THM@SimpleText 2026 - Task 2: Ensemble-based Hallucination Classification in Text Simplification

Julian Dauenhauer[1,†], Nico Hofmann[1,†] and Christin Katharina Kreutz[1,2,*]

[1]TH Mittelhessen - University of Applied Sciences, Gießen, Germany
[2]Herder Institute, Marburg, Germany

Abstract

Task 2 of the SimpleText CLEF Lab tackled controlled creativity with the goal of identifying overgeneration in text simplification on a sentence- and document-level in a binary (Task 2.1) and multi-class (Task 2.2) setting. Our approach consists of an ensemble of BERT classifiers, Random Forests and an LLM-as-a-judge component in a multi-layered pipeline. We submit seven runs for both sub-task, so 14 in total.

Keywords

Overgeneration, Classification, Random Forest, BERT, LLM

1. Introduction

Simplification of scientific content is desirable as it enables non-experts to understand reliable peer-reviewed information and thus tackle dis- and misinformation [1] as well as make informed medical decisions [2].

Current developments in computer technology such as large language models (LLMs) enabled the development of effective automatic text simplification approaches [3, 4]. But usage of LLMs comes with a number of drawbacks, one of them are hallucinations [5].

In the previous year's SimpleText lab around 30% of the submissions show a considerable amount of spurious sentences in produced simplifications [1] which might stem from most teams having used LLMs for producing their simplifications [6].

To tackle this development, the SimpleText lab [1, 7] offers a shared task on *Controlled Creativity: Identify and Avoid Hallucination* as Task 2 [8]. Here, approaches are to be developed for binary (Task 2.1) and multi-class (6 classes, Task 2.2) classification to determine, if overgeneration occurred while biomedical abstracts were automatically simplified. This encompasses cases where the simplification introduces content that goes beyond, contradicts, or corrupts the source.

Our submission to the tasks encompasses 14 runs, seven to both sub-tasks 2.1 and 2.2. Our approach consists of multiple steps and is inspired by previous year's participants [9, 10, 11]. We employ an ensemble of different BERT classifierts, Random Forests and an LLM-as-a-judge component.

2. Related Work

Task 2 of SimpleText@CLEF'25 [12] tackled the identification of hallucinations in simplified text on a sentence-level already. In this shared task, the best-performing runs for detection overgeneration (Task 2.1) [12] were submitted by teams SINAI [11] and DS@GT [9]. SINAI [11] conducted a data analysis as a basis for their threshold selection in a rule-based approach combined with a LLaMA model. This submission informed our sentence-length-based statistical features and stripping thresholds. DS@GT [9]

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany
*Corresponding author.
†These authors contributed equally.
ckreutz@acm.org (C. K. Kreutz)
https://kreutzch.github.io (C. K. Kreutz)
0000-0002-5075-7699 (C. K. Kreutz)

	train	dev	test
abstracts	21,057	1,120	6,728
sentences	350,583	18,638	105,001

Table 1
Number of abstracts and sentences in `train`, `dev` and `test` dataset.

	Non-Spur	Spur	Failure	Leaked	Injection	Repetitive	Overgen
`dev`	91.58 (23.7)	8.42 (104.3)	3.61 (72.89)	2.54 (199.3)	1.77 (47.29)	0.50 (51.11)	0.02 (66.67)
`train`	92.33 (23.87)	7.67 (100.85)	3.23 (69.35)	2.35 (189.36)	1.64 (50.13)	0.42 (49.3)	0.03 (49.15)

Table 2
Percentage of '**Non-Spur**ious' and '**Spur**ious' entries with labels 'generation **Failure**', '**Leaked** instructions', 'ungrounded **Injection**', '**Repetitive** content', 'grounded **Overgen**eration' with their respective (average sentence lengths in characters).

designed a multi-layer detection system with LLM-as-a-judge. This submission directly inspired our BERT council structure and our ensemble stacking.

The best performing run for the overgeneration detection task with the variation of also utilizing pairs of source-prediction content [12] was submitted by team AIIRLAB [10]. AIIRLAB [10] combined LLM-based majority voting with a Random Forest classifier. This submission inspired our two-stage Random Forest verdict architecture.

3. Dataset and Data Analysis

3.1. Dataset

The SimpleText organisers [7, 8] provide a `train`, `dev` and `test` dataset for Task 2, targeting simplification of biomedical abstracts from the Cochrane Library. For all source biomedical abstracts the datasets contain pairs of automatic simplifications of the different sentences and corresponding labels indicating if overgeneration has occurred in the simplification. The simplification resulted e.g., from usage of GPT, Gemini, LLaMA, T5, or BART and the labels have been automatically assigned to these simplifications [8]. If overgenetation has not been detected, sentences are assigned a '`None`' label. Otherwise, the overgeneration is of type '`Generation failure`' if a catastrophic output was produced either by degeneration, truncation or loops in input and output. If task framing, assistant chatter or simplification rationale is detected in the simplification it is labeled '`Leaked instructions`'. In case of unsupported factual or interpretive additions to the content, the simplification is classified as '`Ungrounded injection`'. '`Repetitive content`' indicated repetition loops and lastly, '`Grounded overgeneration`' is assigned in cases where the simplification contains source-supported content beyond the simplification scope. In the binary scenario of Task 2.1 there is only the differentiation between no overgeneration ('`None`') and if there was any overgeneration present. The multi-class scenario of Task 2.2 considers differentiation between all six classes.

Table 1 contains the number of documents and sentences in the train, dev and test datasets.

3.2. Data Analysis

In the SimpleText@CLEF'25 lab [13, 14] team SINAI [11] detected strong correlations between sentence structure and classification, therefore we also consider the properties of the dataset.

Table 2 holds the percentage and lengths of texts of respective classes for the *dev* and *train* sentence datasets and Figure 1 shows the probability of classes dependent on simplified sentence length in characters in the `train` dataset. The `dev` and `train` datasets are well balanced and show little difference. Spurious simplified sentences are on average at least twice as long as non-spurious ones and leaked instructions are the most spurious class. They contains by far the longest average simplified sentences. The scatterplotFigure 1 shows an increase in the probability of spurious generation with

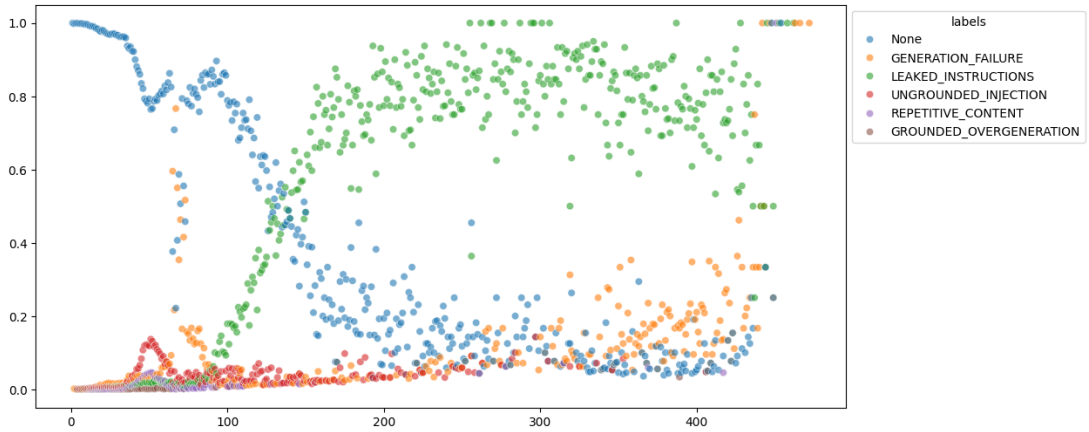

Figure 1: Probability of classes (y axis) per length of simplified sentences (in characters, x axis) in `train`.

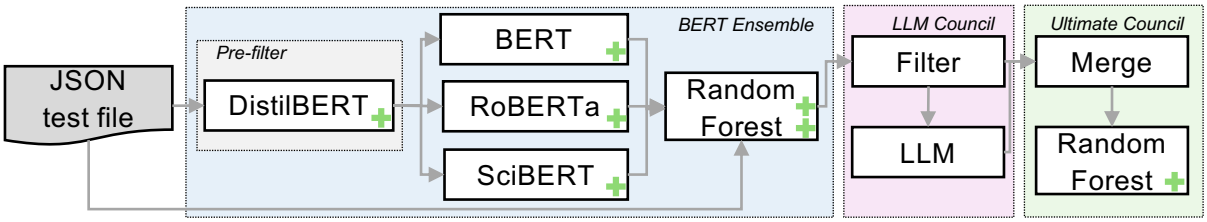

Figure 2: Overview of our ensemble approach. Green plus symbols mark the points that produce runs.

longer sentence length while non-spurious sentences become less common with longer simplified sentences.

4. Approach

Overall, we combine multiple ideas from the previous year's submissions [9, 10, 11] into one unified multi-step ensemble classification approach with the goal of the full multi-class classification. Figure 2 gives an overview of our approach and indicates the ablations leading to our produced runs.

4.1. BERT Ensemble

DS@GT's [9] promising results in last year's iteration of the shared task [12] inspired us to also develop an ensemble-based approach. We selected the following four BERT models: **RoBERTa** [15], which was also used in the previous year and proved stable for multi-label classification for team DS@GT [9], **DistilBERT** [16] due to the model being fast and lightweight, making it suitable in a pre-filter step, **BERT** [17] as a reference model and **SciBERT** [18] due to its pre-training focus on scientific texts, which matches the biomedical domain of the Cochrane source abstracts. We classify with 6 class-labels: `'None'`, `'Generation failure'`, `'Leaked instructions'`, `'Ungrounded injection'`, `'Repetitive content'` and `'Grounded overgeneration'`.

First, a pre-filter step in the classification consists of employing a DistilBERT model on all entries from the `test` dataset. With the output from this classification, the `test` dataset is stripped of all sentences where the `'None'` probability from DistilBERT exceeds 0.9, leading to a dramatically reduced dataset size passed on to subsequent models.

The pre-filtered dataset without entries receiving a `'None'` label with high confidence is given to BERT, RoBERTa and SciBERT for another round of classification. Afterwards, the filtered classifications and the unfiltered DistilBERT classifications are merged back into a single coherent file containing all classification labels produced by the four BERT variants. Entries skipped by the later BERT models

```
"Factuality_Check": (
    "Look at this source: '{src}'. "
    "Does the simplification '{simp}' hallucinate new facts? Answer YES or
       NO.",
),
"Error_Detection": (
    "Analyze the simplification '{simp}' against the source '{src}'.\n"
    "Identify errors from this taxonomy (and only use these):\n"
    "- GENERATION_FAILURE\n"
    "- LEAKED_INSTRUCTIONS\n"
    "- UNGROUNDED_INJECTION\n"
    "- REPETITIVE_CONTENT\n"
    "- GROUNDED_OVERGENERATION\n"
    "If clean, state 'NO ERRORS'.",
),
```

Table 3
Prompt assessing factuality and classification of error type.

receive default probability values of 0.0 for all labels. A Random Forest classifier produces a joint BERT ensemble prediction. Additionally, an alternative Random Forest classifier is run in this case with sentence lengths[1] from the `test` dataset for all entries combined with the produced coherent file of classification labels.

4.2. LLM Council

The LLM council consists of an LLM-as-a-judge component that is presented each entry where the BERT ensemble did not agree on which type of error occurred in the simplification.

The classification results from the first Random Forest are filtered again. Entries surpassing a threshold of 0.9 for the label ‘None‘ assigned by the Random Forest or coming with an unanimous BERT agreement threshold of 0.6 to the same error class with the Random Forest are considered agreed upon and thus disregarded. Only ambiguous cases, entries with different assigned classes or low certainty, are then given to the LLM council. The LLM[2], `GPT-4.1-nano` in our case, is tasked to handle these entries. Table 3 contains the prompt to assess factuality and error-types of ambiguous entries.

In the LLM's responses, entries with placeholders (`"empty, due to none"`) are assigned ‘None‘ labels.

4.3. Ultimate Council

In the last step, all BERT and Random Forest results are merged into a single rich record per entry containing all information. Here, the BERT outputs are compared with the LLM responses to highlight disagreements. A conflict score is computed per sentence according to the following rules:

- $+0.6$ if the LLM indicated a class that BERT assigned low probability (missed detection).
- $+0.2$ if BERT assigned a probability > 0.3 to a class that the LLM did not flag.
- $+0.4$ if the LLM indicated that the sentence is clean while BERT indicated at least one error class.

Table 4 contains a sample entry after the merging of information from LLM and filter.

Sentences that were unambiguous before the LLM stage and ones with conflict score < 0.5 are skipped in the following step. In case of an entry with conflict score ≥ 0.5, all probability scores from

[1]Absolute sentence length in characters, the length ratio between source and simplified sentence, and a binary flag indicating if the simplification contains < 6 words.
[2]The LLM is called for batches of 10 ambiguous entries to reduce the number of API calls.

```
"snt_id":                          "test_00002_1",
"predicted_label":                 "UNGROUNDED_INJECTION",
"None_prob":                       0.0996,
"UNGROUNDED_INJECTION_prob":       0.5052,
"GROUNDED_OVERGENERATION_prob":    0.2809,
"bert_distilbert":       {"None":                    0.9962,
                          "UNGROUNDED_INJECTION":    0.0034},
"bert_bert":             {"None":                    0.9210,
                          "UNGROUNDED_INJECTION":    0.0771},
"bert_roberta":          {"None":                    0.9906,
                          "UNGROUNDED_INJECTION":    0.0075},
"bert_scibert":          {"None":                    0.8955,
                          "UNGROUNDED_INJECTION":    0.1011},
"llm_chatgpt_error_detection":   "UNGROUNDED_INJECTION",
"llm_chatgpt_factuality_check":  "NO",
"conflict_score":                0.0,
"final_decision":                "VALIDATED"
```


Table 4
Sample entry after LLM call with final decision ‘Validated‘. The first Random Forest ("predicted_label") and LLM ("llm_chatgpt_error_detection") reached the same conclusion independently, resulting in a conflict score of 0.0 and a VALIDATED decision. The meta-ensemble assigned UNGROUNDED_INJECTION with a probability of 0.51, and GPT returned UNGROUNDED_INJECTION as well. All BERT models assigned high ‘None‘ probabilities. The Random Forest classification diverges from every individual model’s top prediction. This is the primary motivation for the pipeline: the Random Forest learned that a combination of moderate non-‘None‘ probabilities across several models can outweigh each model’s individual top-1 prediction, particularly for subtle injection cases where no single model is highly confident but the aggregate signal is informative.

the BERT ensemble, the LLM responses and the conflict score are used as input for a second Random Forest classifier to compute the final class label.

The final labels for all input entries stem from DistilBERT in cases where the ‘None‘ class was assigned with high probability. Otherwise the BERT Ensemble’s unanimous decision on the class label was assigned. If the BERT Ensemble did not reach a consensus, the label stems from the LLM Council, if it did not produce a high conflict score. In case a high conflict score was found, a Random Forest classifier finally decided on the class label.

4.4. Implementation

Our code is available via GitHub[3]. In our implementation we used the following BERT models:

- **RoBERTa**: `roberta-base`
- **DistilBERT**: `distilbert-base-uncased`
- **BERT**: `bert-base-uncased`
- **SciBERT**: `allenai/scibert_scivocab_uncased`

All BERT models were fine-tuned for sequence classification using the `dev` dataset. Input sequences were formed by concatenating source abstract and simplified sentence with a `[SEP]` token delimiter, truncated to a maximum of 256 tokens.

Random Forests are trained on the `dev` data. The Random Forest in Ultimate Council is trained on a reviewed subset of sentences stemming from the `dev` dataset where sentences labelled as ‘Skipped‘ are excluded from training and reattached afterwards with zeroed feature values.

[3] https://github.com/kreutzch/THM-SimpleText-26

None	Failure	Leaked	Injection	Repetitive	Overgen
99,950	1,830	1,580	1,281	360	0

Table 5
Number of entries assigned to classes '**None**', 'Generation **Failure**', '**Leaked** instructions', 'Ungrounded **Injection**', '**Repetitive** content', 'Grounded **Overgen**eration' with our full approach.

5. Runs and Results

5.1. Production of Runs

Out of all sentences (105,001) in the `test` set, 95.6% (100,383) never reached the LLM in the LLM Council as they were not considered ambiguous (the probability for the '`None`' class was > 0.9 with DistilBERT or all four BERT models unanimously assigned the same error class with probability > 0.6).

4.1% of all sentences (4293) were reviewed by the LLM and produced a conflict score < 0.5 indicating that BERT models and LLM generally agree, and the ensemble label is accepted as the predicted label.

0.3% of all sentences (325) produced a conflict score of ≥ 0.5 after review by the LLM, indicating a considerable disagreement between BERT and LLM, requiring another pass through the final Random Forest. Specifically a conflict score of 0.6 is added if the LLM assigns a category that BERT models assigned a low probability to.

Table 5 holds the number of entries assigned with the respective classes. The label distribution of the full approach closely mirrors the proportions found in the `dev` set: $\frac{99950}{105001} = 95.2\%$ '`None`', compared to 91.6% in the `dev` set. The slight upward skew towards the '`None`' class is a direct consequence of the DistilBERT pre-filter as sentences removed at the 0.9 threshold are assigned this label by default and inflate count regardless of their true label. '`Grounded overgeneration`' received zero predictions which is consistent with it representing only 0.03% of the `dev` entries and being the label most likely to be removed by the stripping passes given its subtle signal. Of the 6,728 `test` documents, we labeled 1,282 documents (19.1%) as containing at least one error class.

5.2. Submitted Runs

Our runs come with the team prefix `THM` as well as an indication of the task tackled (e.g., `Task21b`), therefore our run submissions are named **THM_Task21b_UltimateCouncil** for instance. We submit seven runs for both Tasks 2.1 and 2.2, so 14 in total. As our approach tackles the multi-class classification scenario, for runs submitted to Task 2.1, the binary setting indicating if overgeneration occurred or not, we replace all overgeneration class-specific labels with a label '`Overgeneration`'.

- `BertOnly_distilbert`: Output file is produced from DistilBERT predictions.
- `BertOnly_bert`: Output file is produced from BERT predictions combined with '`None`' labels with high certainty produced by DistilBERT.
- `BertOnly_roberta`: Output file is produced from RoBERTa predictions combined with '`None`' labels with high certainty produced by DistilBERT.
- `BertOnly_scibert`: Output file is produced from SciBERT predictions combined with '`None`' labels with high certainty produced by DistilBERT.
- `BertOnly_RF`: Output file is produced after only BERT Ensemble including Random Forest is run.
- `BertOnly_RF_2`: Output file is produced after only BERT Ensemble including Random Forest is run. The Random Forest considers length features of simplified sentences.
- `UltimateCouncil`: Output file is produced after the full approach, all BERT models, Random Forest without consideration of sentence lengths, GPT call and the final Random Forest decisions.

	Task 2.1						Task 2.2
	Document-level			Sentence-level			
Configuration	**F1**	**P**	**R**	**F1**	**P**	**R**	**A**
`BertOnly_distilbert`	0.7164	0.7318	0.7016	**0.8151**	**0.816**	**0.8142**	**0.7388**
`BertOnly_bert`	0.738	0.7889	0.6933	0.5902	0.6038	0.5772	0.515
`BertOnly_roberta`	**0.7422**	0.8256	0.674	0.6011	0.6244	0.5794	0.5156
`BertOnly_scibert`	0.7287	**0.8333**	0.6474	0.6031	0.6345	0.5747	0.5133
`BertOnly_RF`	0.7278	0.7067	**0.7502**	0.5699	0.5559	0.5846	0.5211
`BertOnly_RF_2`	0.6788	0.8176	0.5803	0.6617	0.6971	0.6298	0.5612
`UltimateCouncil`	0.709	0.7204	0.6979	0.5802	0.5889	0.5717	0.5145

Table 6
F1, **P**recision and **R**ecall in the binary setting (Task 2.1) on document and sentence-level of submitted runs as well as **A**ccuracy in the multiclass-setting (Task 2.2). Scores produced by Codabench.

5.3. Evaluation

Table 6 contains document- and sentence-level results for our submitted runs from Codabench for the binary (Task 2.1) and multi-class (Task 2.2) tasks. Not one single configuration performs well across all settings with `BertOnly_distilbert` achieving the highest results (Accuracy) for the multi-class as well as the sentence-level setting (F1, Precision, Recall). Our `UltimateCouncil` configuration produces considerably worse results than usage of BERT models alone (`BertOnly_roberta`, `BertOnly_bert` and `BertOnly_scibert`) for Task 2.1 in F1 and Precision on document and sentence level. Only in Recall this configuration was able to surpass some of the single BERT variants. `BertOnly_RF_2` achieves better results than `BertOnly_RF` in the sentence-level binary class and multi-class setting. This highlights the usefulness of features indicating the sentence length of simplifications. As we already saw in subsection 3.2 and Figure 1, considerably longer sentences usually indicate overgeneration.

In all of our configurations distilBERT acts as a filter. Other BERT models were run on the distilBERT-filtered test data, so distilBERT dictated all `‘None‘` labels, meaning all other models had to operate on the the pre-filtered labels. This means, in cases where distilBERT falsely flagged an entry as not spurious, the entry would not be considered by subsequent models.

Our BERT models were only trained on `dev` data to reduce the computational load compared to training on `train` data which potentially heavily impacted our approach's results.

In our approach sentences confidently labeled as `‘None‘` by distilBERT were removed from test data to reduce computational load. This turned out to be a meaningless optimization, as the evaluation speed of BERT-models is quite high compared to their training. Running the filtered data through the `dev`-trained BERT models took around 5 minutes each. With the unfiltered `test` set, the time would likely not have exceeded 20 minutes per BERT model.

6. Conclusion

Our approach combined an ensemble of different BERT models with a Random Forest classifier, and an LLM verification layer and a second Random Forest classifier into a multi-step pipeline for overgeneration classification in simplified biomedical sentences. Due to our filtering steps, the approach is comparably computationally lightweight in the later steps and only very few LLM-calls are issued. Our results point towards major issues with the current pipeline, with employment of the pre-filtering step (plain DistilBERT) achieving the best results by far. The pipeline produced the most reliable results for the higher-frequency labels (`‘Generation failure‘` and `‘Leaked instructions‘`) with least reliability on rare categories. Consideration of linguistic features of simplified sentences seems to hold merit for the classification task.

Our approach should be re-run with the BERT variants and Random Forests trained on the `train` dataset as well as a more powerful LLM in the LLM council to assess the true potential of the pipeline. For the LLM council step, multiple LLMs of different families could be employed. Future works should

focus on length of simplified sentences, mapping simplified sentences to parts of the the source abstract and the incorporation of more linguistic features in simplified sentences or the difference in linguistic features between source and simplified sentences.

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude for generative Code was used for the BERT-Only JSON shaping of the submissions. Claude has also been used in order to get raw textual notes LaTeX-compatible bullet points with according syntax but the produced LaTeX has been completely rewritten. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

Acknowledgments

We thank the SimpleText organisers for continuously organising the shared task and bringing together the community to advance the field of automatic text simplification.

References

[1] L. Ermakova, H. Azarbonyad, J. Bakker, G. K. Shahi, B. Vendeville, J. Kamps, CLEF 2026 SimpleText Track - Simplify Scientific Text (and Nothing More), in: ECIR '26, 2026, pp. 215–224. doi:10.1007/978-3-032-21321-1_31.

[2] M. Shardlow, R. Nawaz, Neural Text Simplification of Clinical Letters with a Domain Specific Phrase Table, in: ACL '19, 2019, pp. 380–389. doi:10.18653/v1/P19-1037.

[3] N. Hofmann, J. Dauenhauer, N. O. Dietzler, I. D. Idahor, C. K. Kreutz, THM@SimpleText 2025 - Task 1.1: Revisiting Text Simplification based on Complex Terms for Non-Experts, in: CLEF Working Notes '25, 2025, pp. 4276–4285. URL: https://ceur-ws.org/Vol-4038/paper_352.pdf.

[4] B. Engelmann, F. Haak, C. K. Kreutz, N. Nikzad-Khasmakhi, P. Schaer, Text Simplification of Scientific Texts for Non-Expert Readers, in: CLEF Working Notes '23, 2023, pp. 2987–2998. URL: https://ceur-ws.org/Vol-3497/paper-250.pdf.

[5] P. J. Denning, The Conundrum of LLMs, Commun. ACM 69 (2026) 31–34. doi:10.1145/3789199.

[6] J. Bakker, B. Vendeville, L. Ermakova, J. Kamps, Overview of the CLEF 2025 SimpleText Task 1: Simplify Scientific Text, in: CLEF '25, 2025, pp. 4167–4185. URL: https://ceur-ws.org/Vol-4038/paper_344.pdf.

[7] L. Ermakova, H. Azarbonyad, J. Bakker, B. Chakar, G. K. Shahi, J. Kamps, Overview of the CLEF 2026 SimpleText track: Simplify scientific texts (and nothing more), in: CLEF '26, 2026.

[8] B. Chakar, J. Bakker, L. Ermakova, J. Kamps, Overview of the CLEF 2026 SimpleText Task 2: Identify and Avoid Hallucination, in: CLEF '26, 2026.

[9] K. C. Marturi, H. H. Elwazzan, Hallucination Detection and Mitigation in Scientific Text Simplification using Ensemble Approaches: DS@GT at CLEF 2025 SimpleText, in: CLEF Working Notes '25, 2025, pp. 4324–4337. URL: https://ceur-ws.org/Vol-4038/paper_356.pdf.

[10] N. Largey, D. Wu, B. Mansouri, AIIRLab Systems for CLEF 2025 SimpleText: Cross-Encoders to Avoid Spurious Generation, in: CLEF Working Notes '25, 2025, pp. 4311–4323. URL: https://ceur-ws.org/Vol-4038/paper_355.pdf.

[11] J. Collado-Montañez, J. A. O. Zambrano, C. Espin-Riofrio, A. Montejo-Ráez, SINAI in SimpleText CLEF 2025: Simplifying Biomedical Scientific Texts and Identifying Hallucinations Using GPT-4.1 and Pattern Detection, in: CLEF Working Notes '25, 2025, pp. 4225–4239. URL: https://ceur-ws.org/Vol-4038/paper_348.pdf.

[12] B. Vendeville, J. Bakker, H. Azarbonyad, L. Ermakova, J. Kamps, Overview of the CLEF 2025 SimpleText Task 2: Identify and Avoid Hallucination, in: CLEF '25, 2025, pp. 4186–4204. URL: https://ceur-ws.org/Vol-4038/paper_345.pdf.

[13] L. Ermakova, H. Azarbonyad, J. Bakker, B. Vendeville, J. Kamps, Overview of the CLEF 2025 SimpleText Track - Simplify Scientific Text (and Nothing More), in: CLEF '25, 2025, pp. 436–463. doi:10.1007/978-3-032-04354-2_23.

[14] L. Ermakova, H. Azarbonyad, J. Bakker, B. Vendeville, J. Kamps, CLEF 2025 SimpleText Track: Simplify Scientific Text (and Nothing More), in: ECIR '25, Cham, 2025, pp. 425–433. doi:10.1007/978-3-031-88720-8_63.

[15] L. Zhuang, L. Wayne, S. Ya, Z. Jun, A Robustly Optimized BERT Pre-training Approach with Post-training, in: Proceedings of the 20th Chinese National Conference on Computational Linguistics, Huhhot, China, 2021, pp. 1218–1227. URL: https://aclanthology.org/2021.ccl-1.108/.

[16] V. Sanh, L. Debut, J. Chaumond, T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, CoRR abs/1910.01108 (2019). URL: http://arxiv.org/abs/1910.01108. arXiv:1910.01108.

[17] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: NAACL '19, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[18] I. Beltagy, K. Lo, A. Cohan, SciBERT: A Pretrained Language Model for Scientific Text, in: EMNLP-IJCNLP '19, 2019, pp. 3615–3620. doi:10.18653/v1/D19-1371.