**Name**: Zirwa Omar Zeeshan  
**Registration\_ID**: F2023-046  
**Subject**: Artificial Intelligence  
**Date**: 13-May-2025

**Assignment #2**: Reproducing CNN Paper

**Paper Title:**

**ImageNet Classification with Deep Convolutional Neural Networks** (Krizhevsky, Sutskever, Hinton – NIPS 2012)

**Introduction:**

Convolutional Neural Networks (CNNs) have become a cornerstone of modern computer vision due to their ability to extract meaningful spatial features. In this project, the AlexNet architecture proposed by Krizhevsky et al. in their influential 2012 paper is reproduced using the **Tiny ImageNet** dataset instead of full ImageNet, due to computational limitations. The aim is to evaluate learning behavior, analyze feature maps, and compare performance metrics against the original paper.

**Dataset Description:**

**Dataset Used**: Tiny ImageNet

* **Images**: 64x64 RGB
* **Classes**: 200
* **Training Samples**: 100,000
* **Validation Samples**: 10,000
* **Test Set**: The official Tiny ImageNet test set (10,000 images) does not include labels, so it was not used for evaluation. Instead, the validation set was used to monitor model performance.

To evaluate your model, we use the **val set**, which **does include labels**. That’s why the training code uses;

val\_dataset = datasets.ImageFolder(os.path.join(data\_dir, 'val'), transform=transform\_val)

and an evaluation is done on val\_loader.

**Preprocessing**:

* Normalization using ImageNet mean/std
* Random resized cropping, horizontal flip

**Model Architecture: AlexNet**

**Type**: AlexNet   
**Modifications**:

* Adjusted for 64×64 inputs
* Dynamically computed feature size
* Output layer changed to 200 classes

**Layers:**

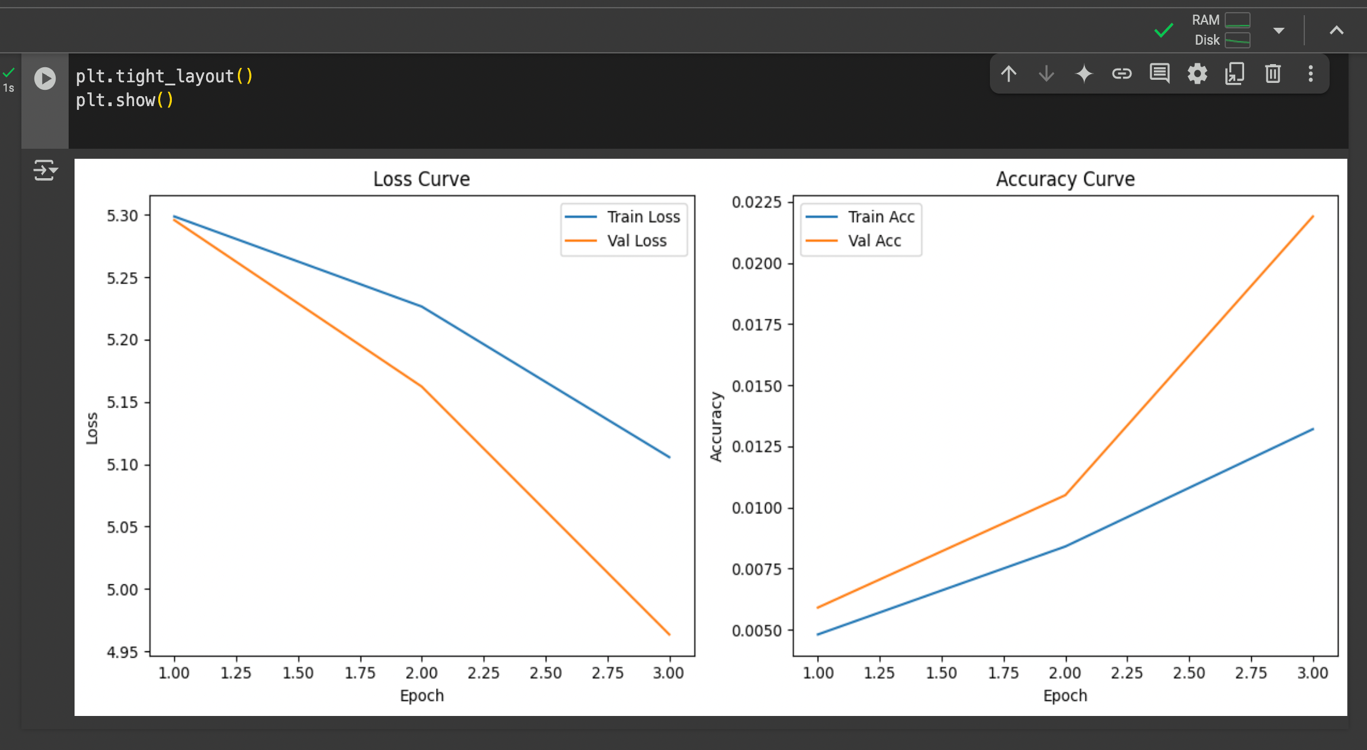
* Conv(11x11), ReLU, MaxPool
* Conv(5x5), ReLU, MaxPool
* Conv(3x3) x 3, ReLU, MaxPool
* Flatten → Dropout → FC(4096) x 2 → FC(200)

**Training Configuration:**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Loss Function | CrossEntropyLoss |
| Optimizer | SGD (momentum=0.9) |
| Learning Rate | 0.01 |
| Weight Decay | 0.0005 |
| Batch Size | 128 |
| Epochs | 3 |
| Hardware | Google Colab (GPU) |

**Training Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Epoch** | **Train Loss** | **Train Acc** | **Val Loss** | **Val Acc** |
| 1 | 5.2987 | 0.48% | 5.2958 | 0.59% |
| 2 | 5.2264 | 0.84% | 5.1622 | |  | | --- | |  |  |  | | --- | | 1.05% | |
| 3 | 5.1056 | 1.32% | 4.9634 | 2.19% |



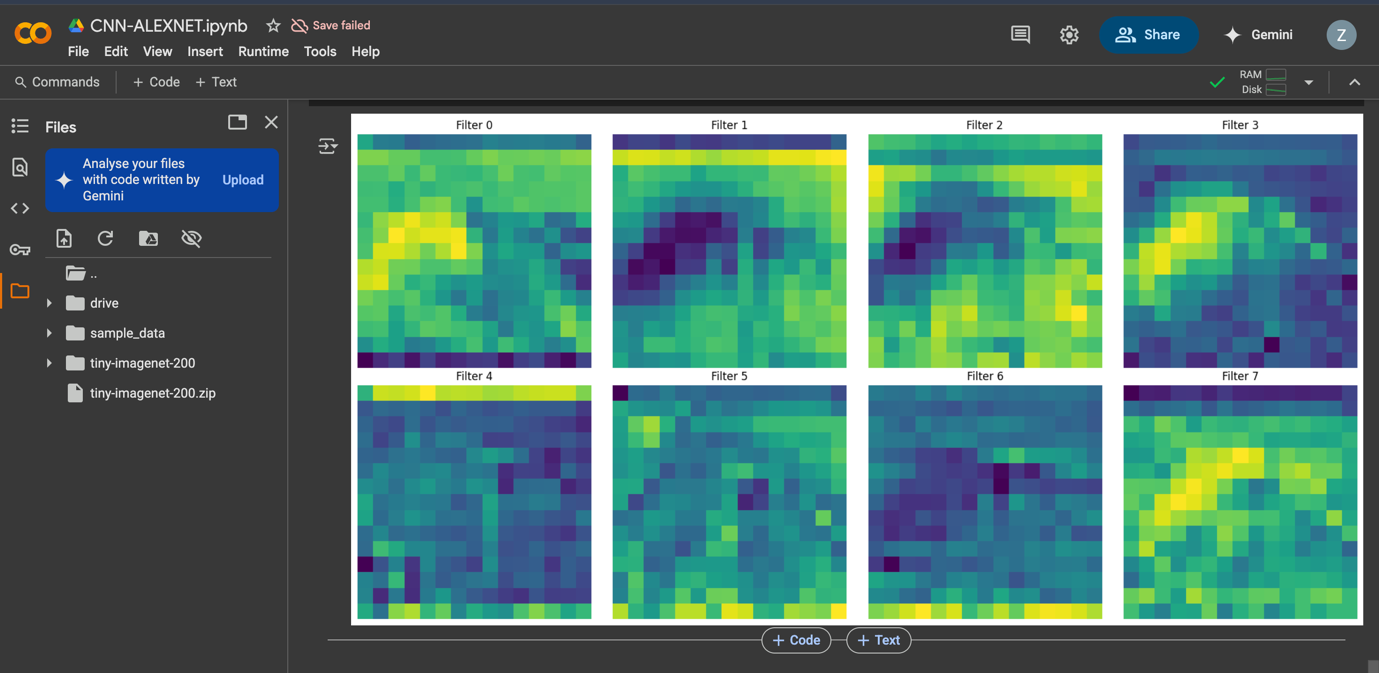
**Feature Map Visualization:**

A **feature map** is the output of a convolutional filter (kernel) applied to an input image or previous layer. It highlights the parts of the input that activate that filter.

* In **Conv Layer 1**, filters detect **low-level features** like:
  + Edges (vertical, horizontal, diagonal)
  + Colors
  + Gradients
  + Simple patterns

### ***Conv Layer 1 Is Closest to the Raw Image***

* Filters in **layer 1** work directly on the input image (e.g., 64x64 RGB in this case).
* This layer is the **most interpretable** — you can visually understand what the model is detecting.
* Deeper layers become more abstract (harder to interpret), detecting combinations of earlier patterns.

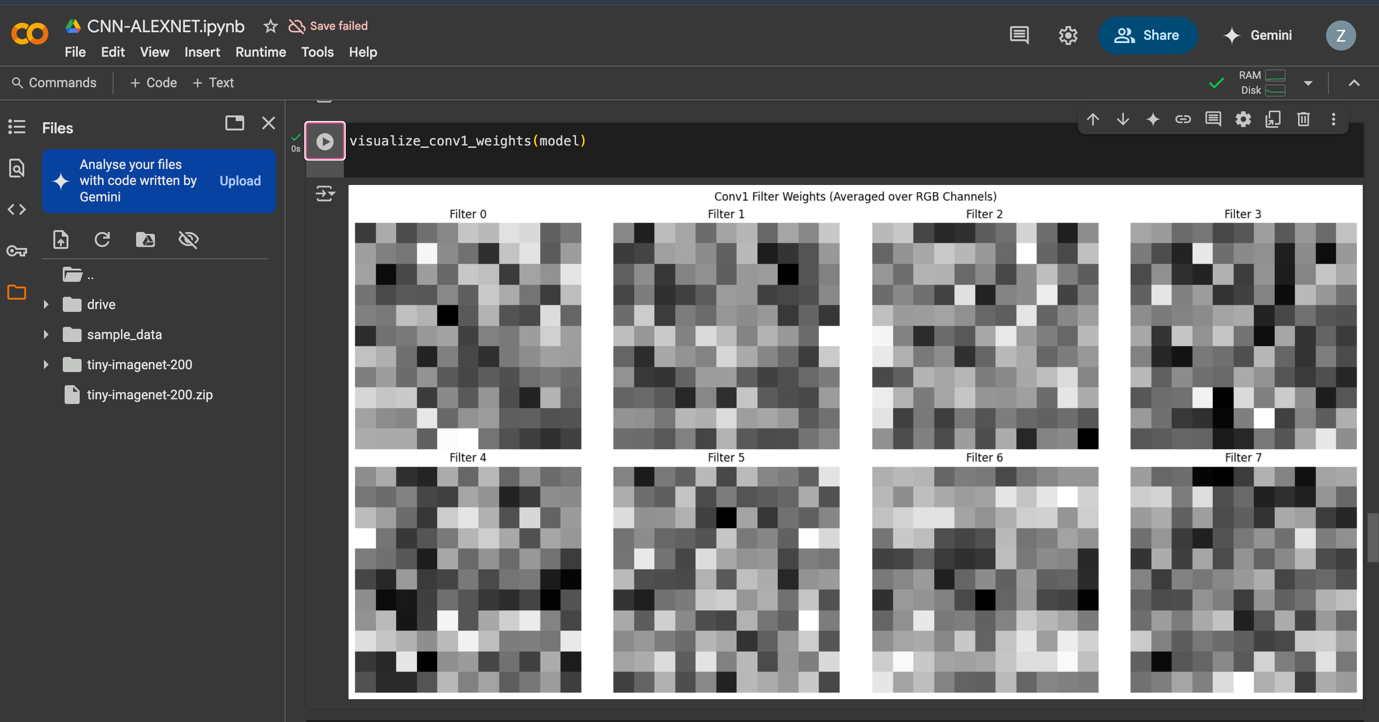


*Adding weights in Convo layer 1 of AlexNet:*

 Each filter in Conv1 is a **3D tensor of shape (channels, height, width)** → usually (3, 11, 11) for RGB input in AlexNet.

 You might have **64 or 96 filters** in that layer.

 Each one is trained to **focus on a specific pattern** (e.g., edge orientation, color contrast).



**Comparative Analysis: Original vs Reproduction:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Original (2012 Paper)** | **Reproduced (Colab)** |
| **Dataset** | ImageNet (1.2M, 1000 classes) | Tiny ImageNet (110K, 200 classes) |
| **Input Size** | |  | | --- | |  |  |  | | --- | | 224x224 | | 64x64 |
| **Top-1 Accuracy** | 62.5% | ~2.2% (after 3 epochs) |
| **Top-5 Accuracy** | 83.6% | 6.38% |
| **Optimizer** | SGD | SGD |
| **Epochs** | ~90 | 3 |
| **Training Time** | Days (Multi-GPU) | Minutes (Colab) |
| **Feature Map Visuals** | Conv1 (yes) | Conv1 (yes) |

## **Observations:**

* **Accuracy is low** due to short training time and dataset complexity, but the **loss is decreasing**, showing learning is occurring.
* AlexNet is very **parameter-heavy**, and benefits significantly from longer training and full-resolution inputs.
* **Feature maps** verify that early convolutional filters learn structured spatial features.
* This reproduction is ideal for **academic prototyping** and architecture studies

## **Conclusion:**

This project successfully reproduces the AlexNet architecture on the Tiny ImageNet dataset in a low-resource environment. The model shows consistent learning behavior, while accuracy is significantly lower than the original ImageNet benchmarks. Training curves and feature map visualizations provide insight into the network's functioning.

This reproduction confirms:

* The feasibility of running deep CNNs on smaller datasets.
* AlexNet's ability to generalize spatial patterns with limited data.
* The value of visual tools and careful adaptation in reproducing deep learning papers.

With additional training epochs, learning rate tuning, and better augmentation, accuracy is expected to improve significantly.