HaRMoNEE at SemEval-2024 Task 6: Tuning-based Approaches to Hallucination Recognition

Timothy Obiso and **Jingxuan Tu** and **James Pustejovsky**
Department of Computer Science
Brandeis University
Waltham, Massachusetts
{timothyobiso, jxtu, jamesp}@brandeis.edu

Abstract

This paper presents the Hallucination Recognition Model for New Experiment Evaluation (HaRMoNEE) team's winning (#1) and #10 submissions for SemEval-2024 Task 6: Shared-task on Hallucinations and Related Observable Overgeneration Mistakes (SHROOM)'s two subtasks. This task challenged its participants to design systems to detect hallucinations in Large Language Model (LLM) outputs. Team HaRMoNEE proposes two architectures: (1) fine-tuning an off-the-shelf transformer-based model and (2) prompt tuning large-scale Large Language Models (LLMs). One submission from the fine-tuning approach outperformed all other submissions for the model-aware subtask; one submission from the prompt-tuning approach is the 10th-best submission on the leaderboard for the model-agnostic subtask. Our systems also include pre-processing, system-specific tuning, post-processing, and evaluation.

1 Introduction

The HaRMoNEE team proposes two architectures to use on the SHROOM (Mickus et al., 2024) task: transformer model fine-tuning and large-scale LLM prompt tuning. First, we pre-process the data. We identify two fields from each example to use in our models.

For the fine-tuning models, the fields are then formatted into a single string with a separator token. We use three different training strategies with SHROOM data to improve performance on the test sets. Finally, we run the model on the test set and post-process the data to get a score. For prompt-tuning, the prompt is constructed around the two selected fields. Using the validation datasets, we experiment with two models and two prompts. We select the best of both and finally evaluate the test sets.

This paper reports our results from these experiments submitted during the SHROOM task's evaluation phase. We discuss the data, the task, our methods, and the experiments we ran. In addition, we analyze and discuss our results and make proposals for future work in hallucination recognition. We make our code and best results publicly available.[1]

2 Related Work

Several approaches have been taken to scoring faithfulness and identifying hallucinations. Laban et al. (2022a) proposed to use Natural Language Inference (NLI) to detect inconsistency in summarization tasks. They applied NLI to sentence pairs and aggregated the scores on the document level to obtain a faithfulness score. Lattimer et al. (2023) adopted a similar method by chunking the whole document into smaller pieces. However, they prompted LLMs to generate their scores. TrueTeacher (Gekhman et al., 2023) leveraged LLMs to generate synthetic data that could augment models' ability to identify factual inconsistencies in summarization tasks. AlignScore (Zha et al., 2023) is a unified evaluation metric for factual inconsistency that is based on the information alignment between two arbitrary text pieces. SelfCheckGPT (Manakul et al., 2023) proposed using LLMs, particularly GPT models, to generate multiple potential consistent/contradictory responses as a task-agnostic method for hallucination detection and fact-checking.

Several datasets for NLI and hallucination are commonly used to pre-train and fine-tune models for these tasks. SNLI (Bowman et al., 2015) is a large NLI dataset collected from image captions. It consists of sentence pairs labeled as entailment, contradiction, or neutral. PAWS (Zhang et al., 2019) proposed a new dataset for paraphrase identification that features non-paraphrase pairs with high lexical overlap. Honovich et al. (2022) stud-

[1] https://github.com/brandeis-llc/shroom

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 1322–1331

Field	Value
task	"DM"
src	"I'm an avid reader. What is the meaning of avid?"
hyp	"Having an intense interest in something."
tgt	"enthusiastic; keen; eager; showing great interest in something or desire to do something"
ref	"tgt"
model	"ltg/flan-t5-definition-en-base"
labels	["Not Hallucination", "Not Hallucination", "Hallucination", "Not Hallucination", "Not Hallucination"]
label	"Not Hallucination"
p(Hallucination)	0.2

Figure 1: Example data

ied the factual consistencies from text generation systems. They proposed TRUE, an automatic factual consistency assessment tool for tasks including summarization, dialogue generation, fact verification, and paraphrase detection. HaluEval (Li et al., 2023) proposed a dataset with human-annotated hallucinated instances specifically for evaluating the performance of LLMs in recognizing hallucinations.

Laban et al. (2022b) and Zha et al. (2023) both fine-tune pre-trained language models to obtain a score for each example indicating faithfulness between two texts or the likelihood of the presence of hallucination in one of the texts. Lattimer et al. (2023), Gekhman et al. (2023), and Manakul et al. (2023) all use LLMs in various ways to perform the same tasks.

3 Data and Task

SHROOM is split into two subtasks, model-aware and model-agnostic, with corresponding datasets. Task participants receive two sets of unlabeled training data, two sets of labeled validation data, and two unlabeled test sets. There is also a smaller labeled trial dataset.

3.1 Datasets

Table 1 shows a breakdown of each dataset. Table 2 shows a breakdown by label for the labeled datasets. The training sets are evenly split by example task. The validation and test sets are split 25:37.5:37.5. In each of the validation and test sets, there are more examples of "Not Hallucination" than "Hallucination".

Figure 1 is an example from the model-aware validation set. Every datapoint in the datasets includes the following fields: `task` (task: the task that the model is trained to perform, e.g. "DM", "MT", "PG"[2]), `src` (source: the text passed to the model), `hyp` (hypothesis: the model output), `tgt` (target: the "gold" text that the model should output), `ref` (reference: which field should serve as a reference for semantic information, e.g. "src" or "tgt").

The model-aware datasets include the additional field `model` (the name of the model used). The labeled data includes `labels` (the list of votes from all annotators), `label` ("Hallucination" or "Not Hallucination"), and `p(Hallucination)` (probability of hallucination: the likelihood that the model output contains hallucinated content).

The validation and test sets were labeled through crowdsourcing. Five annotators annotated each datapoint for the validation and test sets, and three annotators each for the trial set. The label of each datapoint is the label the majority of annotators chose. The probability of hallucination is reported as the ratio of "Hallucination" labels to all labels.

3.2 Task

This task is a binary classification task to determine whether the text generated by the LLM contains any hallucinated content. Accuracy is the main benchmark for this task. This task also uses Spearman's correlation coefficient, ρ, to measure the degree of agreement using `p(Hallucination)`.

From the example in Figure 1, we see the challenges of this task. Annotators were asked to determine whether the hypothesis, "Having an intense interest in something." contains hallucinated information. In addition, annotators know that the hypothesis was generated by the model `tg/flan-t5-definition-en-base`, a Definition Modeling model, from the input "I'm an avid reader. What is the meaning of avid?" Annotators are told to use "enthusiastic; keen; eager; showing great interest in something or desire to do something" as a semantic reference to make their

[2]DM = Definition Modeling, MT = Machine Translation, PG = Paraphrase Generation

Dataset	Trial	Train Agnostic	Train Aware	Val Agnostic	Val Aware	Test Agnostic	Test Aware
PG	9	10,000	10,000	125	125	375	375
MT	35	10,000	10,000	187	188	562	563
DM	36	10,000	10,000	187	188	563	562
Total	80	30,000	30,000	499	501	1,500	1,500

Table 1: Dataset Task Statistics, PG = Paraphrase Generation, MT = Machine Translation, DM = Definition Modelling

Dataset	Val AG	Val AW
Hallucination	218	206
Not Hallucination	281	295
Total	499	501
Dataset	**Test AG**	**Test AW**
Hallucination	611	551
Not Hallucination	889	949
Total	1,500	1,500

Table 2: Dataset Label Statistics, AG = model-agnostic, AW = model-aware

decision.

In this example from the model-aware validation dataset, four out of five annotators label this as "Not Hallucination". A simple probabilistic model may not be able to identify hallucinations. In this example, the target includes a few definitions of avid. Human annotators recognize that only matching one definition indicates no hallucination is present, while a probabilistic model will only see that very few tokens between these fields match. Here, one annotator did believe there was hallucination present, possibly because the hypothesis was not equally diverse or not semantically similar enough.

Another challenge is approximating the diversity of what humans consider a "Hallucination" to be across the three example tasks (PG, MT, DM). With all of this in mind, we formulate methods to identify hallucinations in LLM output.

4 Methods

4.1 Model Fine-Tuning

We first explore how transformer-based models would perform on this task. Due to the similarity of this task to NLI, models made for hallucination detection and NLI models are considered. For our preliminary experimentation, all models are trained and tested using only the validation datasets as they are labeled. Each of the validation datasets was split 80/20 into a train and test subset. The best-performing model is used for evaluation on the test set. Our preliminary experimentation shows that hallucination recognition models significantly and consistently outperform NLI models.

Despite the similarity of NLI to this task, the SHROOM dataset is more diverse than NLI datasets. SHROOM's three example tasks include many different forms of hallucinations that NLI models do not encounter as frequently or at all in their training. Models for hallucination recognition are usually trained on NLI datasets as well as others. This data diversity makes hallucination models especially well suited for the hallucinations and data in the SHROOM task.

After identifying the best model, we experiment with different fine-tuning approaches. We vary the number of epochs, which data the model is fine-tuned on, and the order of fine-tuning. Additionally, we inference these models before any training to serve as a baseline.

All models we fine-tune take in two texts as input. The first text is the frame of reference to determine if hallucinated material is present. The second text is the text that may or may not have a hallucination. The output of the model is a number from 0 to 1. A score of less than 0.5 indicates that a hallucination is "likely" present.

A score of 0 represents high confidence that hallucination is present; a score of 1 represents high confidence that hallucination is not present. Because this scale is inverse to the scale used by the task organizers, where `p(Hallucination)` being 1 indicates all annotators have chosen the label "Hallucination", the model output is subtracted from 1 and the difference is used as `p(Hallucination)`.

For our fine-tuning model architecture (as shown in Figure 3), we pre-process each datapoint by selecting two text fields to pass to the model and inverting the numerical scale to match the model output. After training, we post-process the output by re-inverting the numerical scale to get our `label` and `p(Hallucination)` fields.

Definition modeling is a task to generate a definition for a given word in context. In the example shown below, The source corresponds to the context; The target is the correct definition for this context; the hypothesis is the predicated definition from the model.

Example:
source: The sides of the casket were covered with heavy black broadcloth, with velvet caps, presenting a deep contrast to the rich surmountings. What is the meaning of surmounting?
target: A decorative feature that sits on top of something.
hypothesis: A sloping top.

Your task is to answer whether the hypothesis from the example contains any hallucination (e.g., incorrect semantic information unsupported or inconsistent with the source) and explain why. The target is inferred from the source without any hallucination. You should consider both the source and the target before making a judgment on the hypothesis.

The output should be formatted as a JSON instance that conforms to the JSON schema below.
As an example, for the schema {"properties": {"foo": {"title": "Foo", "description": "a list of strings", "type": "array", "items": {"type": "string"}}}, "required": ["foo"]}
the object {"foo": ["bar", "baz"]} is a well-formatted instance of the schema. The object {"properties": {"foo": ["bar", "baz"]}} is not well-formatted.

Here is the output schema:
{"properties": {"answer": {"title": "Answer", "description": "answer 'Yes' if the hypothesis contains hallucination; answer 'No' if the hypothesis does not contain hallucination", "type": "string"}, "reason": {"title": "Reason", "description": "a brief explanation to your answer", "type": "string"}}, "required": ["answer", "reason"]}

Figure 2: GPT Prompt 1 (Three Fields)

4.2 LLM Prompt-Tuning

Our team also uses LLMs as black-box hallucination detection systems. We first experiment with zero-shot classification on the validation sets. We use GPT-3.5 (Brown et al., 2020) and GPT-4 (OpenAI, 2023) with the prompt shown in Figure 2. If a model performs well on the validation sets, it would also be used to evaluate the test sets.

After choosing a model, we tune the prompt. The prompts we experiment with vary with respect to which fields are provided, the structure/order of the prompt, the inclusion of the task definition, and the overall verbosity of the prompt. One additional experiment we performed was the specific format of the model response. Asking a large-scale LLM to respond to a Yes/No question and give a reason may lead to many different responses. Although we expect the model to answer "Yes" or "No" when told to, it may ignore that part of the instruction and respond in a way that is correct but not directly interpretable by our post-processing such as "This contains a hallucination" instead of "Yes".

Asking the model to respond in JSON format increased the likelihood that the answer would be directly interpretable. Additionally, we ask our models to provide explanations for their "Yes" or "No" responses. While we do not use these to determine the `label` or `p(Hallucination)` for any datapoint, we found that asking LLMs to provide reasoning boosted performance. Shorter prompts such as those seen in Figure 5 were also used with these models. To ensure replicability, we set the hyperparameter `temperature` to 0.0. Setting this hyperparameter as such leads to increased determinism in responses.

Our post-processing assigns the `label` field "Hallucination" or "Not Hallucination" based on the value of the key `answer` found in GPT's JSON output. For these models, `p(Hallucination)` was set naively. Therefore, if the model returns `...{"answer": "Yes"}...`, `label` is "Hallucination" and `p(Hallucination)` is 1. When "Not Hallucination" is the label, `p(Hallucination)` is 0.

5 Experiments

Each datapoint contains up to three semantically relevant text fields, `hyp`, `src`, and `tgt`. In the model-agnostic subtask, these fields are always provided. In the model-aware subtask, `tgt` is left blank if the `task` is Paraphrase Generation (PG).

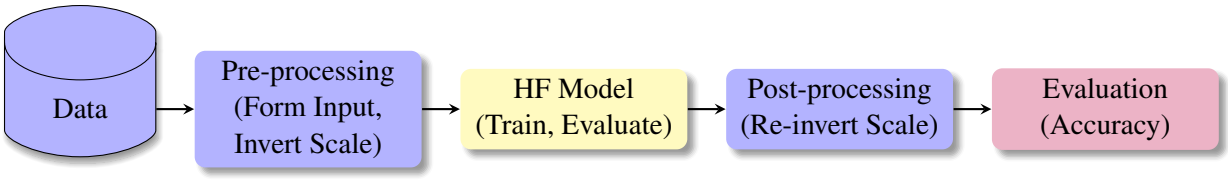

Figure 3: Fine-tuning architecture

Figure 4: Prompt-tuning architecture

Your task is to determine whether the hypothesis contains any hallucinations based on the target. (e.g., incorrect semantic information) and explain why. Only consider target and hypothesis when making the judgement. Your answer must start with 'Yes' or 'No'.

Example:
target: A decorative feature that sits on top of something .
hypothesis: A sloping top .

The output should be formatted as a JSON instance that conforms to the JSON schema below.
As an example, for the schema {"properties": {"foo": {"title": "Foo", "description": "a list of strings", "type": "array", "items": {"type": "string"}}}, "required": ["foo"]}
the object {"foo": ["bar", "baz"]} is a well-formatted instance of the schema. The object {"properties": {"foo": ["bar", "baz"]}} is not well-formatted.

Here is the output schema:
{"properties": {"answer": {"title": "Answer", "description": "answer 'Yes' if the hypothesis contains hallucination; answer 'No' if the hypothesis does not contain hallucination", "type": "string"}, "reason": {"title": "Reason", "description": "a brief explanation to your answer", "type": "string"}}, "required": ["answer", "reason"]}

Figure 5: GPT Prompt 2 (Two Fields)

We experiment by varying which of these fields get passed to our model and the structure of the model input.

We conduct a series of experiments using two approaches to detecting hallucinations. We first fine-tuned existing hallucination detection models using SHROOM validation data. Second, we experiment with a series of prompts to increase determinism and accuracy of LLMs on the same task. We show similarities of note between the best results of each architecture.

5.1 Fine-Tuning Experiments

The model that we find to perform the best on this task is a model to detect LLM-generated hallucinations. We find that the best results were obtained from this model when the input is of the form [CLS]+tgt+[SEP]+hyp for DM and MT. When tgt is not provided for the model-aware PG examples, the input [CLS]+src+[SEP]+hyp is used. This model is vectara/hallucination_evaluation_model on HuggingFace. This model took microsoft/deberta-v3-base (He et al., 2021) and trained it on two NLI datasets, SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2018), as well as one paraphrase dataset, Paraphrase Adversaries from Word Scrambling (PAWS) (Zhang et al., 2019).

We first conducted our preliminary experiments on the split validation sets. After, we took the best-performing model, fine-tuned it on the entire validation set(s), and evaluated it on the test set(s). We experimented with varying the number of epochs, the datasets used, and the training order.

For both the model-agnostic and model-aware subtasks, we experimented with inferencing, 1-5

Dataset	Test AG		Test AW	
Metric	**acc**	ρ	**acc**	ρ
SHROOM Baseline	0.697	0.403	0.745	0.488
Corresponding Dataset	0.783	0.663	**0.813**	0.699
All Data	0.785	0.652	0.808	0.713
All + Corresponding	0.783	0.683	0.810	0.671
GPT-4 (Two Fields)	**0.814**	0.626	0.783	0.614

Table 3: Best results from each approach and the baseline results. Corresponding Dataset, All Data, and All + Corresponding are fine-tuning results; GPT-4 is prompt-tuning results

epochs of training on the corresponding validation dataset, 1-5 epochs of training on all validation data, and 1, 3, or 5 epochs of training on both validation datasets before one epoch of tuning on the corresponding validation dataset.

5.2 Prompt-Tuning Experiments

In addition to fine-tuning off-the-shelf models, we experiment with GPT-3.5 and GPT-4 using several prompts. After evaluating the validation sets on GPT-3.5 and GPT-4, further prompt tuning was done on GPT-4 due to its superior performance.

The first prompt we experiment with includes `hyp`, `src`, and `tgt`, task instructions, the example task definition, and the main prompt. The second prompt we used only included two fields (`hyp` and `tgt` for DM and MT, `hyp` and `src` for PG), task instructions, and the main prompt.

6 Results

Our best results from each training architecture are shown in Table 3. One notable similarity between our best results from each system is that they required pre-processing (data pruning). We obtain our best results by only including two of the three meaningful fields for each datapoint in both fine-tuning and prompt-tuning methods. For the fine-tuning methods, converting three fields into two via concatenation underperformed ignoring one field (`src` was ignored if `tgt` was provided). Additionally, removing the definitions of DM, MT, and PG from the prompt led to improved results.

Despite not making explicit use of the `model` field for the model-aware subtask, our models' best performances earned a higher spot in the model-aware subtask than the model-agnostic one. For the model-aware subtask, we use `src` when `tgt` is not provided for the model-aware PG examples. We also believe it to be due to the differences in other fields. For instance, the DM examples for the

Dataset	Test AG	Test AW
PG	0.789	0.875
MT	0.851	0.837
DM	0.794	0.747
All	0.814	0.813

Table 4: Accuracy of best submission to each subtask split by example task

# of epochs	0	1	2	3	4	5
Agnostic Data	0.756	0.769	0.776	0.776	0.783	0.783
All Data	0.756	0.779	0.777	0.777	**0.785**	0.780
All + Agnostic Data	0.756	0.783		0.775		0.783
Aware Data	0.794	0.804	0.804	0.805	**0.813**	0.803
All Data	0.794	0.796	0.808	0.808	0.808	0.797
All + Aware Data	0.794	0.808		0.810		0.795

Table 5: Table of accuracy data shown in Figure 6

model-aware and model-agnostic tasks are formatted differently. The model-aware examples ask for the definition explicitly at the end of the `src` field (e.g. "... What is the meaning of spoilage ?"). The model-agnostic examples put the word to define in tags (e.g. "The <define> sacrifice bunt </define> was fielded cleanly..."). These different tagging strategies reflect inputs different DM models take in. Based on our results in Table 4, it seems that our systems are better are recognizing hallucinations obtained via <define> tags.

Scores within .001 point of each other were obtained for each subtask using these systems, yet different systems performed the best for each. Our shorter prompt with GPT-4 produced our best results (#10) for the model-agnostic subtask, .117 points above the task organizer baseline. Training on the corresponding SHROOM validation dataset produced the best results (#1) for the model-aware subtask, .068 points above the task organizer baseline.

6.1 Fine-Tuning Results

Our team's first experiments on the test set involved varying the number of epochs, the training set(s), and the training order. These results for the model-agnostic and model-aware subtasks are shown in Figure 6 and Table 5. The 0 epoch results are from inferencing the model before training on SHROOM data. They serve as a baseline for all training strategies. This model's strong performance at inference for both test sets made it a strong contender for more fine-tuning. On the model-agnostic subtask, inferencing resulted in an accuracy of 0.756. On the model-aware subtask, it obtained a score of

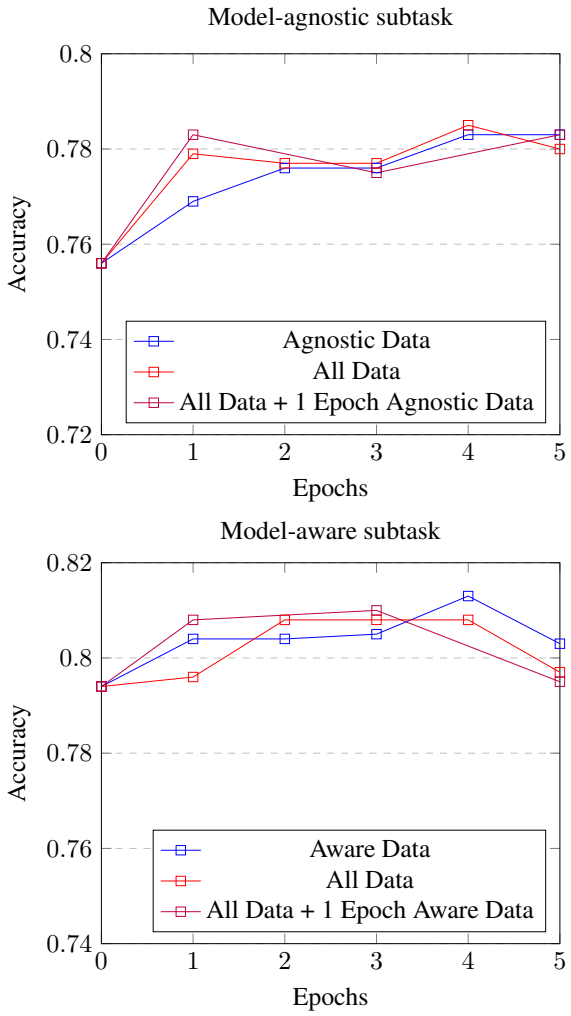

Figure 6: Line graphs showing accuracy on the test sets of both subtasks after fine-tuning

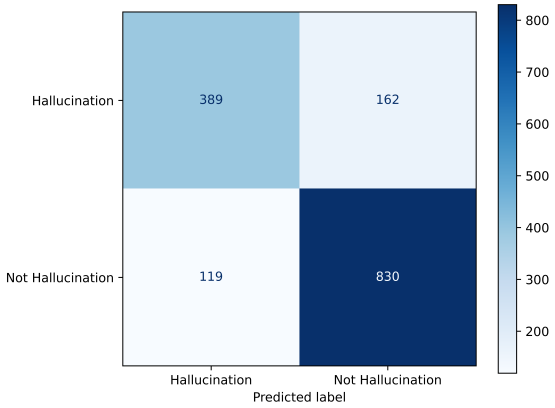

Figure 7: Confusion matrix for the #1 submission on the model-aware dataset (fine-tuning approach)

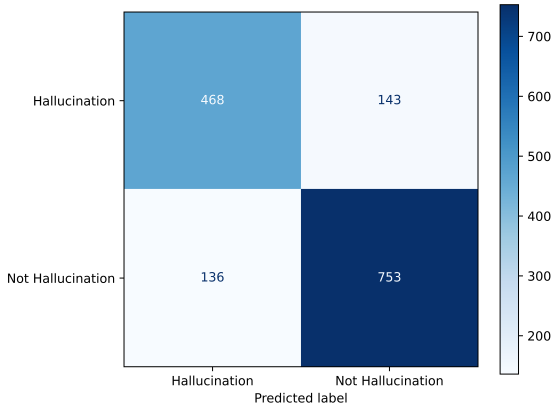

Figure 8: Confusion matrix for the #10 submission on the model-agnostic dataset (prompt-tuning approach)

Dataset	Val AG	Val AW
SHROOM Baseline	0.649	0.707
GPT-3.5	0.661	0.596
GPT-4 (Three Fields)	0.778	0.751
GPT-4 (Two Fields)	**0.782**	**0.773**

Table 6: Comparison of the baseline results to our results on the validation sets using accuracy

0.794.

Training with any SHROOM data led to better performance on the test set. For the model-agnostic subtask, our team's best result came from fine-tuning the model on the model-agnostic and model-aware datasets together for four epochs. This increased the accuracy from our inference baseline by 0.029 to 0.785. On the model-aware subtask, our team's best results were obtained after fine-tuning the model using only the model-aware data for four epochs. This increased the accuracy from the inference baseline by 0.019 to 0.813, our winning submission for the model-aware subtask.

As seen in Figure 7, this architecture performs well on the SHROOM model-aware subtask. Figure 9 and Table 4 show a breakdown by task. This model performed very well on PG and MT but much worse on DM. We believe this task has the lowest accuracy for the same reason we identified earlier in the paper. The target, source, and hypothesis fields may vary much more than in the other two tasks. The target may be much more or much less semantically rich than the hypothesis, which can be interpreted by human annotators and out models in many different ways. For the winning submission, the accuracy for the PG task was .875, MT was .837, and DM was .747.

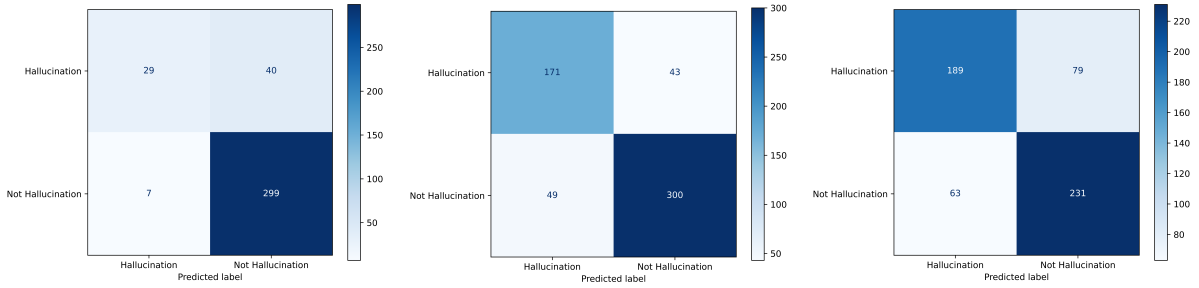

Figure 9: Confusion matrices for the #1 submission on the model-aware dataset (fine-tuning method) split by example task. Left to right: Paraphrase Generation, Machine Translation, Definition Modeling

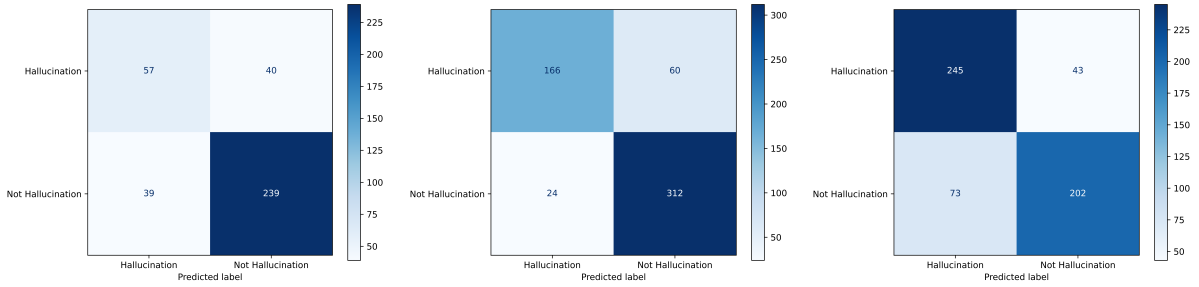

Figure 10: Confusion matrices the #10 submission on the model-agnostic dataset (prompt-tuning method) split by example task. Left to right: Paraphrase Generation, Machine Translation, Definition Modeling

6.2 LLM-Tuning Results

Our LLM experimentation results are shown in Table 6. Because we experimented with prompt-tuning on the validation sets, we were able to directly compare our results to the baselines provided by the task organizers. We found that GPT-3.5 performed worse than the baseline for one subtask and did not pursue further experiments with it. We tested two prompts on the validation sets.

Our first prompt (Figure 2) includes all meaningful fields that annotators had access to when labeling the data. For the model-agnostic subtask, this prompt outperformed the baseline by 0.129 with an accuracy of 0.778. it improved the accuracy in the model-aware subtask by 0.044 with an accuracy of 0.751.

Our second prompt (Figure 5) includes selected fields of meaningful information from each datapoint. It does not explain the example task (DM, MT, or PG) for the datapoint but still explains the shared-task instructions and output formatting instructions. This reduction in verbosity led to improved performance for both subtasks. For the model-agnostic validation set, this change resulted in a .133 point increase in accuracy above the baseline and a .004 point increase compared to the first prompt. For the model-aware validation set, this change resulted in a .066 point increase compared to the baseline and a .022 increase compared to the first prompt.

We also experimented with few-shot learning and found that both random examples and selected examples did not improve performance on either subtask. Overall, we found that a less verbose prompt outperformed a more verbose prompt indicating that GPT has difficulty making connections across large amounts of text. Additional information that GPT may not need in each prompt, such as the example task definition and the third field, adds noise to the prompt and impairs its ability to detect hallucinations.

The confusion matrices in Figures 8 and 10 show the performance of this architecture on the entire test set and on each task for the model-agnostic subtask. For this submission, the accuracy for the PG task was .789, significantly lower than the other example tasks. We believe this is because we used the `tgt` field for all tasks here as it is always provided. In the model-aware subtask, `tgt` is not provided for PG examples, so `src` is used instead. This may indicate that `tgt` is best for MT and DM, but `src` for PG even if `tgt` is provided. The accuracy for the MT task examples was .851, similar to the model-aware subtask. The accuracy for Definition Modeling was .794.

7 Discussion and Conclusion

In this paper, we present two systems for hallucination recognition, one transformer-based model fine-tuned on SHROOM data and one prompt-tuned zero-shot classification model using GPT-4. Our results show that both systems can better handle semantically complex tasks such as hallucination recognition when only semantically relevant information is provided. Pre-processing each example is essential to good performance on this task. From our results, fine-tuning using available labeled data from all tasks improves performance from the baseline. Additionally, pruning information such as over-explicit instructions, irrelevant fields, and definitions from prompts also improves performance from the baseline.

Some avenues we did not fully explore include training on pseudo-labeled training data, training on additional datasets besides SNLI, MultiNLI, and PAWS (specifically adversarial translation or word disambiguation datasets), as well as experimenting with dense paraphrasing and frame saturation methods as proposed by Tu et al. (2023). In this shared task, our team found that it is easier to harmonize by tuning fewer, clearer voices.

Acknowledgements

The authors would like to acknowledge the support and input from Kyeongmin Rim. This work was supported in part by NSF grant 2326985 to James Pustejovsky. The opinions and views reported herein are those of the authors alone.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Zorik Gekhman, Jonathan Herzig, Roee Aharoni, Chen Elkind, and Idan Szpektor. 2023. TrueTeacher: Learning factual consistency evaluation with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2053–2070, Singapore. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. TRUE: Re-evaluating factual consistency evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3905–3920, Seattle, United States. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022a. Summac: Re-visiting nli-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022b. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Barrett Lattimer, Patrick CHen, Xinyuan Zhang, and Yi Yang. 2023. Fast and accurate factual inconsistency detection over long documents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1691–1703, Singapore. Association for Computational Linguistics.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore. Association for Computational Linguistics.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. ArXiv:2303.08896 [cs].

Timothee Mickus, Elaine Zosa, Raúl Vázquez, Teemu Vahtola, Jörg Tiedemann, Vincent Segonne, Alessandro Raganato, and Marianna Apidianaki. 2024. Semeval-2024 shared task 6: Shroom, a shared-task

on hallucinations and related observable overgeneration mistakes. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1980–1994, Mexico City, Mexico. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Jingxuan Tu, Kyeongmin Rim, Eben Holderness, Bingyang Ye, and James Pustejovsky. 2023. Dense paraphrasing for textual enrichment. In *Proceedings of the 15th International Conference on Computational Semantics*, pages 39–49, Nancy, France. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, Toronto, Canada. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.