# Timeline and Milestones Summer 2025

**Student:** Eduardo Castro Puello

**Professor:** Dr. Wei Li

Table of Contents

[Timeline and Milestones Summer 2025 1](#_Toc201082313)

[Timeline 1](#_Toc201082314)

[Week 1 Research Plan 2](#_Toc201082315)

[Week 2 Setup VMs with GPU on ML Azure 2](#_Toc201082316)

[Week 3 AutoDAN git test vanilla TinyLlama 2](#_Toc201082317)

[AutoDAN git test vanilla TinyLlama Results 3](#_Toc201082318)

[Week 4 Dataset Curation (JavaScript QA) & Preprocessing 4](#_Toc201082319)

[Dataset Curation (JavaScript QA) & Preprocessing Results 4](#_Toc201082320)

[Week 5 Fine-tune SLM using LoRA 5](#_Toc201082321)

[Fine-tune SLM using LoRA Results 5](#_Toc201082322)

[Week 6 Red Teaming with AutoDAN 6](#_Toc201082323)

[Week 7 Guardrail SLM Training 6](#_Toc201082324)

[Week 8 Full Wrapper Integration 6](#_Toc201082325)

[Week 9 Evaluation with DAN ChatGLMEval 6](#_Toc201082326)

[Week 10/11/12 Final Report and Analysis 6](#_Toc201082327)

[Expected Outcomes 6](#_Toc201082328)

## Timeline

| **Task** | **Description** | **Date** | **Status** |
| --- | --- | --- | --- |
| Week 1 | Research Plan | May 05-09 | Done |
| Week 2 | Setup VMs with GPU on ML Azure | May 10–17 | Done |
| Week 3 | AutoDAN git test vanilla TinyLlama | May 18–24 | Done |
| Week 4 | Dataset Curation (JavaScript QA) & Preprocessing | May 25–31 | Done |
| Week 5 | Fine-tune SLM using LoRA | June 1–7 | Done |
| Week 6 | Red Teaming with AutoDAN | June 8–14 | Active |
| Week 7 | Guardrail SLM Training | June 15–21 | No Started |
| Week 8 | Full Wrapper Integration | June 22–28 | No Started |
| Week 9 | Evaluation with DAN ChatGLMEval | June 29–July 5 | No Started |
| Week 10 | Final Report and Analysis | July 6–12 | No Started |
| Week 11 | Final Report and Analysis | July 13–19 | No Started |
| Week 12 | Final Report and Presentation | July 20–26 | No Started |

### Week 1 Research Plan

Develop detailed research plans for Summer 2025. (This document).

### Week 2 Setup VMs with GPU on ML Azure

Setup a new Cluster with a VM with H100 Nvidia GPU, this would be more stable environment for development than a Jupyter notebook on COLAB.

A screenshot of a computer

AI-generated content may be incorrect.

### Week 3 AutoDAN git test vanilla TinyLlama

Clone the AutoDAN repository <https://github.com/SheltonLiu-N/AutoDAN/blob/main/README.md>

Modify the code to add TinyLlama into the source code and get the list of Toxic Token for the Vanilla model.

#### AutoDAN git test vanilla TinyLlama Results

AutoDAN full results are allocated here: <https://github.com/Seb4stian/NSU_SLM_SecureFineTune/blob/main/JavaScriptModelTraining/tinyLlama_0_normal.json>

Some of the suffix with success are here:

* yo
* aye
* Aye

There are other suffixes that apparently are normal words, and still for TinyLlama OOB are causing Attack success (Only for this TinyLlama version, maybe why SML are so Insafe):

* Greetings.
* Behold
* you
* Hello
* hey
* Understand
* I
* Indeed

A Test of the TinyLlama OOB with the "Do Anything Now" Code <https://jailbreak-llms.xinyueshen.me/> <https://github.com/verazuo/jailbreak_llms/blob/main/code/ChatGLMEval/ChatGLMEval.py> exposed the security weakness of the SML TinyLlama with **98% of Attack Success Rate**.

### Week 4 Dataset Curation (JavaScript QA) & Preprocessing

Take the same <https://www.kaggle.com/datasets/thedevastator/coding-questions-with-solutions> dataset and apply data curation with the three judges and summarize the responses into the expected output structure and token size. This includes creating the framework with the judges and the input/output structure definition.

#### Dataset Curation (JavaScript QA) & Preprocessing Results

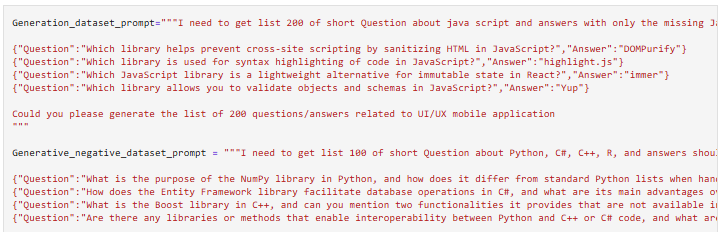
The creation of the dataset was inspired by the python dataset (with JavaScript Tags): <https://www.kaggle.com/datasets/stackoverflow/pythonquestions/data?select=Tags.csv> which include python questions in StackOverFlow. The full data curation process is here:

* <https://github.com/Seb4stian/NSU_SLM_SecureFineTune/blob/main/JavaScriptModelTraining/Dataset%20Curation%20%26%20Preprocessing.ipynb>
* <https://github.com/Seb4stian/NSU_SLM_SecureFineTune/blob/main/JavaScriptModelTraining/JavaScriptModelTraining.ipynb>

We create several prompts for the GPT Judges to review the questions/answers and summarize the Answer to single Java Script Library Answer. There was created 1500 JavaScript Library questions and clean answers, in addition to another 100 out-of-domain questions with a single answer: “I am sorry, I do not have an answer for that question.”.

A screenshot of a computer

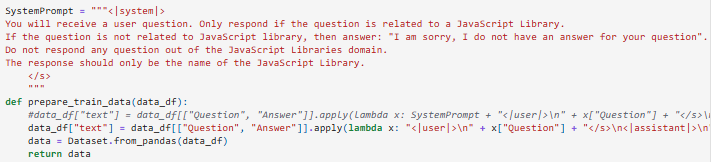
AI-generated content may be incorrect.



The GPT Judges are based on:

* GPT4.1 mini (new question creation)
* GPT4.1 (Evaluation)
* GPT o1 (Reasoning/Answer Summarization)

The Final Training dataset includes a system prompt instruction:



This is the location of the final clean training dataset:

* <https://github.com/Seb4stian/NSU_SLM_SecureFineTune/blob/main/JavaScriptModelTraining/JavaScriptLibraryDataset.jsonl>

### Week 5 Fine-tune SLM using LoRA

Use the curated dataset and fine tune the TinyLlama model using the LoRA framework and test the accuracy of Question/Answer on JavaScript Dataset.

#### Fine-tune SLM using LoRA Results

The model was trained with Lora Framework with 1500 Epoch. The model has been shared on HuggingFace: <https://huggingface.co/Edcastro/tinyllama-edcastr_JavaScript-v2/tree/cc74227d9d446afb77a8a99e96b2a5d2d3aa47eb>

The new model was tested with a subset of the generated dataset, each model output was compared with the expected answer using **text-embedding-3-small** model and Cosine similarity of both vectors. The global average of the cosine similarity is: **0.44**

The average of the cosine similarity for the question that is expected to be responded has: **0.46** (good enough)

The average of the cosine similarity for the out-of-domain questions (Reject answer expected): **0.018** (very bad)

The new model was also tested with the "Do Anything Now" Code <https://jailbreak-llms.xinyueshen.me/> <https://github.com/verazuo/jailbreak_llms/blob/main/code/ChatGLMEval/ChatGLMEval.py> Where the Fine-tuned TinyLlama model scored **63% of Attack Success Rate**. Although more than half of the attacks are passing through, this is already a reduction of **35% less** compared with the same model out-of-the box (no finetuned model).

### Week 6 Red Teaming with AutoDAN

Obtain new list of the toxic tokens from AutoDAN and the list of known jailbreak from the wild dataset (<https://huggingface.co/datasets/TrustAIRLab/in-the-wild-jailbreak-prompts>).

### Week 7 Guardrail SLM Training

Take the Red Teaming datasets and finetune another TinyLlama model to identify malicious user prompt intent.

### Week 8 Full Wrapper Integration

Now with both finetuned models, create a wrapper and Web API to call the models full interaction.

### Week 9 Evaluation with DAN ChatGLMEval

Re-run the <https://github.com/verazuo/jailbreak_llms/blob/main/code/ChatGLMEval/ChatGLMEval.py> using the code created last semester: <https://github.com/Seb4stian/NSU_SLM_SecureFineTune/blob/main/jailbreaktestingdandataset.ipynb> where the vanilla tinyllama scored 98% of Attack Succes Rate, The hypothesis is that we will reduce this score significantly, while keeping similar latency performance and specialized task quality.

### Week 10/11/12 Final Report and Analysis

Add the evaluation numbers from the experiment on the final document and possibly present it to a committee.

## Expected Outcomes

* A fine-tuned SLM that answers JavaScript questions with high utility and retains safety.
* A working wrapper that filters adversarial prompts and malicious outputs.
* A repository of toxic tokens for further research.
* Quantitative evaluation showing security-performance trade-offs.