HairNet: A Hair Type CNN Classifier

DSAN 6600: Neural Networks and Advanced Deep Learning

Morgan Dreiss Viviana Luccioli Satomi Ito
Yashwanth Devabathini

Hair type identification is essential for personalized hair care, yet current methods rely on subjective self-assessment and inconsistent visual comparison charts. No publicly available dataset or deep learning system exists for the full Andre Walker 10-class hair typing framework, which remains the most widely used taxonomy in industry settings. To address this gap, we introduce HairNet, a fine-grained hair type classifier built using EfficientNetV2-M with transfer learning and CORN ordinal regression to model the natural progression from Type 1 through Types 4a-4c. We constructed a custom dataset of ~14,000 images through a multi-stage pipeline integrating SerpAPI retrieval, YOLOv8 person filtering, Keras augmentation, and MediaPipe hair segmentation, followed by extensive manual re-labeling to correct systematic errors. Our model achieved 53.7% top-1 accuracy and 88.7% within-one-class accuracy, demonstrating strong performance on broad curl-pattern categories while highlighting remaining challenges in fine-grained distinctions, especially among Type 4 subtypes. Results indicate that data quality, rather than architecture depth, is the primary bottleneck in this domain. HairNet represents the first end-to-end deep learning system for full 10-class hair type classification, establishing a baseline for future research and offering practical potential for personalized hair-care applications.

Keywords: Neural Networks, CNN, Transfer Learning, Image Classification, EfficientNet-V2-M, Hair Type Classification, Image Processing, Data Augmentation, Deep Learning, Computer Vision
Affiliations: Georgetown University - Data Science and Analytics Masters Program

Table of contents

1 Introduction/Background

Hair care is a deeply personal yet surprisingly complex domain. What works beautifully for one person may leave another with frizz, breakage, or an oily scalp. This variability stems from fundamental differences in hair structure, specifically the shape of the hair follicle and the resulting curl pattern. Understanding one's hair type unlocks a wealth of personalized care knowledge: optimal wash frequency, ideal product formulations, appropriate styling tools, and protective practices that promote long-term hair health. This is especially critical for Type 4 coily hair, where distinctions between 4a, 4b, and 4c subtypes inform daily maintenance, protective styling choices, and moisture retention strategies.

We use the Andre Walker Hair Typing System (Wikipedia 2024), developed in the 1990s, which categorizes hair into four primary types: Type 1 (straight), Type 2 (wavy), Type 3 (curly), and Type 4 (coily/kinky). Each type subdivides into a, b, and c subcategories reflecting curl diameter, density, and texture variations. Despite criticisms, this system remains the most widely recognized framework in the hair care industry.

1.1 Limitations of Existing Approaches

Current hair type identification relies on visual comparison charts, online quizzes, or trial-and-error. These methods suffer from: (1) static comparison images failing to capture within-category variability, (2) self-assessment bias from lighting, styling state, and prior assumptions, and (3) limited access to trained stylists. Computationally, hair analysis research has focused on color, baldness detection, and coarse texture categories. Fine-grained classification using established systems like Andre Walker's has received minimal attention.

1.2 Contributions

1. **Deep learning application:** First application of EfficientNetV2-M, fine-tuned on our custom dataset to ten-category Andre Walker classification
2. **Custom dataset pipeline:** Multi-stage pipeline combining Google search API retrieval, YOLOv8 filtering, Keras augmentation, and MediaPipe segmentation
3. **Web application:** Deployed a usable application using HuggingFace

This paper will follow the following structure. First, we will give a brief overview of previous related works. Second, we will discuss our four-stage dataset pipeline process. Third, we will describe our overall approach: choosing our model and its training implementation choices, followed by our overall iterative experimental process. Lastly, we will share the results of our best fitting model and will discuss implications.

2 Related Work

2.1 Prior Hair Classification Approaches

Borza, Ileni, and Darabant (2018) achieved 92% accuracy in CNN-based hair color recognition. Muhammad et al. (2018) combined CNN features with traditional classifiers for 90% segmentation accuracy. For scalp disorders, Chowdhury et al. (2024) achieved 92% accuracy with modified Xception on 241 images, outperforming VGG19, Inception, ResNet, and DenseNet (50-80% accuracy). Karo Karo et al. (2023) achieved 94.5% with VGG-16 on 4,000 hair disease images.

For hair type classification specifically, Dertli and Koklu (2025) used SqueezeNet, InceptionV3, and VGG19 as feature extractors with ML classifiers on 1,992 images across five categories, achieving 84.9% with InceptionV3-ANN. The Hair-Type-Classifier project achieved 88% validation accuracy with ResNet18 on five categories. Ravishankar (2025) achieved 81.67% with EfficientNet-B2 but used only four classes and 400 images, excluding Types 1 and 4. Meishvili et al. (2024)'s "Hairmony" system showed that strand-based approaches exhibit bias toward straight hair; their DINOv2-based approach demonstrated improved fairness.

2.2 Datasets

The only public dataset (Kavyasree 2023), to our knowledge, classifies images into five categories (Straight, Wavy, Curly, Kinky, Dreadlocks), conflating Andre Walker categories and mixing hairstyles with hair types. Prior work uses 241-2,000 images. This is why our custom dataset (which classifies in the 10 hair types) introduces new possibilities in this domain.

3 Datasets

3.1 Data Collection

We used SerpAPI Google Images to gather images across ten categories (Type 1, Types 2a-2c, Types 3a-3c, Types 4a-4c). Initial testing showed single-query searches yielded repetitive results beyond early pages, so we modified our approach and implemented 25 semantically varied queries per hair type with maximum two shared words between queries. For Type 3b: "type 3b curly hair," "springy curls 3b hair," "corkscrew curls 3b," "3b penny-sized curls."

We retrieved pages 0 and 1 per query with MD5 duplicate detection and rate limiting (0.3s between downloads, 1s between pages), targeting 4,000 images per type.

3.2 Data Preprocessing and Augmentation

YOLO Filtering: YOLOv8 nano filtered images to retain only single-person detections (confidence threshold 0.5), ensuring unambiguous training samples.

Augmentation: Keras-based augmentation targeted 1,200 minimum samples per class using rotation (±15°), brightness adjustment (80-120%), and horizontal flipping. These preserve hair texture while adding realistic variation; aggressive augmentations that could distort curl patterns were avoided.

Segmentation: MediaPipe hair segmentation isolated hair regions using 0.7 confidence threshold. Backgrounds were replaced with white, bounding boxes computed with 5% padding, and images cropped and resized to 600×600 pixels.

3.3 Data Splits

We used 70/15/15 splits: training (11,363 images), validation (2,437), testing (2,440). Stratified splitting via scikit-learn's train_test_split ensured minority classes (Types 2b, 2c, 3a at ~1,200 samples) maintained representative proportions.

4 Proposed Method/Architecture

4.1 Overall Approach

We leverage transfer learning from ImageNet-pretrained CNNs. This approach was selected because: (1) hair texture classification requires recognition of edges, textures, and shapes, so models already trained on ImageNet's diverse data has an advantage in doing this well, (2) our ~16,000 images were insufficient for training from scratch, and (3) pretrained models reduce training time while improving generalization.

We selected EfficientNet for its state-of-the-art performance with computational efficiency. Compound scaling balances depth, width, and resolution, producing superior accuracy with fewer parameters than ResNet or VGG (Jaguuai 2024). We evaluated EfficientNet-B7 and EfficientNetV2-M. The table below compares the two architectures (HackMD 2024):

Table 1: EfficientNet Model Comparison

Feature	EfficientNet-B7	EfficientNetV2-M
ImageNet Accuracy	84.4% top-1	Comparable or better
Training Speed	Slower	Up to 11× faster
Parameters	~66M	~53M

After an iterative experimental process described in section 5.1, we elected EfficientNet V2-M as our final model as it yielded highest performance.

4.2 Model Architecture

EfficientNetV2-M employs MBConv and fused-MBConv blocks combining depthwise separable convolutions with squeeze-and-excitation attention (Tan and Le 2021). Fused-MBConv blocks in early stages replace depthwise convolutions with regular 3x3 convolutions for faster training. Our input resolution for the final model was 512x512 as we ran this on the non-segmented dataset (data only went through YOLO and augmentation but skipped MediaPipe segmentation). 512x512 was chosen as it balances granularity and efficiency. The backbone outputs 1280 dimensional features after global average pooling. We replaced this head with a 10 class linear layer. In total, this was ~52.9M parameters, all fine-tuned.

5 Training Procedure

A key innovation of our approach was the implementation of the **CORN (Conditional Ordinal Regression for Neural Networks)** loss function, which exploits the ordinal structure from Type 1 through Types 4a-4c. Whereas standard multi-class cross entropy treats all misclassifications the same, CORN decomposes K-class classification into K-1 binary tasks determining whether true labels exceed rank k, penalizing predictions proportionally to distance from true class.

Training Configuration:

- **Loss Function:** CORN (Conditional Ordinal Regression for Neural Networks)
- **Optimization:** AdamW
- **Learning rate:** 3×10 , cosine annealing to 1×10
- **Batch size:** 8 (physical) × 4 (gradient accumulation) = 32 effective
- **Epochs:** 20
- **Regularization:** Weight decay via AdamW; no dropout, relying on batch normalization and pretrained initialization
- **Resolution:** 600×600 (segmented) or 512×512 (non-segmented)
- **Mixed precision:** FP16 via PyTorch AMP
- **Normalization:** ImageNet statistics

The model was trained using PyTorch and was run on the hardware configurations of Google Colab NVIDIA Tesla T4 (16GB VRAM) with ~8.5GB memory usage per run. Each epoch trained for ~14 minutes, resulting in ~4.5 hours total training.

6 Experiments

6.1 Experimental Setup

Implementation: PyTorch 2.0+ with torchvision pretrained EfficientNetV2-M (IMAGENET1K_V1), native AMP, ImageFolder dataset, multi-worker prefetching.

Iterative Process:

1. Initial comparison of B7 and V2-M on segmented data with cross-entropy: 42.18% validation accuracy
2. CORN loss introduction: limited improvement on segmented data
3. Frozen-layer experiment (head-only training): significantly worse results, indicating feature extraction was the bottleneck
4. Non-segmented 512×512 images with CORN: 46.68% validation accuracy
5. Dataset inspection revealed systematic labeling errors from Google Images
6. Manual curation and re-processing

Final Configurations: Four configurations tested (B7/V2-M × segmented/non-segmented) with CORN loss on manually-filtered data. Best: EfficientNetV2-M with CORN on non-segmented manually-filtered data.

6.2 Evaluation Metrics

Accuracy: Standard top-1 classification accuracy for model comparison.

Weighted F1 Score: Accounts for class imbalance by computing per-class precision-recall harmonic mean, weighted by support.

Within-One-Class Accuracy: Proportion of predictions within one ordinal rank of true label, recognizing that adjacent hair types exhibit overlapping characteristics.

Confusion Matrix Analysis: Visualizes prediction distributions to identify systematic error patterns, specifically whether errors concentrate near the diagonal.

7 Results

This section presents quantitative and qualitative findings for the two primary model configurations evaluated in this work.

7.1 EfficientNetB7 Model

The following results show the performance of the EfficientNet-B7 model trained on MediaPipe segmented images. It achieved an overall accuracy of 0.53 on the test set. Performance was strongest for Type 1 and Type 2 categories with smooth texture, gentle wave patterns and degraded progressively for tightly clustered curl subtypes. The confusion matrix (Figure 1) shows errors concentrated primarily between neighboring classes, indicating that the model captured coarse texture structure but struggled with fine-grained subclass boundaries.

Table 2: EfficientNetB7 Classification Report

Hair Type	Precision	Recall	**F1-Score**	Support
1	0.85	0.90	**0.87**	291
2a	0.71	0.70	**0.70**	291
2b	0.66	0.65	**0.66**	245
2c	0.58	0.57	0.57	242
3a	0.48	0.38	0.43	239
3b	0.41	0.35	*0.38*	251
3c	0.44	0.54	0.48	240
4a	0.38	0.29	*0.33*	234
4b	0.20	0.08	*0.11*	318
4c	0.44	0.76	0.56	387
Accuracy			**0.53**	2738
Macro avg	0.52	0.52	0.51	2738
Weighted avg	0.51	0.53	0.51	2738

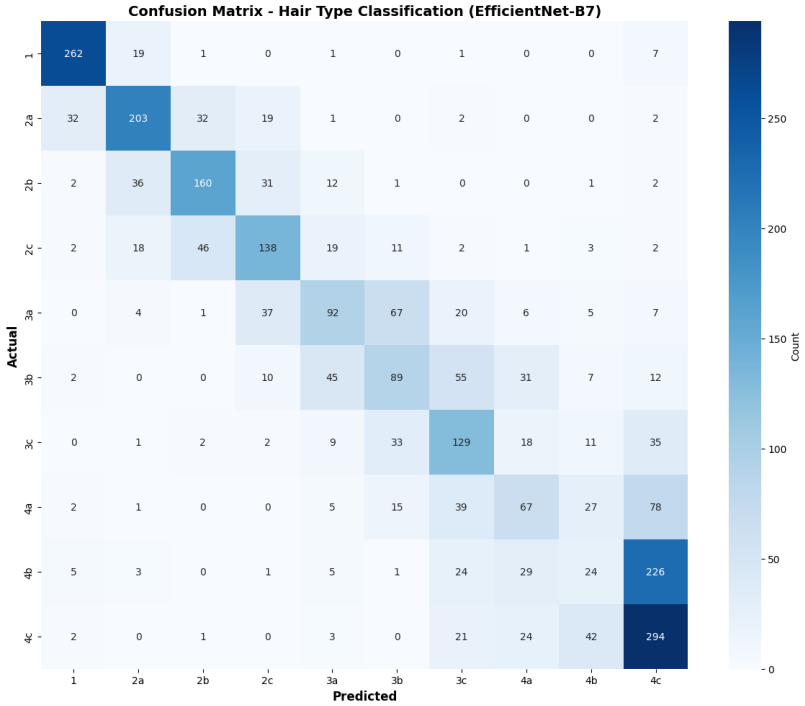

Figure 1: EfficientNetB7 Confusion Matrix

7.2 EfficientNetV2-M Model

Table 3: EfficientNetV2-M Classification Report

Hair Type	Precision	Recall	**F1-Score**	Support
1	0.90	0.91	**0.90**	225
2a	0.71	0.78	**0.74**	219
2b	0.60	0.55	0.57	186
2c	0.58	0.55	0.57	184
3a	0.50	0.39	0.44	180
3b	0.44	0.46	0.45	192
3c	0.47	0.54	0.50	182
4a	0.35	0.46	*0.40*	180
4b	0.32	0.27	*0.30*	260
4c	0.50	0.47	0.48	324
Accuracy			**0.54**	2132
Macro avg	0.54	0.54	0.54	2132
Weighted avg	0.54	0.54	0.53	2132

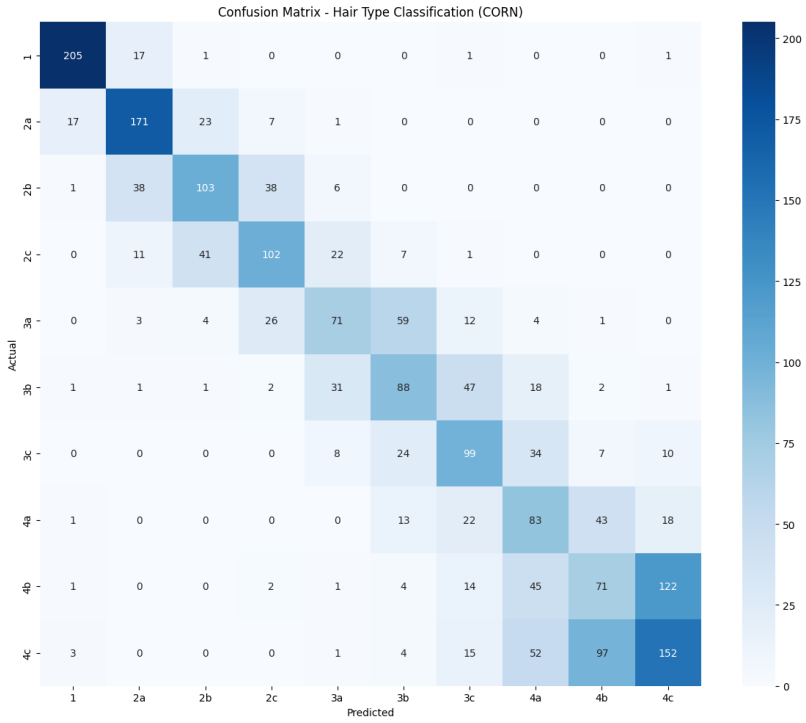

Figure 2: EfficientNetV2-M Confusion Matrix

The EfficientNetV2-M model was trained on both segmented and non-segmented variants of the dataset during the experimental phase. The results shown here are from the non-segmented configuration, which produced superior performance. Using the curated non-segmented images, the model achieved an overall accuracy close to 0.54, outperforming both the segmented data V2-M and B7 models.

The training progression (Figure 3) shows gradual improvement and fine-tuned by 14 epochs with early stopping. The confusion matrix shows a stronger diagonal dominance compared to the segmented configurations, with fewer extreme misclassifications but still struggling with ambiguous subtype boundaries in Types 3, 4.

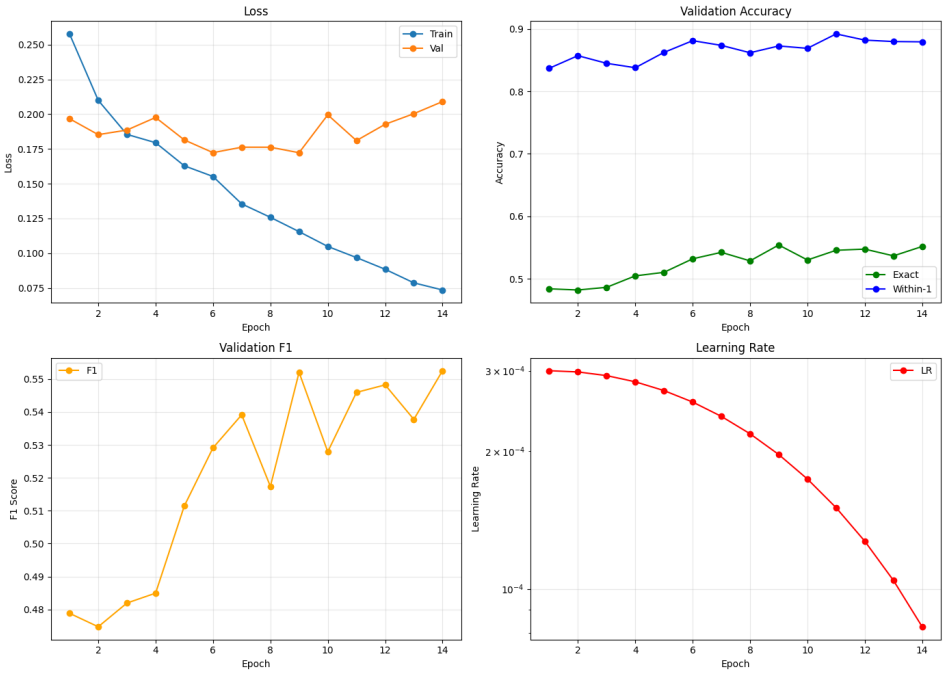

Figure 3: EfficientNetV2-M Training Progression

7.2.1 Computational Analysis

Table 4: EfficientNetV2-M fine-tuned model performance

Metric	Value
Model size	~52.9M parameters, ~208 MB
Training Memory usage	~8.5 GB (Tesla T4 GPU)
Training speed (per epoch)	~550 sec.
Inference speed (per image)	20-30ms; <1 sec. on Hugging Face app

Model Size

Our final model, EfficientNetV2-M, contains approximately 52.9 million parameters. This represents a mid-sized CNN architecture that balances strong representational capacity with computational efficiency and it is significantly smaller than models such as EfficientNet-B7 (~66M parameters), contributing to faster training and inference while preserving accuracy on fine-grained tasks.

Inference Speed

EfficientNetV2-M's fused-MBConv architecture contributes to its fast execution compared to older EfficientNet variants. On an NVIDIA Tesla T4 GPU (16GB VRAM), a single forward pass at 512×512 resolution runs in approximately 20–30 ms. This speed allows for:

- real-time or near–real-time inference
- responsive web applications (e.g., the HuggingFace)
- scalable batch processing

Memory Usage

Full model fine-tuning required about 8.5 GB of VRAM, leaving limited headroom on the T4 GPU. This constraint influenced several design decisions:

- Physical batch size limited to 8, expanded to 32 via gradient accumulation
- Input resolution capped at 512×512 for non-segmented data
- Mandatory use of mixed-precision (FP16) to fit the model and activations into memory

Despite these constraints, the model trained efficiently and remained deployable on standard cloud GPUs.

8 Discussion

8.1 Interpretation of Results

Our best-performing model, EfficientNetV2-M fine-tuned with CORN ordinal regression, achieved 53.7% top-1 accuracy and 88.7% within-one-class accuracy. This large discrepancy highlights an important insight: while the classifier struggles with exact subtype distinctions, it reliably captures the overall curl-pattern progression defined by the Andre Walker system.

Confusion matrix patterns show that most prediction errors occur between adjacent classes (e.g., 3a with 3b, 4a with 4b), rather than distant ones. This behavior is desirable for an ordinal domain, indicating that the model learned meaningful representations of curl diameter, density, and texture, even if exact subtype boundaries remain ambiguous. Thus, our results confirm that hair type classification is better framed as an ordinal task, and CORN effectively supports this structure.

8.2 Strengths and Weaknesses

A central strength of our work is the creation of a full end-to-end dataset pipeline, integrating Google Images retrieval, YOLOv8 filtering, targeted augmentation, MediaPipe segmentation, and extensive manual relabeling. This pipeline enabled us to produce a large-scale datasets covering all ten Andre Walker categories.

Another strength is our methodological insight: architecture choice contributed far less to performance than data quality. The largest improvements came from correcting mislabeled samples, rather than from modifying the CNN design itself.

However, several weaknesses remain. Despite multiple filtering stages, label noise persisted, especially among Type 3 and Type 4 images collected from Google Images. Segmentation sometimes degraded image quality by removing contextual cues (e.g., face outline, hair length) that aided classification. Furthermore, class imbalance and within-class diversity limited the model's ability to distinguish borderline subtypes, reinforcing that data quality, not model capacity, is the primary bottleneck.

8.3 Unexpected Findings

Two findings diverged from our expectations — First, segmented images underperformed non-segmented images, despite our assumption that isolating hair pixels would improve texture recognition. Segmentation occasionally distorted curl patterns or removed structural cues helpful to the model.

Second, the effect of manual relabeling was larger than predicted. Even limited correction of mislabeled samples significantly increased accuracy, demonstrating that fine-grained classification models are extremely sensitive to label noise.

8.4 Real-World Applicability

Although top-1 accuracy is modest, the 88.7% within-one-class performance indicates strong potential for real-world use. Because hair typing is inherently subjective so accurate placement within one adjacent category is typically sufficient for meaningful recommendations. For deployment, we must consider fairness, demographic diversity, and user trust. Future improvements may incorporate more curated datasets, hybrid segmentation–context models, and active learning loops to reduce labeling errors.

Potential applications include:

- personalized hair-care product recommendations
- virtual or mobile-based styling consultations
- automatic tagging for retail or e-commerce
- consumer-facing self-assessment tools

9 Conclusion

9.1 Summary of Contributions

This project represents the end-to-end implementation of a deep learning system capable of classifying all ten categories of the Andre Walker hair typing system. Our team collaborated across multiple components of the pipeline:

- All team members contributed to data scraping, manual curation, and class reassignment, ensuring the dataset's improved accuracy after identifying systematic labeling errors from Google Images
- Yashwanth led the hair segmentation pipeline, implementing MediaPipe-based masking, bounding box extraction, and preprocessing for segmented inputs
- Morgan seveloped the web application and deployment, building an accessible interface for real-time model inference and user interaction
- Viviana implemented and fine-tuned the EfficientNetV2-M model, integrating CORN ordinal regression and running the experiments that yielded the final best-performing configuration
- Satomi implemented the EfficientNet-B7 model, conducted baseline comparisons, and analyzed the effects of different preprocessing strategies and architectures

Together, these contributions resulted in a comprehensive data pipeline, multiple model baselines, and a deployable application demonstrating the feasibility of automated hair type classification.

9.2 Key Takeaways

Our findings show that data quality is the primary bottleneck in fine-grained hair type classification. Improvements from manual relabeling far exceeded gains from architectural upgrades. EfficientNetV2-M, combined with ordinal modeling, performed well at capturing broad curl-pattern categories, while remaining challenged by fine subtype distinctions — an expected outcome given the inherent ambiguity of hair typing. The system's 88.7% within-one-class accuracy demonstrates strong potential for practical use, as approximate classification is often sufficient for product recommendations and self-guided hair care decisions.

9.3 Limitations

This work remains limited by the use of web-sourced images, which introduced noise, inconsistent labeling, and demographic imbalance. Segmentation occasionally removed useful contextual cues, and the size of several subtypes restricted model generalization. Hardware constraints on Colab T4 GPUs also limited batch size and resolution.

Future progress will depend on constructing a more diverse, carefully labeled dataset and exploring hybrid preprocessing strategies that combine segmentation with context-aware learning.

Appendix

Appendix 1: Code

The entirety of our code can be found on Github

Appendix 2: Web Application

An application with our final model is deployed to HuggingFace. The user can upload a JPG file and find out their hair type.

Application Link: https://huggingface.co/spaces/med2106/hairNet

References

Borza, Diana, Tudor Ileni, and Adrian Darabant. 2018. "A Deep Learning Approach to Hair Segmentation and Color Extraction from Facial Images." In *Proceedings of the International Conference on Advanced Concepts for Intelligent Vision Systems*.

Chowdhury, Md Shahin, Tahmina Sultan, Nusrat Jahan, M. F. Mridha, Mejdl Safran, Sultan Alfarhood, and Dunren Che. 2024. "Leveraging Deep Neural Networks to Uncover Unprecedented Levels of Precision in the Diagnosis of Hair and Scalp Disorders." *Skin Research and Technology* 30 (4): e13660.

Dertli, Burak, and Murat Koklu. 2025. "Classification of Hair Types with Deep Learning Methods." In *Proceedings of the 5th International Conference on Trends in Advanced Research*, 463–78. Konya, Turkey.

HackMD. 2024. "EfficientNet-B7 Vs EfficientNetV2-m Comparison." https://hackmd.io/ylIsCTkRRLGFA9WSF5iUnQ.

Jaguuai. 2024. "CNN: VGG, ResNet, DenseNet, MobileNet, EfficientNet, and YOLO." Medium. https://medium.com/@jaguuai/cnn-vgg-resnet-densenet-mobilenet-effecientnet-and-yolo-2329a9fa2d0f.

Karo Karo, Immanuel Marthin, Dedi Kiswanto, Suheri Panggabean, and Aris Perdana. 2023. "Hair Disease Classification Using Convolutional Neural Network (CNN) Algorithm with VGG-16 Architecture." *Sinkron: Jurnal Dan Penelitian Teknik Informatika* 7 (4): 2786–93.

Kavyasree. 2023. "Hair Type Dataset." Kaggle. https://www.kaggle.com/datasets/kavyasreeb/hair-type-dataset.

Meishvili, Givi, James Clemoes, Charlie Hewitt, Zafiirah Hosenie, Xiao Xiao, Martin de La Gorce, Tibor Takacs, et al. 2024. "Hairmony: Fairness-Aware Hairstyle Classification." *arXiv Preprint arXiv:2410.11528*.

Muhammad, Umar Riaz, Michele Svanera, Riccardo Leonardi, and Sergio Benini. 2018. "Hair Detection, Segmentation, and Hairstyle Classification in the Wild." *Image and Vision Computing* 71: 25–37.

Ravishankar, Anjali. 2025. "Making a Neural Network Classify Curly Hair so You Don't Have To." Medium.

Tan, Mingxing, and Quoc V. Le. 2021. "EfficientNetV2: Smaller Models and Faster Training." *arXiv Preprint arXiv:2104.00298*.

Wikipedia. 2024. "Andre Walker Hair Typing System." https://en.wikipedia.org/wiki/Andre_Walker_Hair_Typing_System.