# **GenAI for SD – N-gram Code Completion Lab**

This repository contains the code for the CSCI 420/520 lab “Recommending Code Tokens via N-gram Models” (Spring 2025). The goal is to implement a probabilistic N‑gram language model for Java code completion. The lab involves the following tasks:

1. **Corpus Construction & Preprocessing:**
   * Extract Java methods from GitHub repositories (student data) or use an instructor-provided corpus (teacher data).
   * Preprocess the corpus by removing duplicates, filtering non-ASCII characters, eliminating outlier or boilerplate methods, and tokenizing the Java code.
2. **Model Training & Evaluation:**
   * Train several N‑gram models (varying the context/window size).
   * Select the best model based on evaluation set perplexity.
   * Generate code completion predictions for 100 test methods.
   * Save the best model as a pickle file.
3. **Deliverables:**
   * A JSON file with predictions for the student model (results\_student\_model.json) and the teacher model (results\_teacher\_model.json).
   * A pickle file (trained\_model.pkl) containing the best-performing model.
   * A 1-page document and a README with instructions.

## **Repository Files**

* **extract\_methods.py**Extracts Java methods from a CSV file (for student data) or from a text file (for teacher data) and saves them to a CSV file.
* **train\_model.py**Preprocesses the input corpus, splits it into train/eval/test sets, trains multiple N‑gram models, selects the best one, generates predictions for 100 test examples, and saves the model as a pickle file.
* **evaluate\_model.py**(Optional) Evaluates different N‑gram configurations on the corpus and prints out perplexity values.
* **main\_lab.py (or Lab0.py)**A “clean” main script that reuses helper functions from the other files to run the complete lab pipeline.
* **training.txt**A sample teacher corpus (each line is a pre‑tokenized Java method).
* **results.csv**(Not included in the repository) – A CSV file containing repository names for student data.

## **Environment Setup**

### **Prerequisites**

* **Python 3.8+**
* **Git**

### **Setting Up on macOS or Windows**

**Clone the Repository:**bash  
CopyEdit  
git clone <repository\_url>

cd <repository\_folder>

**Create and Activate a Virtual Environment:**On **macOS/Linux**:  
bash  
CopyEdit  
python3 -m venv venv

source venv/bin/activate

On **Windows**:  
bash  
CopyEdit  
python -m venv venv

venv\Scripts\activate

**Install Dependencies:**Install the required packages using pip:  
bash  
CopyEdit  
pip install pandas pygments javalang pydriller

1. *Note: If you do not have Git installed on Windows, download and install it from* [*git-scm.com*](https://git-scm.com/)*. On macOS, Git is typically available via Xcode Command Line Tools.*

## **Generating the Extracted Methods CSV**

### **For Student Data**

1. Place your results.csv (a CSV containing repository names with a column named **name**) in the repository folder.

Run the following command to generate extracted\_methods\_student.csv:  
bash  
CopyEdit  
python extract\_methods.py --input results.csv --output extracted\_methods\_student.csv --mode student

1. This script will iterate through the repository names listed in results.csv, extract Java methods using PyDriller and javalang, and output a CSV with columns: *Commit Hash, File Name, Method Name, Method Code, Commit Link*.

### **For Teacher Data**

1. Ensure you have your training.txt file (each line is a pre-tokenized Java method).

Run the following command to generate extracted\_methods\_teacher.csv:  
bash  
CopyEdit  
python extract\_methods.py --input training.txt --output extracted\_methods\_teacher.csv --mode teacher

1. This will read each line from training.txt and write it as a row in the CSV (with a single column **Method Code**).

## **Running the Lab**

There are two main modes: **student** and **teacher**. Both modes run the same pipeline – the only difference is the input corpus.

### **Running the Student Version**

Use the CSV file generated from your student extraction (e.g., extracted\_methods\_student.csv).

bash

CopyEdit

python main\_lab.py --input extracted\_methods\_student.csv --mode student

This command will:

* Load and preprocess the CSV (removing duplicates, filtering, tokenizing, etc.).
* Split the data into training (80%), evaluation (10%), and test (10%) sets.
* Train several N‑gram models (with different context sizes).
* Select the best-performing model based on evaluation perplexity.
* Generate predictions for 100 test Java methods and save them to results\_student\_model.json.
* Save the best model as trained\_model.pkl.

### **Running the Teacher Version**

Use your instructor-provided text file (e.g., training.txt or the CSV extracted\_methods\_teacher.csv if already converted).

bash

CopyEdit

python main\_lab.py --input training.txt --mode teacher

This command will:

* Load the plain text file and split each line into tokens.
* Follow the same pipeline as above.
* Save the predictions to results\_teacher\_model.json and the model as trained\_model.pkl.

*Note: If you converted the teacher file to CSV using extract\_methods.py, then use that CSV file with mode teacher.*

## **Additional Evaluation**

If you want to inspect the perplexity of various N‑gram configurations on your corpus, run:

bash

CopyEdit

python evaluate\_model.py --input extracted\_methods\_student.csv

Replace extracted\_methods\_student.csv with your teacher CSV if needed.

## **Summary**

To complete the lab assignment:

1. **Extract Methods:**Generate your CSV files using extract\_methods.py (for student and/or teacher).
2. **Train & Evaluate:**Run main\_lab.py (or Lab0.py) with the appropriate input file and mode to train the N‑gram model, generate predictions, and save the model.
3. **Report Results:**The JSON output (e.g., results\_student\_model.json or results\_teacher\_model.json) and the pickle file (trained\_model.pkl) are the deliverables. Additionally, document the dataset creation process, model training methodology, evaluation results, and include links to your code in a 1‑page write-up.

If you have any questions or issues setting up the environment or running the scripts, please contact your teaching assistant.