**Approach for "Can You Tell the Difference?" Kaggle Competition**

**1. Understanding the Problem Statement**

* The goal is to classify images as **AI-generated or human-created**.
* Dataset consists of **real and AI-generated images** with a balanced distribution.
* Evaluation metrics likely include **accuracy, F1-score, and fairness**.

**2. Dataset Exploration**

* **Download & Inspect:** Visualize a few samples to understand patterns.
* **Check Labels:** Verify data integrity (no missing values, incorrect labels, or duplicates).
* **Class Distribution:** Ensure balance between real vs. AI-generated images.
* **Metadata Analysis:** Look for unique AI-generation patterns (e.g., artifacts, textures, inconsistencies).

**3. Choosing an Approach**

**(A) Baseline Model**

Start with a **simple CNN model** (e.g., ResNet50, EfficientNet) to establish a benchmark:

* Train with minimal augmentations.
* Use **Cross-Entropy Loss**.
* Track metrics like **accuracy & F1-score**.

**(B) Advanced Model (Custom Approach)**

1. **Experiment with Different Models:**
   * Test **Vision Transformers (ViTs), ConvNeXt, or Swin Transformers** for detailed feature extraction.
   * Consider **CLIP or SAM** models for multimodal learning.
2. **Enhance Data Augmentation:**
   * **Geometric transforms:** Random flips, rotations, and crops.
   * **Color distortions:** Brightness, contrast, hue adjustments.
   * **Feature Augmentations:** MixUp, CutMix.
3. **Fine-Tune a Pretrained Model:**
   * Use **RegNet, ConvNeXt, or Swin Transformer** pretrained on ImageNet.
   * Adjust last layers for fine-tuning.
4. **Ensemble Learning:**
   * Combine multiple models (CNN + Transformer) for robustness.

**4. Training Strategy**

* **Hyperparameter tuning:** Try different **batch sizes (16–64), learning rates (1e-4 to 5e-5), and optimizers (AdamW, RAdam).**
* **Scheduler:** Use **CosineAnnealingLR** or **ReduceLROnPlateau**.
* **Early stopping:** To prevent overfitting.

**5. Evaluation & Testing**

* Use **stratified k-fold cross-validation** to ensure fairness.
* Verify **balanced performance across different image types**.
* Analyze misclassified images to improve model robustness.

**6. Submission Strategy**

* **Thresholding & Post-processing:** Adjust decision thresholds for better F1-score.
* **Ensembling multiple models:** Blend predictions using majority voting or weighted averaging.
* **Check Edge Cases:** Visualize failure cases and refine model accordingly.

**7. Deployment & Final Submission**

* Once satisfied with performance, generate final predictions.
* Ensure the submission format is correct.
* Compare with top solutions for further improvements.

**Next Steps:**

* **First, load and explore the dataset** to understand its structure.
* Implement a **baseline ResNet model** before moving to **RegNet + augmentations**.
* Keep improving based on results from data insights and model evaluations.

┌────────────────┐

│ Input Image │

└──────┬─────────┘

↓

┌────────────────┐

│ Preprocessing │ (Resize, Normalize)

└──────┬─────────┘

↓

┌────────────────┐

│ Encoder │ (Feature Extraction)

└──────┬─────────┘

↓

┌────────────────┐

│ Decoder │ (Reconstruct Image)

└──────┬─────────┘

↓

┌───────────────────────┐

│ ConvNeXt + Swin Trans │ (Extract Deep Features)

└────────┬──────────────┘

↓

┌───────────────────────┐

│ Dual Branch │

│ GELU + ReLU Activ. │ (Different Activation Tests)

└────────┬──────────────┘

↓

┌───────────────────────┐

│ Fully Connected Layers │ (Classification)

└────────┬──────────────┘

↓

┌───────────────────────┐

│ Final Prediction (0/1) │ (Fake or Real)

### ****Table: Step-by-Step Model Workflow****

| **Step** | **Component** | **Purpose** | **Key Notes** |
| --- | --- | --- | --- |
| **1** | Preprocessing | Normalize, resize, augment | Ensures consistency |
| **2** | Encoder A | Basic feature extraction | Standard encoder-decoder |
| **3** | Encoder B (VAE) | Captures deepfake-style artifacts | Latent representation learning |
| **4** | ConvNeXt | Extracts low-level to high-level features | Improves spatial understanding |
| **5** | Swin Transformer | Captures global & local dependencies | Efficient vision transformer |
| **6** | Activation Layers | GELU & ReLU for non-linearity | Helps in decision boundaries |
| **7** | Fully Connected Layer | Binary classification (Fake/Real) | Predicts the final label |
| **8** | Output | AI or Human-generated prediction | Interpretable binary output |

|  |
| --- |
|  |