Main Independent Study Specifications:

The Evolution of Neural Machine Learning

Enrollment Request: CS 496 (3 credits)

Semester: Spring 2025

Advisor: Professor Jaime Dávila

Student: Aryan Sajith

Descriptive Title:

Tracing the Evolution of Neural Machine Learning: From Artificial Neurons to Transformers and Beyond

Statement of Objectives:

Neural networks form the backbone of modern artificial intelligence(AI), enabling advancements in diverse applications such as ChatGPT-based text generation, disease detection with computer vision, and self-driving cars. However, as neural architectures have grown increasingly complex, incorporating techniques such as convolutions, self-attention, among others, it has become increasingly difficult to fully grasp how and why these architectures came about.

As such, this study aims to trace the conceptual and architectural evolution of neural-based machine learning, starting from the fundamental idea of artificial neurons to state-of-the-art architectures like Transformers, and emerging alternatives. By examining how different architectures have evolved in response to specific challenges, we can uncover the motivations behind their development, refine our understanding of their core mechanics, and gain deeper insight into their strengths, limitations, and trade-offs.

Beyond historical analysis, this exploration will provide a structured framework for selecting appropriate neural problem-solving techniques based on problem-solving demands, clarifying the intuition behind key design choices, and identifying potential areas for future innovation. By bridging the gap between theoretical understanding and practical application, this study will not only demystify neural networks but also equip us with the knowledge to contribute meaningfully to the next generation of machine learning advancements.

Planned Activities:

Throughout:

- Add implementations/comparative analyses to a **Github repository** throughout the study

Weeks 1–2: Core Foundations – Perceptron & Multilayer Perceptron (MLP)

- **Objective:** Understand the origin of neural networks and why deeper models became necessary.
- Read: McCulloch-Pitts neurons (1943) and Rosenblatt's Perceptron (1958).
- Implement a **Perceptron from scratch** (NumPy).
- Study the **universal approximation theorem** and why MLPs matter.
- Read: Rumelhart et al. (1986) on **Backpropagation**.
- Implement a **1-2 layer MLP without PyTorch/TensorFlow** to solidify understanding.
- **Cross-Comparative Testing:** Train both perceptron/shallow MLP on simple linearly separable challenges(ex: separating clusters of linearly separable points) for effectiveness. Also, train both on simple non-linearly separable challenges(ex: XOR problem) to demonstrate why deeper architectures and backpropagation became necessary.

Weeks 3–4: Transition to Deep Learning – Backprop & Convolutional Neural Networks (CNNs)

- **Objective:** Understand why CNNs revolutionized vision tasks.
- Read: LeCun et al. (1998) on **LeNet-5** (first successful CNN).
- Implement a **basic CNN from scratch**
- Understand and compare:
    - **Why CNNs over MLPs?**
        - **Key operations:** Convolutions, pooling, parameter sharing.
        - Briefly study modern CNN architectures (**AlexNet, ResNet**) but **don't implement them**. Could mention these more modern variations in final paper
- **Cross-Comparative Testing:** Train a basic CNN and simple MLP(from previous task) on a small image dataset(MNIST, CIFAR-10, etc.) to compare their performance. Analyze accuracy, parameter efficiency, overfitting tendencies, and more demonstrating how CNNs outperform MLPs in vision tasks via benefits like spatial hierarchy recognition.

Weeks 5–6: Recurrent Neural Networks (RNNs) – Why Sequence Models?

- **Objective:** Learn why standard feedforward networks struggle with sequential data.
- Read: Hochreiter & Schmidhuber (1997) on **LSTMs**.

- Implement a **vanilla RNN** and **simple LSTM** from scratch.
- Discuss:
- The **vanishing gradient problem** in RNNs.
    - Why **LSTMs > Vanilla RNNs** for long sequences.
    - Brief look at **GRUs**, but **no implementation** (similar to LSTMs). Again could be mentioned as more modern variations in the final paper.
- **Cross-Comparative Testing:** Train a standard feedforward MLP(from previous task), vanilla RNN, and LSTM on simple sequential tasks(eg: character-level text prediction or time-series forecasting). Compare the ability to capture long-range dependencies, highlighting the vanishing gradient issue in RNNs and demonstrating why LSTMs perform better on longer sequences.

Weeks 7–8: Transformers – The Breakthrough

- **Objective:** Understand how **self-attention solved RNN limitations**.
- Read: **"Attention Is All You Need"** (Vaswani et al., 2017).
- Implement **a minimal Transformer model from scratch**.
- Understand **how Transformers outperform RNNs** in NLP and vision.
    - Briefly study **Bert/GPT** for modern use of transformers.
- **Cross-Comparative Testing:** Train an LSTM and a minimal Transformer on a sequence-to-sequence task(eg: machine translation, text generation, or something similar). Compare training efficiency, ability to handle long-range dependencies, and performance on increasing sequence lengths, demonstrating how self-attention mitigates the vanishing gradient problem and improves parallelism over recurrent models.

Weeks 9–10: Mamba – The Transformer Alternative

- **Objective:** Investigate Mamba's efficiency and trade-offs vs. Transformers.
- Read: **Mamba paper**(Mamba: Linear-Time Sequence Modeling with Selective State Spaces) and analyze its key innovations.
- Implement **a simplified Mamba model** to compare its architecture to Transformers.
- Discuss:
    - Why Mamba **eliminates self-attention bottlenecks**.
    - Where **state-space models shine over Transformers**.
    - Why **haven't SSMs overtaken Transformers?**
- **Cross-Comparative Testing:** Train both a minimal Transformer and a simplified Mamba model on a sequence modeling task(eg: language modeling, time-series forecasting, or something similar). Compare their computational efficiency, memory usage, and

performance on long-context sequences to analyze how Mamba's state-space approach mitigates challenges associated with self-attention with trade-offs in generalizability, expressivity, and such.

Weeks 11–13: Final Synthesis & Experimentation

- **Objective:** Connect historical insights with practical takeaways.
- Compare:
    - **MLPs** → **CNNs** → **RNNs** → **Transformers** → **Mamba** (big-picture evolution).
- Summarize:
    - **What problem did each architecture tackle?**
    - **Why did certain ideas fade away while others thrived?**
- Write a structured **research paper** synthesizing insights/findings: Reflect on findings from previous cross-comparative tests, analyzing how each architecture addresses its predecessor's limitations. Discuss trade-offs in performance, scalability, efficiency, and such across MLPs, CNNs, RNNs, Transformers and Mamba, providing a holistic view on deep learning's evolution from simple artificial neurons to modern artificial neural networks.

Evaluation Criteria:

1. **In-Depth Paper**: Upon completion, the project will be documented in a well-structured paper that provides strong comparative analyses and implementation details for the various neural architectures tackled and references for further learning. The paper will also highlight challenges encountered during comparison, key insights gained and potential future directions where applicable.

2. **GitHub Repository**: A well-organized GitHub repository will be maintained throughout the project's development, containing all relevant files. It will feature a detailed README outlining the project's objectives, implementation details, and step-by-step instructions for implementing project components.

3. **Meetings**: Bi-weekly meetings will be held to discuss project progress, share updates, and address any challenges encountered.