Machine Learning II

- Assignment #3 -

Course	Department	Student ID	Name
SOI1010	Automotive Engineering	2021048140	Dongmin Kim

December 1, 2025

1. Introduction

1.1 Environment

All experiments in this assignment were conducted on a local Linux workstation due to **Google Colab's short TPU session duration**, which was insufficient for long-epoch training.
The environment specifications are as follows:

- **OS:** Ubuntu
- **CPU:** Intel i7
- **GPU:** NVIDIA GeForce GTX 1650
- **RAM:** 16GB

To ensure reproducibility, the full implementation has been provided as a **.py** file. The same codebase has also been uploaded to the Colab notebook for verification and easy re-execution.

Google Colab:
https://colab.research.google.com/drive/1DycXA28OdAuMGsQdHn7m6_KrnKV3hzke?usp=sharing

1.2 Background

The Artworks Artist Classification challenge requires distinguishing 5,311 unique artworks with significantly varying styles. Major difficulties include:

- **High intra-class variance**: the same artist may produce radically different artworks
- **High inter-class similarity**: multiple artists share visual techniques
- **Severe long-tail distribution**: ~**81%** of classes fall under the **"Other"** category
- **High-resolution texture dependency**: small-scale brush stroke patterns matter

Our model must learn all visual primitives from scratch, requiring:

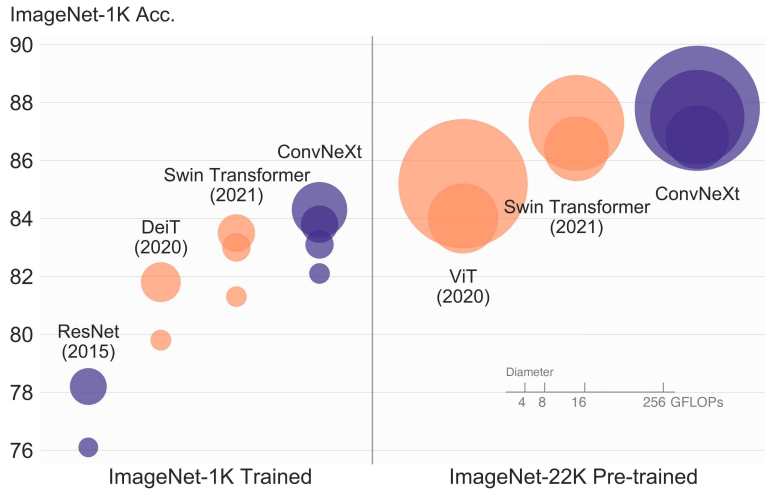
- **Stable Regularization**
- **Strong Augmentation**
- **K-fold validation**
- **Higher input resolution**

These constraints motivated the design choices described below.

1.3 Advanced Architectures

In addition to traditional convolutional networks, I examined more recent CNN architectures designed to improve efficiency and representational power. Among them, **ConvNeXt** stands out as a modernized convolutional architecture that incorporates design elements inspired by contemporary high-performing models while preserving the hierarchical structure of classical CNNs.
Benchmark comparisons on **ImageNet** indicate that ConvNeXt achieves significantly **higher accuracy**

than conventional ResNet variants across multiple scales.



Given its strong performance and architectural improvements, **ConvNeXt** was considered as a **potential candidate** for this project. An implementation was attempted; however, the model's computational demands exceeded the available hardware capacity. Training ConvNeXt from scratch resulted in **Out-of-Memory** error**s** even when the batch size was reduced to 4. As a result, full training was not feasible.

This exploration provided useful insights into the **trade-off** between **architectural complexity, memory requirements**, and **training stability**. While modern architectures offer higher theoretical performance, practical constraints highlight the importance of selecting a balanced model. Among the ResNet, ResNet-50 offered the best compromise between depth, training stability, and generalization, outperforming both shallower (ResNet-18/34) and deeper (ResNet-101) models under the same training setup.

2. Dataset Analysis

2.1 Dataset Composition

- **Train samples: 5,311 artworks**
- **Labels:** artist
- **Distribution:** Van_Gogh (11%) / Degas (8%) / **4,308 artists (81%)**

2.2 Observations

(1) Overfitting Risk

The dataset contains **5,311 artworks** but more than **4,300 artists** belong to the **'Other' category**, meaning many classes appear only 1–3 times. Such sparsity makes the model memorize individual images rather than learn artist-level features, creating a strong tendency toward overfitting during scratch training. This requires **strong regularization**, **augmentation**, and **stable optimization**.

(2) Data Augmentation

Because most artists exhibit **only a few examples,** the model must rely heavily on transformations to simulate intra-class variation. RandomResizedCrop, rotation, and horizontal flips significantly reduce overfitting by producing a broader distribution of training samples. Augmentation serves as an implicit regularizer, compensating for the lack of samples per artist.

(3) High Validation Variance: K-Fold Cross-Validation

Due to the highly imbalanced label distribution, **different train/validation splits** produce noticeably different artist compositions. Some folds may contain more samples of dominant artists (e.g., Van Gogh, Degas), leading to artificially inflated validation scores. **Stratified K-Fold** reduces this variance and provides more reliable validation metrics and stable ensemble performance.

(4) Higher Input Resolution

Artist identification depends heavily on **fine visual details** such as brush strokes, canvas texture, and subtle shading patterns.

Lower resolutions (e.g. 224×224) **lose information** necessary for distinguishing artists. Experiments showed that **288×288 resolution** strikes the optimal balance between texture preservation and stable generalization.

3. Method

3.1 Model Architecture: ResNet-50

ResNet was chosen due to its stable optimization properties when training from scratch. **Residual skip-connections** mitigate vanishing gradients and allow deeper models to be trained without collapsing.

To determine the optimal depth, several variants were implemented and evaluated:

- **ResNet-18 / ResNet-34: insufficient** capacity and consistently underfitted the dataset
- **ResNet-50**: provided **the best balance** between model capacity and generalization
- **ResNet-101**: showed strong training accuracy but severe **overfitting**
- **ConvNeXt**: failed to train due to **Out-of-Memory** even when batch-size was reduced to 4

3.2 Resolution

Since artist identification depends on **fine-grained textures** (e.g., brush strokes, canvas details), input resolution strongly influences performance.
Three resolutions were tested:

- **224×224**: too low to preserve high-frequency texture, **underfitting**
- **320×320**: captured more detail but also **learned noise** and background artifacts, **overfitting**
- **288×288**: provided the best balance between texture preservation and **stable generalization**

Thus, **288×288** was chosen as the final resolution.

3.3 Optimization

The optimization setup is as follows:

(1) Optimizer: Adam
- Chosen for its stability in poorly conditioned optimization landscapes typical in scratch training

(2) Loss Function: Cross-Entropy Loss
- Standard for **multi-class classification**

(3) Learning Rate Scheduler: CosineAnnealingLR
- Helps avoid late-stage oscillation and improves final convergence

This combination consistently produced smooth loss curves and stable training.

3.4 Label Smoothing

A smoothing factor of **0.05** was applied. Due to extreme class imbalance, standard Cross-Entropy tends to produce overly confident predictions. Label smoothing **reduces overconfidence**, stabilizes training, and improves generalization in long-tail distributions.

3.5 K-Fold Cross-Valitation: k=4

Stratified K-Fold was used to reduce validation variance caused by the imbalanced dataset. Because training is **computationally expensive**, k=4 was selected as a practical compromise. During training, some folds showed **unstable validation loss** depending on artist distribution in that split.

As a result, when generating the final test.csv, the Fold 3 model was excluded from the ensemble to avoid injecting noise.

4. Experiments

4.1 Experimental Setup

All experiments were conducted using the **ResNet** models implemented from scratch.
Training was performed using:

- **Resolution:** 224, 288, 320
- **Optimizer:** Adam
- **Scheduler:** CosineAnnealingLR

- **Regularization:** Label smoothing (0.05), strong augmentations
- **Cross-validation:** Stratified K-Fold (K = 4)

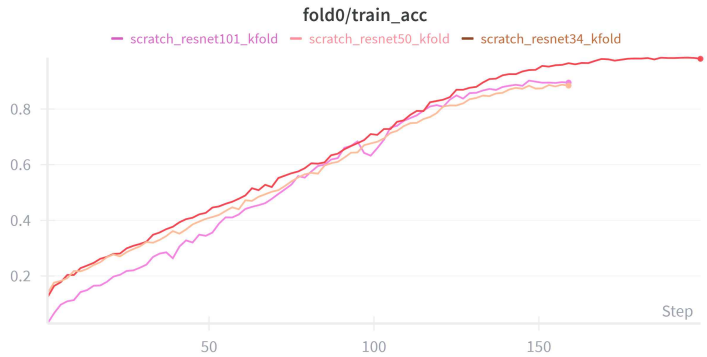
Fold 0 performance is mainly presented for clarity, as the relative comparison trends were consistent across folds.

4.2 Comparison: ResNet-34 vs ResNet-50 vs ResNet-101

To determine the **optimal depth**, all three scratch models were trained with identical settings.

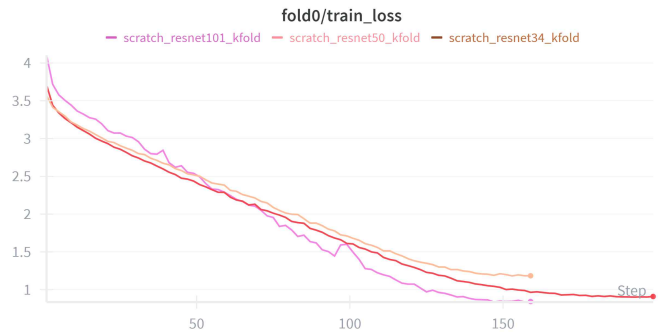
(1) Training Accuracy

- ResNet-34 and ResNet-50 show similar training curves
- ResNet-101 increases more slowly early on but eventually catches up
- However, the ResNet-101 curve shows visible instability



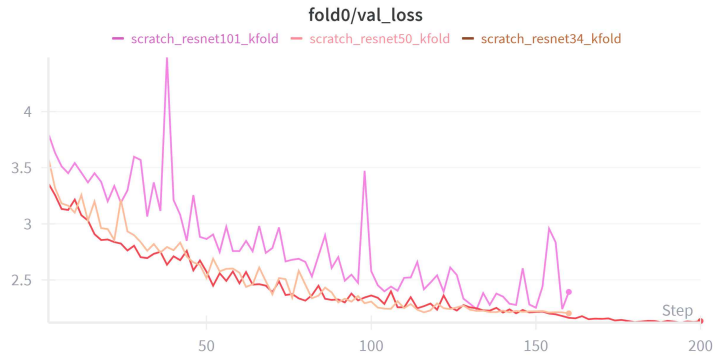
(2) Training Loss

- ResNet-101 exhibits the fastest loss reduction but with noticeable oscillations
- ResNet-50 reduces loss smoothly and consistently
- ResNet-34 reduces loss the slowest



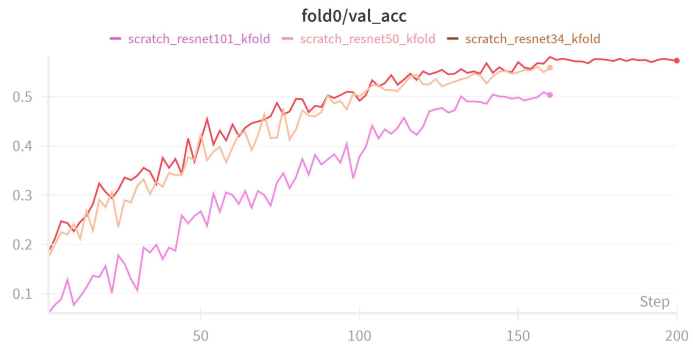
(3) Validation Loss

- ResNet-50 shows the lowest and most stable validation loss
- ResNet-34 performs slightly worse but reasonably stable
- ResNet-101 is **highly unstable** with large spikes, indicating severe overfitting



(4) Validation Accuracy

- ResNet-50 achieves the highest validation accuracy
- ResNet-34 is slightly lower but consistent
- ResNet-101 significantly underperforms despite high training accuracy



Conclusion:

- **ResNet-34: Underfits** slightly
- **ResNet-50: Best trade-off between capacity & generalization**
- **ResNet-101: Optimization unstable + poor generalization**

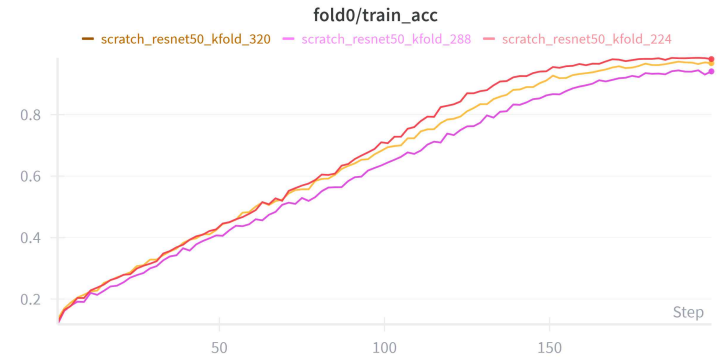
Therefore, **ResNet-50** was selected as the final architecture.

4.3 Resolution Comparison: 224 vs 288 vs 320

To analyze the effect of image resolution, **ResNet-50** was trained under three input sizes.

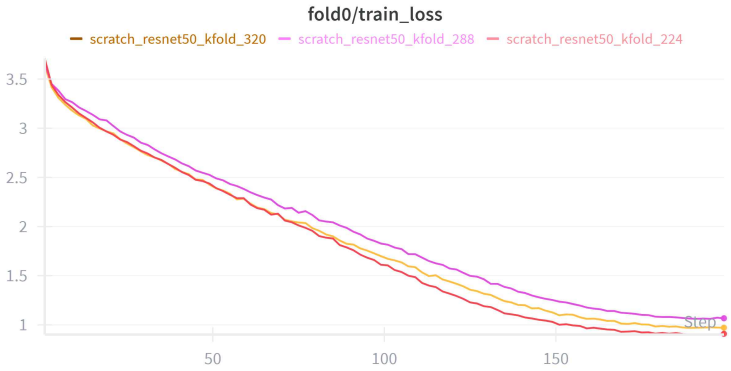
(1) Training Accuracy

- 224 learns fastest
- 320 and 288 follow, with 288 showing the most stable curve



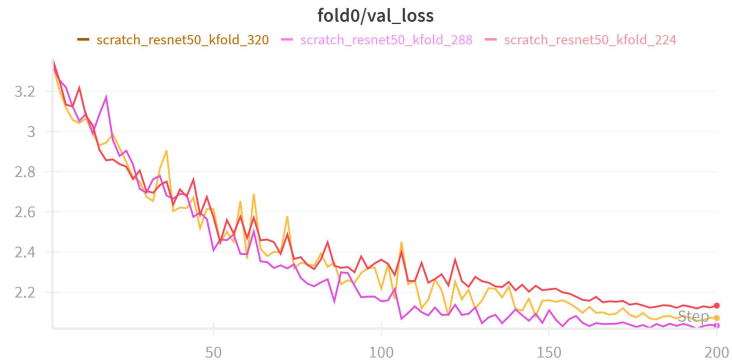
(2) Training Loss

- 224 drops fastest
- 320 drops moderately fast
- 288 shows the cleanest and most stable reduction



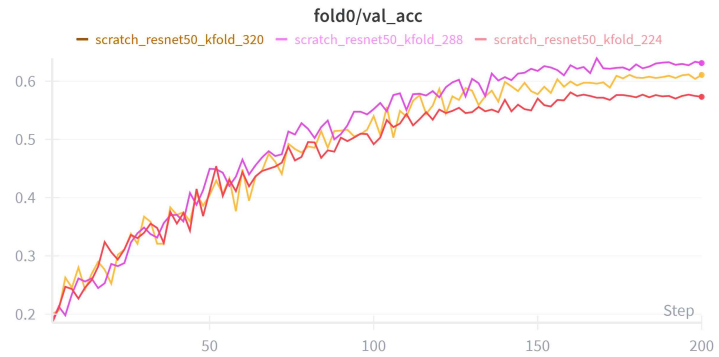
(3) Validation Loss

- 288 produces the lowest validation loss
- 320 performs moderately well but fluctuates due to noise learning
- 224 consistently underperforms



(4) Validation Accuracy

- 288 achieves the best final validation accuracy
- 320 is second but unstable
- 224 is the lowest due to insufficient high-frequency detail



Conclusion

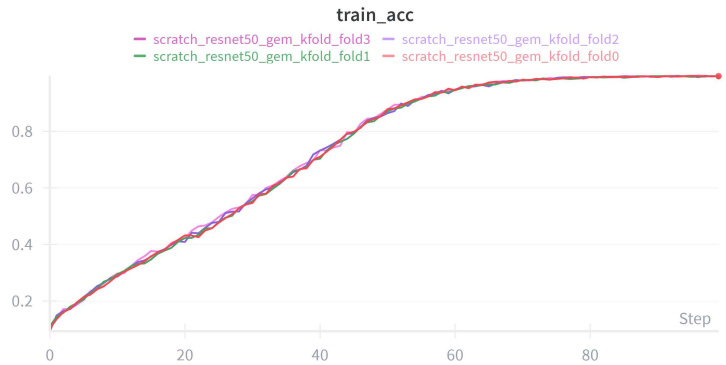
288×288 provides the optimal balance between texture preservation and stable generalization, while 320×320 causes overfitting to noise and 224×224 lacks critical artist-level cues.

4.4 GeM Pooling

To investigate whether a more expressive pooling strategy could improve fine-grained artist classification, a ResNet-50 model with **Generalized Mean (GeM)** pooling was trained under **4-fold cross-validation**. GeM emphasizes high-frequency texture patterns, which can be beneficial in retrieval tasks but may increase overfitting on small or long-tailed datasets.

(1) Training Accuracy

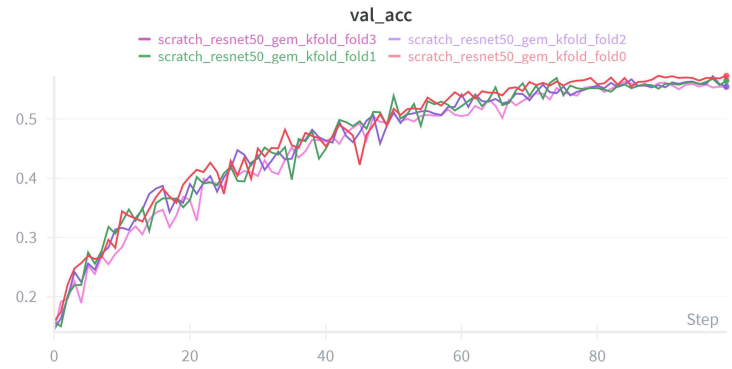
Across all folds, the GeM model rapidly reached **train accuracy** ≈ **0.99**, far higher than the standard average-pooling ResNet-50. This indicates that the model memorized the training set rather than learning robust style features.



(2) Validation Accuracy

Despite the near-perfect training accuracy, **validation accuracy** consistently saturated around **0.54–0.56** across all folds. This is significantly lower than the baseline ResNet-50 with average

pooling (0.61–0.63). The large discrepancy between train and validation accuracy confirms severe overfitting.



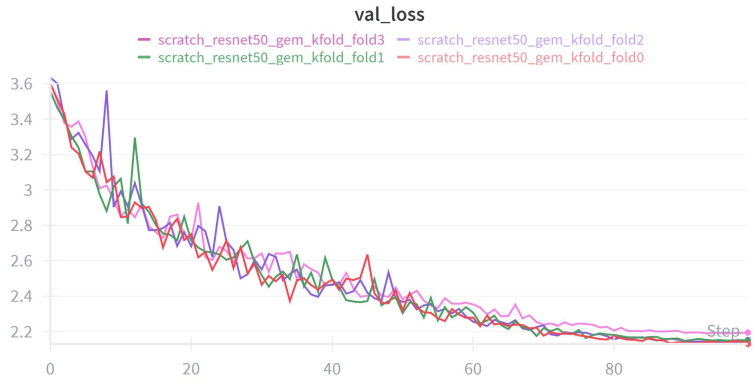
(3) Validation Loss

The validation loss curves showed:

- **noticeable oscillations**
- **unstable behavior during mid–late epochs**

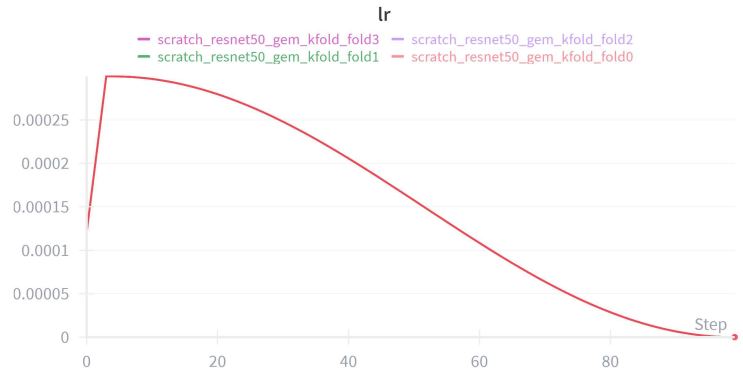
Even when the learning rate decreased, the loss remained noisy.

This suggests the model was overly sensitive to high-frequency texture noise.



(4) Learning Rate

Cosine Annealing typically stabilizes late-stage training, producing smooth convergence. However, the GeM model **did not stabilize** when the LR entered the low region. This indicates that GeM's architectural bias causes structural overfitting that optimization alone cannot resolve.



Conclusion

GeM pooling increases expressiveness but also makes the model highly prone to **overfitting** on this long-tailed, small-sample dataset. Despite perfect training accuracy, validation accuracy remained ~7% lower than the baseline. Validation loss and LR-dependent stability further confirmed that GeM was unsuitable for this task.

Therefore, GeM pooling was excluded from the final system, and standard average pooling was retained as the most stable and generalizable choice.

4.5 Optimization

Across all experiments, the combination of **Adam, Cross-Entropy loss,** and **CosineAnnealingLR scheduler** produced smooth and reliable learning curves. No divergence, gradient explosion, or collapse occurred, confirming the chosen optimization setup is appropriate for scratch training on this dataset.

(1) Adam

- Adam maintains exponential moving averages of gradients and squared gradients:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla\mathcal{L}_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla\mathcal{L}_t)^2$$

$$\theta_t = \theta_{t-1} - \alpha \frac{m_t/(1-\beta_1^t)}{\sqrt{v_t/(1-\beta_2^t)} + \epsilon}$$

(2) Cross-Entropy loss with Label Smoothing

- Standard CE loss:

$$\mathcal{L}_{CE} = -\sum_{c=1}^{C} y_c \log p_c$$

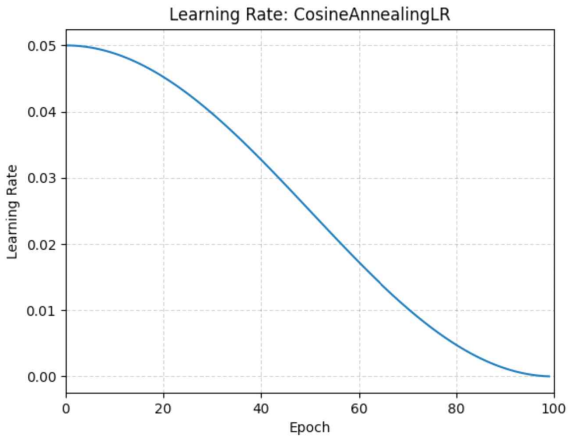
- Label smoothing with ε = 0.05:

$$y_c^{smooth} = (1-\epsilon)y_c + \frac{\epsilon}{C}$$

This reduces overconfidence and stabilizes training in long-tailed distributions.

(3) Cosine Annealing scheduler

- PyTorch's cosine schedule gradually decreases the learning rate following:



$$\eta_{t+1} = \eta_{\min} + (\eta_t - \eta_{\min}) \cdot \frac{1 + \cos\left(\frac{(T_{cur}+1)\pi}{T_{max}}\right)}{1 + \cos\left(\frac{T_{cur}\pi}{T_{max}}\right)}$$

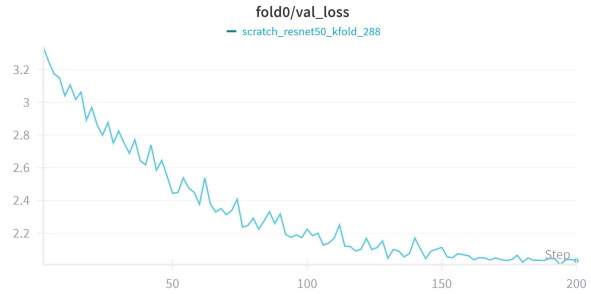
This schedule enables fast exploration in early epochs and stable fine-tuning in later epochs.

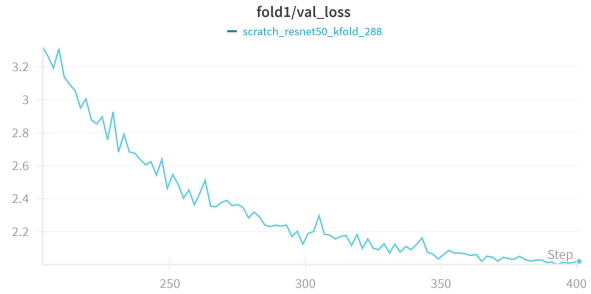
4.6 K-Fold Variance

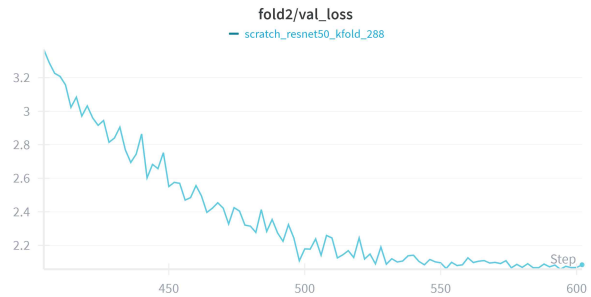
Due to the extreme class imbalance, the validation split varies significantly depending on which rare artists appear in each fold. This leads to noticeable differences in model behavior across folds.

Observations:

- Fold 0, 1, and 2 exhibit stable training curves
- Fold 3 shows occasional instability in validation loss due to distribution shift
- To avoid performance degradation during test-time inference, **Fold 3** was **excluded** from the final **ensemble**







Conclusion:

- Using **k = 4** was sufficient to reduce variance and provide reliable validation estimates, but **individual folds** can still be **noisy** depending on sample distribution.
- Excluding an underperforming fold improved ensemble robustness and produced more consistent final predictions.

5. Result

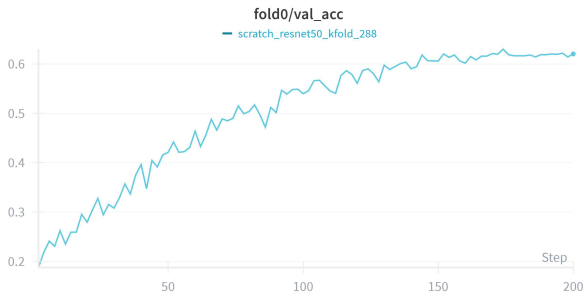
This section reports the performance of the final selected model:

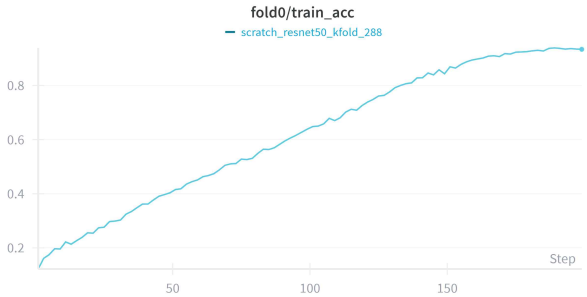
- ResNet-50 + 288×288 resolution + Adam + CE + Cosine Annealing + K=4 folds with ensemble

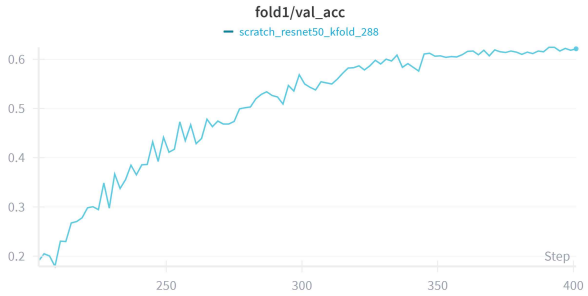
5.1 Training and Validation Curves

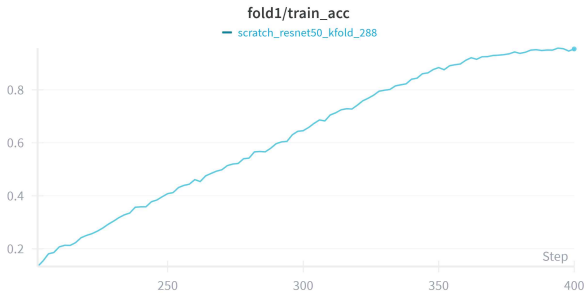
The final model exhibited stable convergence across most folds.

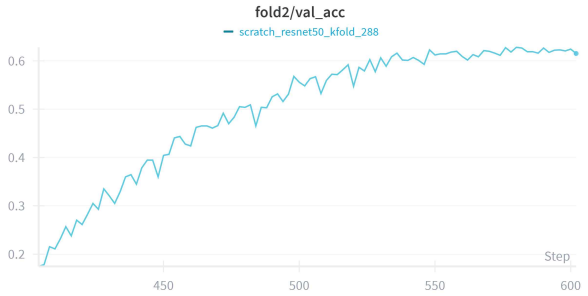
- Training loss smoothly decreased
- Validation accuracy steadily increased and plateaued around 0.60–0.63 for stable folds
- No divergence or collapse was observed

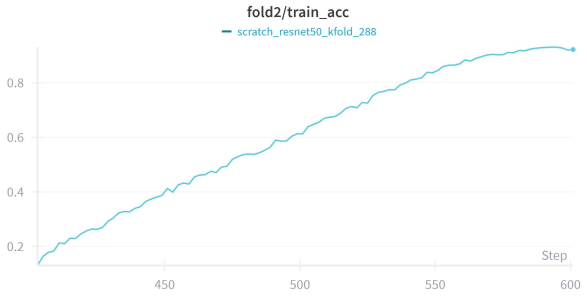


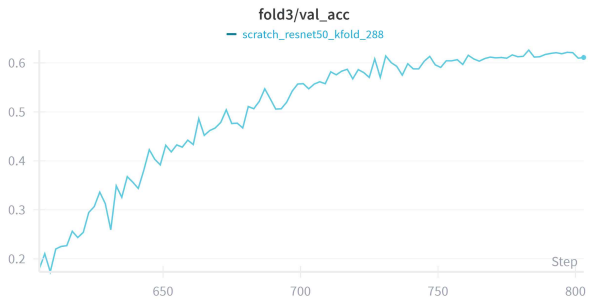


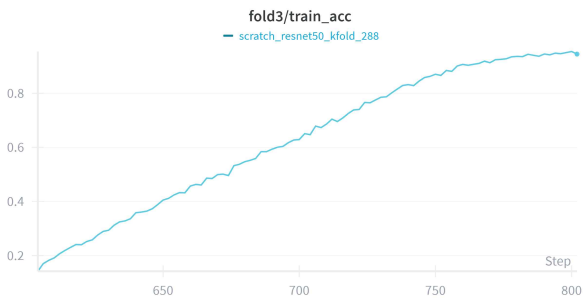












5.2 Performance

Fold	Best Val Acc	Notes
Fold 0	0.63	stable
Fold 1	0.62	stable
Fold 2	0.62	stable
Fold 3	0.62	slightly unstable

Across the four folds, the final model exhibited stable and consistent validation accuracy. Fold 0, 1, and 2 showed **nearly identical performance** in both loss reduction and accuracy progression, indicating reliable generalization under different train–validation splits. Fold 3 reached a similar peak accuracy but displayed **minor instability** in the validation loss curve, suggesting sensitivity to the specific distribution of artist samples in that fold.

Despite this slight variation, all folds **converged to a similar performance range** (0.62–0.63), confirming the robustness of the selected architecture and training setup. For test-time inference, the ensemble was constructed using the three most stable folds (0, 1, 2), resulting in improved reliability and reduced variance across predictions.

5.3 Ensemble

For test-time inference, the following ensemble strategy was used:
- logits from the **three stable folds (Fold 0, 1, 2)** were averaged
- horizontal flip **TTA (Test-Time Augmentation)** was applied
- final predictions were generated from the softmax-averaged probabilities

This ensemble approach improved robustness by reducing per-fold variance and mitigating the effect of distribution shift in individual folds.

The final predictions were saved in submission.csv and submitted to Kaggle

5.4 Kaggle

The final submission (ResNet-50 scratch, 288×288 resolution, Fold 0–2 ensemble with horizontal-flip TTA) achieved **8th place** on the official Kaggle leaderboard, confirming that the selected configuration generalizes effectively to unseen data.

6. Discussion

The experiments revealed several characteristics of the dataset and the task that directly influenced the final design choices.

6.1 Architectural Difference

Despite using the same training pipeline, **small architectural** or **hyperparameter changes** (e.g., pooling choice, resolution, depth) produced disproportionately large shifts in validation behavior. This indicates that the dataset is **sensitive**, largely due to extreme class imbalance, many classes with only 1–3 samples, and strong style variability within the same artist.

As a result, models that normally behave similarly (e.g., ResNet-34 vs ResNet-50) showed clearer performance separation than expected.

6.2 Detail of Texture

The **resolution comparison** highlighted a **non-linear relationship**:

- **too low** → critical style cues lost
- **too high** → the model begins learning canvas noise

This suggests the task is **texture-dependent** but **noise-sensitive**.
288×288 happened to be the "**sweet spot**" where meaningful high-frequency features were preserved without destabilizing training.

6.3 Pooling

The **GeM pooling** experiment exposed a key **limitation:**
- the model learned exact texture statistics of the training set
- but these statistics do not generalize to other works by the same artist

This behavior implies that artist identity is **not solely defined by textures**,
and increasing texture sensitivity **reduces generalization** on classes with few samples.

6.4 Optimization

The same optimizer **(Adam + LR decay)** produced:
- smooth convergence for standard ResNet-50
- unstable oscillations for GeM and deeper networks

This shows **the architecture,** not the optimizer, was the **bottleneck.**
In other words, optimization instability was a symptom of over-expressive representations, not poor training hyperparameters.

6.5 K-Fold

The folds behaved differently even though all splits were stratified.
This confirms that:
- rare artists introduce **fold-dependent distribution shifts**
- validation accuracy alone can be misleading depending on which rare classes appear
- ensembles should be built selectively, not by blindly averaging all folds.

The decision to exclude Fold 3 from the ensemble is a direct consequence of this observation.

7. Conclusion

This assignment investigated scratch-based artist classification on a highly imbalanced and fine-grained dataset. Through systematic experimentation across **architectures, resolutions, pooling layers,** and **optimization strategies,** several consistent patterns emerged.
First, **model capacity** must be carefully controlled. Deeper networks such as ResNet-101 produced high training accuracy but failed to generalize, while shallower networks underfit. ResNet-50 offered the most stable balance between representation power and generalization.

Second, **input resolution** strongly influenced performance. Artist identification relies on fine-grained textures, but excessive resolution caused the model to learn noise. A resolution of **288×288** provided the optimal trade-off.

Third, **pooling strategy** played a crucial role. **GeM pooling** increased sensitivity to high-frequency texture statistics, resulting in severe overfitting and unstable validation behavior. Standard average pooling proved more reliable for this dataset. **Optimization** choices—Adam, Cross-Entropy with label smoothing, and Cosine Annealing—consistently yielded stable convergence across most folds.

Finally, **K-Fold** evaluation revealed that stratification alone cannot fully correct distribution shifts caused by rare classes. Some folds remained unstable, and selectively ensembling only the reliable folds resulted in better robustness than averaging all folds.

Overall, the findings highlight that **balanced model capacity, noise-aware resolution selection, and stability-focused training strategies** are essential for fine-grained classification with limited and imbalanced data. Future improvements could include more sophisticated class-balancing methods, synthetic data augmentation for rare artists, or adaptive sampling strategies to mitigate long-tail effects.