AMATU@Simpletext2024: Are LLMs Any Good for Scientific Leaderboard Extraction?

Notebook for the SimpleText Lab at CLEF 2024

Moritz Staudinger[1,*,†], Alaa El-Ebshihy[1,2,*,†], Annisa Maulida Ningtyas[1,*,†], Florina Piroi[1,2] and Allan Hanbury[1]

[1]Technische Universität Wien, Austria
[2]Research Studios Austria, Data Science Studio, Austria

Abstract

In this paper, we present our approach to solve the SOTA challenge of the SimpleText shared task at CLEF 2024. The objective of the challenge is to extract all (Task, Dataset, Metric, Score) tuples from scientific papers which report model score on benchmark datasets. In this work, we propose a rule-based classification model to identify papers that reports score information. We then apply different methods to extract TDMS using: (1) a baseline model from the literature, and (2) two Large Language Models (LLMs), GPT-3.5 and Mistral. Results show that the baseline model outperforms the LLMs in most cases, especially in zero-shot settings, with improvements seen in few-shot settings. Manual investigation shows that extracting TDMS from paper text is challenging, particularly for "Dataset" and "Score" extraction.

Keywords

Scientific Text Extraction, State-of-the-art, Entity Extraction, Relation Extraction

1. Introduction

In our data-driven world, the volume of published literature, including newspaper articles, social media posts, and scientific publications, is rapidly increasing. Since technological and scientific advancements are generally communicated through scientific publications, it is important to find and keep track of significant advances and challenges in various scientific fields. With the never-ending flow of new scientific publications (e.g. 1,000 new ML publications per month on arXiv[1] alone), it is becoming increasingly difficult to keep updated with the state-of-the-art for a given scientific task and compare new research contributions with previous ones.

In the particular area computer science of experimentation and evaluation of ML/IR models, assessing the effectiveness of a new model or algorithm is difficult due, not least, to heterogeneous reporting styles. To address this, a possibility is to create machine-readable results by either (1) creating machine actionable publications (that is, publications prepared in such a way that they contain further specifically formatted data which can be automatically processed and harvested correctly by algorithms) or (2) standardizing the evaluation and experimentation environment.The first path, if established, would allow creating and extracting comparable results with a minimal overhead for researchers, as the results could be collected in a standardized format, along with the written submissions. This is, though, unlikely to happen in the near future, as it requires a critical mass of scientists to alter their documentation habits, as stated by Kabongo et. al. [1].

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France
*Corresponding author.
†These authors contributed equally.
moritz.staudinger@tuwien.ac.at (M. Staudinger); alaa.el-ebshihy@tuwien.ac.at (A. El-Ebshihy); annisamaulidaningtyas@ugm.ac.id (A. M. Ningtyas); florina.piroi@tuwien.ac.at (F. Piroi); allan.hanbury@tuwien.ac.at (A. Hanbury)
0000-0002-5164-2690 (M. Staudinger); 0000-0001-6644-2360 (A. El-Ebshihy); 0000-0002-5041-0230 (A. M. Ningtyas); 0000-0001-7584-6439 (F. Piroi); 0000-0002-7149-5843 (A. Hanbury)

[1]https://arxiv.org/stats/monthly_submissions

Standardizing experimentation and evaluation environments is done through *shared tasks* or *evaluation labs/tracks*, which standardize the evaluation environment to evaluate the state-of-the-art performance of predefined tasks, with given metrics and hidden evaluation datasets. Although the results of these challenges are valuable to the community, they only provide comparable results for a few selected tasks and datasets, but not for the vast variety of research. Therefore, many research publications are not comparable, as they do not follow standardized evaluation strategies, use variations of datasets [**?**] or propose new tasks and datasets.

Therefore, extracting scientific entities from scholarly articles is currently the best option for enabling comparable and machine-actionable results throughout the scientific community. Although platforms like PapersWithCode, AI-metrics, NLP-Progress, and the Open Research Knowledge Graph [2] enable the comparison of research results on given datasets, they are manually curated, therefore limited by the crowd-sourcing resources of the community and subject to human error.

Automated extraction of predefined scientific entities, such as *Task-Dataset-Metric-(Score)* (TDMS) [3, 1], can help automatic knowledge base population and comparability of research contributions.

In this work, we present our approaches for the SimpleText State-of-the-Art (SOTA) Extraction Challenge at CLEF 2024 [4, 5]. The task is to determine whether a given scientific paper reports model scores on benchmark datasets and, if so, extract all TDMS tuples. We use a large language model (LLM)-based approach to extract and combine dependent scientific entities (e.g. a score depends on a given dataset, task, and metric) in terms of the TDMS objective, and discuss improving this task's performance with two distinct rule-based prefiltering systems for faster and more accurate extraction. As a baseline, we use the extraction tool presented by Kardas et al. [6].

The remainder of this work is structured as follows. In Section 2, we discuss related work in the area of scientific entity extraction. In Section 3, we discuss which methods we applied to approach the task challenge. In Section 4, we present our results for the shared task test sets. In Section 5, we discuss our approaches and limitations, which we then follow up with a summary and a brief outlook in Section 6.

2. Related work

In the area of IR and ML evaluation, one way to follow advancements in the area is the automatic generation of leaderboards by extracting data from scholarly articles, with the CLEF Simpletext State-of-the-Art extraction Task contributing to the evaluation of these efforts. Augenstein et al. [7] organized a Shared Task at the SemEval Workshop 2017, where participants had to extract three types of entities from scientific paragraphs: *Task*, *Method*, and *Material*. This work was extended by Gabor et al. [8], where additionally to analyzing paragraphs, the authors have also used annotated abstracts.

Independent of an evaluation lab or shared task, research on information extraction from publication texts usually create specific datasets on which they experiment and evaluate their models. Jain et al. [3] introduced the SciREX dataset and model, which extracts *dataset*, *metric*, *task*, and *method* entities from a corpus of 1,170 ML articles from PapersWithCode. Hou et al. [9] published the first datasets for *Task, Dataset, Metric, and Score* (TDMS) extraction in the NLP domain using distant supervision annotations. Kardas et al. [6] developed the AxCell pipeline to process LaTeX source code and extract TDMS. Kabongo et al.[1, 10, 2] focused on mining for TDMS tuples, with the goal to automatically populate the Open Research Knowledge Graph (ORKG) with this information. Automating this process has the potential to accelerate the growth of the ORKG, permitting an easier comparison of research results across scholarly articles. Yang et al. [11] analyzed existing solutions assessing their limitations and proposed an approach that does not require LaTeX sources, is not limited to a closed taxonomy (e.g. not limited to extracting TDMS), and requires less supervision than previous solutions, like, for example, Kardas et al. [6].

3. Methods

The SOTA challenge aims to extract all TDMS tuples from a given arXiv scientific paper, labeling the article as "answerable" if no TDMS tuple is found. We first inspected the training and validation sets to understand the distribution of the articles the contain TDMS tuples and "unanswerable" articles. The training dataset includes 9,352 articles with LaTeX sources and annotations: 5,274 with TDMS tuples and 4,078 labeled as "unanswerable". Similarly, the validation set is composed of 100 articles: 50 articles with TDMS tuples and 50 articles labeled as "unanswerable". The training and the validation set are evenly distributed between articles that contain TDMS and "unanswerable".

Therefore, our approach to solving the challenge consists of two modules: (1) *Filtering unanswerable documents* by applying a rule-based binary classification model to identify papers that do not contain TDMS, and (2) *TDMS extraction* to identify all TDMS in a given paper. For the TMDS extraction we experiment with AxCell [6] as a baseline model, and with GPT-3.5 and Mistral LLMs.

3.1. Filtering out unanswerable documents

We apply a rule-based binary classification method to recognize papers which are classified as "unanswerable". These papers are excluded from further processing and, therefore, reduce the costs for running the advanced models. For this, we evaluated three rule-based settings with similar configurations but different outcomes, aiming for high recall to ensure only clearly unanswerable documents are filtered out. Each setting was evaluated based on *Precision*, *Recall*, and *Accuracy* in identifying "unanswerable" articles in the validation dataset. Table 1 shows the configurations tested. The first

Table 1
Results of Prefiltering Methods on the Validation Set

Method	Precision	Recall	Accuracy
Result Section Exists	0.685	0.96	0.76
Result Section Exists with add. terms	0.67	0.98	0.75
Result Table Exists	0.85	0.80	0.83

two methods assess section titles to determine if a paper is "unanswerable". The first approach (*Result Section Exists*) checks if any section title includes the terms *result*, *experiment*, or *evaluation*, indicating the presence of scores. The second approach (*Result Section Exists with add. terms*) extends on this idea, with the two additional terms *comparison* and *performance*. To build on the idea of section title detection, we further scan for any tables in the result section (*Result Table Exists*). This was done by heuristically looking if the result section of the LaTeX source code contains the phrase *\begin{table*, instead of only looking at the section name. Although this improved Precision and Accuracy, the Recall dropped significantly.

As a result, we chose the first method presented, which filters on the basis of the three section names. This method yields a similar Recall as the second approach, with only three papers of the validation data set are differently classified.

3.2. TDMS extraction models

We experimented with the TDMS extraction using different models (see Table 2). We divide the experiments into two types: Baseline model and Large Language Models (LLMs). As a baseline model we utilize the AxCell presented by Kardas et al. [6]. For the LLMs models, we utilize GPT-3.5 and Mistral (an open source LLM) in zero-shot and few-shot settings and different sources as input for the prompt text. We detail the implementation in the following sections.

Table 2
Summary for the systems used in the experiments.

Type	Id	Name	Filtered	zero- or few-shot	fulltext or az	PwC information
Baseline	1	AxCell	✓	–	fulltext	✗
LLMs	2	GPT35-zero	✗	zero	fulltext	✗
	3	GPT35-fil-zero	✓	zero	fulltext	✗
	4	GPT35-few	✗	few	fulltext	✗
	5	GPT35-fil-few	✓	few	fulltext	✗
	6	GPT35-info-few	✗	few	fulltext	✓
	7	GPT35-az-few	✗	few	az	✗
	8	GPT35-az-info-few	✗	few	az	✓
	9	Mistral-fil-zero	✓	zero	fulltext	✗
	10	Mistral-fil-info-zero	✓	zero	fulltext	✓

3.2.1. Baseline Model - AxCell

We use the implementation of the AxCell system [6] (model Id 1 in Table 2). AxCell is a machine learning pipeline, which extracts TDMS tuples from scientific papers, by combining neural networks for text extraction and table extraction. The extracted linking candidates are then verified through *caption classification*, *mention lookup* and *table segmentation*, before being merged into TDMS tuples.

To use AxCell for this shared-task, we first download the eprint version of the arXiv publications from the test datasets to create the same data structure as in the original paper. Then we filter out "unanswerable" papers using the previously mentioned rule-based classification system (Section 3.1). The eprint is then processed by the AxCell Extraction script to obtain all the unpacked LaTeX sources, graphics, and an HTML version of the article. The processed articles are fed to a Neural Network which first extracts scientific entity candidates (Task, Dataset, Metric) and then tries to link them with the extracted scores from the tables. Therefore, each table cell is annotated with meta-information, as the TDM combination used to obtain a given score.

3.2.2. LLMs

We use two LLMs in our experiments: GPT-3.5 [2] (models 2-8) and MistralAI (models 9-10) as an OpenSource alternative to OpenAI's GPT model. We divide our experiments using different criteria.

Instructions to the model: We set up models in a zero-shot setting, where we give the LLM instructions, only (models 2, 3, 9, and 10), and models in a few-shot setting, where we give instructions and a small number of examples (models 4 to 8).

Filtering out unanswerable papers: In some of our models (ids 3, 5, 9 and 10), we filter out unanswerable papers with our selected rule-based classification (Section 3.1). In others, we use the complete test-sets without filtering, letting the model decide if the paper contains TDMS or unanswerable (models 2, 4, and 6 to 8).

Input text to the model: In models 2 to 6, 9 and 10 we use the full paper text as input to the LLM to extract the TDMS. Models 7 and 8, inspired by Argumentative Zoning (AZ) [12] which defines the main rhetorical structure in scientific articles, we extract only the text from sections referring to experiments and results, in addition to the abstract. We believe that these sections contain the TDMS information and, thus, avoid, processing the full paper text. We refer to the modules that utilize only the experiments and results sections text to using **az** as input while others use full text input (Column 6 in Table 2).

[2] we used "gpt-3.5-turbo-0125" variant in our experiments

Table 3
Models used for submissions for each phase of the competition.

Type	**Phase 1 - Few-shot**	**Phase 2 - Zero-shot**
Baseline	AxCell	AxCell
LLMs	GPT35-zero GPT35-fil-zero	GPT35-fil-zero GPT35-few GPT35-fil-few GPT35-info-few GPT35-az-few GPT35-az-info-few Mistral-fil-zerp Mistral-fil-info-zero

Additional Information from Paper with Code (PwC): We use PwC as a knowledge base to collect lists of dataset and task names. These information are exhaustive lists of available datasets and task names, which we then use to check if any of these datasets or any of these tasks are mentioned in the full text of the paper. Although these lists contain all names of datasets and tasks of the test set (as it was not specified at the time of evaluation that PwC was used as a ground truth), they do not contain any information if a dataset or task was used in a specific paper, an information which can also be part of the training data of LLMs. We used especially PwC as it is one of the most frequently used platforms for the comparison of research results and provides an API to easily access the data.

After extracting the matching datasets and tasks, they are then appended to the input prompt as helping materials for the LLMs.
We construct our prompts to the LLMs from the following components: «*task description*», «*examples*», «*output format*», «*additional instructions*», «*input text*», and «*PwC information*». In «*task description*», we describe the task, inputs, and outputs. «*examples*» are provided only in the few-shot setting, showing input and expected output. In «*output format*», we describe the expected output format, which is a list of JSON objects covering the TDMS in the paper. «*additional instructions*» emphasize key points (e.g., Score should be a numerical value). «*input text*» is a placeholder for the paper text, either the full text or sections describing results and experiments. «*PwC information*» includes lists of datasets and tasks found in the paper using PwC. The prompts used in the experiments are in Appendix A.

4. Experiment and Results

In this section, we describe the experiments submitted during each phase of the competition using the models described in Section 3. The competition consists of two phases: *Few-shot Phase* (Phase 1) and *Zero-shot Phase* (Phase 2). We give information about the test datasets, experiments, and results for each phase.

4.1. Test datasets

We run the experiments in each phase on the test datasets provided by the competition organizers. This test dataset comprises the LaTeX source of articles from arXiv. The test data includes 1.401 articles for Phase 1 and 789 articles for Phase 2.

4.2. Experiments

In Table 3, we show the models used for experiments submitted in each phase of the competition (see Section 3). For both phases, we consider the AxCell system as a baseline model. We use two and nine LLM variations for Phase 1 and Phase 2, respectively.

Table 4
The Accuracy and Summary results of our Phase1 submissions

Model	Accuracy	Summary			
		Rouge 1	Rouge 2	Rouge L	Rouge Lsum
AxCell	**75.59**	**58.34**	**12.98**	**57.34**	**54.4**
GPT35-zero	**82.8**	**53.11**	**11.71**	**51.25**	**51.32**
GPT35-fil-zero	45.82	45.05	0.34	44.87	44.98

Table 5
The overall results of all TDMS for our Phase 1 submissions

Model	Exact			Inexact		
	P	R	F1	P	R	F1
AxCell	**44.02**	**13.9**	**21.13**	**47.11**	**14.88**	**22.62**
GPT35-zero	8.64	**11.43**	**9.84**	**12.38**	**16.18**	**14.02**
GPT35-fil-zero	5.57	0.26	0.5	7.31	0.34	0.66

Phase 1 Our aim in this phase is to compare the performance of the GPT-3.5 LLM in zero-shot settings against the AxCell baseline. We also consider the effect of filtering "unanswerable" papers (Section 3.1) to observe performance changes when reducing the number of papers processed by GPT-3.5.

Phase 2 In this phase, we compare the performance of LLM models in few-shot settings against zero-shot settings, specifically comparing **GPT35-fil-zero** with **GPT35-fil-few**. We also compare performance using the full paper text as input (i.e. **GPT35-few**) versus only sections referring to experiments and results (i.e. **GPT35-az-few**). Additionally, we compare different LLM models (i.e. **GPT35-fil-zero** vs. **Mistral-fil-zero**). Lastly, observe performance of the LLMs when providing external helping materials representing datasets and tasks (i.e **GPT35-info-few**, **GPT35-az-info-few**, and **Mistral-fil-info-zero**).

4.3. Results

In this section, we present the results of our submissions for each phase. The submissions are evaluated based on: (1) *Accuracy* – measures whether the system can distinguish articles containing TDMS, (2) *Summary* – measures quality of the extracted TDMS using *Rouge1*, *Rouge2*, *RougeL*, and *RougeLsum*, and (3) *precision*, *recall*, and *F1* – for each element in the TDMS (Task, Dataset, Metric, Score) tuples and the overall average[3]. In the following, for each phase, we report the performance of our submissions for *Accuracy*, *Summary* measures, and overall *Precision*, *Recall*, and *F1*.

Phase 1 In Table 4 and Table 5, we show the performance of our submissions for Phase 1. Values in **bold** are above the average for all submissions. The results in Table 4 show that the AxCell system outperforms GPT-3.5 submissions in all measures except *Accuracy*, indicating that GPT-3.5 better identifies papers containing TDMS. Generally, in Table 5, the *Precision*, *Recall*, and *F1* for GPT35-zero are lower than AxCell.

Phase 2 In Table 6 and Table 7, we show the performance of our submissions for Phase 2. Values in **bold** are above the average for all submissions. Similar to the previous phase, the AxCell system outperforms the LLM submissions in all measures except for some cases in *Accuracy*. With regard to our aimed experiments, we notice the following:

[3]Due to space, we report only the overall values, the full results can be found here: https://docs.google.com/spreadsheets/d/1k82FmlztEBiNkKuskAZsNaovkblHeqmpKzD5C63Mn5Q/

Table 6
The Accuracy and Summary results of our Phase 2 submissions

Model	Accuracy	Summary			
		Rouge 1	Rouge 2	Rouge L	Rouge Lsum
AxCell	**83.4**	**75.25**	**4.56**	**74.85**	**73.7**
GPT35-fil-zero	67.05	66.61	0.11	66.54	66.44
GPT35-few	**85.93**	**73.72**	**6.07**	**72.72**	**72.57**
GPT35-fil-few	69.07	68.81	0.14	68.76	68.66
GPT35-info-few	72.75	59.22	2.48	59.06	58.99
GPT35-az-few	**79.09**	**71.07**	**3.56**	**70.82**	**70.62**
GPT35-az-info-few	**75.41**	**71.59**	1.71	**71.46**	**71.35**
Mistral-fil-zero	**75.79**	**68.92**	2.18	**67.51**	**66.48**
Mistral-fil-info-zero	71.23	56.63	3.7	55.14	53.09

Table 7
The overall results of all TDMS for our Phase 2 submissions

Model	Exact			Inexact		
	P	R	F1	P	R	F1
AxCell	**36.36**	**6.21**	**10.6**	**40.85**	**6.97**	**11.9**
GPT35-fil-zero	2.63	0.15	0.29	3.08	0.18	0.34
GPT35-few	**12.82**	**9.89**	**11.16**	**16.74**	**12.81**	**14.52**
GPT35-fil-few	4.04	0.15	0.3	4.04	0.15	0.3
GPT35-info-few	4.46	2.24	2.99	6.82	3.43	4.56
GPT35-az-few	**11.78**	**3.77**	**5.71**	**18.16**	**5.7**	**8.68**
GPT35-az-info-few	**13.45**	**1.38**	2.5	**23.12**	**2.33**	4.23
Mistral-fil-zero	**8.05**	**4.42**	**5.71**	**10.74**	**5.89**	**7.61**
Mistral-fil-info-zero	**11.64**	**8.37**	**9.74**	**14.62**	**10.5**	**12.22**

1. **Zero-shot vs. Few-shot settings (GPT35-fil-zero vs. GPT3.5-fil-few):** The performance of GPT35-fil-few is better than GPT35-fil-zero, showing that providing examples helps the model detect TDMS. This is confirmed by comparing GPT35-few with AxCell, where GPT35-few is better than AxCell in most cases.

2. **Full text input vs. AZ input only (GPT35-few vs. GPT35-az-few):** Generally, the performance of the GPT35-few model better than that of GPT35-az-few except for the case of the Inexact *Precision*. The significance difference in the results need to be verified by further experiments.

3. **Providing the LLM with helpful material (GPT35-few vs. GPT35-info-few, GPT35-az-few vs. GPT35-az-info-few, and Mistral-fil-zero vs. Mistral-fil-info-zero):** Models given helpful materials about datasets and tasks (GPT35-info-few, GPT35-az-info-few, and Mistral-fil-info-zero) perform worse than their counterparts (GPT35-few, GPT35-az-few, and Mistral-fil-zero). We expect that the reason behind that is the helpful materials may be misleading to the models.

5. Discussion

In this section, we present a discussion through conducting a manual investigation to compare the output of the proposed models with each other and the ground truth annotation.

AxCell vs. GPT-3.5 The AxCell system generally outperforms the LLM submissions, particularly the GPT-3.5 submissions, in all metrics except for *Accuracy*. This suggests that GPT35-zero and GPT35-few are better at identifying papers containing TDMS.

Table 8
Sample output of TDMS extracted from Arxiv ID 2012.02818v2

Data source	Task-Dataset-Metric-Score
Ground Truth	Image Classification - CIFAR-10 - Percentage correct - 95.02 Image Classification - CIFAR-100 - Percentage correct - 76.85
AxCell	Image Classification - CIFAR-10 - Percentage error - 4.98 Image Classification - CIFAR-100 - Percentage error - 20.7 Semantic Segmentation - KITTI Semantic Segmentation - Mean IoU (class) - 89.08
GPT35-zero	Image classification - CIFAR-10, CIFAR-100 - Accuracy - 96.74% Image classification - CIFAR-10, CIFAR-100 - AUC - 0.9803 Stability analysis of Bayesian Neural Networks - CIFAR-10 - Epoch Divergence - None OOD detection - CIFAR-10, CIFAR-100 - ECE - 0.0520 Semantic segmentation - StreetHazards, BDD-Anomaly - mIoU - 56.12%

Through manual analysis, we identify common causes for TDMS extraction errors **GPT35-zero** and compare to **AxCell** and the ground truth data in Table 8. From the analysis, we found that **GPT35-zero** extracts a broad range of information but with inconsistencies and potential errors. For the "Task" entity, it predicts a mix of task names, some matching the ground truth and others deviating. As a result, further investigation is needed for the potential hallucination. For the "Dataset" entity, it sometimes combines multiple datasets despite correctly identifying individual ones. **GPT35-zero** also predicts unconventional "Metric" not present in the ground truth data, such as *Epoch Divergence*. The "Score" entity are mixed with percentages, raw scores, and string values.

While **AxCell** outperforms **GPT35-zero** on the "Task" and "Dataset" entities, showing consistency with the ground truth data, it struggles with accurately predicting the "Metric" entity. Both **AxCell** and **GPT35-zero** often predict different metric names, such as *Percentage error* and *Accuracy* instead of *Percentage correctness*. Additionally, **AxCell** predict "Score" entity differ from the ground truth data. Despite these drawbacks, it is evident the AxCell system could produce better results in the entity-level evaluation compared to the **GPT35-zero** model.

We noticed both model extracted new information of Tuple(s) respect to the ground truth data. Thus, we observe whether these tuples existed in the original paper or not. **AxCell** and **GPT35-zero** could accurately predict the task name mentioned in the paper, such as *Semantic Segmentation* along with the correct associated "Metric" name. However, for **AxCell**, the predicted "Dataset" name was incorrect, which might be due to the parsing error that require further investigation. On the other hand, **GPT35-zero**, could partially predict the correct "Dataset" name (*StreetHazard*). Nevertheless, the "Score" entity remained challenging for both models. Additionally, according to our observation, the annotated ground truth data is based on data collected from the results on PaperWithCode. As a result, we consider that further work needs to be done to expand the annotated ground truth data source.

GPT-3.5 vs. Mistral Our experiments revealed that the **Mistral-fil-zero** language model outperforms the GPT35-zero model across all evaluation metrics during Phase 2. Through manual analysis, we observed that **Mistral-fil-zero** could more effectively identify TDMS from the input text which present also in the ground truth data compared to **GPT35-fil-zero**. For this experiment, we utilized the similar prompt for both LLMs model. However, we noticed that in many cases, **GPT35-fil-zero** return "unanswerable" results, as presented in Table 9. We hypothesize that these cases, where **GPT35-fil-zero** failed to extract the TDMS, caused its overall score to be lower than **Mistral-fil-zero**.

From the analysis, **Mistral-fil-zero** accurately predicts the "dataset" entities according to the ground truth. However, inconsistencies arise with the "Task", "metric", and "Score" entities. Although **Mistral-fil-zero** can extract some "Score" entities similar to the ground truth, there were differences in its "Score" prediction compare to the ground truth. Potential hallucinations were observed, such as in the last row of the table where it extracted "Metric" and "Score" entities not found in the original paper.

Table 9
Sample output of TDMS extracted from various articles

Arxiv ID	Data source	Task-Dataset-Metric-Score
1903.06150v2	Ground Truth	Fine-Grained Image Classification - iNaturalist - Top 1 Accuracy - 68.2 Fine-Grained Image Classification - Stanford Cars - Accuracy - 93.8% Fine-Grained Image Classification - CUB-200-2011 - Accuracy - 87.9
	GPT35-fil-zero	unanswerable
	Mistral-fil-zero	Fine-grained image recognition - CUB-200-2011 - Recognition accuracy - 85.3
1902.10814v1	Ground Truth	Image Classification - iNaturalist - Top 1 Accuracy - 31.12% Image Classification - iNaturalist - Top 5 Accuracy - 52.76% Image Classification - ImageNet - Top 1 Accuracy - 68.29%
	GPT35-fil-zero	unanswerable
	Mistral-fil-zero	k-Nearest-Neighbor (kNN) search - ImageNet - Top-1 accuracy - 68.29 k-Nearest-Neighbor (kNN) search - ImageNet - Top-5 accuracy - 87.75 k-Nearest-Neighbor (kNN) search - iNaturalist - Top-5 accuracy - 52.76 k-Nearest-Neighbor (kNN) search - iNaturalist - Top-1 accuracy - 31.12 Triplet evaluation - PIT - Triplet accuracy (m=0.2) - 87.16

Table 10
Sample output of TDMS extracted from Arxiv ID 1711.05225v3

Source Data	Task - Dataset - Metric - Score
Ground Truth	Pneumonia Detection - ChestX-ray14 - AUROC - 0.844 Pneumonia Detection - ChestX-ray14 - Params - 7.0M Pneumonia Detection - ChestX-ray14 - FLOPS - 2800M
GPT35-fil-zero	Pneumonia detection from chest X-rays - ChestX-ray14 - F1 Score - 0.435 Pneumonia Detection - ChestX-ray14 - AUROC - 0.7680 Effusion Detection - ChestX-ray14 - F1 Score - 0.83 Pneumonia detection from frontal-view chest X-ray images - ChestX-ray14 - Accuracy - exceeding practicing radiologists
GPT35-fil-few	Pneumonia Detection - ChestX-ray14 - F1 Score - 0.435 Pathology Classification - ChestX-ray14 - AUROC - increase in AUROC Pneumonia Detection - ChestX-ray14 - Accuracy - exceeding practicing radiologists Multi-Disease Detection - ChestX-ray14 - Accuracy - outperforms previous state of the art

Further investigation is needed to understand this issue. Nevertheless, **Mistral-fil-zero** consistently predicts the "Task" and "Dataset" entities.

GPT-3.5 Zero-shot vs. Few-shot Settings The performance of **GPT35-fil-zero** and **GPT35-fil-few** models is comparable, as presented from Tables 6 and 7. While **GPT35-fil-few** model performs slightly better than **GPT35-fil-zero** in all evaluation metrics, further experiments might be needed to generalize the results. Table 10 presents a sample of the output generated by both models for the ArXiv ID 1711.05225v3, in comparison to the ground truth data.

We observe that both models extract a wide range of information, including some inconsistent and inaccurate information. This likely contributes to their lower performance, despite errors from the filtering process. For the "Task" entity, both models predict different task names, one of which matches the ground truth data. Both models could extract the "Dataset" and the "Metric" entity accurately. However, similar to the previous discussion, extracting the entity "Score" is challenging, often resulting in string values. Besides, we argue that the filtering process may mislead the model so that both models,

affecting their performance compared to other proposed models in Phase 2.

To sum up Extracting entities from the TDMS task is challenging, particularly for the "Score" and "Dataset" entities. The proposed models struggle with extracting the "Score" entity. This difficulty arises from the diverse formats authors use to present results, such as tables, graphs, or plain text. Additionally, the variability in naming conventions for the "Dataset" entity across papers poses a challenge, as noted by [2]. However, our models exhibit less variability in extracting the "Dataset" entity compared to the ground truth, which may use naming conventions not found in the papers. Further investigation is needed to understand the construction of the ground truth data. Consistent with [2], extracting the 'Task' entity is relatively more straightforward, as it is rarely referenced differently across papers addressing the task.

Overall, the AxCell system performance shows better performances in compared to the LLM submissions both GPT-3.5 and Mistral in all submission. However, the two variants of the GPT-3.5 model, GPT35-zero and GPT35-few, surpass AxCell in terms of *Accuracy*, indicating that these models are better at identifying papers containing TDMS information. While AxCell consistently extracts each entity of TDMS, particularly "Task", "Metric", and "Dataset", there was an error where the dataset name was not present in the publication.

This could be due to the underlying taxonomy, which maps the dataset to the best fitting dataset in this taxonomy. Additionally, we discovered after the deadline that there is a small overlap (around 5%) between AxCell's training dataset and the test data of the shared task, as both used PapersWithCode as ground-truth data. Such contamination are quite frequently nowadays, as many LLMs are not disclosing their training data, and therefore many baselines are already compromised [13, 14]. In the future, we plan to investigate how this bias affects results and explore ways to mitigate such contamination.

Moreover, Large Language Models (LLMs) show performance comparable to the AxCell system. Incorporating one or more examples (few-shot learning) into the prompt improves TDMS extraction quality. Nevertheless, the experiment heavily relies on prompts, which may influence the models' output during evaluation phases. This observation aligns with findings that the quality of outputs from conversational LLMs is directly influenced by the quality of the prompts provided by users [15]. Therefore, further investigation and refinement in the process of prompt engineering are essential.

6. Summary and Future Work

We have presented our approach to solve the SOTA challenge of the SimpleText shared task, composed of two modules: (1) applying rule-based classification to verify if a paper contains TDMS, and (2) extracting TDMS from papers with result information. We use the AxCell implementation as a baseline for TDMS extraction and experimented with GPT-3.5 and Mistral as LLMs in zero-shot and few-shot settings with different input information. The results show that AxCell outperforms LLMs when the zero-shot prompting paradigm is applied. LLMs on the other hand surpass AxCell's performance in few-shot settings. We conducted a manual investigation and showed that the LLM instructions can be misleading in the "Dataset" and "Score" extraction in zero-shot settings, with improvements seen in few-shot settings. We argue that the LLMs output are sensitive to the instructions given through the prompts.

Our discussion (Section 5) shows different direction of future work, including expanding the ground truth dataset with data from papers and investigating potential hallucination that may occur from the LLMs extraction. Additionally, our findings suggest that performance of the LLMs which are given the sections from the paper text referring to experiments and results only is comparable to those given the full paper text. Additionally, the results of LLMs using few-shot settings are comparable to, and sometimes better than, the AxCell system. The open-source Mistral model outperforms the GPT-3.5 model. To verify these assumptions, we plan to repeat the experiments and conduct statistical analysis.

References

[1] S. Kabongo, J. D'Souza, S. Auer, Automated mining of leaderboards for empirical AI research, in: H.-R. Ke, C. S. Lee, K. Sugiyama (Eds.), Towards Open and Trustworthy Digital Societies, Springer International Publishing, 2021, pp. 453–470. doi:10.1007/978-3-030-91669-5_35.

[2] S. Kabongo, J. D'Souza, S. Auer, ORKG-leaderboards: a systematic workflow for mining leaderboards as a knowledge graph, International Journal on Digital Libraries 25 (2023) 41–54. URL: https://doi.org/10.1007/s00799-023-00366-1. doi:10.1007/s00799-023-00366-1.

[3] S. Jain, M. van Zuylen, H. Hajishirzi, I. Beltagy, SciREX: A challenge dataset for document-level information extraction, in: D. Jurafsky, J. Chai, N. Schluter, J. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2020, pp. 7506–7516. URL: https://aclanthology.org/2020.acl-main.670. doi:10.18653/v1/2020.acl-main.670.

[4] L. Ermakova, E. SanJuan, S. Huet, H. Azarbonyad, G. M. D. Nunzio, F. Vezzani, J. D'Souza, J. Kamps, Overview of the CLEF 2024 simpletext track — improving access to scientific texts for everyone, in: L. Goeuriot, P. Mulhem, G. Quénot, D. Schwab, L. Soulier, G. M. D. Nunzio, P. Galuščáková, A. G. S. de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science (LNCS), Springer, Heidelberg, Germany, 2024.

[5] J. D'Souza, S. Kabongo, H. B. Giglou, Y. Zhang, Overview of the CLEF 2024 simpletext task 4: SOTA? tracking the state-of-the-art in scholarly publications, in: G. Faggioli, N. Ferro, P. Galuščáková, A. G. S. de Herrera (Eds.), Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum, CEUR Workshop Proceedings, CEUR-WS, Online, 2024.

[6] M. Kardas, P. Czapla, P. Stenetorp, S. Ruder, S. Riedel, R. Taylor, R. Stojnic, AxCell: Automatic extraction of results from machine learning papers, in: B. Webber, T. Cohn, Y. He, Y. Liu (Eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 8580–8594. URL: https://aclanthology.org/2020.emnlp-main.692. doi:10.18653/v1/2020.emnlp-main.692.

[7] I. Augenstein, M. Das, S. Riedel, L. Vikraman, A. McCallum, SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications, in: S. Bethard, M. Carpuat, M. Apidianaki, S. M. Mohammad, D. Cer, D. Jurgens (Eds.), Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 546–555. URL: https://aclanthology.org/S17-2091. doi:10.18653/v1/S17-2091.

[8] K. Gábor, D. Buscaldi, A.-K. Schumann, B. QasemiZadeh, H. Zargayouna, T. Charnois, SemEval-2018 task 7: Semantic relation extraction and classification in scientific papers, in: M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, M. Carpuat (Eds.), Proceedings of the 12th International Workshop on Semantic Evaluation, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 679–688. URL: https://aclanthology.org/S18-1111. doi:10.18653/v1/S18-1111.

[9] Y. Hou, C. Jochim, M. Gleize, F. Bonin, D. Ganguly, Identification of tasks, datasets, evaluation metrics, and numeric scores for scientific leaderboards construction, Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019) 5203–5213. URL: https://www.aclweb.org/anthology/P19-1513. doi:10.18653/v1/P19-1513, conference Name: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics Place: Florence, Italy Publisher: Association for Computational Linguistics.

[10] S. Kabongo, J. D'Souza, S. Auer, Zero-shot entailment of leaderboards for empirical AI research, in: 2023 ACM/IEEE Joint Conference on Digital Libraries (JCDL), IEEE, 2023, pp. 237–241. URL: https://ieeexplore.ieee.org/document/10265895/. doi:10.1109/JCDL57899.2023.00042.

[11] S. Yang, C. Tensmeyer, C. Wigington, IN: Table entity LINker for extracting leaderboards from machine learning publications, in: T. Ghosal, S. Blanco-Cuaresma, A. Accomazzi, R. M. Patton, F. Grezes, T. Allen (Eds.), Proceedings of the first Workshop on Information Extraction from

Scientific Publications, Association for Computational Linguistics, 2022, pp. 20–25. URL: https://aclanthology.org/2022.wiesp-1.3.

[12] S. Teufel, et al., Argumentative zoning: Information extraction from scientific text, Ph.D. thesis, Citeseer, 1999.

[13] S. Balloccu, P. Schmidtová, M. Lango, O. Dusek, Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source LLMs, in: Y. Graham, M. Purver (Eds.), Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, St. Julian's, Malta, 2024, pp. 67–93. URL: https://aclanthology.org/2024.eacl-long.5.

[14] O. Sainz, J. Campos, I. García-Ferrero, J. Etxaniz, O. L. de Lacalle, E. Agirre, NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark, in: H. Bouamor, J. Pino, K. Bali (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2023, Association for Computational Linguistics, Singapore, 2023, pp. 10776–10787. URL: https://aclanthology.org/2023.findings-emnlp.722. doi:10.18653/v1/2023.findings-emnlp.722.

[15] X. Ma, J. Li, M. Zhang, Chain of thought with explicit evidence reasoning for few-shot relation extraction, 2024. arXiv:2311.05922.

A. Prompts for LLMs

A.1. Zero-Shot Prompt

```
"""
You are an expert in machine learning who can identify if a research paper contains
     certain key elements related to tasks, datasets, metrics, and scores.
    Specifically, you need to look for the following:
Task: A phrase describing the research problem or focus, often found in the title,
    abstract, introduction, or results tables/discussion.
Dataset: A mention of the dataset(s) used for the machine learning experiments,
    usually located near the task mentions.
Metric: Phrases referring to the evaluation measures used to assess the models'
    performance on the given task and dataset. These are commonly found in results
    tables/figures and the discussion section.
Score: The numeric value(s) representing the model's performance on a specific
    metric. Multiple scores may be reported for a single metric, in which case the
    best score should be identified.

Your task is to determine if the given scholarly article contains mentions of all
    four elements (task, dataset, metric, score) together as a tuple. If the same
    task is evaluated on multiple datasets or with multiple metrics, each
    combination should be represented as a separate JSON object within the array, as
     long as the task, dataset, and metric are related to each other.


<< FORMATTING >>
Answer in the form of a list of JSON objects as follows.

The output should be a markdown code snippet formatted as a list of JSON objects in
     the following schema, including the leading and trailing "```json" and "```":

```json
[
  {
    "Task": string  // Extract the research problem or focus mentioned in the paper
        . Use '' if not available.
    "Dataset": string  // Extract the dataset(s) used for the machine learning
        experiments. Use '' if not available.
    "Metric": string  // Extract the evaluation measure(s) used to assess the
        models' performance. Use '' if not available.
    "Score": string  // Extract the best numeric value(s) representing the model's
        performance on a specific metric. Use '' if not available.
  }
]
```

Additional formatting instructions:
- Provide the output as a list of valid JSON objects, using the above description.
- If a value is not available or applicable, return an empty list.
- Make sure the JSON output is properly formatted.
- Example 'Task': 'Keyword Spotting', 'Dataset': 'hey Siri', 'Metric': 'Error Rate
    ', 'Score': '0.45%'
- Don't extract text or information in the 'Score' field. Extract only the numeric
    values that indicate the 'Score'.
- Be strict when returning the score. Only include numeric values.
- Do not combine multiple datasets or metrics or scores for the same task into a
    single JSON object.
<< INPUT >>
{input_content}

<< OUTPUT >>
"""
```

A.2. Few-Shot Prompt

```
"""
You are an expert in machine learning who can identify if a research paper contains
     certain key elements related to tasks, datasets, metrics, and scores.
    Specifically, you need to look for the following:
Task: A phrase describing the research problem or focus, often found in the title,
    abstract, introduction, or results tables/discussion.
Dataset: A mention of the dataset(s) used for the machine learning experiments,
    usually located near the task mentions.
Metric: Phrases referring to the evaluation measures used to assess the models'
    performance on the given task and dataset. These are commonly found in results
    tables/figures and the discussion section.
Score: The numeric value(s) representing the model's performance on a specific
    metric. Multiple scores may be reported for a single metric, in which case the
    best score should be identified.

Here are a few examples of the input and expected output format:
<< Example 1 >>
Input: Streaming keyword spotting is a widely used solution for activating voice
    assistants. We apply our method for 'hey Siri' detection. Compared to the best
    of the two prior works, our method reduces the FRR from 1.7% to 0.45%, which
    yields about 73% relative FRR improvement.
Expected Output Format:
[
  {{
    "task": "Keyword Spotting",
    "dataset": "Hey Siri",
    "metric": "Error Rate",
    "score": "0.45"
  }}
]

<< Example 2 >>
Input: In this paper, we present the Lbl2Vec approach, which provides the retrieval
     of documents on predefined topics from a large corpus based on unsupervised
    learning. We use the two publicly available classification datasets, 20
    Newsgroups and AG's Corpus, described in Table tab:datasets. Compared with an
    unsupervised classification baseline, we increased F1 scores from 76.6 to 82.7
    and from 61.0 to 75.1 on the respective datasets.
Expected Output Format:
[
  {{
    "Task": "Unsupervised Text Classification",
    "Dataset": "AG News",
    "Metric": "F1 score",
    "Score": "82.7"
  }},
  {{
    "Task": "Unsupervised Text Classification",
    "Dataset": "20NewsGroups",
    "Metric": "F1 score",
    "Score": "75.1"
  }}
]

<< Example 3 >>
Input: This paper is concerned with the form of typed name binding used by the
    FreshML family of languages. Its characteristic feature is that a name binding
    is represented by an abstract (name,value)-pair that may only be deconstructed
    via the generation of fresh bound names. The paper proves a new result about
    what operations on names can co-exist with this construct. In FreshML the only
    observation one can make of names is to test whether or not they are equal.
    This restricted amount of observation was thought necessary to ensure that there
```

```
    is no observable difference between alpha-equivalent name binders. Yet from an
    algorithmic point of view it would be desirable to allow other operations and
    relations on names, such as a total ordering. This paper shows that, contrary
    to expectations, one may add not just ordering, but almost any relation or
    numerical function on names without disturbing the fundamental correctness
    result about this form of typed name binding (that object-level alpha-
    equivalence precisely corresponds to contextual equivalence at the programming
    meta-level), so long as one takes the state of dynamically created names into
    account.
Expected Output Format:
[]

<< FORMATTING >>
Answer in the form of a list of JSON objects as follows. The output should be a
    markdown code snippet formatted as a list of JSON objects in the following
    schema, including the leading and trailing "json" and "":
[
  {
    'Task': string // Extract the research problem or focus mentioned in the paper.
        Use '' if not available.
    'Dataset': string // Extract the dataset(s) used for the machine learning
        experiments. Use '' if not available.
    'Metric': string // Extract the evaluation measure(s) used to assess the models
        ' performance. Use '' if not available.
    'Score': string // Extract the best numeric value(s) representing the model's
        performance on a specific metric. Use '' if not available.
  }
]

Additional formatting instructions:
Provide the output as a list of valid JSON objects, using the above description.
If a value is not available or applicable, return an empty list.
Make sure the JSON output is properly formatted.
Don't extract text or information in the 'Score' field. Extract only the numeric
    values that indicate the 'Score'.
Be strict when returning the score. Only include numeric values.
Do not combine multiple datasets or metrics or scores for the same task into a
    single JSON object.

Now, given a new scholarly article, your task is to determine if it contains
    mentions of all four elements (task, dataset, metric, score) together. If the
    same task is evaluated on multiple datasets or with multiple metrics, each
    combination should be represented as a separate JSON object within the array, as
    long as the task, dataset, and metric are related to each other. Generate your
    own output based on the input text, using the examples only as a reference for
    the desired format:

<< INPUT >>
{input_content}
<< OUTPUT >>
"""
```

A.3. Few-Shot Prompt with data from PwC

```
"""
You are an expert in machine learning who can identify if a research paper contains
    certain key elements related to tasks, datasets, metrics, and scores.
    Specifically, you need to look for the following:
Task: A phrase describing the research problem or focus, often found in the title,
    abstract, introduction, or results tables/discussion.
Dataset: A mention of the dataset(s) used for the machine learning experiments,
    usually located near the task mentions.
```

```
Metric: Phrases referring to the evaluation measures used to assess the models'
    performance on the given task and dataset. These are commonly found in results
    tables/figures and the discussion section.
Score: The numeric value(s) representing the model's performance on a specific
    metric. Multiple scores may be reported for a single metric, in which case the
    best score should be identified.

Here are a few examples of the input and expected output format:
<< Example 1 >>
Input: Streaming keyword spotting is a widely used solution for activating voice
    assistants. We apply our method for 'hey Siri' detection. Compared to the best
    of the two prior works, our method reduces the FRR from 1.7% to 0.45%, which
    yields about 73% relative FRR improvement.
Expected Output Format:
[
  {{
    "task": "Keyword Spotting",
    "dataset": "Hey Siri",
    "metric": "Error Rate",
    "score": "0.45"
  }}
]

<< Example 2 >>
Input: In this paper, we present the Lbl2Vec approach, which provides the retrieval
     of documents on predefined topics from a large corpus based on unsupervised
    learning. We use the two publicly available classification datasets, 20
    Newsgroups and AG's Corpus, described in Table tab:datasets. Compared with an
    unsupervised classification baseline, we increased F1 scores from 76.6 to 82.7
    and from 61.0 to 75.1 on the respective datasets.
Expected Output Format:
[
  {{
    "Task": "Unsupervised Text Classification",
    "Dataset": "AG News",
    "Metric": "F1 score",
    "Score": "82.7"
  }},
  {{
    "Task": "Unsupervised Text Classification",
    "Dataset": "20NewsGroups",
    "Metric": "F1 score",
    "Score": "75.1"
  }}
]

<< Example 3 >>
Input: This paper is concerned with the form of typed name binding used by the
    FreshML family of languages. Its characteristic feature is that a name binding
    is represented by an abstract (name,value)-pair that may only be deconstructed
    via the generation of fresh bound names.  The paper proves a new result about
    what operations on names can co-exist with this construct.  In FreshML the only
    observation one can make of names is to test whether or not they are equal.
    This restricted amount of observation was thought necessary to ensure that there
     is no observable difference between alpha-equivalent name binders.  Yet from an
     algorithmic point of view it would be desirable to allow other operations and
    relations on names, such as a total ordering.  This paper shows that, contrary
    to expectations, one may add not just ordering, but almost any relation or
    numerical function on names without disturbing the fundamental correctness
    result about this form of typed name binding (that object-level alpha-
    equivalence precisely corresponds to contextual equivalence at the programming
    meta-level), so long as one takes the state of dynamically created names into
    account.
Expected Output Format:
```

```
[]

<< FORMATTING >>
Answer in the form of a list of JSON objects as follows. The output should be a
    markdown code snippet formatted as a list of JSON objects in the following
    schema, including the leading and trailing "json" and "":
[
  {
    'Task': string // Extract the research problem or focus mentioned in the paper.
        Use '' if not available.
    'Dataset': string // Extract the dataset(s) used for the machine learning
        experiments. Use '' if not available.
    'Metric': string // Extract the evaluation measure(s) used to assess the models
        ' performance. Use '' if not available.
    'Score': string // Extract the best numeric value(s) representing the model's
        performance on a specific metric. Use '' if not available.
  }
]

Additional formatting instructions:
Provide the output as a list of valid JSON objects, using the above description.
If a value is not available or applicable, return an empty list.
Make sure the JSON output is properly formatted.
Don't extract text or information in the 'Score' field. Extract only the numeric
    values that indicate the 'Score'.
Be strict when returning the score. Only include numeric values.
Do not combine multiple datasets or metrics or scores for the same task into a
    single JSON object.

Now, given: (1) a new scholarly article, (2) list of data sets, and (3) list of
    tasks that we manually identified also in the article as helping materials to
    you. Your task is to determine if the article contains mentions of all four
    elements (task, dataset, metric, score) together using the lists of the datasets
     and the task. You can also extract other datasets and tasks outside the given
    lists. If the same task is evaluated on multiple datasets or with multiple
    metrics, each combination should be represented as a separate JSON object within
     the array, as long as the task, dataset, and metric are related to each other.
    Generate your own output based on the input text, using the examples only as a
    reference for the desired format:

<< INPUT >>
scholarly article:
{input_tex_content}

datasets list:
{dataset_list}

tasks list:
{tasks_list}

<< OUTPUT >>
"""
```