NLP Final Project Proposal

Final Project Proposal — *Modern LLM: Default Project, Frontier Additions*

Student: Ayman Mahfuz

Course: CS 371N — Natural Language Processing (Fall 2025)

Project Type: Custom final project fully satisfying the “Default Project” structure while extending it with frontier-level methods and evaluations

Timeline: Oct 24 – Dec 11, 2025

Hardware: 64 GB RAM, 22-core CPU, RTX 3060 (12 GB)

---

1) Overview and Motivation

This project will implement a **modern decoder-only language model** from scratch and evaluate it against several finetuned Hugging Face baselines. It satisfies the *Default Final Project* requirements—(1) a custom mini Transformer, (2) pretrained HF models finetuned on downstream tasks, and (3) rigorous evaluation and analysis—while extending them with contemporary frontier innovations.

The goal is to reproduce, at small scale, the essential features of modern LLMs: advanced normalization and activation, rotary embeddings, alignment via preference optimization, self-verification, and efficient finetuning. The final deliverables will include a runnable codebase, reproducible experiments, and an ACL-format report designed to meet both course and professional research standards.

---

2) Default Project Requirements and Compliance

The default CS371N final project expects:

1. **A mini Transformer implementation** trained on a language modeling task (e.g., WikiText-2 or TinyStories).

2. **A pretrained Hugging Face model** finetuned on a downstream task (e.g., classification or summarization).
3. **Evaluation and analysis**, including quantitative metrics (perplexity, accuracy, ROUGE) and qualitative insights (error and interpretability analysis).

Compliance:

This project will meet all three requirements. My from-scratch model will serve as the mini Transformer; I will finetune and evaluate Hugging Face models such as GPT-2, DistilGPT2, TinyLlama, and T5-small on tasks like SST-2, SAMSum, and GSM8K. The results will be compared along multiple axes—accuracy, perplexity, parameter efficiency, and verifier impact—accompanied by attention visualizations and ablations.

---

3) Frontier Techniques and Additions

This project expands the default assignment by incorporating high-level architectural and training innovations used in state-of-the-art systems. No low-level kernel optimizations are included.

Baseline feature set (must-deliver)

1. **RoPE (Rotary Positional Embeddings)** — replaces sinusoidal embeddings; standard in LLaMA and GPT-NeoX.
2. **RMSNorm + SwiGLU blocks** — improves stability and efficiency (used in PaLM and LLaMA).
3. **Attention sinks for streaming and long-context stability** — allows extrapolation beyond trained window lengths.
4. **Parameter-efficient finetuning (LoRA/QLoRA)** — enables efficient adaptation of HF models within 12 GB GPU memory.
5. **Prompting baselines** — compare zero- and few-shot prompting to finetuned performance.
6. **Direct Preference Optimization (DPO)** — performs preference alignment on a small-scale dataset without full RLHF complexity.

7. **Universal-style verifier** — lightweight model trained to score or rerank outputs for math and QA tasks, enabling post-hoc correction.

Strong stretch goals (attempt after baseline is stable)

- **Grouped Query Attention (GQA):** reduce KV cache memory by sharing keys/values among heads.
- **Mixture-of-Experts (MoE):** top-1/2 routing for selective activation in feedforward layers.
- **Test-time compute scaling:** self-consistency (sample N outputs, majority vote) or verifier-gated "revise and retry."
- **Instruction tuning (SFT):** supervised instruction pass before DPO to form a complete SFT → DPO → verifier alignment pipeline.
- **Structured outputs:** calculator/tool stub for symbolic math problems.

These features replicate the development trajectory of frontier LLMs—SFT for instruction following, DPO for preference alignment, verifier for reflection, and architectural efficiency via RoPE, RMSNorm, SwiGLU, and GQA.

---

4) Hugging Face Finetuning Plan (Comparison Framework)

To satisfy the requirement of using pretrained HF models and comparing them with the mini Transformer, I will finetune several small open models using LoRA/QLoRA.

Models to finetune:

- GPT-2 small (124M) and/or DistilGPT2 (82M)
- TinyLlama (1.1B, quantized with QLoRA if feasible)
- FLAN-T5-small for encoder-decoder comparison

Tasks and evaluation:

- **Classification (SST-2):** compare scratch model vs. GPT-2 (finetuned) vs. GPT-2 (prompted).

- **Summarization (SAMSum):** compare T5-small (finetuned) vs. scratch model (prompted).
- **Math reasoning (GSM8K subset):** compare all models before and after verifier reranking.

Comparison dimensions:

- Quantitative: Accuracy, ROUGE, Exact Match (math)
- Qualitative: error categories, attention behavior, verifier effects
- Efficiency: parameter count, convergence rate, GPU hours
- Alignment effects: DPO or verifier improvement over base finetune

This ensures a rigorous “scratch vs. pretrained” comparison aligned with course expectations.

---

5) Evaluation Benchmarks and Metrics

Core benchmarks (from default project):

- **Language Modeling:** WikiText-2 or TinyStories (Perplexity)
- **Classification:** SST-2 (Accuracy, F1)
- **Summarization:** SAMSum (ROUGE-1, ROUGE-L)
- **Qualitative analysis:** attention visualization, misclassification examples, interpretability probe

Math benchmarks (verifier-aware):

- **GSM8K subset (~500 examples):** report Raw EM, Verifier-reranked EM, and Pass@N.
- **SVAMP/MAWPS:** arithmetic word problems for ablations.
- **Symbolic arithmetic:** synthetic expressions like `(2+3)*4`; verifier checks with programmatic evaluation.
- **Analysis:** report when the verifier improves or harms performance; measure false acceptance/rejection rates.

---

6) Hardware-Aware Training Plan (Maximizing the 3060)

Target: Train as large a model as possible while maintaining stability on a single 12 GB GPU.

Optimization strategies:

- Mixed precision (FP16/BF16) for 2× memory savings
- Gradient checkpointing and accumulation (micro-batch=2, accumulate=16)
- CPU offloading for optimizer states when needed
- Efficient scaled dot-product attention (no FlashAttention dependency)

Expected model sizes:

- **Baseline:** 50–80M parameters (1–2K context) — stable, reproducible
- **Aggressive:** 120–160M parameters (comparable to GPT-2 small) — heavy optimization
- **Stretch:** 200M+ parameters (only if stable training achieved)

HF finetunes: QLoRA-based finetuning on GPT-2, DistilGPT2, and optionally TinyLlama.

Attention sinks experiment: demonstrate generation beyond trained length (train at 1K, generate 4K+ tokens) to show extrapolation stability.

---

7) RL and Alignment Pipeline (Course-Aligned)

This component integrates content from “Prompt Engineering,” “RLHF,” “DPO,” and “Alignment” lectures.

Three-stage alignment:

1. **Supervised Finetuning (SFT):**

   A small instruction-tuning pass on curated QA/dialog data to improve adherence to task prompts.

2. **Direct Preference Optimization (DPO):**

Train on a pairwise preference dataset (Anthropic HH-RLHF subset or synthetic).

Single-GPU implementation—no reward model or PPO required.

3. **Verification & Self-Correction:**

   Train a verifier on labeled math/QA pairs.

   Use it to rerank or trigger a second pass for low-confidence generations.

Evaluation:

Quantitatively compare base → SFT → DPO → +Verifier pipelines on downstream and math tasks. This will demonstrate incremental improvements from each alignment stage and show an understanding of the full modern RLHF/DPO/Verifier stack.

---

8) Datasets

- **Language Modeling:** WikiText-2 or TinyStories
- **Classification:** SST-2
- **Summarization:** SAMSum
- **Math:** GSM8K subset, SVAMP, MAWPS
- **Preference optimization:** synthetic or open pairwise preference data

  All datasets are small, open-source, and feasible on the proposed hardware.

---

9) Baseline vs. Stretch Goals

Baseline Deliverables (guaranteed):

- From-scratch Transformer with RoPE + RMSNorm + SwiGLU
- Attention sinks demonstration
- Finetuned HF models on multiple tasks
- Prompting vs. finetuning evaluation
- DPO run on small preference dataset

- Verifier applied to math/QA tasks
- Full evaluation suite and interpretability analysis

Stretch Deliverables (attempt if time allows):

- GQA or MoE layer integration
- Instruction-tuning pre-stage (SFT)
- Self-consistency or verifier-gated rethinking
- Structured outputs for calculator-style verification

---

10) Timeline (Milestone-Driven)

Week 1 (→ Nov 6 Proposal)

- Base Transformer with RoPE, RMSNorm, SwiGLU implemented and training on WikiText-2/TinyStories
- Attention sinks demonstration (generation >2K tokens)
- First HF finetune (GPT-2 LoRA on SST-2)
- Deliverable: proposal submission and working repo with reproducible training

Week 2 (→ Nov 18 Progress Report)

- All HF finetuning complete (SST-2, SAMSum, GSM8K subset)
- Prompting vs. finetuning comparison tables
- DPO run finished and verified
- Verifier trained and active in math reranking
- Initial attention visualizations and error analysis
- Deliverable: progress report and repo link

Weeks 3–4 (→ Dec 11 Final Submission)

- Ablations (with/without RoPE, sinks, DPO, verifier)
- Optional stretch goals (GQA, MoE, instruction tuning)
- Final evaluation tables, interpretability results, polished report

- Deliverable: ACL-format final report and complete repo with reproducibility script

---

11) Success Criteria and Evaluation Rubric

Minimum Viable (baseline completion):

- Scratch Transformer reaches reasonable perplexity (within 2× GPT-2 small)
- HF models finetuned and compared on at least two tasks
- At least three frontier features functional (RoPE, RMSNorm/SwiGLU, attention sinks)
- DPO implemented and logged
- Verifier improves math EM by ≥5%
- Complete report and reproducible repository

Strong Performance (target):

- Scratch model competitive with GPT-2 small on one task
- Verifier achieves 10–15% EM boost on GSM8K subset
- Attention sinks stable up to 4K+ context length
- At least one stretch goal implemented (GQA or SFT)

Outstanding (portfolio-grade):

- All baseline + multiple stretch goals
- Clear novel findings from ablations (e.g., RoPE doubles length extrapolation)
- Professional documentation, clean figures, one-button reproducibility
- Work demonstrates deep understanding of frontier LLM architecture and alignment pipelines

---

12) Deliverables Summary

- **Code:** runnable scripts for training, finetuning, DPO, verifier, and evaluation
- **Documentation:** clear README with setup and reproduction instructions

- **Reports:** proposal, progress report, final ACL-format paper (8–12 pages)
- **Evaluation:** tables for all tasks, error analysis, attention visualization, and verifier impact
- **Demo notebook:** optional interactive generation interface

---

13) Concluding Vision

This project is intentionally designed as “the default project—modernized.” It fulfills every required course element while layering frontier-level techniques that reflect how today’s top AI labs build and align their models. The combination of architectural innovations (RoPE, RMSNorm, SwiGLU), alignment strategies (SFT, DPO, verifier), and rigorous evaluation (HF comparisons, math benchmarks, ablations) creates a comprehensive demonstration of both technical competence and research maturity.