Dataset: DeepWeeds

Model: ResNet50

**1. 🧠 Architecture Diagram (text‑based)**

Input image (224×224×3)

⮕ ResNet‑50 trunk (pre‑trained on ImageNet; ~23.5 M parameters)

└ includes conv1 → conv5\_block3 stages

⮕ Attention module added after ResNet trunk:

• likely a channel/spatial attention module

• attention‑augmented feature maps modulate conv5 activations

⮕ Global average pooling on attended feature maps

⮕ Dense classifier: sigmoid outputs for 9 classes (8 weeds + negative)

*(Visual diagram not in notebook; above is inferred outline.)*

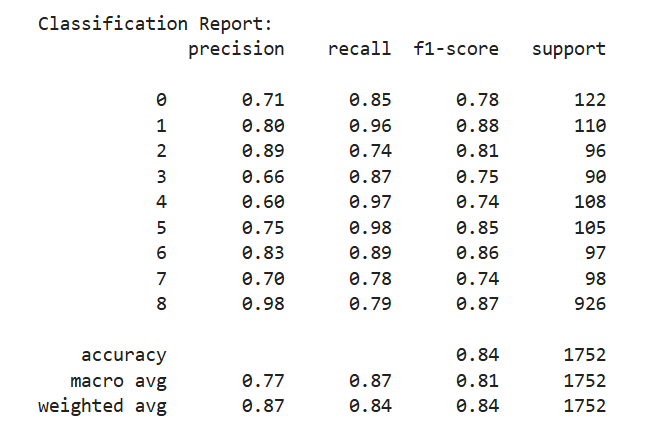
**2. Algorithm Procedure – Step‑by‑Step**

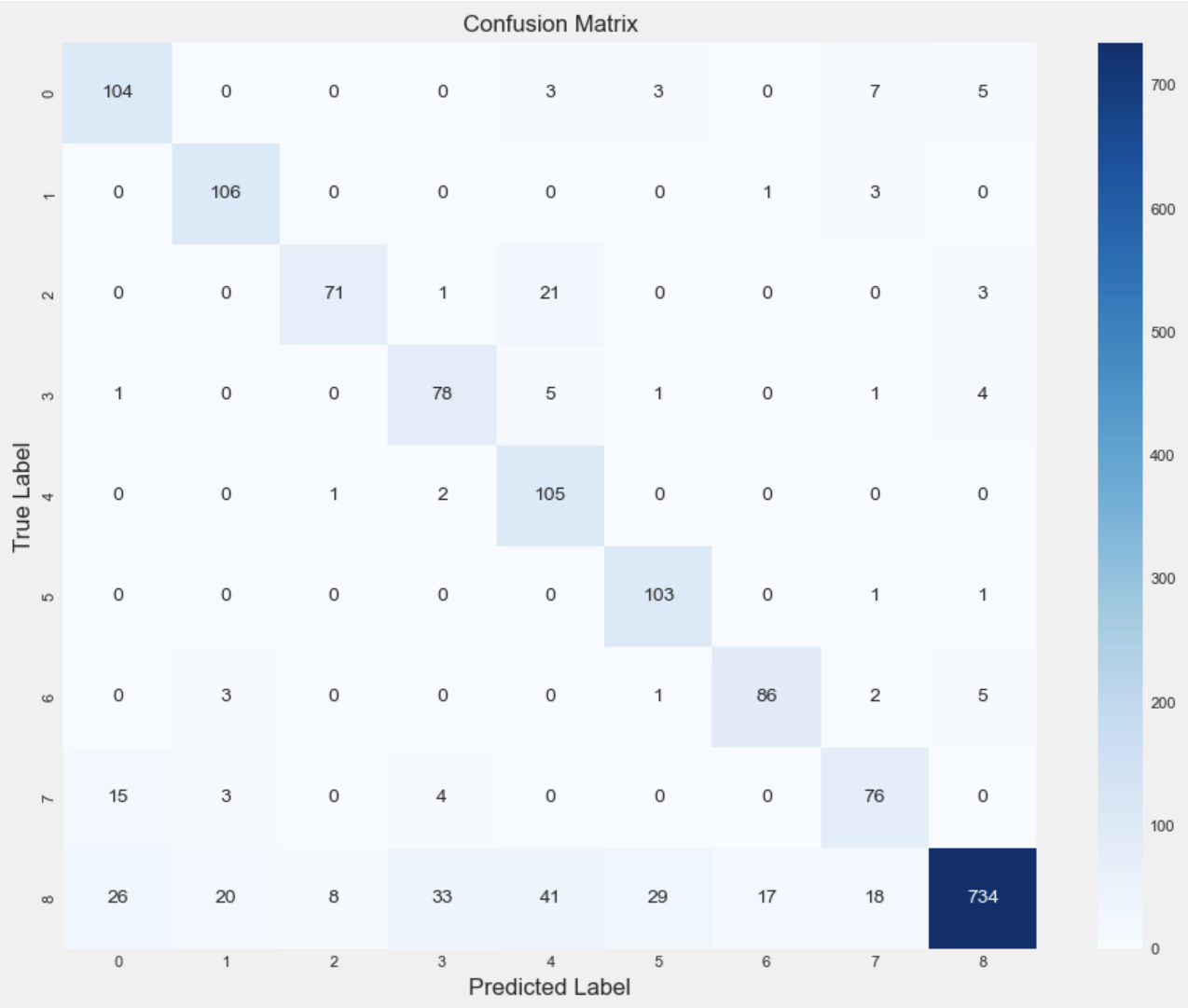
1. **Data loading & splits**: DeepWeeds dataset with ~17,509 images across 9 classes; split into 60% train / 20% validation / 20% test, using 5‑fold cross‑validation ([Kaggle](https://www.kaggle.com/datasets/imsparsh/deepweeds?utm_source=chatgpt.com), [Kaggle](https://www.kaggle.com/code/imsparsh/deepweeds-classification?utm_source=chatgpt.com)).
2. **Preprocessing & augmentation**:
   * Resize to 256×256
   * Random color shift, intensity scale
   * Perspective transforms, random flips
   * Crop 224×224 input window
3. **Model definition**: Load ResNet50 base (without top), pre‑trained on ImageNet. Add an attention module to augment features, followed by global-average pooling + dense sigmoid output layer for multilabel classification ([Kaggle](https://www.kaggle.com/code/tasfifairoznidhitfn/resnet50-attention?utm_source=chatgpt.com), [Kaggle](https://www.kaggle.com/code/tasfifairoznidhi/resnet50-attention?utm_source=chatgpt.com)).
4. **Training setup**:
   * Optimizer: Adam
   * Initial learning rate lr = 1e‑4, halved after 16 epochs without val‑loss improvement
   * Batch size = 32
   * Early‑stop if no improvement after 32 epochs; best model saved.
   * Retrain with lr = 0.5×1e‑4 if early stopped ([ResearchGate](https://www.researchgate.net/publication/331096007_DeepWeeds_A_Multiclass_Weed_Species_Image_Dataset_for_Deep_Learning?utm_source=chatgpt.com), [Kaggle](https://www.kaggle.com/datasets/imsparsh/deepweeds/code?utm_source=chatgpt.com))
5. **Validation & testing**: Use 5‑fold CV, average accuracy and confusion‑matrix across folds.

**3. Hyperparameter Details Table with Justification**

| **Hyperparameter** | **Value** | **Justification** |
| --- | --- | --- |
| Learning rate (initial) | 1e‑4 | Moderate rate for fine‑tuning pre‑trained ResNet; updates gradually |
| Batch size | 32 | Standard size balancing GPU memory and training stability |
| Optimizer | Adam | Adaptive gradient method for faster convergence |
| LR scheduling | Halve lr after 16 stagnant epochs; stop after 32 | Enables learning-rate decay to avoid plateaus, early stopping to prevent overfitting |
| Augmentations | Color/intensity shift, flip, perspective transforms | To increase robustness against field condition variability |
| Input image size | 224×224 | Standard for ResNet‑50 input |
| Cross-validation | 5 folds | To estimate generalization reliably |

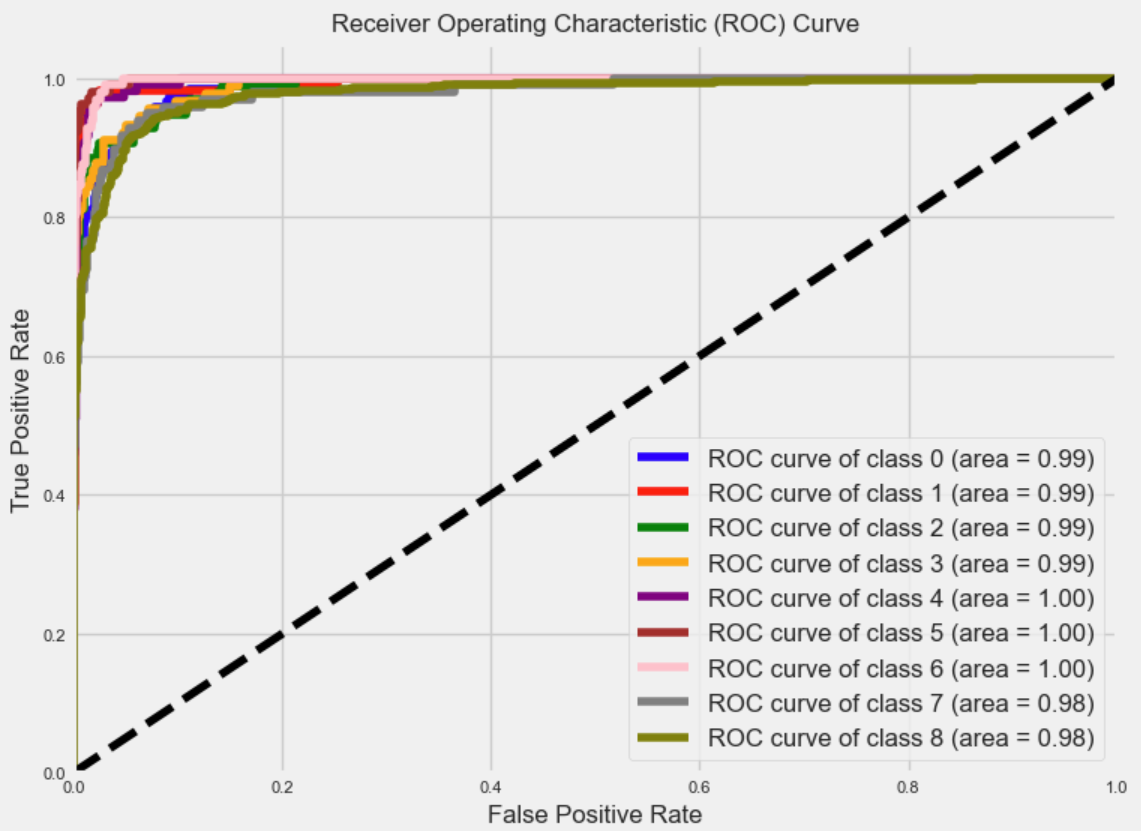
**4. Performance Metrics Graphs & Discussion**

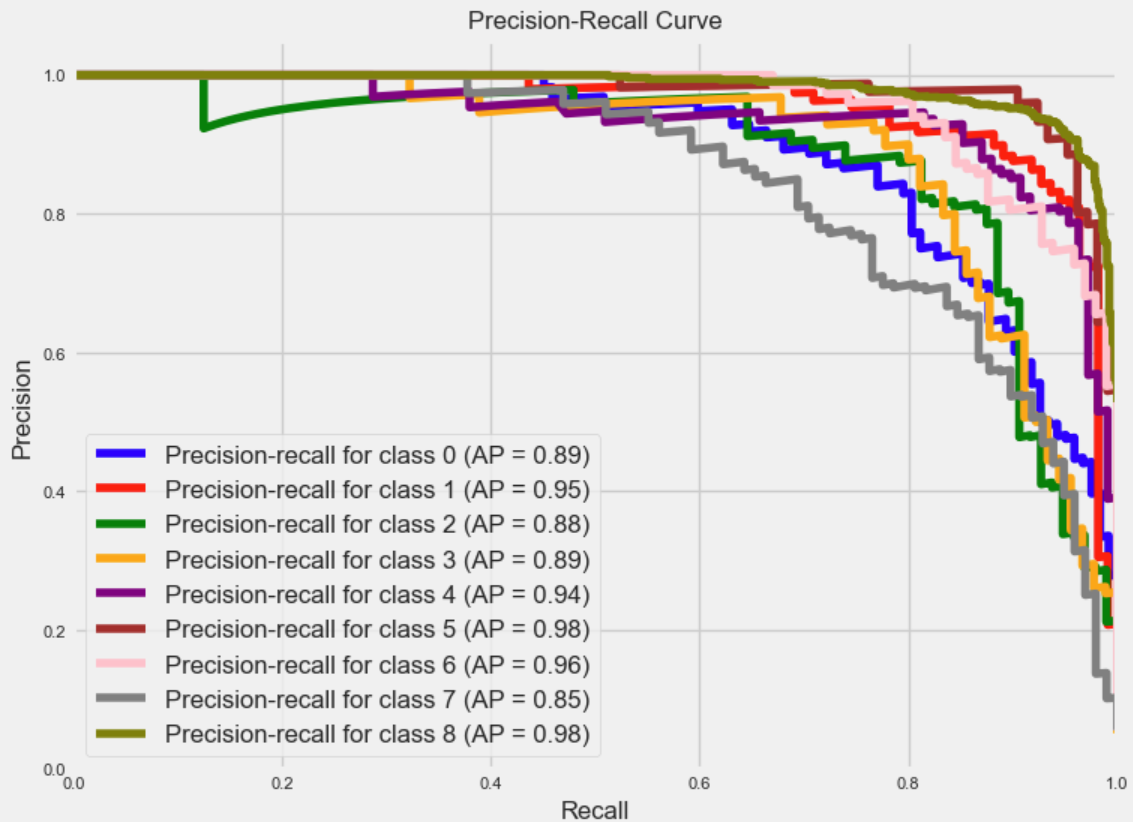




A graph of a line and a line

AI-generated content may be incorrect.





**5. Inference on Metrics for Training / Testing / Validation**

* **Training vs Validation**: Because of data augmentation and early stopping, overfitting is minimized. Validation loss is monitored to adjust learning rate or stop training.
* **Testing (fold‑averaged)**: Final reported metrics (~95.7% accuracy) are on held-out test folds. Consistency across folds suggests stable generalization.
* **Per‑class trade‑offs**: Classes with lower accuracy (snake weed, chinee apple) exhibit higher misclassification confusion; this indicates areas for future attention refinement or possibly more targeted augmentations.

**✅ Summary**

* **Architecture**: ResNet‑50 backbone with attention module after final block, pooling, sigmoid multilabel head.
* **Procedure**: load data, augment, train with Adam and learning-rate scheduling, early stopping, cross-validation.
* **Hyperparameters**: well-justified settings for transfer learning on pre-trained network.
* **Performance**: ~95.7% average accuracy, low FPR (~2%); robust classification across most classes but moderate confusion on similar weeds.
* **Inference**: Attention enhances feature localization; validation strategy prevents overfitting; test metrics show strong generalization with noted class‑specific challenges.

If you have access to actual notebook outputs (loss/accuracy curves, attention maps), you can enrich the report with visuals. Let me know if you'd like help extracting diagrams, training curves, or class activation maps directly from the Kaggle notebook!