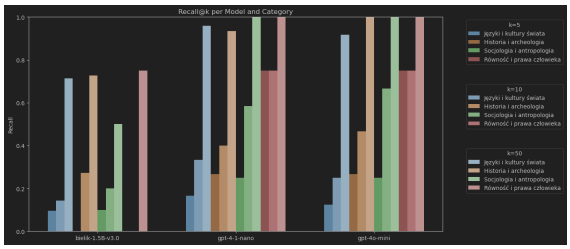

(c) Recall of filtering with use of lingual data

Figure 3: All models achieved an accuracy above 50%. However, the **Bielik** model showed significantly lower precision and recall overall. Interestingly, Bielik achieved a perfect precision score (100%) in setups with many user preferences. This may be attributed to its pretraining on legal texts, potentially making it more effective at recognizing topics related to equality, human rights, and law. Despite this isolated strength, both GPT-based models clearly outperformed Bielik across most evaluation metrics. **GPT-4.1-nano** achieved the highest overall precision, indicating strong performance in identifying only the most relevant courses. In contrast, **GPT-4o-mini** achieved better recall, suggesting it is more effective at identifying a broader set of related samples. This supports the interpretation that the general-purpose model (GPT-4.1-nano) excels in precise filtering, while the reasoning-optimized model (GPT-4o-mini) is better suited for broader semantic understanding.

## A.3 Course Filtrating Agent, Metrics for numerical data




(a) Precision@k of filtering with use of lingual data




(b) Recall@k of filtering with use of lingual data.

Figure 4: We computed both precision@k and recall@k for $k = 5, 10, 50$. The **Bielik** model performed significantly worse compared to the GPT-based models across all values of $k$, reaffirming its limitations in this task. In contrast, both **GPT-4.1-nano** and **GPT-4o-mini** demonstrated comparable performance in this evaluation, with only minor differences between them across the evaluated $k$-values.

### A.4 Course Filtrating Agent, Mean score per response

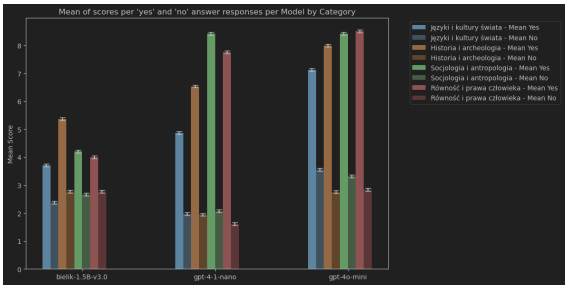



Figure 5: It is evident that the numerical similarity scores strongly correlate with the categorical (lingual) outputs for the **GPT-based models**, indicating consistent and reliable behavior across both response formats. However, for the **Bielik** model, this correlation appears weaker. This suggests that the numerical values it generates may not reliably reflect the corresponding categorical judgments, reducing their utility for downstream evaluation or ranking.

## B Metrics and plots for the Recommendation-Making Agent

On the Recommendation-Making Agent's evaluation plots, the following categories correspond to specific types of user preferences used as inputs in the prompts:

- **Course Conduct Mode**
- **Passing Criteria**
- **Preferred Course Topics**
- **All Categories Combined**

Each category reflects the type of preference from which the prompt's input was selected. The associated preference options for each category are as follows:

- **Course Conduct Mode**: `['zdalnie', 'mieszany:  w sali i zdalnie', 'w sali']`
- **Passing Criteria**: `['Esej/praca pisemna', 'Obecność/aktywność', 'Test/egzamin']`
- **Preferred Course Topics**: `['Języki i kultury świata', 'Historia i archeologia', 'Socjologia i antropologia']`
- **All Categories Combined**:
  - `['zdalnie', 'Test/egzamin', 'Języki i kultury świata']`
  - `['mieszany:  w sali i zdalnie', 'Test/egzamin', 'Historia i archeologia']`
  - `['w sali', 'Test/egzamin', 'Języki i kultury świata']`

By evaluating the Recommendation-Making Agent's performance separately on these categories, we can assess how well it adapts to different user preferences both individually and in combination. This helps us understand which types of preferences the agent handles best and informs future improvements.

## B.1 Recommendation-Making Agent, Models erratic responses

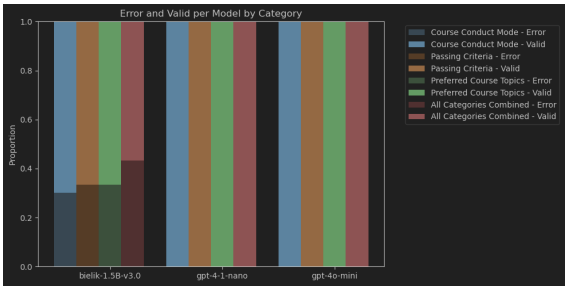
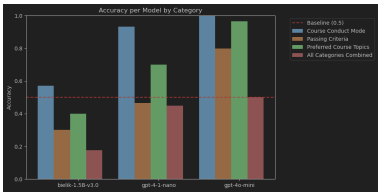
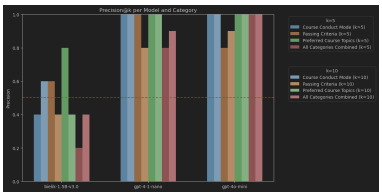



Figure 6: The GPT-based models did not produce erratic responses, whereas the **Bielik** model exhibited nearly one-third of responses with errors. Notably, more errors were observed in the "All categories combined" scenario, suggesting that handling multiple simultaneous preferences posed greater difficulty for Bielik in reasoning about recommendation accuracy.

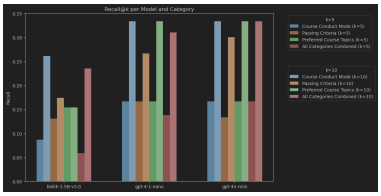
## B.2 Recommendation-Making Agent, Metrics




(a) Accuracy of recomendations




(b) Precision@k




(c) Recall@k of recomendations

Figure 7: The accuracy of recommendations was highest for the GPT reasoning model (**GPT-4o-mini**) and lowest for the **Bielik** model. It is worth noting that the baseline accuracy for the "All categories combined" scenario may not be entirely reliable due to the complexity of aggregating multiple preference categories. Precision@k and Recall@k metrics were computed for $k = 5, 10$. Both GPT models demonstrated comparable performance in these metrics. The higher overall accuracy of the reasoning-focused GPT-4o-mini suggests that it may be better suited for the recommendation ranking task.

## B.3 Recommendation-Making Agent, Mean score per response

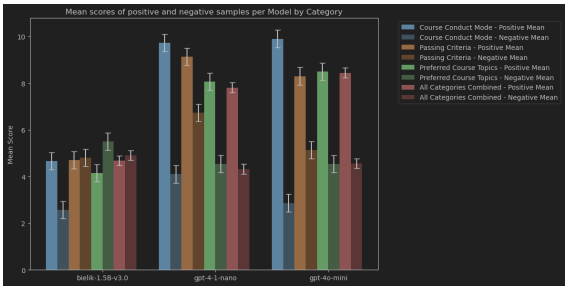



Figure 8: It is evident that the numerical scores strongly correlate with the categorical (lingual) outputs for the GPT models, indicating consistent behavior across both output formats. In contrast, for the Bielik model, this correlation appears weaker, suggesting that the numerical values may not reliably correspond to the lingual responses.