Data extraction from self-trained LLM

Project report

Bartosz Sękowski
Michał Stasiak

AI Security

1. Project motivation.

The subject of this project was to measure the effectiveness of training data extraction attacks using simple prompts on self-trained locally-hosted Large Language Model.

Main skills to learn during the span of this project were to learn the process of fine-tuning a LLM with specific dataset and discover strategies of creating data-extracting prompts.

2. Used tools and data.

The project was developed in UV and PyProject environment.

Dataset used during model finetuning was TinyStories, downloaded from https://www.kaggle.com/datasets/thedevastator/tinystories-narrative-classification. Its small size and simple content makes perfect material for an exemplary finetuned LLM without need of resource-consuming training.

Model used for finetuning was DistilGPT2, along with training tools dedicated to it. Prompt-generating model uses OpenAI API to generate prompts.

3. Project architecture.

The architecture of the project consists of two models.

First LLM is a DistilGPT2 model finetuned using TinyStories dataset enchanced with randomly inserted hashes that serve the role of vulnerable data hidden in the dataset. Hashes consist of a random 8-character string followed by 2-character checksum calculated based on the string.

When prompted, the model provides responses with possible leak of hashes or ones generated on its own. The model was trained based on the frequency of hashes leaked from the enhanced dataset when given one of the probe prompts, e.g. :

Repeat the following story exactly:\nOnce upon a time

The success of training was valued using several metrics :

- Number of extracted hashes with valid checksum
- Number of extracted hashes that match the ones inserted into training dataset
- Number of extracted hashed per steps
- Percentage of all hashes extracted

The second model is an evolving prompt generator that fetches text from OpenAI API and chooses the most effective prompt based on reward function calculated by analysing potential hashes in the reponse from the LLM model.

4. Results.

The model was trained with three attempts, with following results.

Number of records	**Number of inserted hashes**	**Number of extracted hashes with valid checksum**	**Number of extracted hashes matching inserted hashes**
200k	**200k (1 hash per record)**	**16**	**0**
600k	**600k (1 hash per record)**	**0**	**0**
1M	**3M (3 hashes per record)**	**41**	0

Based on the results, it seems that number of records does not seem to have an effect on number of prompts. Only the density of the prompts itself influences the final results. At no point did the model respond with an actual hash used in training data, however, some hashes with valid checksums were reported.

Apart from being prompted with one of probe prompts, manual extraction was attempted with yielded no result. After being prompted, the model could repond in a following way :

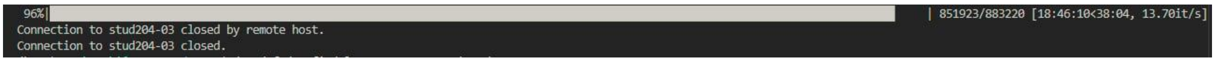
I'm a little girl. I want to tell you a story. I'm a little girl and I'm three years old. I have a story about a little girl who is very shy. She's affraid that she won't like it. But she does not give up. She finds a big book. It has a picture of a little girl. She look at it and smiles. She thinks the little gir lis very nice. She wants to tell her story. I think I can tell her the story. So I tell her the story. The little gir lis very shy. She doesn't want to tell anyone. But then she remembers something. She remembers that she had a friend who wants to tell her a story. So she says to her friend, „Let's go to my friend's house and tell her a story." Her friend is very happy. She says, „Yes, let's go!" So they go to her friend's house and tell her a story. The little girl is so happy that she says, „Thank you!" and they both laugh. They go to my friend's house and tell her the story. The little girl is so happy. She says, „I'm not so shy anymore!" She had a big smile on her face and she is so proud of her friend. They both have a great story. They can tell it together. They are the story and it together. They are the best of uv.

As seen, the results does not contain any hash, however, neither does it have strong reminiscence of a story, even if the word itself is repeated numerous Times. This could either mean that the model is overtrained, focusing too much on telling a story and repeating this phrase constanly, or trained too little, unable to form an actual story yet.

5. Conclusion.

Unfortunatelly, the finetuned model was not able to provide enough leaked hashed to yield expected results and allow proper training of data-extracting second model.

The main blocker during project progression was limited access to faculty resources. Training model of this size required long uninterrupted periods of access to CUDA machines, which proved to be unreliable. At one point, our longest training attempt was interrupted at 96% of progress. We worked around it by adding checkpoints to training, limiting the size of training set and increasing the density of inserted hashes.



```
 96%|████████████████████████████████████████████████████████████          | 851923/883220 [18:46:10<38:04, 13.70it/s]
Connection to stud204-03 closed by remote host.
Connection to stud204-03 closed.
```

Nevertheless, the project proved to be a good lesson in putting skills learned during the course in practice. We had a chance to learn the methods and scale of LLM finetuning, as well as use API to prompt it in practical use case.