

Assignment 3 Handout

Post-Training from a Base Model with SFT + RLVR

Scalable AI: Bridging Theory, Understanding, and Practice

EE 290 / 194 · Spring 2026

Released	March 13, 2026
Due	April 4, 2026
Where you work	Course-provided node, $8 \times \text{H100}$ per group. Use only this allocation.
Compute policy	No external compute. No extra GPUs. You may use the full two-week window on your assigned node.
Grading split	Part A: learning notebook, 20% Part B: final post-trained model quality, 80%
What you submit	Notebook artifacts for Part A + report, configs, logs, and evaluation outputs for Part B.
Checkpoint	Do not upload the checkpoint by default. You must save the final checkpoint and any adapter artifacts, and be able to provide them if we ask to verify evals.
How to submit	Submit a single ZIP on Gradescope.

High-level goal

In two weeks, your team will start from a **base language model** and produce a **post-trained model** using a two-stage recipe:

base checkpoint → **SFT** → **RLVR in NeMo Gym** → **evaluation**

This is an end-to-end post-training assignment: data design → SFT → RLVR → evaluation.

Starting checkpoint options. Your starting point must be exactly one of the following:

1. **Your own Assignment 2 base model** (the checkpoint you pre-trained from scratch), or
2. NVIDIA's released base checkpoint: [nvidia/NVIDIA-Nemotron-3-Nano-30B-A3B-Base-BF16](#).

You must start from a **base model**, not from an instruct, chat, reasoning, or otherwise post-trained checkpoint.

Important grading policy. We do **not** normalize for which starting checkpoint you choose. You will be evaluated on the **quality of the final post-trained model you produce** under the course compute budget. If your Assignment 2 base model is not your strongest starting point, you are free to use the NVIDIA base checkpoint instead.

Part A: Learning notebook, 20%

This part is about **post-training data engineering**, especially the design of synthetic data that can be useful for SFT. Use the provided **NeMo Data Designer lab**: `data-designer-lab.ipynb`. The notebook is the required hands-on exercise for this assignment.

Purpose. The point of this notebook is to practice the workflow for designing, previewing, inspecting, and exporting synthetic data. That workflow is directly relevant to post-training, even if your final Part B recipe uses a different or broader data mixture.

What to do:

- Run the notebook end-to-end.
- Answer the questions in the notebook.

- Save the requested outputs and short written responses.
- Export the artifacts referenced by the notebook (for example, generated datasets or summaries).

What to submit:

- The completed notebook file.
- Any generated artifacts referenced by the notebook.

Connection to Part B. You may use outputs from this notebook in your SFT stage for Part B, but you are not required to. If you do use them, document clearly how you converted or reformatted them for your final SFT pipeline.

Part B: Post-training project, 80%

Compute constraints

You must use only your course-provided node with $8 \times \text{H100}$. This is a fixed compute budget. You are expected to decide how to divide that budget across SFT, RLVR, evaluation, and iteration. There is no single correct allocation. We care about the final post-trained model quality you achieve within the available compute.

Choose a coherent target capability

Your goal is a **broadly post-trained model**: one that is competent across many capability verticals, not narrowly specialized in a single task. Think of your target as producing a general-purpose chat or instruct model that performs well on a wide range of evaluations—reasoning, instruction following, coding, knowledge, tool use, etc.—rather than maximizing a single benchmark.

How you evaluate is part of your grade. The breadth and rigor of your evaluation suite matter. We expect you to select benchmarks that cover multiple capability dimensions and to justify why those benchmarks collectively demonstrate the quality of your post-trained model. A narrow evaluation that only measures the specific reward you trained on will not receive full credit.

Look at how leading labs evaluate. Study the evaluation suites used in recent model releases such as Qwen, DeepSeek, and Nemotron. These releases demonstrate what a comprehensive, multi-vertical evaluation looks like for broadly capable models. Use them as a reference when designing your own benchmark suite.

Required post-training recipe

1. **Mandatory SFT stage in NeMo RL.** You must run **supervised fine-tuning (SFT)** using **NeMo RL**. Your SFT data can be synthetic, human-written, public, private, or mixed, but you must document all sources and preprocessing. If your raw data is not already chat-formatted, you must convert it into the format expected by your NeMo RL pipeline and describe that conversion.
2. **Mandatory RLVR stage in NeMo RL.** After SFT, you must run **RL with verifiable rewards (RLVR)** using **NeMo RL** with environments released in NeMo Gym. A smoke test alone does *not* satisfy this requirement: the RLVR stage must produce a non-trivial updated checkpoint and measurable validation results. You should train on **as many NeMo Gym environments as you can support** to build broad capability.
3. **DPO or other extras are optional, not substitutes.** Preference optimization, DPO, reward-model training, distillation, or other extra stages are welcome, but they do **not** replace the required **SFT** stage or the required **RLVR** stage. But it can go a long way in improving your model quality.

4. **Base-model policy.** If you use your Assignment 2 checkpoint, you are responsible for any conversion, wrapping, or loading steps needed to train it in your NeMo RL stack. If you use the NVIDIA checkpoint, it must be the **base** checkpoint linked above, not a post-trained derivative.
5. **Final artifact retention.** Keep the final checkpoint and any tokenizer, chat-template, adapter, or merge artifacts needed to reproduce your results. If you train with adapters, you must clearly explain how to load or merge them.

Recommended default path. The simplest acceptable recipe is:

base model → **SFT in NeMo RL** → **GRPO in NeMo RL on NeMo Gym environments** → **evaluation**

You are welcome to use a different RLVR algorithm supported by NeMo RL (or contribute a PR to NeMo RL for a novel one), but the burden is on you to make it work and document it clearly.

What we grade

We grade Part B on **final post-trained model quality**, as demonstrated by the benchmark suite you choose and justify. The starting checkpoint itself is **not** separately graded. We are also **not** directly grading throughput, MFU, or raw tokens/day. However, systems efficiency still matters indirectly because it determines how much useful SFT and RLVR you can fit into the fixed compute budget.

Required evaluations

You must evaluate the **same benchmark suite** on:

1. the starting base model,
2. the post-SFT checkpoint, and
3. the final post-SFT+RLVR checkpoint.

At minimum, your evaluation suite must include:

- **At least a few external benchmarks spanning different capability verticals** (e.g. reasoning, instruction following, coding, knowledge, tool use). Your final score should not be justified purely by optimizing the training reward itself.

The more comprehensive your evaluation suite, the better. A model evaluated on a broad, well-justified set of benchmarks will receive more credit than one evaluated on a single narrow metric, all else being equal.

Examples of evaluation dimensions to consider:

- held-out environment validation aligned with the RLVR reward,
- math or reasoning benchmarks (e.g. AIME, MATH),
- coding benchmarks (e.g. HumanEval, MBPP, LCB),
- instruction-following or schema-following benchmarks,
- tool-calling or agentic evaluations (e.g. Tau),
- knowledge or general QA benchmarks,
- text-to-SQL

We recommend looking at the evaluation suites of recent model releases such as Qwen, DeepSeek, and Nemotron to get ideas.

Deliverables and submission format

Submit a single ZIP on Gradescope.

```
submission/
• partA/
  - completed data-designer-lab.ipynb
  - any notebook artifacts
• partB/
  - report.pdf
  - base_model_choice.txt with the exact starting checkpoint and any conversion steps
  - sft/ SFT data manifests, configs, logs, and checkpoint or adapter information
  - rl/ RLVR configs, environment choices, logs, and checkpoint or adapter information
  - eval/ raw evaluation outputs and an eval_summary.pdf
  - repro/ one command or script for SFT, RLVR, and evaluation
  - checkpoint_availability.txt describing where the final checkpoint is saved and how to provide it if requested
  - Wandb links or tensorboard files for your training runs for SFT/RLVR/any other stage — make sure they are accessible
```

Required report structure

Keep it concise: **2–6 pages**, appendix allowed.

1. **Starting base model and target capability.** State which base checkpoint you used, why you chose it, and what capability bundle you targeted.
2. **SFT data.** Describe your SFT mixture, how it was collected or generated, and why it matches your target capability. If you used Data Designer outputs, explain how.
3. **SFT recipe.** Describe chat formatting, templates, key hyperparameters, and any parameter-efficient adaptation choices. Include training curves.
4. **RLVR recipe.** Describe the NeMo Gym environment(s), the NeMo RL algorithm, the reward or success signal, and the key hyperparameters. Explain why this RLVR stage should improve the target capability. Include training curves.
5. **Benchmark suite and justification.** Explain why your chosen benchmarks are the right ones for this model. Be explicit about which metrics are held-out validation versus external evaluation.
6. **Results.** Provide a table comparing the **base**, **after SFT**, and **final after RLVR** checkpoints on the same benchmark suite.
7. **Failure analysis and reproducibility.** Discuss what did not work, what you would change next, and include exact reproduction details.

Grading rubric

Component	Weight
Part A: learning notebook	20%
Part B: final post-trained model quality on your chosen benchmark suite	80%
Extra credit: open-source contributions to NeMo Gym / NeMo RL	up to 30%

Extra credit (up to 30%): contribute to NeMo Gym or NeMo RL

We **strongly encourage** open-source contributions. Opening a pull request into **NeMo Gym** or **NeMo RL** is an excellent way to earn up to **30% extra credit**.

Examples of high-value contributions:

- A new, interesting NeMo Gym **environment** that trains a capability not already covered by the released environments.
- Bug fixes, documentation improvements, or feature additions to **NeMo RL** or **NeMo Gym**.

Your contribution should be functional, well-documented, and submitted as an actual PR. Include a link to the PR in your report. Contributions that are merged or receive positive maintainer feedback will receive the most credit.