DermaDeiT: Advancing Skin Image Classification via Knowledge Distillation

Aya Elgebaly, Muhammad Alberb

Methodology

Data Processing

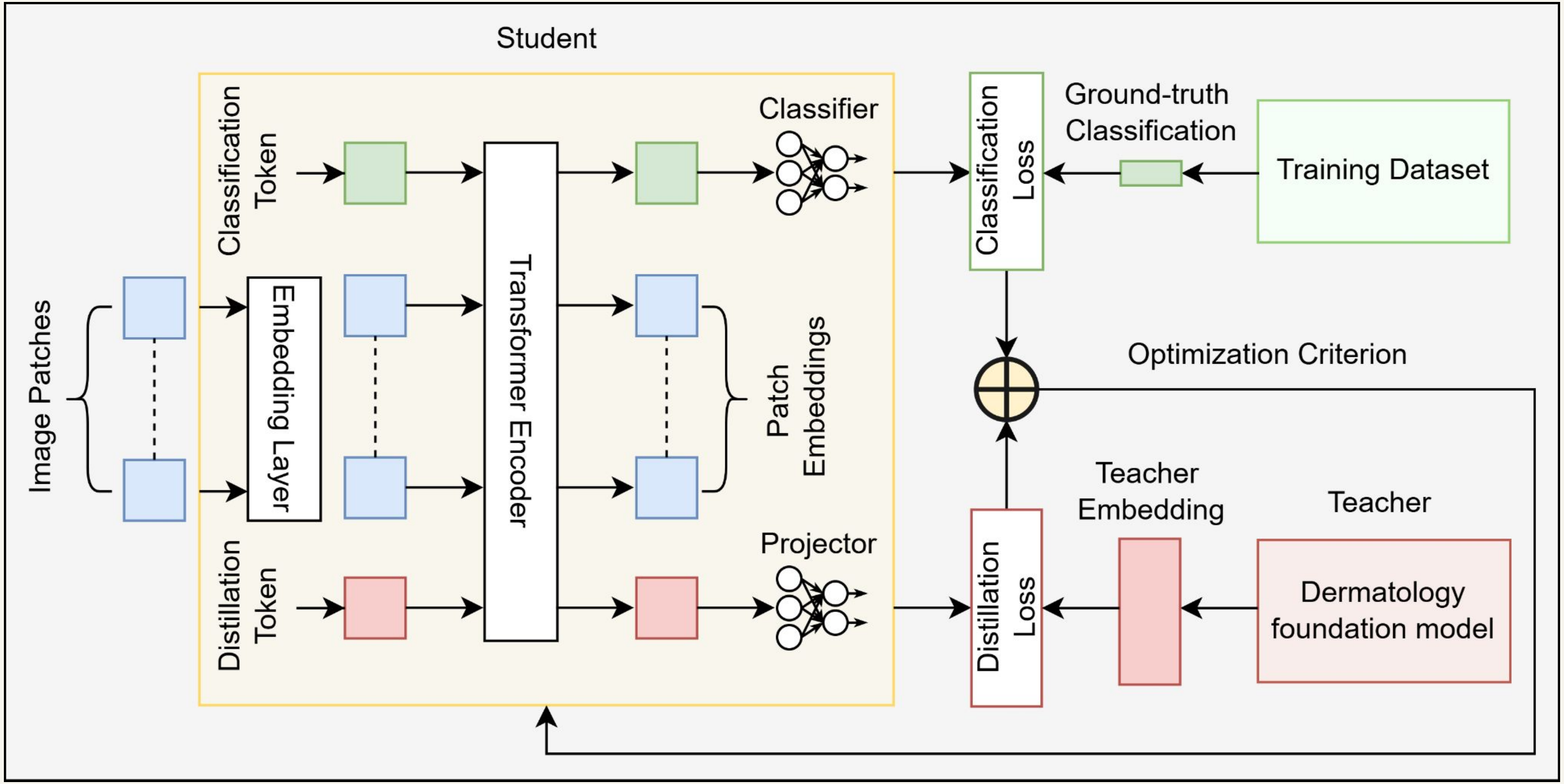
Basic Preprocessing

- Downsampling
- Normalization
- Class Balancing

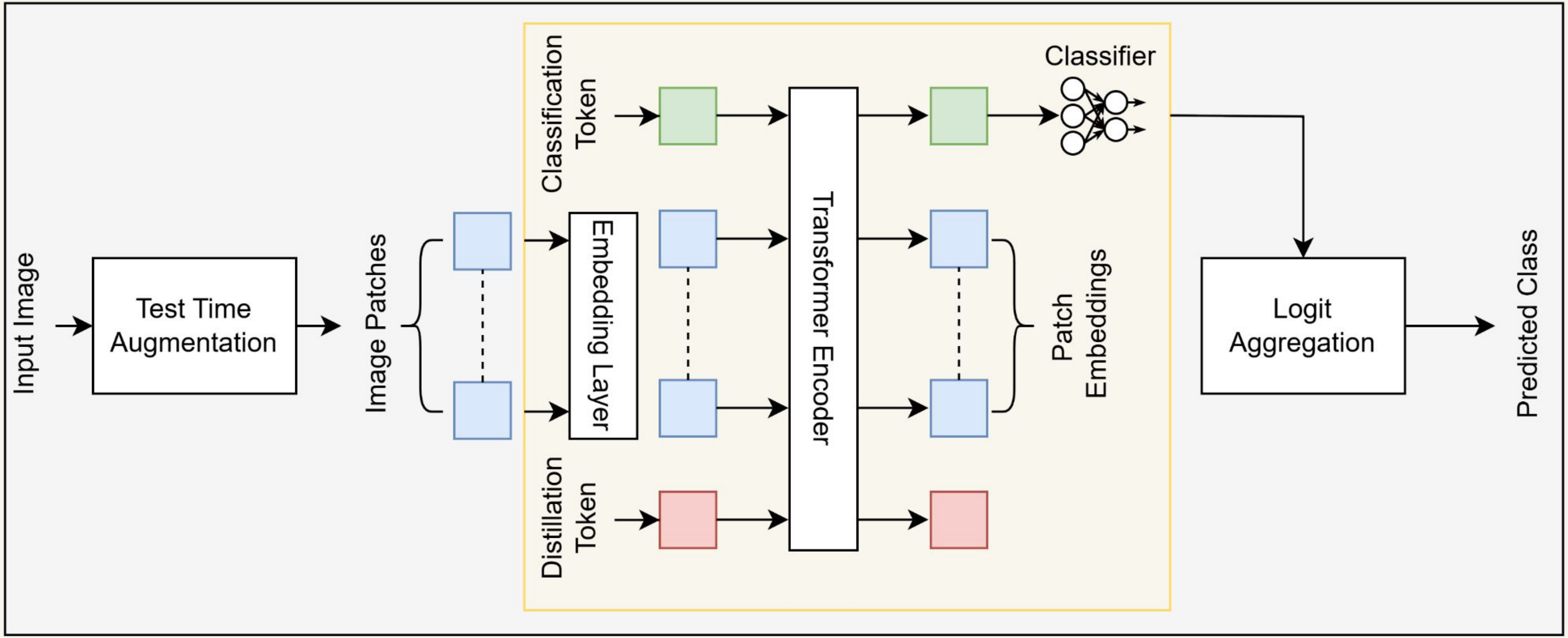
Data Augmentation

- Flipping
- Rotation
- Elastic Deformation

Training Pipeline



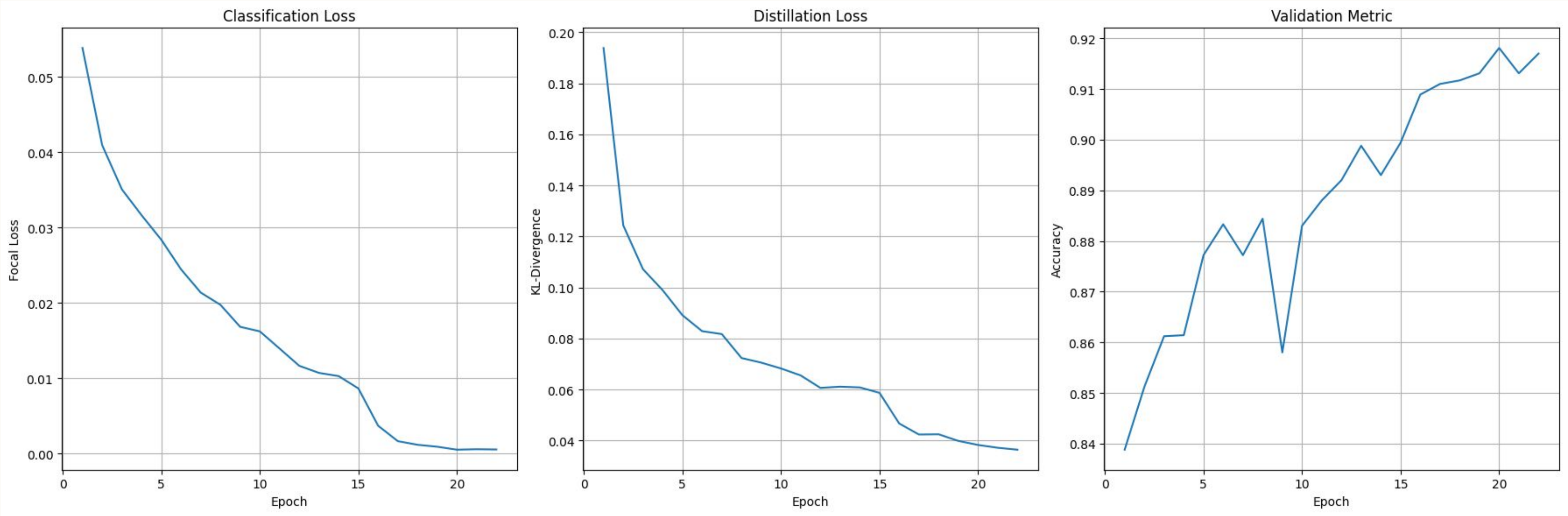
Inference Pipeline



Evaluation

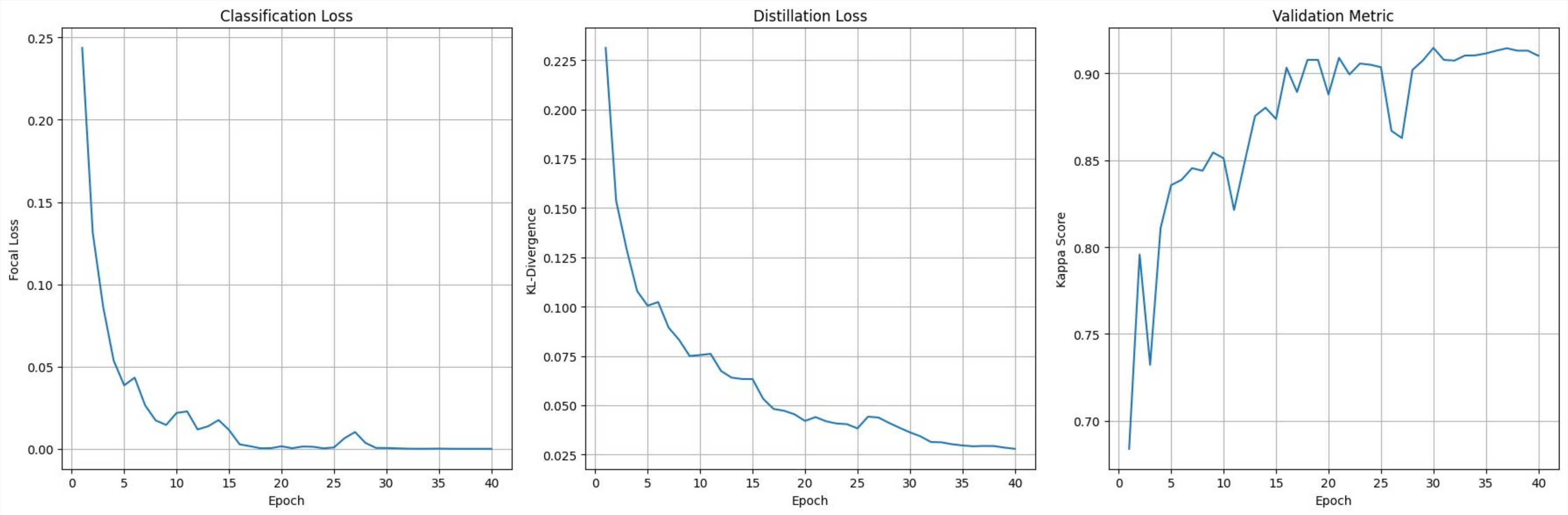
Training Curves

- Binary Challenge



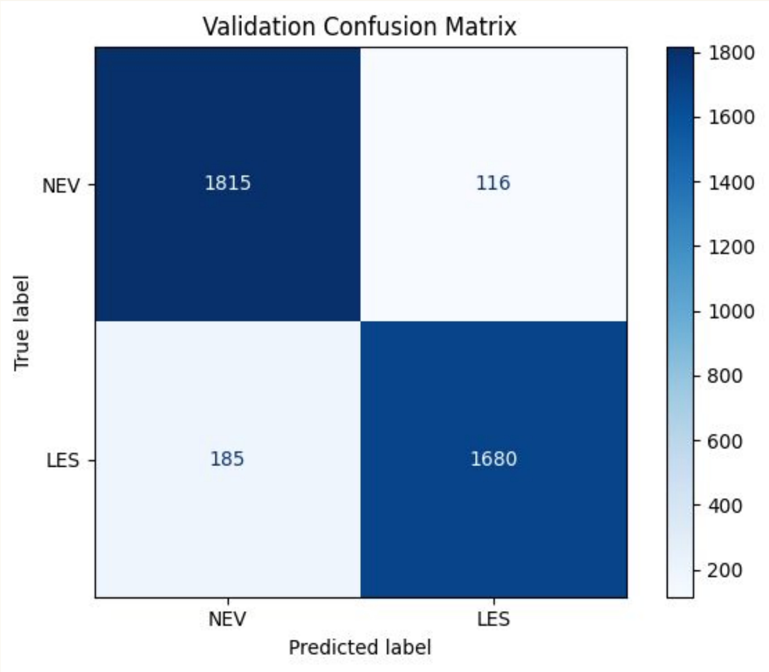
Training Curves

- Multi-class Challenge

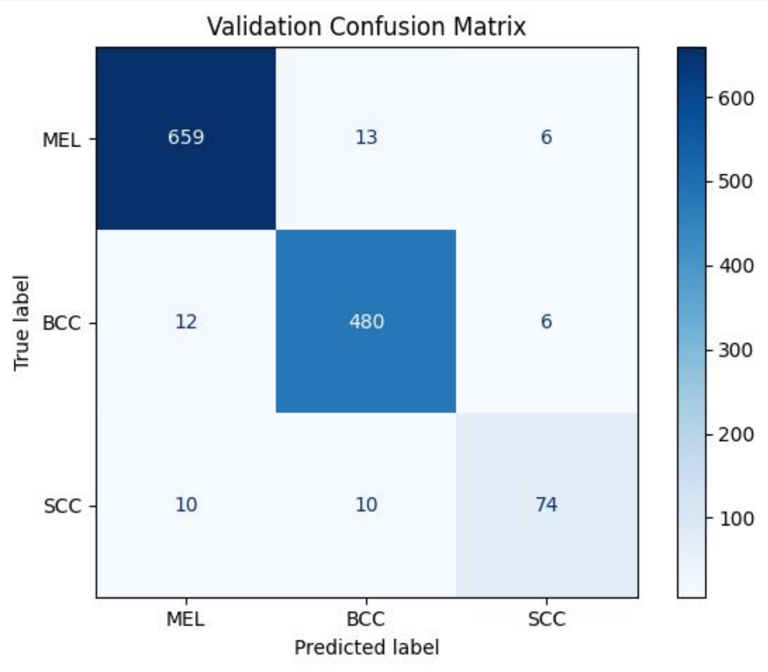


Confusion Matrices

- Binary Challenge



- Multi-class Challenge



DermaDeiT vs Baselines

- Binary Challenge

Method	Accuracy	Precision	Recall	F1-Score	Kappa
MobileNetV2	0.8512	0.8575	0.8359	0.8466	_
DeiT	0.9131	0.9132	0.9129	0.9130	0.8260
DINOv2	0.8775	0.8842	0.8638	0.8739	0.7548
GoogleDerm	0.8855	0.8859	0.8854	0.8854	0.7709
DermaDeiT	**0.9207**	**0.9215**	**0.9204**	**0.9206**	**0.8413**

DermaDeiT vs Baselines

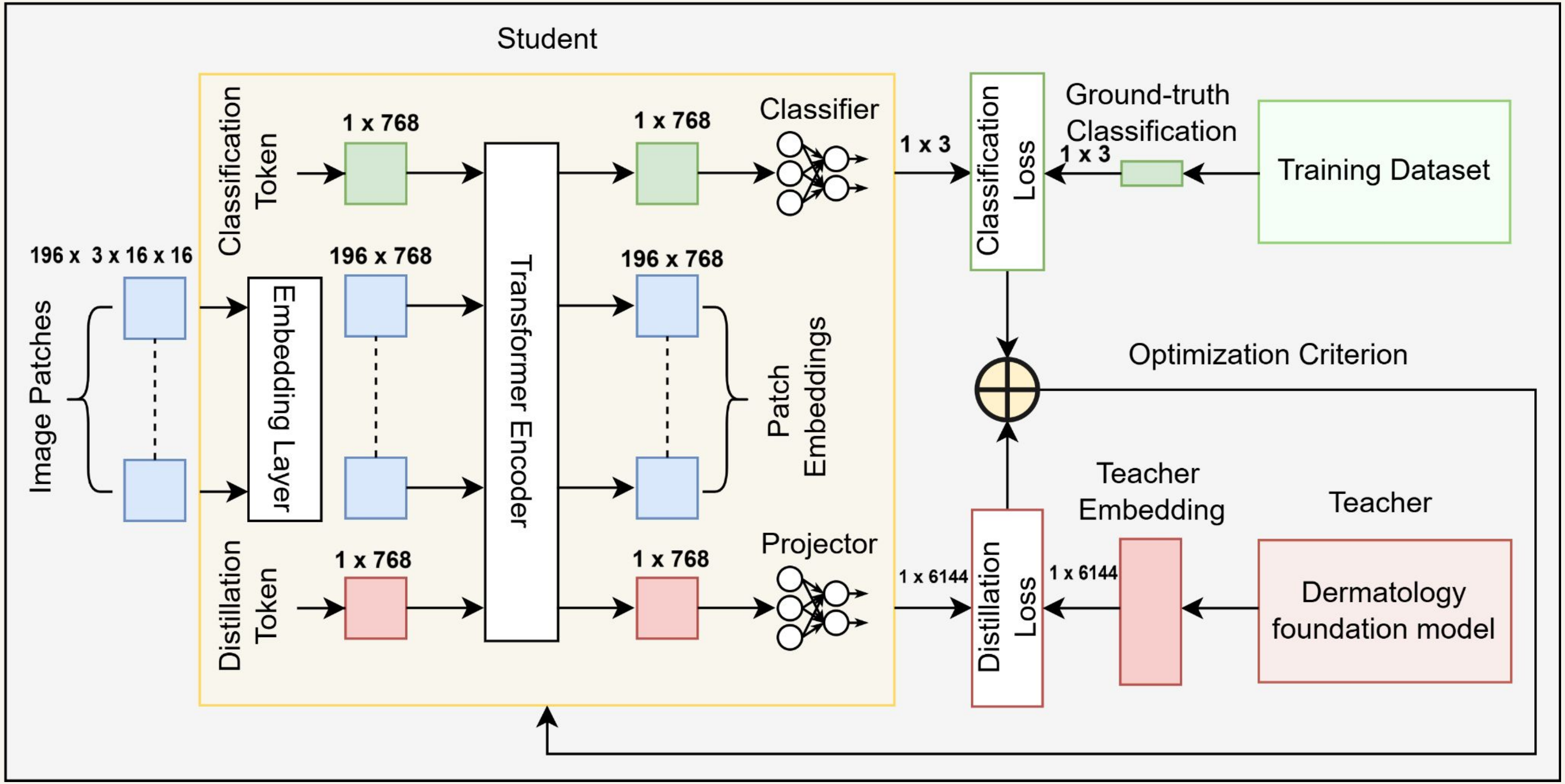
- Multi-class Challenge

Method	Accuracy	Precision	Recall	F1-Score	Kappa
EfficientNetV2	0.9157	0.8638	0.8702	0.8669	0.8487
DeiT	0.9488	**0.9289**	0.8890	0.9065	0.9073
GoogleDerm	0.9167	0.8680	0.8278	0.8451	0.8493
DermaDeiT	**0.9551**	0.9275	**0.9077**	**0.9170**	**0.9189**

ألف شكر😁

Appendix

Training Pipeline



Loss Functions

- Focal Loss for classification

$$L_c = -(1 - p_s)^{\gamma} \cdot \log(p_s)$$

- KL-Divergence for distillation

$$L_d = E_t \cdot \log\left(\frac{E_t}{E_s}\right)$$

- Total loss is a weighted-average

$$L_t = L_c + \alpha \cdot L_d$$

Training Details

- StepLR vs CosineAnnealing schedulers
- Adam vs AdamW
- 16-32 batches based on model size
- Nvidia T4 via Google Colab (free version😂)
- CPU for Embedding extraction (Aya’s M4 Pro)
- PyTorch, HuggingFace, and MONAI Implementation