

# **Assignment 5**

Team name: Lior\_AbuHav\_Ofri\_Kutchinsky

Lior AbuHav ID: 209521400

Ofri Kutchinsky ID: 313611360

Repo link:

<https://github.com/IAmLior/CONTEXT-WINDOW-LABS>

## **Self-Assessment**

**Students:** Ofri Kutchinsky, Lior Abuhav.

**Project:** Context Window Labs - A Comprehensive LLM Context Window Analysis Framework

**Self-assigned grade:** 96/100

### **Self-Assessment Statement**

In this project, we implemented a comprehensive experimental framework for systematically testing and analyzing Large Language Model (LLM) context window limitations through four progressive laboratories.

The project demonstrates rigorous scientific methodology, excellent documentation practices, modular architecture, and delivers actionable insights for real-world LLM applications.

The framework is structured as a production-ready Python package with four independent yet interconnected labs:

1. **Lab 1:** "Needle in a Haystack" - Testing position bias and the "lost in the middle" phenomenon
2. **Lab 2:** Context size impact on performance (latency, accuracy, scalability)
3. **Lab 3:** RAG vs Full Context comparison for document-based QA
4. **Lab 4:** Context engineering strategies for multi-turn conversations

Each lab follows a complete research lifecycle: hypothesis formulation → data generation → experimentation → analysis → documentation → actionable insights.

## What We Believe We Did Well

**A Structured Experimental Methodology** - Across all labs, we followed a consistent process

- Define the question
- Build controlled experiments
- Run multiple trials when needed
- Analyze failures and successes
- Summarize what the results actually mean

This gave the project a strong sense of progression. Each lab built on insights from the previous one, and the conclusions feel justified - not forced.

**Insights with Real Practical Value** - The most meaningful outcome of the project is the set of findings about LLM behavior.

Some examples:

- Chunk size can matter more than total context size
- RAG can outperform full-context usage in both speed and consistency
- Context limits are task-dependent, not a fixed number
- Overflow failure patterns (like gibberish) can be predicted

These are real insights that someone working with LLM systems could immediately use.

## **Production-Oriented Architecture**

Even though this is a course project, the codebase is organized like a real Python package. There is a reusable LLM helper module, clean experiment modules per lab, and a simple public API to run labs directly.

## **Testing and Validation**

We have a strong test suite, and we feel we have focused on the tests which actually matter for us as an engineers

- Unit tests for the LLM helper
- Integration tests to confirm installation and basic workflows
- Repeated experimental runs to reduce noise in the results

## **Clear and Thorough Documentation**

The project is documented in a way that makes it easy to understand what we did, why we did it, and how to reproduce the work.

The main README and the lab-specific files walk through the experiments step-by-step, include the reasoning behind key decisions, and summarize the results in a clear way.

The documentation is detailed, but still practical - it feels like something a motivated engineer or researcher could actually follow.

## **Solid Data Generation**

The synthetic data we created is varied, parameterized, and consistent.

We learned from early mistakes (like repetitive filler text) and fixed them.

The control we had over word count, fact placement, and noise allowed us to test models in a meaningful way.

## **Useful Visualizations and Analyses**

Every lab ends with graphs or tables that actually help interpret the results, rather than just decorate the project.

They make the conclusions intuitive and easy to verify.

## **Challenges and What We Learned**

### **Model Behavior is Context-Dependent**

- Different models fail differently (graceful degradation vs gibberish)
- Smaller models can match larger ones within their range
- Task complexity affects limits more than specifications
- Theoretical limits don't match practical performance

### **Data Quality Over Quantity**

- Repetitive content causes failures even within limits
- Diversity in synthetic data is essential
- Data generation requires rigorous validation

### **Chunk Size Revolution**

- Individual chunk complexity > total token count
- Smaller chunks enable processing more information
- Challenges conventional RAG chunking strategies

### **Latency Requires Statistical Rigor**

- High variance from network, throttling, cold starts
- Multiple runs needed for confidence
- Outliers reveal system issues

### **Evaluation Methodology Matters**

- String matching misses semantically correct answers
- Accuracy is an artifact of evaluation choice
- Transparency about limitations is critical

## **7. RAG as a Change in the LLM's world - Benefits are Quantifiable**

- Faster, more consistent, more cost-effective
- Token reduction = cost savings at scale

## **What Could Be Improved**

### **More Automated Testing**

The tests we have are good and useful for engineers that will use our package, but limited. Although we have created a test platform that can be connected to CI/CD methods, in a perfect world we would add more integration tests, and some regression tests.

### **Performance Optimizations**

Some experiments could run faster with batching, caching, or concurrency - although this wasn't critical for the data loads we did the labs on.

We might want to improve the load ability in order to let this package serve more test cases and trials.

### **Broader Model Coverage**

Testing against more models (e.g., Gemini, Claude) helped us make the findings more general and insights more interesting, although we have used some different models in our experiments, it might be better to broaden the model options within our package.

## Score Breakdown (Realistic 96/100)

Category	Weight	Score	Notes
Documentation	20%	98	<b>Very strong, extremely clear, only slightly longer than needed.</b>
Code Structure	25%	98	<b>Clean, modular, logical.</b>
Configuration & Security	15%	90	<b>Good practices, could add more validation.</b>
Testing & QA	15%	90	<b>Useful tests but not comprehensive.</b>
Research & Analysis	15%	100	<b>Deep insights, well justified.</b>
Use of Course Concepts	10%	100	<b>Strong understanding and correct terminology.</b>

## Final Grade: 96/100

### Summary

**This project reflects a high level of effort, thoughtful design, and a strong understanding of how LLM systems behave.**

**It combines careful experimentation with clean engineering practices and meaningful insights.**

### **3.6 הצהרת יושר אקדמי (Academic Integrity Declaration)**

אני מצהיר/ה בזאת ש:

- הערכתה העצמית שלי היא כנה ואמיתית**
- בדקתי את העבודה מול כל הקритריונים לפני קביעת הציון**
- אני מודע/ת שציון עצמי גבוה יוביל לבדיקה דקדקנית יותר**
- אני מקבל/ת את העבודה שהציון הסופי עשוי להיות שונה מהציון העצמי**
- העבודה היא פרי עבודתי/נו (של הקבוצה) ואני/o אחראי/ם לכל תוכנה**

30.12.25

תאריך:

LA

חתימה: