Proposal - Google Summer of Code 2025

By Mayukh Tunga

Table of Contents

Project Title: Implement Open LLM Models with JAX and Flax
Student Name: Mayukh Tunga
Email: tungamayukh@gmail.com
GitHub: https://github.com/MayukhTunga
University: SRM Institute of Science and Technology
Degree Program: Computer Science Engineering

---

1. Synopsis

This project aims to build clear and modular implementations of open-source large language models (LLMs) using JAX and Flax. **The initial focus will be on porting GPT-2 (small variant) and OPT-125M**, two foundational open-source LLMs, into a clean, educational codebase. These models were selected for their educational value, compatibility with JAX’s functional paradigms, and availability of pre-trained weights. The end result will provide reference implementations that other researchers and developers can adapt or extend, with a focus on reproducibility, clarity of code, and comprehensive documentation.

---

2. Benefits to the Community

- **Educational Value:**
  The project will generate well-documented, modular implementations with accompanying Jupyter notebooks. These resources will help demystify LLM architectures and serve as an excellent learning tool for developers new to JAX and Flax.
- **Research & Development:**
  Having reproducible reference implementations will accelerate research and experimentation with LLMs, as well as allow for benchmarking, performance tuning, and integration of pre-trained weights from established models.
- **Open Source Contribution:**
  The resulting GitHub repository will act as a community resource, inviting collaboration, review, and further

enhancement of the implementations. This aligns with the broader goals of open science and reproducibility in machine learning.

---

3. Deliverables

1. **Codebase:**
    - Modular Flax implementations of **GPT-2 (small variant) and OPT-125M**.
    - Integrated tools for loading pretrained weights (using HuggingFace conversion tools or direct weight conversion scripts).
    - Utility modules for autoregressive text generation, including attention, MLP blocks, and positional encodings.
2. **Educational Notebooks:**
    - Jupyter notebooks demonstrating step-by-step usage of each model (inference, performance profiling, and optional training loops).
    - Clear explanations of architectural design decisions, including attention to how JAX's functional paradigms are leveraged.
3. **Documentation & Testing:**
    - Detailed documentation covering installation, usage, and contribution guidelines.
    - Unit tests for core modules to ensure code correctness and reproducibility.
4. **Benchmarks & Evaluation:**
    - Initial performance benchmarks on TPU/GPU setups.
    - **Comparative analysis of inference outputs (e.g., perplexity, generated text quality)** against PyTorch/HuggingFace baselines.

---

4. Timeline

Community Bonding Period (May 8 – June 1)

- **Orientation & Model Study:**
  - Familiarize with JAX and Flax best practices.
  - Analyze reference implementations (HuggingFace models, miniGPT, etc.) and **finalize model choices with mentors.**

Coding Phase 1 (June 2 – July 14)

- **Implementation of GPT-2 (Small Variant):**
  - Develop a Flax implementation of GPT-2.
  - Integrate pretrained weight loaders and validate inference correctness.
  - Build a comprehensive Jupyter notebook demonstrating model usage.

Coding Phase 2 (July 19 – August 25)

- **Porting OPT-125M:**
  - Extend the codebase to implement OPT-125M.
  - Enhance utility modules for scalability and performance (e.g., multi-device support with jax.pmap).
- **Benchmarking & Validation:**
  - Profile models on TPU/GPU and compare outputs with PyTorch/HuggingFace baselines.

Final Submission (August 25 – September 1)

• Complete final repository version, including all notebooks, documentation, and performance reports.

•Submit final work product and mentor evaluation.

Note: I remain flexible for extended contributions if longer timelines are preferred.

---

5. Technical Approach

Model Selection Rationale

Why GPT-2 and OPT-125M?

- GPT-2: Well-documented architecture, widely used in tutorials, and pre-trained weights are readily available (e.g., HuggingFace).
- OPT-125M: Designed for reproducibility and openness by Meta, with publicly accessible weights.

Modular Architecture

- **Reusable Blocks:**
  Build Flax modules for standard transformer components (e.g., multi-head self-attention, feed-forward networks, and positional embeddings) to ensure easy reuse.
- **Weight Conversion & Loading:**
  Develop scripts to convert and load pretrained weights from existing models. This might leverage HuggingFace's conversion utilities and custom preprocessing as needed.
- **Inference Module:**
  Construct an autoregressive decoder that implements a simple generation loop. Use JAX's just-in-time compilation (jit) to optimize runtime performance.
- **Scalability Considerations:**
  Optimize for multi-device use with `jax.pmap` to harness the power of TPU and GPU clusters for inference and optional fine-tuning.

Documentation & Educational Notebooks

- Provide clear, step-by-step explanations within interactive notebooks that illustrate:
    - Model architecture and design rationale.
    - How to load and test the models with provided examples.
    - Techniques for benchmarking and optimizing performance.

Tools & Libraries

- **JAX & Flax** for core model implementation.
- **Optax** for optimization routines.
- **TensorFlow Datasets / HuggingFace Datasets** for lightweight data loading during evaluation.
- **GitHub Actions** for continuous integration and testing.

---

6. About Me

I have a strong foundation in deep learning and machine learning frameworks, with hands-on experience in multiple projects:

- **Research Internship:**
  At Cheng Chung University (Taiwan) in collaboration with SRM University, I developed CNN-based models for image and audio classification, created custom preprocessing pipelines, and achieved competitive accuracy on challenging datasets. Below is a simplified version of the CNN model I developed during the internship:

```
class PineappleCNN(nn.Module):
    def __init__(self, num_classes=4):
        super().__init__()
        self.net = nn.Sequential(
            nn.Conv2d(1, 32, kernel_size=3, padding=1),
            nn.BatchNorm2d(32),
            nn.ReLU(),
            nn.MaxPool2d(2),

            nn.Conv2d(32, 64, kernel_size=3, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(2),

            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(),
            nn.MaxPool2d(2),

            nn.AdaptiveAvgPool2d((4, 4)),
            nn.Flatten(),
            nn.Linear(128*4*4, 256),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(256, num_classes)
        )

    def forward(self, x):
        return self.net(x)
```

- **Hackathon Project – Image-to-Comic Strip Generation:**

  Github Link
  I played a key role in developing an AI system that integrated LLM-based storytelling with computer vision (using Stable Diffusion) to generate comic strips, emphasizing modular code and clear documentation.

- **Current Project – Text-to-GIF Generation in JAX:**

  Github Link
  I am developing a model that uses a CLIP text encoder and a custom 3D U-Net diffusion model in JAX/Flax. This project involves TPU parallelism, pmap-based training pipelines, and comprehensive end-to-end system design, deepening my understanding of JAX's functional paradigms and performance optimization.

I have experience with PyTorch, TensorFlow, and have recently embraced JAX/Flax for its high-performance computing capabilities. My work is always focused on building clear, reproducible solutions—traits that I intend to bring to this GSoC project.

I have also created a GitHub repository dedicated to this project which contains the proposal PDF and will serve as a home for all implementations, notebooks, and resources I build during GSoC.
Repository Link: https://github.com/MayukhTunga/Gsoc2025-jax-flax-llm

---

7. Commitment & Availability

I am committed to dedicating 35–40 hours per week throughout the GSoC period. I have thoroughly reviewed the GSoC timeline and, aside from my end-semester examinations from **May 16 to May 26**, during which I anticipate allocating around 1-3 hours per day to GSoC, I have no conflicting commitments following that. I will be available full time during the coding phases and will keep in close communication with my mentors and the community via GitHub, Slack, and regular email updates.

---

8. Conclusion

I am excited by the opportunity to implement open LLM models in JAX and Flax, a project that sits at the intersection of cutting-edge research and educational outreach. My background in deep learning, hands-on project experience, and passion for clear, accessible code make me a strong candidate for this role. I look forward to contributing to the open-source community and advancing the state-of-the-art in LLM development with JAX and Flax.

Thank you for considering my proposal. I am excited to collaborate and bring this project to fruition.
