Ilaria Tangorre 5307483
Greta Riva 5306762

Skin Diseases Classification

Deep Learning for AI Project

Project Goal

Develop a model capable of classifying skin diseases images using a deep learning architecture

Data exploration

9 Skin Diseases

Basal Cell Carcinoma	500
Melanoma	505
Pigmented Benign Keratosis	500
Vascular Lesion	290
Acitinic Keratosis	500
Nevus	500
Squamous Cell Carcinoma	414
Dermatofibroma	400
Seborrheic Keratosis	500

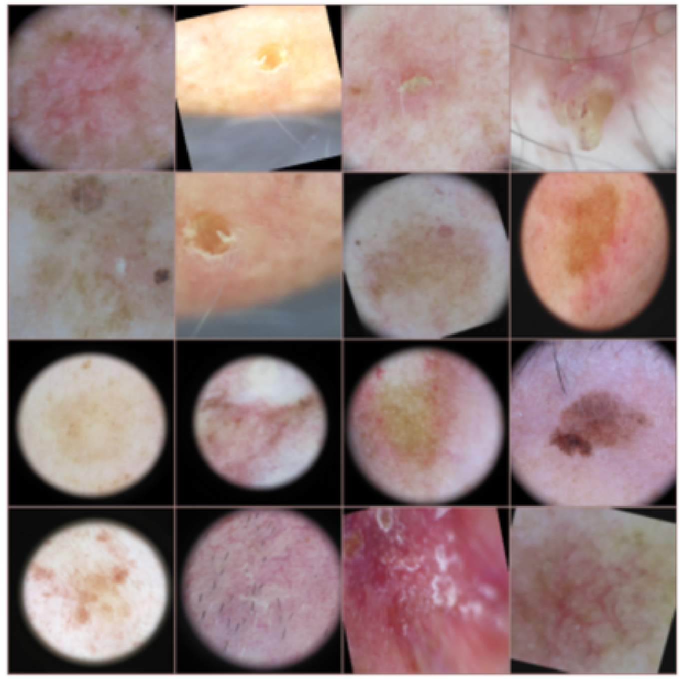
→ 4107 images

WELL BALANCED!

Data preprocessing

1. Data augmentation

These transformations increased variability, making the model more robust to input variations



- Random Transformation
- Data normalization

We performed normalization using the calculated mean and standard deviation for the RGB channels to standardize pixel intensities

2.Data oversampling

We replicate minority class samples by cycling through the dataset multiple times, controlled by oversampling factor 5

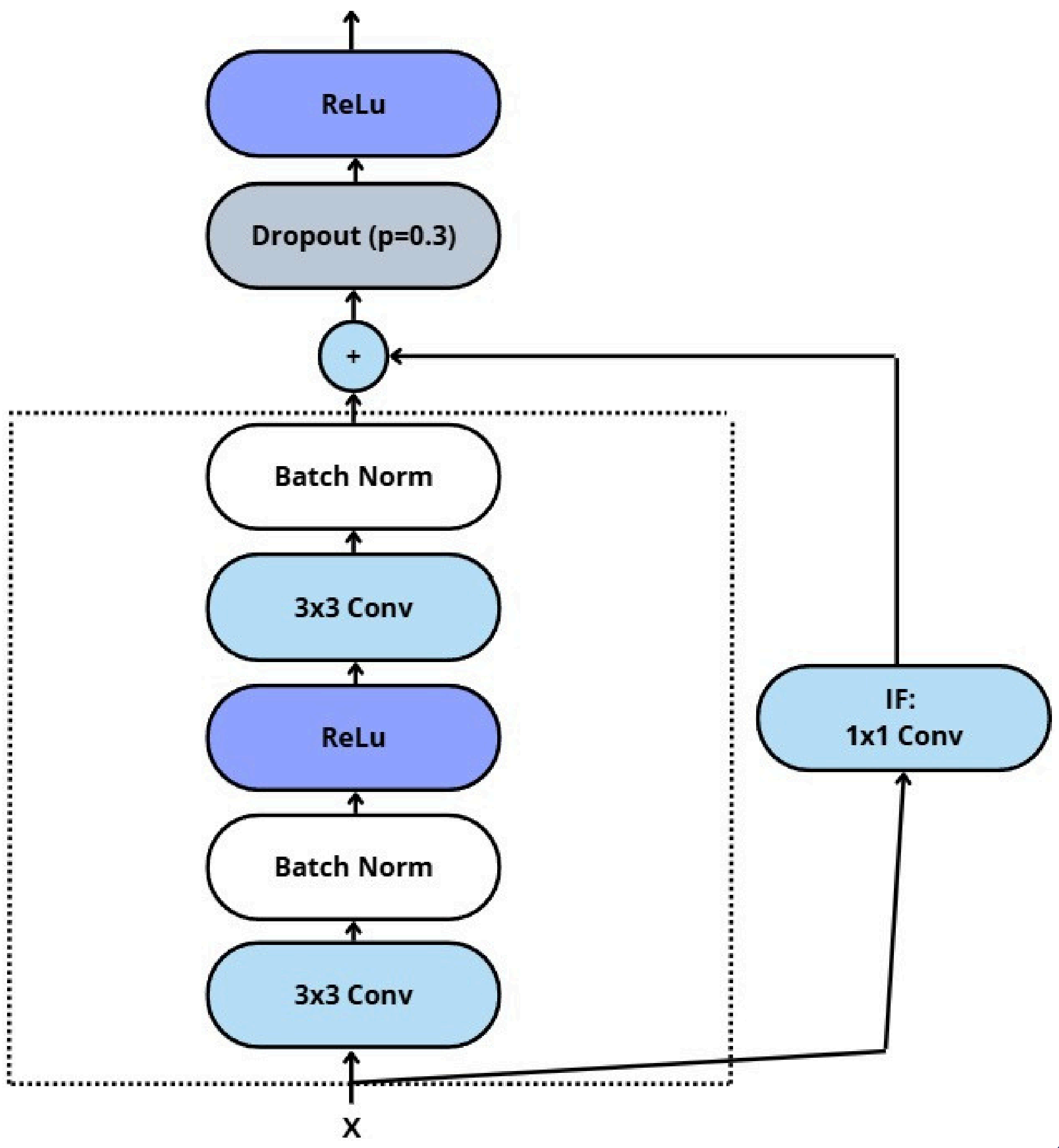
Why Resnet18 ?

- **Robust Learning:** Residual blocks solve the vanishing gradient problem, enabling efficient training in deep networks.
- **Balanced Complexity:** ResNet-18 strikes a perfect balance between model depth and computational efficiency, ideal for our dataset and Google Colab environment.
- **Generalization to Noise:** Effectively handles noise and artifacts in medical images, focusing on meaningful patterns.

Architecture

1. **Pass the input through:**
7x7 Convolution with stride 2 to extract initial features.
Batch Normalization to stabilize training.
ReLU activation function to introduce non-linearity.
MaxPooling with 3x3 kernel and stride 2 to reduce spatial dimensions.

2. **Apply 4 blocks:**
Layer1: 2 residual blocks with 64 channels.
Layer2: 2 residual blocks with 128 channels.
Layer3: 2 residual blocks with 256 channels.
Layer4: 2 residual blocks with 512 channels.



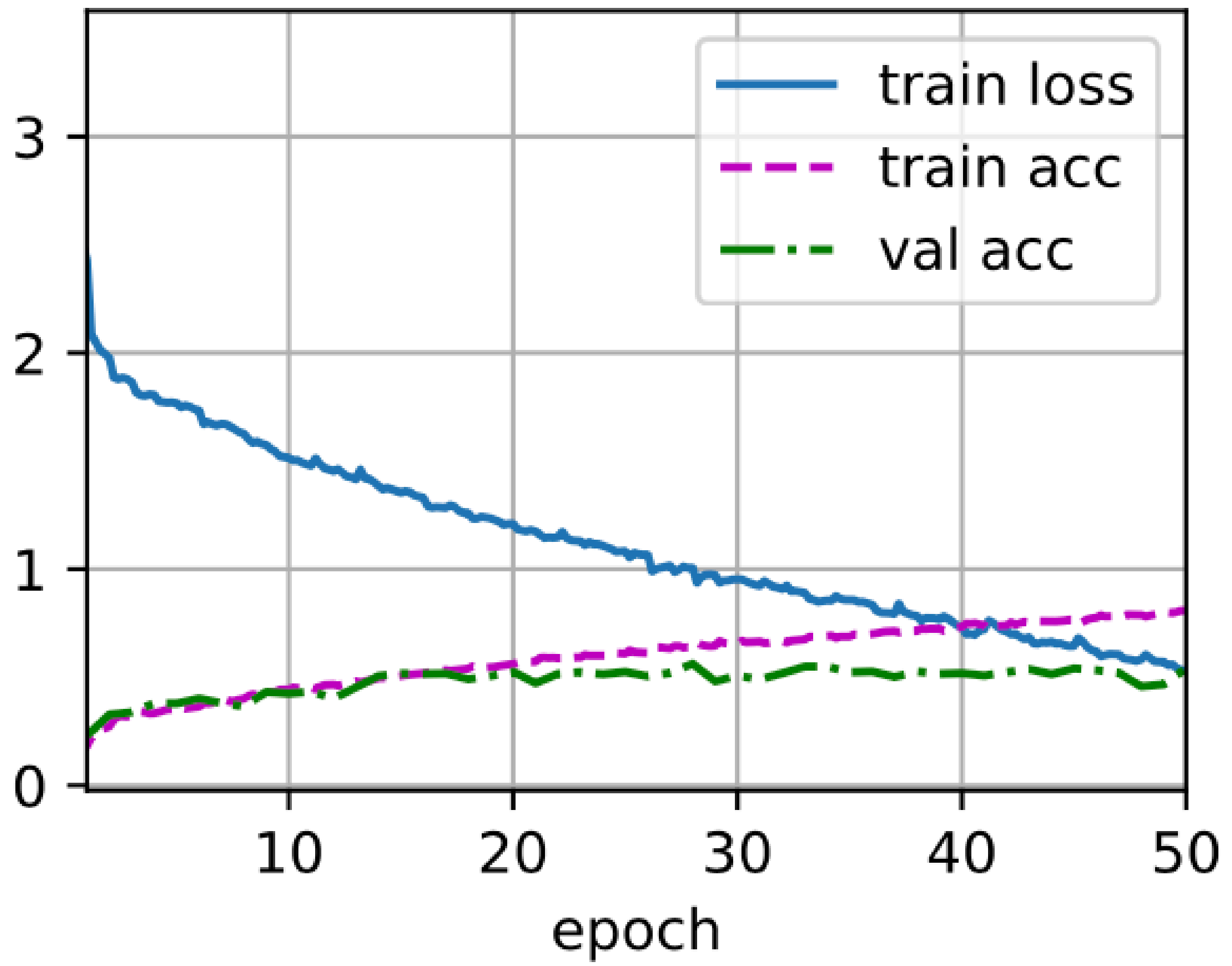
3. **Perform Adaptive Average Pooling** to reduce the spatial dimensions of the feature maps to 1x1, regardless of the input image size.

4. **Flatten the data** into a 1D vector to feed it into the fully connected layer.

5. **Add Dropout** for regularization to prevent overfitting.

6. **Pass through the Fully Connected Layer** to get the final predictions, outputting the classification result for 9 classes.

Training



Optimizer : Adam

it adapts the learning rate for each parameter, improving convergence and stability

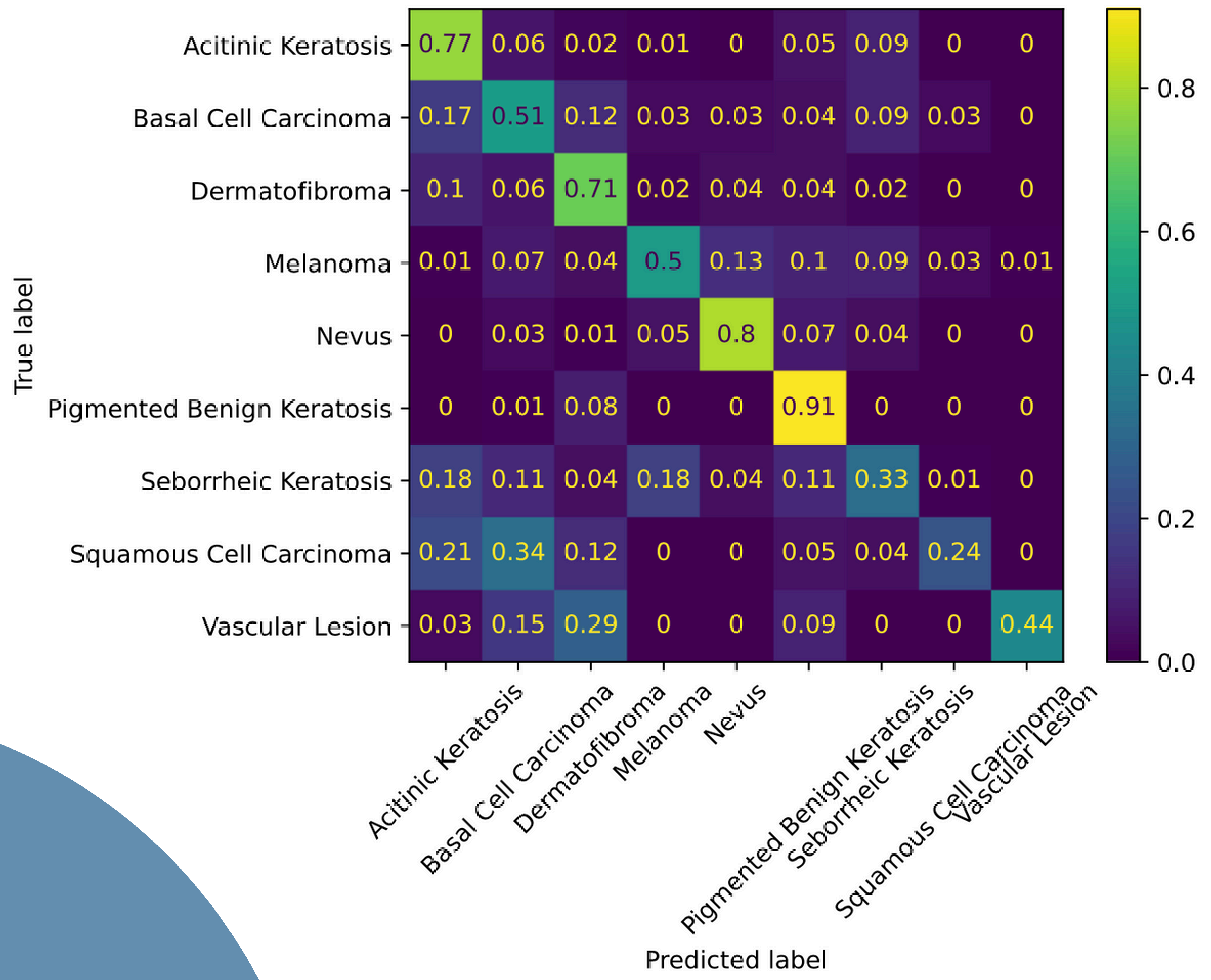
Loss Function: Cross Entropy Loss

is used in classification tasks to measure the difference between the predicted probabilities and the actual class labels, penalizing incorrect predictions more heavily as they diverge from the true labels.

Learning rate : 0.01

Number of Epochs: 50

Batch size: 16



Results

ACCURACY: 0.584415584415584

While this is not an excellent result, it aligns with the small size of the initial dataset. The results show that the model performed better with classes containing a larger number of images