NLP Milestone 2 - WS 2024

Czernin Nikolaus,

Gergő Kapás,

Gábor Lakatos,

Gábor Zeke

Contents

1 Introduction

2 Problem Overview

This project tackles this year's MuSHROOM challenge [2]. Any LLM output may contain hallucinations. The task at hand is to detect not only whether or not an output contains a hallucination at all, but rather to detect the spans of the hallucinations within the response, not unlike a teacher would highlight false information in a written essay with a red marker. The task data are multilingual, containing notoriously complex languages like Mandarin and Finnish. Moreover, the provided training data are not labeled by default, part of the challenge is to generate one's own training labels to fit a model on, and thus, hopefully, be able to make testing data predictions that perform well according to the following evaluation criteria: The intersection of characters marked as hallucinations of the true and predicted validation labels, as well as the degree to which the probabilities of predicted spans correlate with the annotated spans [2].

3 Baselines

As a baseline to tackle the problem, we propose two approaches. Using a supervised deep learning approach and an unsupervised approach using an existing LLM.

3.1 Baseline 1: Supervised sequence classification using mBERT

For this approach, the main question was how to set up a sequence classification architecture that incorporates the right information to make context-aware decisions for each token in a given sequence. Given two preprocessed sequences of lemmatized tokens, the model query and the model's response output, the latter of which contains hallucination spans to detect, how can we detect the spans of tokens that are hallucinations. Initially, we proposed a token-wise training and predicting workflow: Iteratively predicting a single binary label for each token of the response, given a query, the features being the vectorized feature space data of the query and the token concatenated. This raises the following issue: Take for example the query "When did Albert Einstein live?". The response "Albert Einstein was born in 1955 and dies 1879." contains two hallucinations, the dates are swapped. On a token level classification without the response's context, a model would judge the response token "1955" to be correct given the feature space of the query. Thus, leaving out the rest of the response is not a viable option.

Instead, we propose the architecture, visualized in figure 1. The pillar of this approach is the multilingual BERT token classifier [1], a transformer-based language model which is able to take

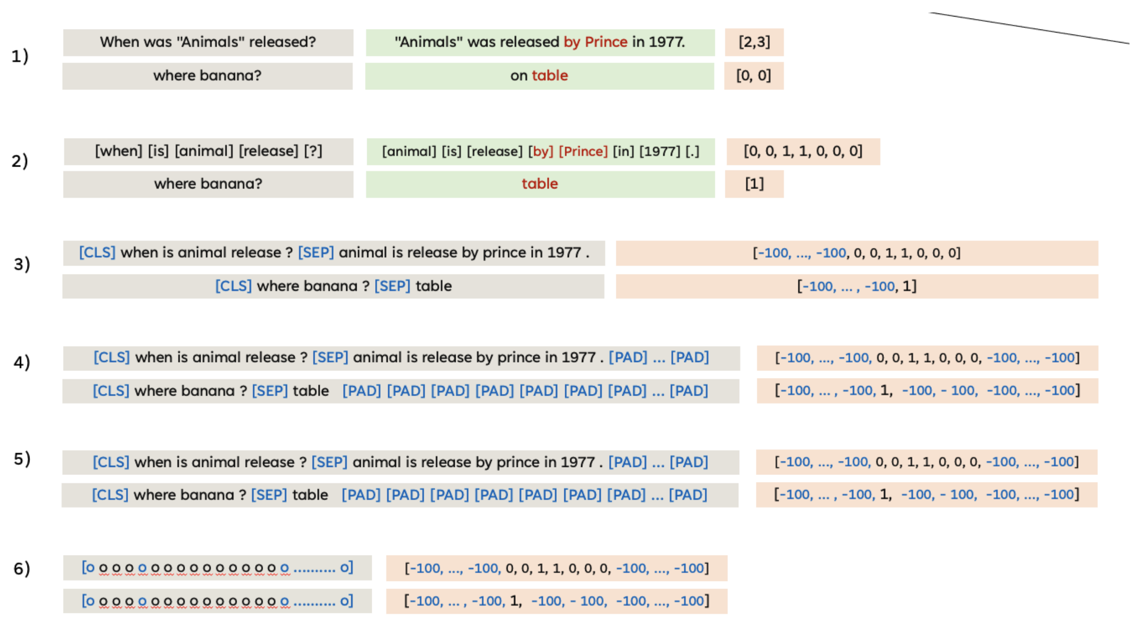
1)
When was "Animals" released?
"Animals" was released by Prince in 1977.
[2,3]
where banana?
on table
[0, 0]
2)
[when] [is] [animal] [release] [?]
[animal] [is] [release] [by] [Prince] [in] [1977] [.]
[0, 0, 1, 1, 0, 0, 0]
where banana?
table
[1]
3)
[CLS] when is animal release ? [SEP] animal is release by prince in 1977 .
[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0]
[CLS] where banana ? [SEP] table
[-100, ... , -100, 1]
4)
[CLS] when is animal release ? [SEP] animal is release by prince in 1977 . [PAD] ... [PAD]
[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0, -100, ..., -100]
[CLS] where banana ? [SEP] table [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] ... [PAD]
[-100, ... , -100, 1, -100, - 100, -100, ..., -100]
5)
[CLS] when is animal release ? [SEP] animal is release by prince in 1977 . [PAD] ... [PAD]
[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0, -100, ..., -100]
[CLS] where banana ? [SEP] table [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] ... [PAD]
[-100, ... , -100, 1, -100, - 100, -100, ..., -100]
6)
[o o o o o o o o o o o o o o o o .... o]
[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0, -100, ..., -100]
[o o o o o o o o o o o o o o o o .... o]
[-100, ... , -100, 1, -100, - 100, -100, ..., -100]

Figure 1: Baseline 1: Supervised sequence classification architecture workflow

into account the context of a token and assign labels to each token in the sequence.

In step 1, the gray fields are examples of queries, the green fields the model responses, and the orange fields are given labels. The hallucinations are marked red in the responses. In step two, the input data is lemmatized and tokenized. The labels are transformed to binary arrays, containing zeros for tokens outside the hallucination spans. Since we did not intend to participate in the challenge, we took the liberty of modifying the problem to suit our preferred approaches, namely, we changed the given sentence character level indices pointing to hallucination spans to word level indices. Although this transformation is reversible after inference, we thus far have not implemented this functionality. In step 3, the two sequences of input data are concatenated and the special token "[CLS]" and "[SEP]" are prepended. Here it becomes apparent, that if we feed not only the response but also the query into the model, it will also train and make predictions on the query tokens. In order to avoid the model being fitted to correctly make predictions on unnecessary data points, namely the input query and the special tokens, we need to make the loss function ignore those tokens. We used CrossEntropyLoss, which can be prompted to ignore a token by giving it a training label of -100, which you can see prepended to the label sequence in step 3. At this point we need to address that the chosen model requires each observation's lengths to be of equal length, after all we are using a single neural network for a multitude of texts of different lengths. After setting a maximal length for the input and output vectors of the model, in step 4, both the input sequence and the label sequence are padded with special "[PAD]" tokens and -100 labels respectively. We implemented 4 different strategies of handling data rows too long to fit within the max length. The most simple strategy was skipping any rows with where overflow would happen to avoid confusing the model with the issues raised in the other two strategies. The next strategy was to truncate the training inputs to fit within the maximum length, which forfeits some response context, just as does the strategy of splitting an overflowing row into multiple rows, where each contains the same query head, but with different parts of the response. The former increases the number of training observations, but forfeits some individual context. The fourth strategy was to add the option of leaving out the query in addition to one of the former 3 strategies. This allowed for more response context within a shortened observation, at the expense of losing the context of the query. Lastly, in step 5, the input sequences are transformed into a vector feature space using the mBERT multilingual tokenizer.

The now numeric input and label data are now used for model training. Afterward, to make predictions, prediction data is processed in the same way as the training data in figure 1. After getting the probabilities of being hallucinated of each token in the sequence, the probabilities of only the tokens of the response need to be extracted from this sequence of max length tokens. By using the tokenizer to get the original tokens of the input sequence, we can find the locations of the special token enclosing the model output response and extract the relevant label predictions.

3.1.1 Results

For the time of creation of the baseline, no training data was available, so we trained on the labelled validation dataset. We did 5-fold cross validation and applied 3 different overflow handling strategies: splitting overflowing responses into multiple rows containing the same query, truncating overflowing responses to fit within the maximum length, and lastly, skipping any overflowing rows to restrict the model to full context training only.

Overflow strategy	Training observations	Accuracy	Recall
Split	822	0.6605	0.4868
Truncate	399	0.6426	0.4277
Skip	234	0.6703	0.4785

Table 1: Prediction evaluation of Baseline 1 on validation dataset

3.2 Baseline 2: Unsupervised labeling using GPT-4o

Experiments were also conducted that tested the viability of zero shot prompting a pre-existing LLM - GPT-4o in this case - to see how well such a complex model could perform even without any training on our data.

The Input-Output pairs were passed in one at a time in order to provide the most clear task

definition for the LLM and ensure the best response, which was expected to be a list containing the start and end index of the hallucinated tokens. OpenAI's API was used to query GPT-4o. Attempts to use GPT-4o-mini instead were also made, but the lighter model performed significantly worse than the full 4o version, regularly returning not only wrong spans, but also ones that were outside the range of the input text it was supposed to be processing.
If one wishes to obtain competitive results with this methodology, fine-tuning the model is almost certainly necessary. This was not the goal of this experiment, but utilizing (potentially preprocessed) labeled training data to fine-tune such a powerful LLM would certainly yield far better results than what was observed with zero-shot prompting.

3.3 Next steps

- Fine-Tuning GPT Models: While zero-shot prompting with GPT-4o showed promise, fine-tuning on the provided training data would likely yield more competitive results. Fine-tuning enables the model to adapt to specific nuances in hallucination detection tasks across multiple languages.
- Hybrid Approaches: Combining the strengths of supervised and unsupervised methods could improve accuracy. For instance, supervised mBERT predictions can serve as inputs for GPT-4o to refine predictions in an iterative framework.
- Multilingual Challenges: Although mBERT handles multilingual inputs, we might get better results picking more specialized models instead.
- Post-Processing Improvements: Enhancing post-processing techniques to better map predictions from word-level back to character-level indices will align results with the challenge requirements.

4 Conclusion

This project addressed the challenge of hallucination span detection in multilingual model-generated responses. We implemented two baselines: a supervised mBERT token classification model leveraging contextual information to classify tokens, and an unsupervised GPT-4o zero-shot prompting approach. mBERT achieved promising accuracy, GPT-4o only sparingly demonstrated the potential of unsupervised methods. Future work will focus on fine-tuning, hybrid approaches, and data augmentation to further improve performance and address these limitations.

References

[1] google-bert. *BERT multilingual base model (cased)*. Accessed: 2024-12-12. URL: https://huggingface.co/google-bert/bert-base-multilingual-cased.

[2] Language Technology at the University of Helsinki. *Welcome to SemEval-2025 Task-3 — Mu-SHROOM, the Multilingual Shared-task on Hallucinations and Related Observable Overgeneration Mistakes.* Accessed: 2024-12-12. URL: https://helsinki-nlp.github.io/shroom/.