

Assignment 4: Multilingual Knowledge Distillation

COL772/COL7372: Natural Language Processing - Spring 2026

Release Date: 10/04/2026 | **Due By:** 24/04/2026

Objective

You have to design and implement an end-to-end Knowledge Distillation (KD) pipeline. The core challenge is to extract the deep, latent reasoning capabilities (Chain-of-Thought) of a larger teacher model and successfully instill them into two distinct, smaller student models. A major focus of this assignment is preserving the quality and structure of reasoning across non-English languages. You are highly encouraged to explore, justify, and implement a non-standard or novel distillation strategy to address the cross-lingual and cross-architecture challenges. We will be evaluating your distillation strategy and performance on a private test set.

Models and Datasets

You will be working with the following models to understand the dynamics of knowledge transfer:

- **Teacher Model:**
 - Qwen2.5-7B-Instruct
- **Student Models:**
 - In-Family (Qwen2.5-1.5B-Instruct)
 - Cross-Family (LLaMA-3.2-1B-Instruct)

Your distillation training data has to be sampled from the provided `data/dataset.jsonl`. It contains MCQ-style questions from different languages and in the following languages: English, Hindi, Bengali, Kannada, and Tamil. The distribution of languages in the dataset is as follows:

Language	English	Hindi	Bengali	Kannada	Tamil
Instances	8200	4200	3100	2500	2000

Each line of the JSONL file contains the following fields for each instance: question, options, answer, subject, language, source. A sample instance from the dataset is given as follows:

```
{"question": "What is the value of log 0.0148?", "options": ["3.0000", "-0.1703", "-2.1703", "-0.8297", "-1.1703", "-1.8297", "0.0148", "1.8297", "0.1703", "2.0000"], "answer": "F", "subject": "stemez-Chemistry", "language": "en", "source": "mmlu_pro"}
```

Part A: Distillation Data Creation (Teacher Prompting)

For this part of the assignment, your primary goal is to build your training dataset entirely from scratch using samples from the benchmarks. You are not allowed to download a pre-distilled dataset.

1. **Sampling:** Write a robust script to sample questions from the provided dataset. You are allowed to sample only 10K samples for training. You can sample a different number of examples for each language, as long as the size of the training set does not exceed 10K. You may also create a validation set for tuning your training hyperparameters.
2. **Generation:** Once you have compiled your questions, your next step is to prompt the teacher model to solve them. You must explicitly instruct the teacher to generate a detailed reasoning trace before arriving at the final answer. To keep the training data structured, you may find it useful to have custom tokens to encapsulate this thinking process. You are allowed to generate at most 2048 tokens from the teacher, for each question. Reasoning in the same language as the question will fetch you full marks for this part.
3. **Formatting:** Format the outputs into a JSONL file. For each question, the distillation training instance should have the teacher generated response along with the original question and ground truth. The following format should be followed:

```
{"question": "Suppose $ 100 is deposited into a bank. ... total  
addition to demand deposits is $100.", "final_answer": "J",  
"gold_answer": "J", "language": "en", "subject":  
"stemez-Economics", "teacher_generation": "To solve this problem,  
we need ... #### ANSWER: (J) 1 and $100"}
```

Hint: Consider how you will handle incorrect answers from the teacher. Will you filter them out, or distill them anyway?

Part B: Knowledge Distillation (Training Pipeline)

In this section, you will develop the training pipeline to transfer the reasoning behavior to both student models.

1. **Architecture Handling:** Address the challenge of cross-architecture distillation. Since Llama-3.2 and Qwen2.5 have different tokenizers, classical knowledge distillation will require vocabulary projection if attempted. You will likely focus on text-based CoT distillation strategies (e.g., sequence-to-sequence loss on reasoning traces, or alternatives of your choosing).
2. **Training and Optimization:** Implement your chosen training loop. You may use Parameter-Efficient Fine-Tuning (PEFT) methods like LoRA/QLoRA if constrained by compute.
3. **KD Variants:** You can try offline distillation, online distillation or their combination for improving the student.

Part C: Student Inference and Evaluation

1. **Inference Script:** Write a custom inference script that loads a trained student model, accepts a raw prompt, and generates the reasoning trace similar to the teacher, followed by the final answer.
2. **Evaluation:** We will evaluate your distilled models on a held-out test set using accuracy. Ensure your evaluation metrics are clearly segmented and reported individually for English, Hindi, Bengali, Kannada, and Tamil. Student performance on the test set will be used for final grading.

Submission

Starter code is available [here](#). Implement the code for each part in the corresponding script. Ensure that your script supports the required CLI arguments, we will be using these during evaluation. Feel free to add or modify functions and additional CLI arguments in the individual scripts. Use the provided utility functions to generate text from the models using vLLM. Submit a single ZIP file named `[EntryNumber].zip` containing the following:

1. **dataset_generation.py:** Script to query the teacher and build the training corpus. You can select the samples on which you want to infer the teacher.
 - `--teacher_model`: HuggingFace model ID or path to the teacher model
 - `--num_samples`: Comma-separated sample counts for English, Hindi, Bengali, Kannada, and Tamil
 - `--output_file`: Path to save the final train.jsonl
2. **train_distill.py:** Your training/distillation strategy. You may use the teacher model in this code. You are allowed to use the full training set in addition to the teacher generated set.
 - `--teacher_model`: HuggingFace model ID or path to the teacher model.
 - `--student_model`: HuggingFace model ID or path to the student model.
 - `--train_data`: Path to training data (train.jsonl) generated by the teacher.
 - `--output_dir`: Path to save the distilled student model
3. **inference_eval.py:** Script to run the student models and calculate accuracy.
 - `--base_model`: Path to distilled student model
 - `--adapter_path` (optional): Path to LoRA adapter, if you use PEFT
 - `--test_data`: Path to test data (test.jsonl)
 - `--output_predictions`: Path to save predictions (predictions.jsonl)
 - `--report_file`: Path to save metrics (distilled student accuracy by language)
4. **predictions_<student_model>.jsonl:** Predictions from the student model along with the question, gold answer, and the parsed answer.

```
{"language": "en", "subject": "ori_mmlu-conceptual_physics",
"question": "The magnetic field lines about a current-carrying
wire form", "gold_answer": "H", "predicted_answer": "H",
"generation": "To determine the correct answer, we need ... ####
ANSWER: (H) "}
```

5. **metrics_<student_model>.txt:** Accuracy by language for the student model.

<LANGUAGE> ACCURACY: xx.xx

6. **run_distillation.sh**: Shell script that runs the entire pipeline - from training data generation to evaluation of the distilled student.
7. **requirements.txt**: All dependencies used. Allowed libraries: Python standard libraries, transformers, datasets, vllm, peft, trl, torch, tqdm, pandas, numpy. Any additional packages must be requested over Piazza and are permitted only after verification
8. **Project Report (Max 4 pages)**:
 - Detail your chosen distillation strategy, emphasizing its novelty or suitability for cross-lingual transfer.
 - Discuss the specific challenges of distilling into a cross-family model (Llama-3.2-1B) versus an in-family model (Qwen2.5-1.5B).
 - Present your evaluation metrics comparing base students, distilled students, and the teacher, with a comparative analysis of English vs. non-English performance.
 - You must include the **honour code** in the report: “Even though I have taken help from the following students and LLMs in terms of discussing ideas and coding practices, all my code is written by me.” -- List all students and LLMs you discussed with, or write “None”.

Evaluation Environment

We will evaluate your distilled student model on a private test set on an NVIDIA V100 node with a Python 3.10 environment.

Time Limits:

- Dataset Generation: 240 minutes
- Distillation: 240 minutes
- Evaluation: 60 minutes

Scripts that exceed these limits will be terminated.

Clarifications

Post all questions to the Piazza forum. It is your responsibility to seek clarification before the deadline.