SIRCSS-CTA

Lab 4: Large Language Models

1 * Prompt engineering with ChatGPT

You can choose to either do this task or the task above to get VG on this assignment.

This task is only necessary to get one point toward a *pass with distinction* (VG) grade. Hence, if you do not want to get a *pass with distinction* (VG) point, you can ignore this part of the assignment.

For this assignment, you need to have a user account at Open AI to generate an API token. See [this link] for details on how to get an API token to use.

Warning: Do not upload any files containing your API key to public repositories (Github, public google colab, etc.).

You also need to buy credits at Open AI. I think 1-2 dollars should be sufficient for this assignment.

As a second step you need to install the `ropenaiapi` R package. This can be done as follows:

```
library(remotes)
remotes::install_github('MansMeg/ropenaiapi')
```

In the next step we set the API key as an environment variable with:

```
library(ropenaiapi)
set_openai_api_key("[YOUR KEY GOES HERE]")
```

We can now call the API with the `openai_chat` function.

```
x <- openai_chat("Who is Gustav Vasa? Give a short answer.",
                 model = "gpt-3.5-turbo")
x
```

```
## assistant:
## Gustav Vasa was a Swedish king who led a successful rebellion
## against Danish rule and became the founder of modern Sweden in
## the 16th century.
```

See the `ropenaiapi` documentation with "'?openai_chat"' in R for further information on how to interact with the Open AI API.

Note! If you get "'Error: You exceeded your current quota, please check your plan and billing details."' that means that you don't have any credits to use for the calls. You then need to buy some credits from Open AI, 1-2 dollars should be sufficient for this task.

1.1 Make ChatGPT hallucinate

We are now going to understand how and when transformer-based decoder models 'hallucinate', i.e. when the models give factual incorrect information. See (Zhao et al., 2023,

, Section 7.1.2) for an introduction and Huang et al. (2023) for details on hallucinations in LLMs.

Now, create 3 different hallucinations with three different strategies (i.e. its not ok to use the same prompt for three incorrect generations) from the chatGPT models (you can choose which one you like). For each generated hallucination return:

1. The prompt and a the `seed` supplied to the API. You are free to both create hallucinations through a longer conversation and tuning parameters.
2. The response from the LLM.
3. Why the response is factually incorrect and weather it is an intrinsic or extrinsic hallucination.
4. Describe the strategy you use and the analysis why the hallucination happened.

As a final step, when you have succeded to get the LLM to hallucinate, change the model to GPT-4 (include exact which model you use) and run the same prompt that you could get to hallucinate.

1. Return the response from GPT4 for each prompt.
2. Is the response still factually incorrect? If not, see if you can get at least one hallucination for GPT-4. If you don't succeed, just state that.

1.2 In-Context Learning for few-shot classification

In-context learning is the idea to use large language models to complete tasks in a few-shot or zero-shot way. In this task we are going to try to classify political quotes in Swedish to weather they belong to the social democratic party manifesto or the conservative party (Moderaterna) manifesto.

Note! This data is included in `uuml` version 0.4.0 and later.

```
library(uuml)
data("pc_test")
data("pc_train")
```

Below is an example on how you can combine the data to a text string that can be combined to be used in a prompt. Note, you need to develop further on this.

```
example_string <- paste0(pc_train$quote,
                         ": ",
                         pc_train$party, collapse = "\n\n")
cat(example_string)
```

See (Zhao et al., 2023, , Section 8) for details on how to best engineer prompts to solve the classification task.

Now use the techniques (Zhao et al., 2023, , Section 8) to generate as good prompts as possible to classify weather a quote is from the social democratic or the conservative manifesto.

1. Start with zero-shot learning, ie only prompt the LLM without any demonstrations. Test to classify the quotes in the test set.

2. Test how far you can get by using better prompt descriptions to classify the quotes. Try to get responses as classes from the model.

3. Now add demonstrations by adding examples from the training set. Does that improve the accuracy on the testset?

4. Try at least three different strategies from (Zhao et al., 2023, , Section 8) to improve the quality. Discuss your conclusions.

References

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models, 2023.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, 2023.