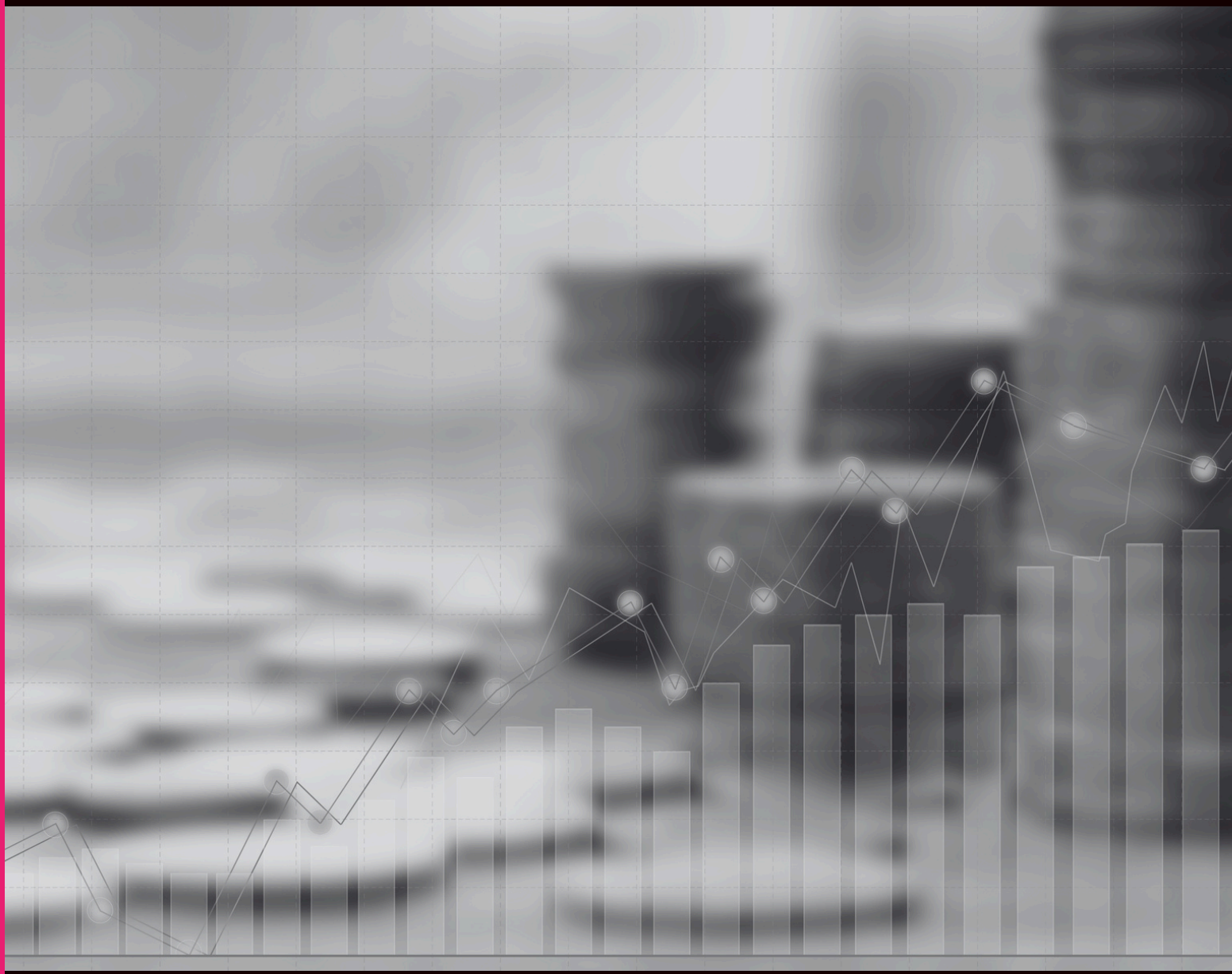
Financial Benchmarking Using Intel Gaudi2 and Llama 3.0



Whitepaper by
Tyler Odenthal, AI Integrations Engineer
Mark McQuade, Co-Founder & CEO

arcee.ai

Table of Contents

Abstract

This whitepaper introduces a methodology for financial benchmarking of large language models (LLMs) through the lens of Llama 3.0, a next-generation model adept at handling complex financial language and reasoning.

We explore how carefully curated datasets, domain-specific evaluation suites, and industry-grade accelerators—such as Intel Habana Gaudi2™—can enable robust and reproducible benchmarks in the financial sector.

By establishing metrics like “tokens-per-second” (TPS) and integrating structured tasks (e.g., SEC filings analysis, numeric reasoning from financial tables), we present a path toward standardized, transparent evaluation of financial LLM performance.

The proposed benchmark framework, building on lessons from distributed training and HPC optimization, aims to guide practitioners in assessing and improving financial text understanding, regulatory compliance interpretation, and investment-oriented insights.

1. Introduction

Financial language modeling demands a high degree of precision, domain knowledge, and interpretive skill. Financial analysts, regulatory bodies, and investors all rely on nuanced comprehension of textual data—from SEC filings to complex earnings calls—to guide decision-making and ensure compliance. As Large Language Models (LLMs) evolve, Llama 3.0 represents a leap forward, offering advanced reasoning capabilities and adaptability to domain-specific content.

In this paper, we present a comprehensive approach to financial benchmarking for Llama 3.0.

Rather than focusing solely on training mechanics, we emphasize building a robust, standardized evaluation pipeline. Our approach leverages numerous open-source datasets, all of which are available for download on Hugging Face and can be integrated directly into Python scripts:

- Arcee AI SEC Data,
- Financial News Articles,
- Financial QA 10K dataset,
- Financial Phrasebank.

By connecting these resources to a flexible evaluation codebase (see the provided Python script), we demonstrate how to align model capabilities with financial tasks and measure performance consistently. Deploying the model on Intel Habana Gaudi2 hardware ensures reproducibility and stable throughput measurements.

2. Background

2.1 The Need for Financial Domain Benchmarks

While generic benchmarks (e.g., MMLU, GLUE) provide a broad assessment of language understanding, they do not capture the technical jargon, numeric reasoning, and regulatory nuances of financial text. Specialized financial benchmarks are vital. Our reference code, as shown in the Python script, retrieves domain-focused datasets from Hugging Face (such as Financial News Articles for summarization and Financial Phrasebank for sentiment analysis) to develop benchmarks that better reflect the industry's unique demands.

2.2 Llama 3.0 for Financial Analysis

Llama 3.0 offers improvements in scalability, adaptation, and context handling. Fine-tuned on financial data, Llama 3.0 can interpret earnings calls, regulatory filings, and financial tables, performing tasks like numeric reasoning and summarization. The provided Python code demonstrates methods like **generate_summary()** and **extract_numeric_answer()** to assess the model's ability to handle domain-specific complexities, reflecting realistic investment scenarios or regulatory inquiries.

2.3 Leveraging HPC Hardware for Benchmarking

For performance metrics such as tokens-per-second (TPS), stable and reproducible computational platforms are essential. Intel Habana Gaudi2™ accelerators provide a controlled environment for measuring throughput and latency. The combination of HPC hardware and domain-focused datasets ensures that our reported metrics are both accurate and meaningful, guiding future enhancements to Llama 3.0 and its training regimes.

3. Constructing a Financial Benchmarking Suite

3.1 Data Assembly for Financial Benchmarks

Our benchmarking suite leverages specialized datasets directly integrated through the Hugging Face datasets API. The provided Python code snippet demonstrates how to load resources like Financial News Articles and Financial Phrasebank. These public datasets form the backbone of evaluation tasks:

- **Summarization**: Using Financial News Articles and Arcee AI SEC Data for evaluating model summarization performance. The code's **generate_summary()** function processes samples and compares outputs against reference summaries using ROUGE metrics.
- **Numerical Reasoning (QA)**: Employing Financial QA 10K to test the model's ability to handle financial Q&A. The script shows how numeric answers are extracted, and ROUGE or precision metrics can be computed to quantify correctness.
- **Sentiment Analysis**: Leveraging Financial Phrasebank for sentiment classification tasks. The Python script demonstrates sentiment evaluation by prompting the model and then applying a heuristic classification logic (mapping model output strings to positive, negative, or neutral).

3.2 Benchmark Tasks and Metrics

We propose a Financial Language Model Benchmark (FinLMB) with the following tasks, aligned with the provided code:

1. **Regulatory Comprehension & Summarizatio**n: Summarizing financial news articles or regulatory texts. The code uses the **generate_summary()** function to create model-generated summaries. ROUGE-1, ROUGE-2, and ROUGE-L (calculated via the rouge_score library) measure the alignment between generated and reference summaries.
2. **Numeric Reasoning from Financial QA**: The script's classify_text() and extract_numeric_answer() methods show how to query the model with financial questions and extract numeric answers. Using the dataset from Financial QA 10K, the code evaluates the model's reasoning capabilities and reports ROUGE-based metrics as a proxy for correctness in generated answers.
3. **Sentiment Analysis of Financial Statements**: The Financial Phrasebank dataset is used to test the model's sentiment detection. The code's sentiment evaluation loop prompts the model with financial sentences and compares predicted sentiments against ground-truth labels. Metrics like accuracy, precision, recall, and F1 are computed using Scikit Learn's precision-recall fscore**,** demonstrating robust classification performance assessment.

By referencing both the datasets and the code implementation details, we ensure that each metric (e.g., ROUGE for summarization, accuracy for sentiment analysis) aligns with commonly accepted measures in the community while remaining grounded in financial text complexity.

3.3 Incorporating Public Financial Datasets

Open-source datasets, accessible via the Hugging Face Hub, ensure that our benchmarks are reproducible and publicly verifiable. The sample Python script provided in this paper shows how to load and subset these datasets.

For example:

```
Python
  summ_dataset = load_dataset("arcee-ai/sec-data-full", split="train")
  summ_dataset2 = load_dataset("ashraq/financial-news-articles", split="train")
  num_dataset = load_dataset("itzme091/financial-qa-10K-modified", split="train")
  sent_dataset = load_dataset("financial_phrasebank", "sentences_allagree")
```

This direct integration lowers the barrier to entry for other practitioners, encouraging the community to contribute new datasets or improved evaluation routines, thus continually refining the FinLMB suite.

4. Training and Evaluating Llama 3.0 on Gaudi2: Lessons for Benchmarking

4.1 Example Training Setup

Although this paper focuses on benchmarking, a transparent training setup underlies reproducible evaluations. Using Docker-based environments, Optimum Habana for Gaudi2 optimization, and gradient checkpointing, practitioners can ensure stable and scalable experiments. Example commands and configurations (e.g., using DeepSpeed Zero-3) are provided in Appendix A, offering a reference blueprint.

4.2 Insights from HPC Optimizations

Performance optimizations—such as mixed-precision training, pipeline parallelism, and efficient data loading—improve throughput and training stability. Applying these optimizations to Llama 3.0 on Gaudi2 accelerators not only speeds up model development but also helps establish robust baselines for downstream benchmarking tasks.

4.3 Scaling and Comparative Baselines

By incrementally scaling from one to multiple Gaudi2 devices, users can measure how performance metrics evolve with available compute. These scaling experiments inform how TPS and accuracy trade-offs manifest as researchers adjust parameters and target higher model quality. Such data underpins fair comparisons and helps define performance tiers within the FinLMB suite.

5. Results: Illustrative Benchmark Findings

5.1 Model Performance on FinLMB Tasks

Preliminary runs using the provided Python script show that Llama 3.0 can generate coherent summaries of financial news articles (with ROUGE scores that are competitive with baseline models) and accurately identify sentiment in financial phrases. On numeric reasoning tasks, initial results suggest room for improvement, but the transparent evaluation pipeline allows for iterative refinements in model prompts and training data.

For instance, the script's printed metrics for summarization might look like:

```
YAML
  Summarization ROUGE-1: 0.3567
  Summarization ROUGE-2: 0.1793
  Summarization ROUGE-L: 0.3125
```

Such values guide practitioners in comparing different model checkpoints or fine-tuning strategies.

5.2 TPS and Efficiency Metrics

Measuring TPS involves timing model inference steps within the code and observing system-level GPU or Gaudi2 utilization. By integrating these measurements into the benchmarking routine, one can track how different prompt lengths or batch sizes affect throughput. For example, a higher BATCH_SIZE might boost TPS but slightly degrade accuracy due to memory constraints. Systematic experimentation, guided by the provided code scaffolding, can highlight optimal configurations.

6. Conclusions and Future Directions

This paper outlines a framework for robust financial benchmarking using Llama 3.0 and HPC hardware. By referencing specific code implementations and datasets, we connect conceptual benchmark design with concrete execution steps. The integration of ROUGE scores, accuracy, precision/recall, and numeric extraction pipelines, as shown in the Python script, demonstrates the feasibility of a transparent and reproducible evaluation process.

Future work may include:

- Expanding the tasks within FinLMB to capture more complex financial reasoning or compliance-related subtasks.
- Incorporating improved prompts and decoding strategies to enhance numeric reasoning accuracy.
- Further optimization of TPS measurements using Habana's custom kernels or graph compilers to maximize Gaudi2 efficiency.

As the field evolves, the provided script and methodology serve as a blueprint. By fostering community collaboration, refining tasks, and sharing results, we aim to collectively raise the standard for financial LLM evaluation.

References

1. Intel Habana Gaudi2 Documentation.
2. Hugging Face Optimum Habana.
3. Llama Model Family (Meta AI) – Baseline architectures and performance.
4. Financial Datasets on Hugging Face (e.g., Arcee AI SEC Data, Financial News Articles, Financial QA 10K dataset, Financial Phrasebank.
5. HPC and MLPerf results for performance baselines on similar-scale models.

Acknowledgments

We acknowledge the contributions of:

- the financial AI research community
- Intel Habana Labs for their hardware insights
- and the open-source contributors who made the financial datasets publicly available.

Their work underpins the benchmarks, metrics, and methodologies detailed in this paper.

As technology and modeling techniques advance, we anticipate continuous improvements to the FinLMB benchmark suite, more refined TPS measurements, and a richer ecosystem of tools and datasets for financial LLM benchmarking.

About Arcee AI

Arcee AI delivers purpose-built AI agents, powered by industry-leading small language models (SLMs), for enterprise applications.

Our offering, Arcee Orchestra, is an end-to-end agentic AI solution that enables businesses to create AI agents for complex tasks. The solution makes it easy to build custom AI workflows that automatically route tasks to specialized SLMs to deliver detailed, trustworthy responses, fast.

Find us at arcee.ai, on Linkedin, and on X.

arcee.ai