# Model Architecture:

The model, PasticheModel, is a **feedforward network** that takes an input image and applies a style transformation. It consists of:

1. **Conditional Convolutions (CondConvolution)**
   * Instead of a standard convolution layer, it uses a **style-conditioned convolution**, where the transformation is dynamically adjusted based on a learned gamma (scale) and beta (shift) for each style.
   * nn.Parameter is used to store these **style-specific parameters**.
2. **Instance Normalization**
   * Each convolution layer normalizes feature maps using **InstanceNorm2d** with affine transformations, helping to **remove content information** while keeping style-related statistics.
3. **Residual Blocks (ResBlock)**
   * The model includes **four residual blocks** to preserve content structure while applying stylization.
   * These blocks use **conditional convolutions** as well.
4. **Upsampling Layers**
   * After feature extraction and residual transformation, **two upsampling layers** restore the image resolution.
5. **Final Output Layer**
   * Uses a 9x9 convolution to generate the **styled output image**.
6. **Activation Functions**
   * **ReLU** is used in most layers.
   * The final layer uses a **sigmoid activation** to keep pixel values between **0 and 1**.

# Training Pipeline

1. **Feature Extraction via VGG16**
   * A **pre-trained VGG16 model** extracts **style features** from target images.
   * These features serve as ground truth for training.
2. **Content and Style Loss Calculation**
   * **Content Loss**: Measures the difference between feature maps of the input and generated images (using MSE Loss).
   * **Style Loss**: Compares Gram matrices of the generated and reference styles.
3. **Optimization**
   * Uses **Adam optimizer** with content and style loss balancing factors.
   * **Backpropagation updates the style-specific gamma and beta parameters**, making the model adaptively learn multiple styles.
4. **Model Saving & Evaluation**
   * After training, the model can apply any learned style by specifying its **style index**.

# Inference

During inference, the model requires:

* **An image** to be transformed.
* **A style index** (e.g., style\_no) to condition the transformation.
* The model applies **pre-learned transformations** without additional feature extraction.

# Why Conditional Convolutions?

Unlike traditional style transfer models that require **separate models per style**, this approach:

* **Allows multiple styles in one model** (via gamma and beta).
* **Supports style interpolation** by blending between styles (using alpha and style\_no2).
* **Is more memory-efficient** since it shares weights across styles.