Neural Network Training and Optimization: Classical vs Evolutionary Approaches

What is Neural Network Training?

- Neural networks are a type of artificial intelligence inspired by how the brain works.
- Training a neural network means teaching it to make predictions or decisions by adjusting its internal settings (called **weights**).
- The goal is to find the best set of weights so the network performs well on new (unseen) data.
- In this work, we use the MNIST handwritten digits dataset as an example.

Why Optimization is Needed

- The process of finding the best weights is called **optimization**.
- Neural networks have many weights, and the best combination can be hard to find.
- Optimization helps us search for weight values that make the network as accurate as possible.

Classical vs Evolutionary Optimization

- **Classical optimization** (like SGD) makes small, step-by-step changes to weights, using mathematical rules.
- **Evolutionary optimization** uses ideas from natural evolution, like "survival of the fittest."
- These methods try many possibilities and use randomness to search for good solutions.

Optimizing Neural Network Weights

- Instead of calculating how to change each weight using math, evolutionary algorithms test many sets of weights.
- The best-performing sets are kept, changed, and combined to create new candidates.
- Over time, the population of weight sets improves.

Differential Evolution (DE): Global Optimization

- DE works like a global explorer, searching the entire solution space.
- Each solution is a candidate set of weights (a vector).
- Main functions:
  - **Choosing Vectors**: Randomly pick other candidates to guide changes.
  - **Mutation**: Create new candidates by mixing others with random differences.
  - **Crossover**: Combine candidates to try new possibilities.
  - **Extinction**: Occasionally remove poor solutions to keep diversity.

Genetic Algorithm (GA): Local Optimization

- GA acts as a local optimizer, refining the best candidates found so far.
- Uses concepts like:
  - **Selection**: Pick the best candidates for reproduction.
  - **Crossover**: Mix parts of two candidates to create new ones.
  - **Mutation**: Change parts randomly for diversity.
- Helps to fine-tune solutions in promising regions.

Hybrid Approach: Combining DE and GA

- Start with DE for broad, global exploration.
- Switch to GA for detailed, local refinement of the best solutions.
- This combines the strengths of both: wide search first, then focused improvement.

Technologies Used

- Python
- TensorFlow and Keras
- Evolutionary Algorithms (Differential Evolution, Genetic Algorithm)
- Classical Optimization (SGD)
- Data Science Libraries (NumPy, Matplotlib, Plotly)
- Dataset: MNIST Handwritten Digits