MDPI

Article

Research on the Application of Clothing Product Image Generation Based on Stable Diffusion Model

Lingyun KONG [1]*, Haiyang WU [1]

[1] School of electronic information, Xijing University, Xi'an, Shaanxi, 710123, China.
* Correspondence: 1400100383@qq.com; Tel.: +86-15619076627

Abstract: Image generation is a pivotal research area in computer vision and graphics, with wide-ranging applications, particularly in enhancing product sales and brand promotion through visual impact. This paper focuses on the application of image generation technology for creating clothing product images, addressing the high photography costs faced by individual merchants and small manufacturers. It introduces two novel algorithms for background and character generation. The paper integrates Segment Anything and Grounding DINO for clothing and character image segmentation, utilizing the ViTMatte module for fine-grained mask segmentation to improve accuracy. It also combines YOLOv8, Depth-fm, Canny edge detection, and normal mapping algorithms with the ControlNet model to guide the Stable Diffusion model's generation process. By synthesizing these algorithms, the paper proposes new methods for background and character generation and restoration, achieving automated creation of clothing product images. This advancement is crucial for the commercialization of generative models and the development of the clothing industry.

Keywords: Stable Diffusion, ControlNet, LoRA, Clothing Product Images, Character Generation Task, Background Generation Task.

1. Introduction

Facing the dual pressures of a contracting domestic consumer market and diminishing user traffic dividends, exploring lower-tier markets offers new development opportunities for e-commerce platforms. The e-commerce industry is now focusing on cost advantages and service excellence. With over 60% of China's population residing in lower-tier markets, their evolving social structure and e-commerce infrastructure promise significant growth potential.

As domestic consumption levels decline, the barriers to becoming an e-commerce seller have significantly lowered. One can start with just a supplier and customer service. Additionally, affordable domestic clothing performs exceptionally well in Southeast Asia and Africa due to its low prices and good quality, relying on high sales volumes for profitability. Quick shipping and trend adaptation are crucial for market capture, with some sellers in China going from design to stocking in just one week. High-quality main product images are essential to highlight product features, increase click-through rates, and drive conversions.

Affordable clothing sellers typically hire professional models and photographers, rent studios, or use simple setups with backdrops, though high-quality images come with significant costs. Major e-commerce platforms require detailed product images for algorithm recommendations and exposure. While detail shots are less expensive, high-quality studio shots are costly, posing challenges for sellers with low profit margins.

To address this, the paper proposes a background and character generation algorithm based on Stable Diffusion, allowing e-commerce sellers to quickly create platform-compliant main product images at low cost. This advancement is crucial for the commercialization of generative models and the development of the clothing industry.

Citation: . Title. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

2. Related work

2.1. Diffusion Models and Their Evolution

The Denoising Diffusion Probabilistic Models (DDPM)[1], proposed in 2020 by Jonathan Ho and his team from the University of California, Berkeley, represents a novel type of generative model. In contrast to traditional Generative Adversarial Networks (GANs)[2] and flow-based models, diffusion models have rapidly gained prominence in the field of computer vision. This model has significantly elevated the standards in generative modeling.

Denoising Diffusion Probabilistic Models primarily involve a forward noising process and a reverse denoising process. In the forward process, the model gradually adds Gaussian noise to transform a real image into a disordered Gaussian noise. In the reverse process, the model progressively denoises the Gaussian noise to restore the original image. The core of training a denoising diffusion model lies in training a noise predictor.

During training, the noise predictor takes an image that has not been fully denoised and the current denoising step as inputs and predicts the noise for the next step. Subsequently, by subtracting the predicted noise from the current image, the next step's denoised image is generated. In the noising process, the model randomly generates Gaussian noise and adds it to the image, resulting in the next noisy image. By using the next noisy image and the current step as inputs, and the noise generated at the current step as outputs, a neural network is trained to predict the noise. The training process of the diffusion model involves training a neural network to accurately predict the noise, while the inference process entails progressively removing all noise through reverse inference to generate high-quality images that approximate the real image. Each iteration brings the generated image distribution closer to that of the real image.

During training, the random noise added at each step also serves as the ground truth data required for training. Consequently, only real images are needed to create the training dataset, eliminating the need to generate additional noisy images. The forward and reverse processes of the diffusion model are illustrated in Figure 1.

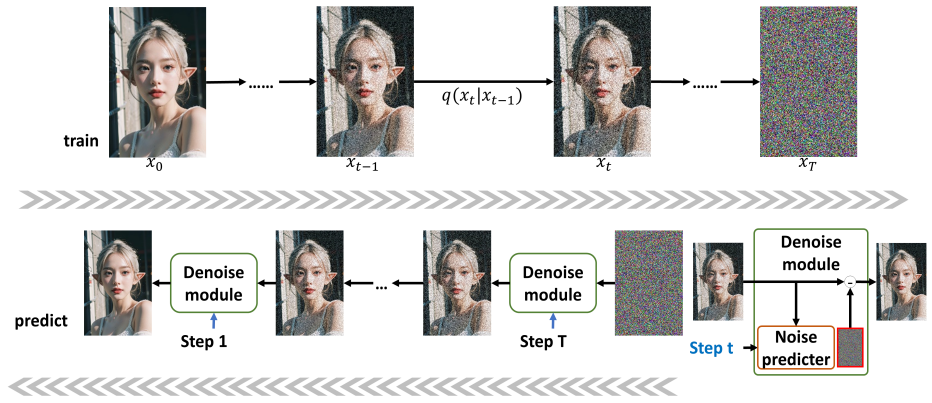


Figure 1. Diagram of the Forward and Backward Processes of the Diffusion Model.

In Figure 1, the diffusion model is divided into the training process and the inference process, which can also be understood as the forward process and the reverse process. In the forward process, Gaussian noise is progressively added to the input real image x_0, turning the real image into complete Gaussian noise after T steps. Assuming the initial distribution of the real image is $q(x_0)$, a real image can be randomly sampled from the training set and denoted as $x \sim q(x_0)$. In the forward process, the result from the previous noising step x_{t-1} is subjected to additional Gaussian noise to obtain the result for the next step x_t. The standard deviation of the Gaussian noise is determined by a fixed value β_t, while the mean is determined by the fixed value β_t and the current state x_t at time t. Thus, the mean μ_t and variance Σ_t of the Gaussian noise added at each step are given by

$\sqrt{1-\beta_t} \cdot x_{t-1}$ and $\beta_t I$, respectively. Therefore, the conditional distribution of the noisy image x_t given x_{t-1} is:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t} \cdot x_{t-1}, \beta_t I)$$

According to[3], the objective of the image generation model is to train a neural network that can take arbitrary Gaussian noise as input and output images. By inputting a large amount of Gaussian noise, the distribution of generated images should approach the distribution of real images. Let the parameters of this neural network be θ, the distribution of generated images by this network be $P_\theta(x)$, and the distribution of real images be $P_{\text{data}}(x)$. Suppose m images are sampled from real images to form a dataset, and $P_\theta(x^i)$ represents the probability of generating image x^i. The training objective of the network is to find a set of parameters that maximizes the probability of generating images x^i in the generated image distribution, i.e.,

$$\theta^* = \arg\min \mathrm{KL}(P_{\text{data}} \| P_\theta)$$

This analysis indicates that the desired parameters θ^* maximize the probability of any real image appearing in the generated image distribution, which corresponds to minimizing the KL divergence between P_{data} and P_θ.

Assuming that each denoising step generates new Gaussian noise, when the KL divergence between the denoised output noise and the predicted noise is minimized, the value of P_θ is maximized, i.e.,

$$P_\theta(x_{t-1}|x_t) \propto \exp\left(-\|G(x_t) - x_{t-1}\|_2^2\right)$$

Therefore, the probability of the model generating any real image can be expressed as:

$$\log P_\theta(x_0) = \mathbb{E}_{q(x_1:x_T|x_0)}\left[\log \frac{P(x_0 : x_T)}{q(x_1 : x_T|x_0)}\right]$$

To maximize $\log P_\theta(x_0)$, we need to maximize the expectation in the expression above. Since $q(x_1 : x_T|x_0)$ represents the forward process of the diffusion model, the values of $q(x_t|x_{t-1})$ are known. Before each noise addition, the model samples a vector from a two-dimensional standard normal distribution $\epsilon_{1:T} \sim \mathcal{N}(0, I)$, and with the parameter β_t and x_{t-1}, we can compute:

$$x_t = \sqrt{1-\beta_t} \cdot x_{t-1} + \sqrt{\beta_t} \cdot \epsilon_{t-1}$$

In the diffusion model, defining α_t as $1 - \beta_t$, this simplifies to:

$$x_t = \sqrt{\alpha_t \alpha_{t-1}} \cdot x_{t-2} + \sqrt{1-\alpha_t \alpha_{t-1}} \cdot \epsilon_{t-2}$$

From this analysis, we see that the noise distribution at any step in the noising process can be directly predicted from x_0. Using the isomorphic formula, we get:

$$x_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1-\bar{\alpha}_t} \cdot \epsilon_t$$

By calculating the obtained expression for x_t, we can derive the values for $q(x_1 : x_T|x_0)$ and $P(x_0 : x_T)$. Substituting these into the above equation, the final objective function of the diffusion model [4] is:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \cdot \epsilon_\theta(x_t, t)\right) + \sigma_t z$$

where $\sigma_t z$ represents random noise. The DDPM paper mentions that without random noise, the output would be repetitive and lack diversity. In diffusion models, images

generated without additional noise may turn into plain colors or incomprehensible images, which are significantly different from the real image distribution. Hence, the paper includes random noise at the end to ensure diversity and quality in generated images.

The primary model used in this paper is the Latent Diffusion Model (LDM)[3], which is an improvement upon the Denoising Diffusion Probabilistic Models. The main contribution of this model is the introduction of diffusion learning in the latent space to model feature distributions, rather than performing diffusion learning directly in the image space. Additionally, the model incorporates external conditions through the use of self-attention mechanisms[5] to achieve conditional generation. The advent of latent space diffusion models has made it possible to control the generation process. For example, with text prompts, the input text is encoded into a format that can be processed by self-attention mechanisms, thereby allowing control over the generated images, as illustrated in Figure 2.

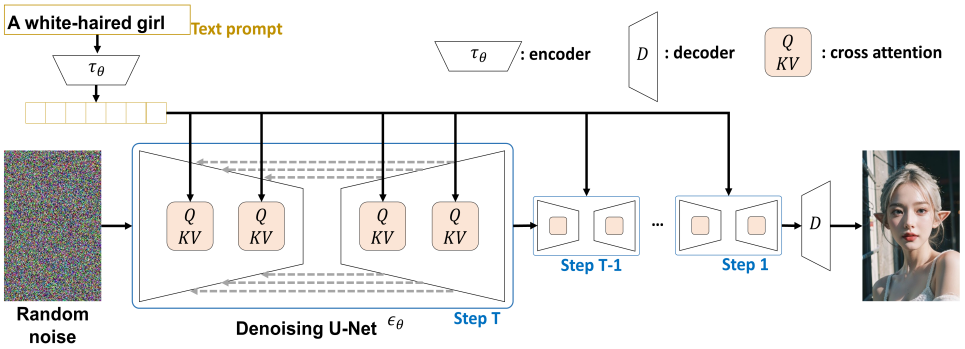


Figure 2. Schematic Diagram of Training the Denoising Diffusion Probabilistic Model.

The training process of the Latent Diffusion Model (LDM) involves two main steps. First, a Variational Autoencoder (VAE)[6] is trained to encode an input image x into an intermediate latent variable Z_T and to reconstruct the output image from this latent variable. During this phase, the encoder of the VAE maps the input data x into the latent space Z_T, while the decoder converts the latent variable Z_T back into image data. This step ensures that the model can effectively represent the input data within the latent space.

After completing the training of the VAE, the next step is to train the latent space diffusion model. This model aims to learn the generative process from Gaussian noise to the intermediate latent variable Z_T. The generative process is structurally similar to that of Denoising Diffusion Probabilistic Models (DDPM) . However, unlike traditional DDPM models, the latent space diffusion model incorporates encoder-decoder structures both before and after the diffusion process, enabling the model to accept a broader range of inputs beyond just Gaussian noise.

In the latent space diffusion model, the architecture of the encoder-decoder is largely inspired by Transformer and Vision Transformer (ViT) designs. Due to the structural characteristics of the attention mechanism, the processed feature vectors have the same dimensions regardless of whether the input data is text or images. Thus, a single model can be trained on different types of data.

As shown in Figure 2, the denoising process of the diffusion model involves controlling through self-attention modules. Therefore, appropriate input needs to be encoded into feature vectors and fed into the attention modules. Building on this, Min Zhang from Stanford University proposed the ControlNet model[7], which enables deep control over the diffusion model.

2.2. Contributions of ControlNet

With the advent of text-to-image generation models, it is now possible to create visually compelling images from brief descriptive prompts [8]. However, this raises three key issues:

- Can text-based prompts meet practical production requirements?

- In image processing tasks, many long-standing tasks have well-defined solutions. Can these large models reference and advance these specific tasks? What framework should be constructed to unify various problems and user controls?
- In specific tasks, can large models retain the advantages and capabilities gained from datasets containing billions of images?

To address these questions, Min Zhang, the author of ControlNet, investigated various image processing applications and made the following three findings:

- The available dataset sizes for specific task domains are often limited, with many datasets for specific tasks (e.g., object shapes, pose understanding) having fewer than 100,000 samples. This necessitates robust neural network training methods to avoid overfitting and maintain generalization when large models are trained for specific problems.
- Large model computing clusters are not always available, making it crucial to optimize large models for specific tasks within acceptable time and memory constraints (e.g., on personal devices). This requires leveraging pre-trained weights, as well as fine-tuning strategies or transfer learning [9].
- Various image processing problems come with different forms of problem definitions, user controls, or image annotations. While image diffusion algorithms can manually adjust parameters, such as constraining the denoising process or editing multi-head attention activations, certain tasks like depth-to-image conversion or pose estimation require interpreting the raw input image as object-level or scene-level understanding. Achieving a general learning solution across many tasks necessitates end-to-end learning.

To address these challenges, the author proposed an end-to-end neural network: the ControlNet model. ControlNet involves cloning the weights of large diffusion models into a “trainable copy” and a “locked copy.” The locked copy retains the network capabilities learned from billions of images, while the trainable copy is fine-tuned on specific task datasets to learn conditional control.

Trainable and locked neural network blocks are connected through a unique type of convolutional layer known as "zero convolution", where convolutional weights gradually evolve from zero to optimized parameters in a learnable manner. This approach retains the weights, thus providing robustness across datasets of varying scales. Since zero convolution does not add new noise to the deep features, the training speed is comparable to fine-tuning a diffusion model, and is as fast as training from scratch.

We trained multiple ControlNet models under various conditions, including datasets with Canny edge detection, Hough transforms, user scribbles, human keypoint detection, segmentation maps, and depth maps. Experiments were conducted on datasets with both small sample sizes (less than 50k or even 1k) and large sample sizes (up to millions of samples). The results indicated that for certain tasks, such as image generation from depth maps, ControlNet models trained on personal computers (equipped with an Nvidia RTX 3090TI) achieved performance nearly indistinguishable from that of commercial models trained on large computing clusters.

ControlNet employs a specialized convolutional layer known as "zero convolution." Early neural network research extensively discussed the issue of network weight initialization, including the rationale behind Gaussian distribution initialization and the potential risks associated with zero initialization of weights. Relevant literature such as ProGAN[10], StyleGAN[11], and Noise2Noise[12] addresses the operations involved in initial convolutional weights.

ControlNet further regulates the behavior of the entire network by manipulating input conditions of neural network modules, such as "ResNet" blocks[13], "Conv-bn-relu" blocks, multi-head attention blocks, and Transformer modules. For instance, consider a two-dimensional feature with $x \in h \times w \times c$, where {h, w, c} represent height, width, and channels, respectively. Let the neural network module be $F(\cdot;\Theta)$, where Θ denotes the network parameters. If all parameters are locked to Θ and cloned to a trainable copy Θ_c,

training the cloned Θ_c with an external condition vector c yields the original parameters Θ as the "locked copy" and the new parameters Θ_c as the "trainable copy". The motivation for this copy mechanism is to avoid overfitting on smaller datasets while retaining the capabilities learned from large-scale datasets.

Neural network modules are connected via a unique convolutional layer called "zero convolution", specifically a 1×1 convolutional layer where both the weights and biases are initialized to zero. Let the zero convolution be denoted as $Z(\cdot;\cdot)$. For a ControlNet example containing two modules, the input feature is x and the output feature is y_c, which can be expressed as:

$$y_c = F(x;\Theta) + Z(F(x + Z(c;\Theta_{z1});\Theta_c);\Theta_{z2})$$

Since the weights and biases of the zero convolution layer are initialized to zero, during the first training step, we have:

$$\begin{aligned} &Z(c;\Theta_{z1}) = 0, \\ &F(x + Z(c;\Theta_{z1});\Theta_c) = F(x;\Theta_c) = F(x;\Theta), \\ &Z(F(x + Z(c;\Theta_{z1});\Theta_c);\Theta_{z2}) = Z(F(x;\Theta_c);\Theta_{z2}) = 0. \end{aligned}$$

The above equations demonstrate that, in the initial training step, the parameters of the trainable copy and the locked copy of the neural network module are identical, meaning that before optimization, ControlNet does not alter the deep neural network. It is only during subsequent training steps that the parameters are optimized. Since the trainable copy has fewer parameters compared to the original parameters, the training process is more efficient.

Stable Diffusion employs a preprocessing method similar to VQ-GAN (Vector Quantized-Consistent Adversarial Networks)[14], converting input images of size 512×512 into smaller latent space noise of size 64×64 for training. To match the latent space noise dimensions of Stable Diffusion, ControlNet needs to convert image-based conditional prompts into latent space features of size 64×64. ControlNet utilizes a miniature network composed of four 4×4 convolutional kernels with a 2×2 stride to encode image-based conditional prompt features c_i into feature maps $c_f = E(c_i)$, where c_f represents the transformed feature map. The network architecture of ControlNet is illustrated in Figure 3.

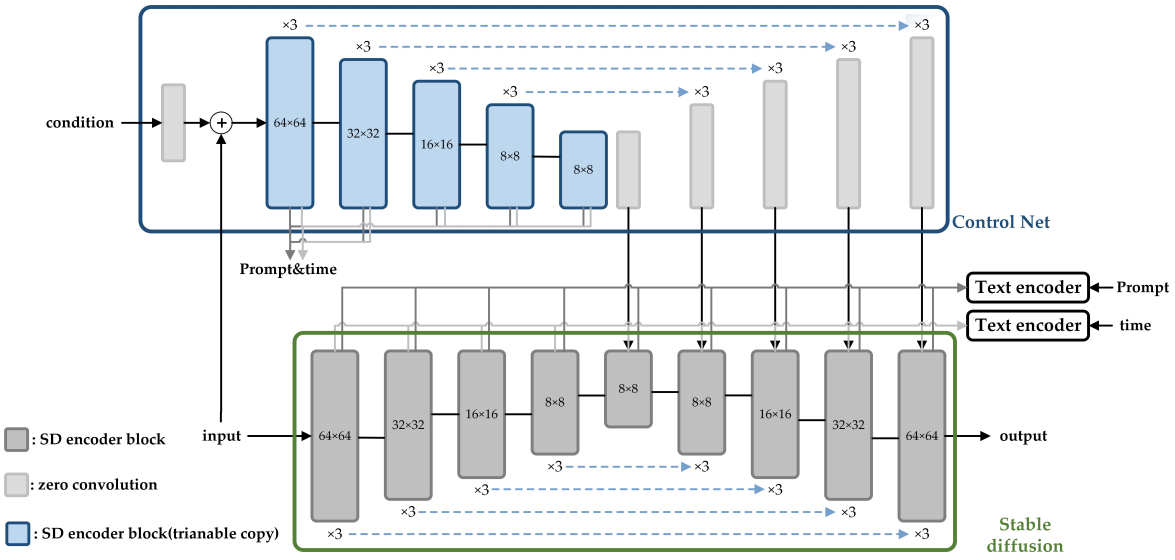


Figure 3. Network Architecture of ControlNet.

The computation method of ControlNet is efficient. Since the original weights are locked, gradient computation on the original encoder is not required during training. This

speeds up the training process and saves GPU memory, effectively avoiding the need for gradient computation that would otherwise account for half of the original model's computations.

3. Preprocessing Methods

In this study, to accomplish background generation and character generation tasks, we primarily use image-to-image generation from diffusion models. In the image-to-image generation task, uploading an image and providing a prompt primarily achieves style transfer. However, the tasks in this study include areas in the original image that cannot be modified. For example, in background generation tasks, the foreground characters must remain unchanged, and in character generation tasks, the clothing must remain unchanged. Therefore, this study employs masking and inpainting techniques from image-to-image generation, which involve uploading the original image and a mask, and then regenerating the areas outside the mask while keeping the masked areas unchanged.

Masks can be obtained in two ways. One method is manual mask drawing, which can be achieved through the UI interface of Stable Diffusion. However, manual mask drawing has significant limitations: edges are often not clear, and the drawing may be inaccurate. Additionally, each image requires manual drawing, which is inefficient, labor-intensive, and difficult to scale for batch processing. The other method is to use image segmentation models to automatically obtain masks. However, most existing image segmentation models cannot segment all objects, especially when dealing with product images, as the variety of products and complexity of segmentation areas make it challenging for standard segmentation models. Therefore, this study proposes an image segmentation task that can segment arbitrary objects and regions.

After obtaining the masks, the ControlNet model is used to control the generation process, including image depth, character pose, and background content. The authors of ControlNet, such as Zhang Lümin, have published various control network models on Hugging Face, including edge control, pose control, and depth control models. It is important to note that different control network models require specific input images. For instance, edge detection maps are used for edge control, and pose estimation maps are used for pose control. These images are not generated automatically and require additional acquisition. Therefore, this study proposes a preprocessing model task capable of generating edge detection maps, pose estimation maps, and depth maps as inputs for ControlNet.

In character generation tasks, where there is a need to generate detailed areas such as faces and hands, Stable Diffusion may encounter issues such as extra fingers, missing fingers, or distorted faces during the inpainting of these areas. Therefore, a repair method for hands and faces is necessary. This study proposes an image repair method that includes detecting head and hand regions, segmenting the skin areas of these regions, regenerating them, and pasting them back into the original image. Hence, a model is required to accomplish the tasks of detecting heads and hands as well as segmenting skin regions.

In summary, the three preprocessing tasks to be implemented in this chapter are:

- An image segmentation task capable of segmenting arbitrary objects and regions.
- A preprocessing model task capable of generating edge detection maps, pose estimation maps, and depth maps as inputs for ControlNet.
- Detection of heads and hands, as well as segmentation of skin regions.

3.1. Experimental Environment

The experimental environment is configured as follows: the operating system is Ubuntu 22.04, the deep learning framework used is PyTorch, and the hardware includes an NVIDIA A40 GPU (with 40GB of VRAM), a Xeon(R) Platinum 8358P processor, and 80GB of memory. The CUDA version is 11.8, and the Python version is 3.10.8.

3.2. General Object and Region Segmentation Methods

Grounding DINO[15] is an open-vocabulary object detection approach that combines the Transformer-based detector DINO[16] with grounding pretraining techniques. The key aspect of open-vocabulary detection is incorporating natural language into closed-set detectors to enable the detection of novel categories and specific attribute targets. On the COCO[17] dataset, Grounding DINO achieved an average precision (AP) of 52.5% in zero-shot detection and 63% AP after fine-tuning.

Most open-vocabulary detectors extend closed-set detectors to open-set scenarios with language information. A typical closed-set detector usually consists of three main modules: a backbone network for feature extraction, a neck network for feature enhancement, and a head network for region refinement. By learning language-aware region embeddings, closed-set detectors can be adapted for detecting new objects, allowing each region to be categorized into new classes within the language-aware semantic space. The key to this adaptation is aligning the language features and region outputs of the neck using contrastive loss. CLIP[18] performs early fusion in the neck module, while OV-DETR uses language-aware queries as inputs to the head. Grounding DINO opts for more feature fusion throughout the pipeline to improve model performance. The proposed location-based DINO model extends DINO to open-vocabulary object detection, enabling it to detect any object based on given text queries.

Segment Anything[19] is an image segmentation model introduced by the Meta AI research team in 2023, designed to achieve strong generalization across various datasets through pretraining. This model can perform zero-shot and few-shot learning[20] via prompting techniques. The authors proposed a promptable segmentation task, aimed at returning effective segmentation masks given any segmentation prompt.

The promptable segmentation task imposes constraints on the model architecture as it needs to support flexible prompt inputs. The authors presented a simplified design to meet these requirements: a powerful image encoder for computing image embeddings, a prompt encoder for embedding prompts, and a lightweight mask decoder that combines the two information sources to predict segmentation masks. By dividing the SAM model into an image encoder and a rapid prompt encoder, the model can reuse the same image embeddings with different prompts. To ensure robust generalization to new data distributions, the model was trained on a large-scale and diverse dataset exceeding any existing segmentation datasets.

However, the Segment Anything model, as a zero-shot segmentation approach, also has certain limitations. This model can accept three types of input prompts: points, boxes, and text. When using point or box annotations for image segmentation, batch processing becomes challenging. To achieve batch segmentation, text prompts are required. However, the CLIP encoder used by Segment Anything for text input does not achieve precise segmentation through sentence inputs. Therefore, this paper recommends using box annotations as input for Segment Anything. By integrating Segment Anything with Grounding DINO, text prompts and images to be segmented are input into Grounding DINO to obtain bounding box coordinates. These coordinates are then used as input for Segment Anything to perform fine-grained image segmentation tasks.

Grounding DINO sometimes generates multiple outputs, which can complicate batch segmentation tasks. Thus, after obtaining the output from Grounding DINO, Non-Maximum Suppression (NMS)[21] is necessary to remove lower probability and smaller area bounding boxes of the same class. Given that the output bounding box coordinates are in the form of the bottom-left corner (x_0, y_0) and the top-right corner (x_1, y_1), the NMS formula is:

$$\sigma = \max_{k \in (1,n)} \left| (x_1^k - x_0^k)(y_1^k - y_0^k) \right|$$

where n represents the total number of bounding boxes, and σ is the final bounding box coordinate output.

When the input image size is relatively small, the output mask may exhibit unclear edges and poor segmentation quality, often resulting in excessive noise. Therefore, after obtaining the mask, it is necessary to detect all connected components in the mask and filter out those with an area smaller than 50, treating these as noise. Let M be the input mask matrix, where $M(i,j) \in \{0,1\}$, and C_k denotes a connected component with size $|C_k|$. The denoised mask matrix M' can be defined as:

$$M'(i,j) = \begin{cases} 0 & \text{if } (i,j) \in C_k \text{ and } |C_k| < 50 \\ M(i,j) & \text{otherwise} \end{cases}$$

In the field of image segmentation, there are two primary refinement methods: Trimap-based[22] and Trimap-free[23]. Examination of the Segment Anything model revealed that it does not use a Trimap[24] for segmentation refinement. Consequently, this paper decides to use ViTmatte[25] to refine the mask.

After obtaining the coarse segmentation mask from Segment Anything, it undergoes dilation and erosion. In the dilated mask, white pixels (value 255) are converted to gray (value 128). Pixels in the new mask are then categorized: values greater than 128 are set to 200, less than 128 to 0, and greater than 200 to 0. Pixels marked as 200 are set to 255, labeling unknown regions where values less than 128 indicate background, values greater than 200 indicate foreground, and values equal to 200 indicate unknown regions. The original mask is converted to a numpy array, denoised with low-pass filtering, and combined with the unknown regions to produce the final Trimap.

The Trimap and original image are input into ViTMatte, refining segmentation in conjunction with the Segment Anything model to achieve Trimap-based image segmentation. This paper proposes an enhanced image segmentation method using an improved Segment Anything model, achieving zero-shot fine-grained segmentation of any object through text prompts. The detailed structure of this method is shown in Figure 4.

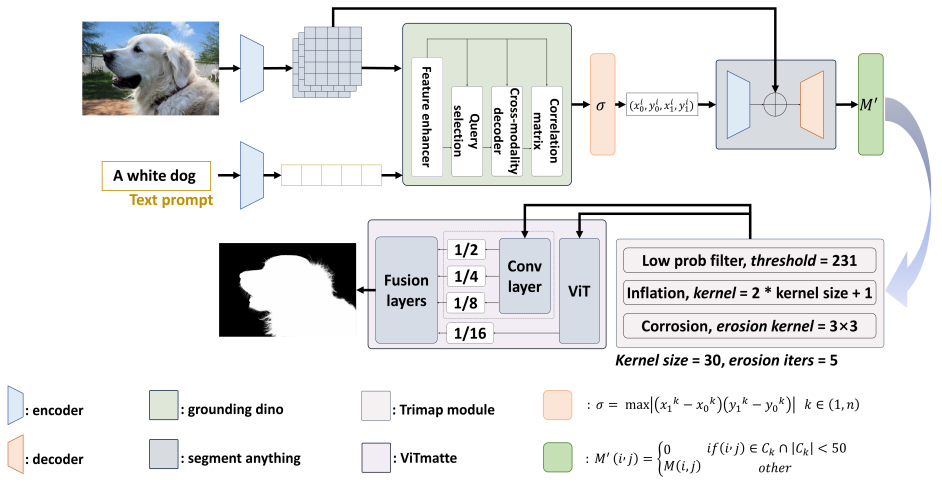


Figure 4. Schematic diagram of the universal object and region segmentation method.

3.3. Generation and Preprocessing of Input Images for Control Net

3.3.1. Edge detection task

The Canny edge detector[26] is a widely used method for edge detection, designed to transform the edge detection problem into one of detecting local maxima in the unit step function. Based on the Gaussian model and the concept of image filtering, the Canny operator introduces three key criteria for evaluating edge detection performance:

- Low Error Rate
  Minimizes the omission of true edges and reduces the incidence of non-edges being incorrectly classified as edges.
- High Localization Accuracy

The detected edges should be precisely aligned with the actual boundaries.

- Single-Pixel Edge
  Each edge should produce a unique response, resulting in a boundary of single-pixel width.

To meet these criteria, the Canny operator defines three principles: signal-to-noise ratio, localization accuracy, and single-edge response.

The workflow of the Canny operator consists of the following four steps:

- Gaussian Filtering
  Apply a Gaussian filter to smooth the image and reduce noise. Environmental interference and device noise, such as salt-and-pepper and Gaussian noise, are common during image acquisition. Gaussian filtering slightly blurs the image, aiding in the clearer detection of subtle yet significant edges.
- Sobel Operator
  Use the Sobel operator[27] to detect the magnitude and direction of the gradient in the filtered image. The Sobel operator is effective in extracting edge information from the image.
- Non-Maximum Suppression
  Refine the detected edge pixels to form precise boundaries by examining small regions in the gradient magnitude map. Compare the central pixel with its neighboring pixels along the gradient direction. If the central pixel's value is not greater than its neighbors, set it to zero; otherwise, retain it as a local maximum.
- Hysteresis Thresholding
  Determine the final edge points using hysteresis thresholding. Select high and low thresholds. Mark gradient values above the high threshold as edges and those below the low threshold as non-edges. Points between thresholds are edges only if connected to high-threshold points; otherwise, they are non-edges.

When processing building edges, to ensure that the overall structure of the buildings in the generated image remains intact, line detection techniques are employed. The Hough Transform is an effective method for line detection, converting values from the Cartesian coordinate system to Hough space[28] using polar coordinates. In Hough space, if multiple pixels in the Cartesian coordinate system lie on the same line, the curves corresponding to these points will intersect at a single point in Hough space. An accumulator is used to count the number of intersections at each point in Hough space, and those intersections occurring more frequently than a certain threshold are selected. These points are then converted back to Cartesian space to determine the desired lines. The results of edge detection and line detection are illustrated in Figure 5.

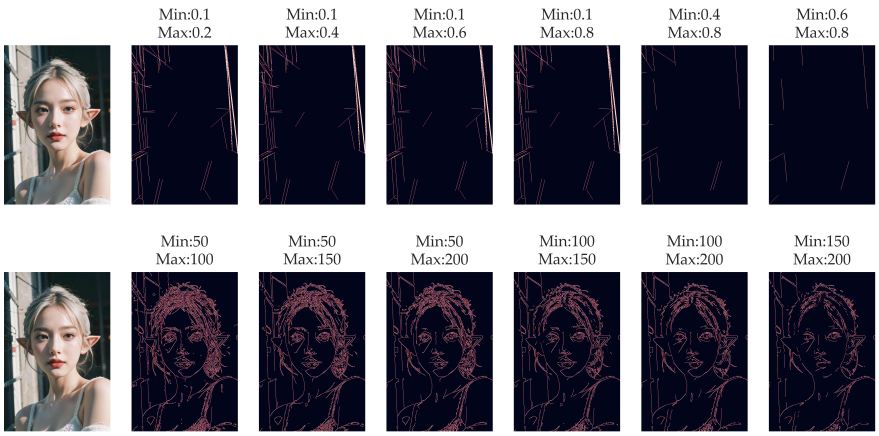


Figure 5. Results of Canny Edge Detection and Line Detection at Different Thresholds.

3.3.2. Pose Estimation Task

In the task of generating human figures, it is essential to use pose estimation with ControlNet to manage the pose as well as the details of the hands and face. This requires generating pose estimation maps. To reduce inference time and ensure accuracy, this paper opts to use YOLOv8[29] for pose estimation. To meet the demands for rapid inference while maintaining fundamental accuracy, the model has been parameter-optimized.

Specifically, to reduce the model's parameter count, this paper replaces the backbone network of YOLOv8 with an improved Vanilla Net[30]. This enhancement optimizes the model structure by replacing the original convolutional modules with the more efficient ODConv[31] modules, and substituting the C2f[32] modules with the parameter-efficient VanillaNet modules, without altering the input and output dimensions of each module. The revised network structure is illustrated in Figure 6.

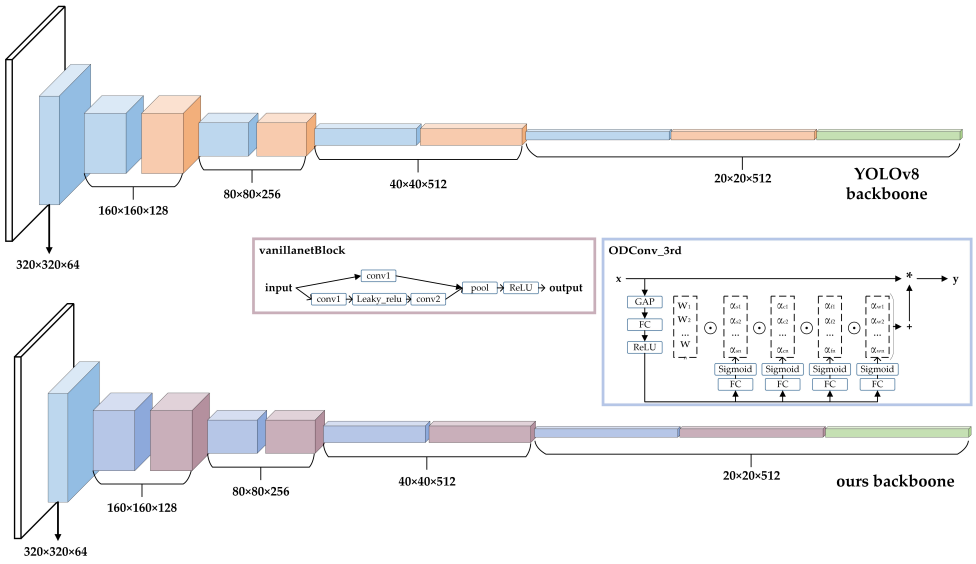


Figure 6. Schematic Diagram of the Improved Backbone Network of YOLOv8.

After training for 100 epochs on the COCO2017 dataset, it was observed that the modified model reduced the parameter count by 620,424 and the GFLOPs by 2.612G, while still achieving a 0.6% increase in accuracy. This ensures both the accuracy and inference speed of the model are maintained, alongside a reduction in model size. The parameter comparison between YOLOv8 and the improved model is shown in Table 1.

Table 1. Parameter Comparison between YOLOv8 and the Improved Model.

model	map@0.5	FLOPs	Parameters	preprocess	inference	postprocess
YOLOV8	0.747	9.2G	3289964	5.9ms	132.9ms	28.8ms
ours	0.785	6.6G	2671652	4.9ms	108.1ms	15.0ms

3.3.3. Depth Map Estimation Task

In background generation tasks, accurately controlling the spatial relationship between the foreground and background in generated images is crucial. To ensure harmonious integration of foreground and background, depth maps are used. For depth map estimation, this paper employs the depth-fm[33] model, which is one of the state-of-the-art monocular depth estimation models. The depth-fm model excels in depth estimation tasks by effectively transferring powerful image prior knowledge from diffusion models to flow matching models, with minimal reliance on training data and real-world images.

It is important to note that while the depth-fm model outputs pseudo-color images (mapped to RGB space) rather than direct depth maps, ControlNet requires single-channel

depth maps as input. Therefore, the output of the model must be converted from pseudo-color images to single-channel depth maps. This conversion process involves transforming the pseudo-color images into a format that meets ControlNet's input requirements.

Furthermore, once the depth map is obtained, it can be used to generate normal maps[34]. Normal maps are a technique for simulating surface lighting effects, essentially implementing bump mapping. Specifically, normal maps are created by calculating the components of each pixel along the X, Y, and Z axes and mapping these components to the RGB color space, resulting in a map that reflects the surface texture of the image. This technique enhances the realism of the lighting effects on the image, better simulating surface details. An example of a depth map generated by depth-fm is shown in Figure 7.

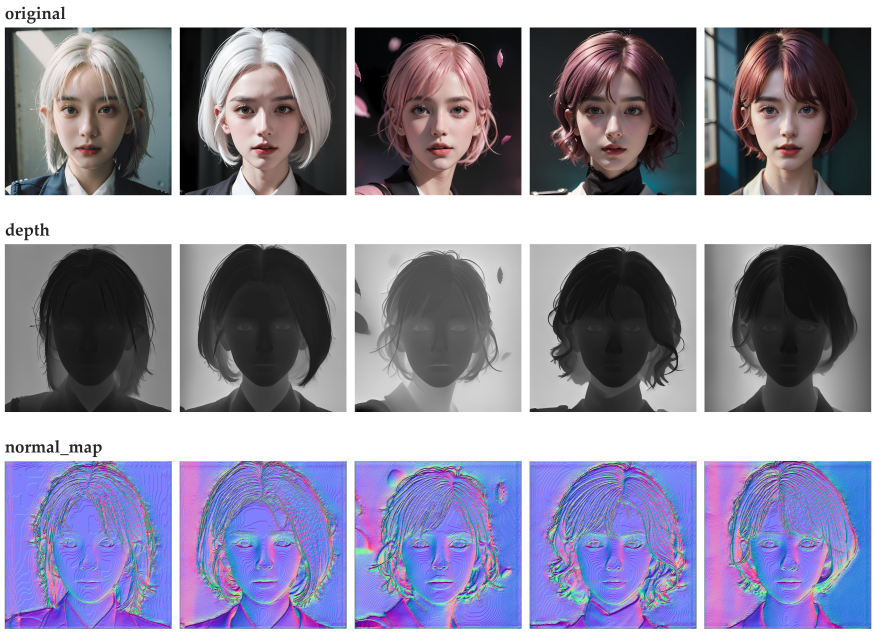


Figure 7. Example of a Depth Map Generated by depth-fm.

3.4. Head and Hand Detection and Skin Region Segmentation Techniques

In generating human figures, it is crucial to address the deterioration of details in the head and hands. This repair process involves two main steps: first, obtaining and enlarging images of the head and hand regions to facilitate detailed refinement; second, regenerating the skin region and re-integrating the repaired images into the original image. This section describes the implementation of head and hand detection tasks and skin image segmentation tasks.

- Head and Hand Detection
  In the detection tasks, this study uses an improved Vanilla Network to replace the original YOLOv8 backbone. The enhanced model shows superior performance across multiple evaluation metrics: compared to the original model, it achieves a 2.855% improvement in mean Average Precision (mAP50(B)), reaching 94.992%; a 2.845% increase in precision, reaching 93.634%; and a 4.523% improvement in recall, reaching 90.755%. These improvements enhance the model's ability to detect head and hand regions, ensuring the accuracy and reliability of the repair tasks.
- Skin Image Segmentation
  For skin segmentation, a lightweight skin segmentation model is introduced. The model selected is a project released by Will Brennan on GitHub[35], which uses FCNResNet101[36] and BiSeNetV2[37] as backbone networks for semantic segmentation of skin. Although the project was trained on a custom dataset of only 150 images from the COCO dataset, it has effectively implemented skin segmentation.

In the previous step, head and hand detection was completed. In this step, the detected regions are segmented to obtain separate hand and head images, which are then input into the skin detection model for initial skin segmentation. However, in practice, it was observed that this model performs suboptimally on smaller images, particularly in terms of foreground-background separation and edge clarity. This limitation affects its practical application in mask redrawing tasks.

To address these issues, this study adopts the ViTmatte model to refine the skin mask. As a mask refinement module, ViTmatte enhances accuracy and improves edge clarity beyond the original segmentation. The implementation process is illustrated in Figure 8, showing how ViTmatte refines the skin mask to achieve higher-quality repair results.

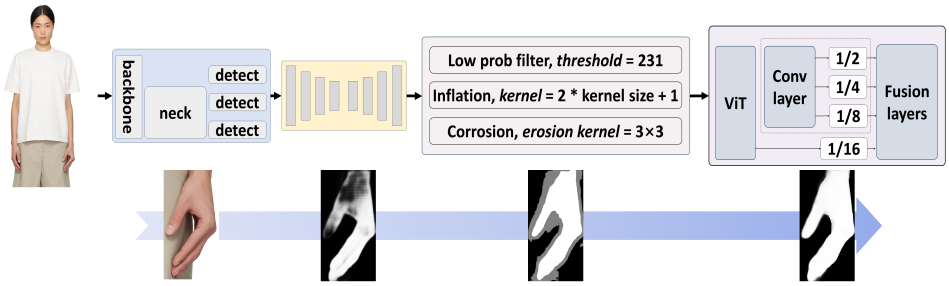


Figure 8. Schematic Diagram of the Skin Segmentation Process.

In summary, this subsection presents an effective solution for the repair of head and hand regions in human generation tasks by utilizing improved object detection techniques and high-precision skin segmentation methods.

4. Algorithm Design and Implementation

4.1. Comparison of Mainstream Generative Models

Current AI image generation technologies on the market are primarily divided into two categories: paid online platforms, such as Midjourney[38] and DALL·E[39], and locally deployable models, such as Stable Diffusion, Diffusers[40], and ComfyUI[41].

4.1.1. Online Platforms

Paid online platforms like Midjourney and DALL·E offer the advantage of requiring no model deployment or understanding of diffusion model principles. Users can simply input keywords and click generate to obtain the desired images. These platforms typically provide various services, including prompt correction, automatic prompt generation, and model switching. While these services are flexible and convenient, their primary drawback is the high cost. For example, Midjourney's basic subscription costs 8 per month, allowing only 200 images to be generated. Additionally, although these platforms support features such as mask redrawing, image-to-image generation, text-to-image generation, and LoRA[42] fine-tuning, users cannot upload custom LoRA models or freely choose diffusion models. This lack of flexibility and high cost can be limiting, especially in commercial applications.

4.1.2. Locally Deployable Models

Locally deployable models such as Stable Diffusion, Diffusers, and ComfyUI each have unique characteristics and limitations. Stable Diffusion, developed by Stability AI, features a rich array of plugins, an active community, and various functionalities including text-to-image, image-to-image, and video generation. It has a user-friendly visual interface but requires substantial VRAM and RAM, making deployment complex and more suitable for servers than laptops for advanced functions.

Diffusers are known for their active development community and flexibility, encapsulating nearly every step of AI image generation into pipelines. This allows users to call image encoders, text encoders, and image decoders directly, but the drawback is the increased model size due to retraining requirements. For example, the Stable Diffusion

1.5 model expands from 2GB to 11GB in Diffusers. Additionally, Diffusers lack a visual interface and plugin support, requiring functionalities to be implemented through code, increasing development difficulty.

ComfyUI, as the workflow version of Stable Diffusion, offers faster inference speed and lower VRAM requirements with relatively straightforward deployment. However, it demands a higher theoretical understanding from users as they need to manually select components like encoders and decoders. ComfyUI lacks an active community and plugin support due to its recent development. In summary, while paid online platforms provide convenient services, they are costly and less flexible. Locally deployable models offer greater freedom but pose higher deployment and development challenges, requiring users to choose solutions based on their needs and technical capabilities.

To determine the final development framework, this subsection compares the inference speed and model size of three different projects under the same task. To eliminate interference from other parameters on the experimental results, the same input parameters are used across the three models, including prompt words, diffusion models, generated image size, sampler selection, and iteration steps. Since the selectable samplers in each model differ and their specific implementations vary, the basic DDIM[43] sampling method is chosen to ensure consistent performance under the same conditions. The specific experimental results are shown in Table 2.

Table 2. Comparison of Parameters for Three Mainstream Generative Models.

model	environment	inference	Memory usage	Image size	sampler	steps
Stable diffusion	8.2G	10.1s	4264MB	512*768	DDIM	30
Comfy UI	3.1G	2.8s	2974MB	512*768	DDIM	30
diffusers	3.4G	3.2s	5140MB	512*768	DDIM	30

The experimental results indicate significant differences in inference speed and memory usage among the different models. Despite the varied strengths and weaknesses of each model, Stable Diffusion stands out in terms of image quality due to its earlier development and well-established community support. Consequently, Stable Diffusion was chosen as the generative model for this study based on these experimental data.

Currently, there are several generative models similar to Stable Diffusion available on the market, such as ComfyUI. However, research indicates that most of these models merely modify the front-end interface or reorganize and remove existing functions. For instance, the Fooocus model does not offer substantial improvements in implementation or inference speed over Stable Diffusion. Additionally, the lack of an active development community for these models, due to either a small user base or high model complexity, further complicates development efforts. This is one of the key reasons for excluding these models from consideration.

4.2. Background Generation Task

In the implementation of the background generation task, it is crucial to ensure that the foreground, specifically the product image, remains unchanged. This prevents alterations to the product image while generating a background that meets the requirements of e-commerce platforms. The steps for generating the foreground mask were detailed in a section 3.2, and will not be elaborated upon here. Once the mask is obtained, directly using Stable Diffusion for image generation may result in issues such as background edge overflow, incorrect foreground-background relationships, uncoordinated background content, and logical errors in the generated background. Therefore, integrating the ControlNet model to manage the background generation content is essential.

To address the issue of background edge overflow, ControlNet can be introduced with Canny edge detection to control the contours of the generated background. By inputting

the edge-detected image of the mask, ControlNet can effectively manage the background edges, preventing content overflow. However, solely relying on Canny edge detection cannot resolve issues related to uncoordinated background content or logical errors. Hence, incorporating ControlNet based on normal maps is necessary. This approach can control the lighting, depth, and structure of the generated background, ensuring a more natural and coordinated final output with the foreground.

In the e-commerce market, while background generation often involves complex backgrounds, most merchants require product images with plain backgrounds. Given that original images usually have complex backgrounds, controlling the depth and complexity of the background is crucial. For this purpose, this study employs depth maps and the mask's Canny image as input to ControlNet, enabling the transformation from a complex background to a simple one. The specific experimental results are shown in Figure 9, demonstrating the effectiveness of this method in background generation.

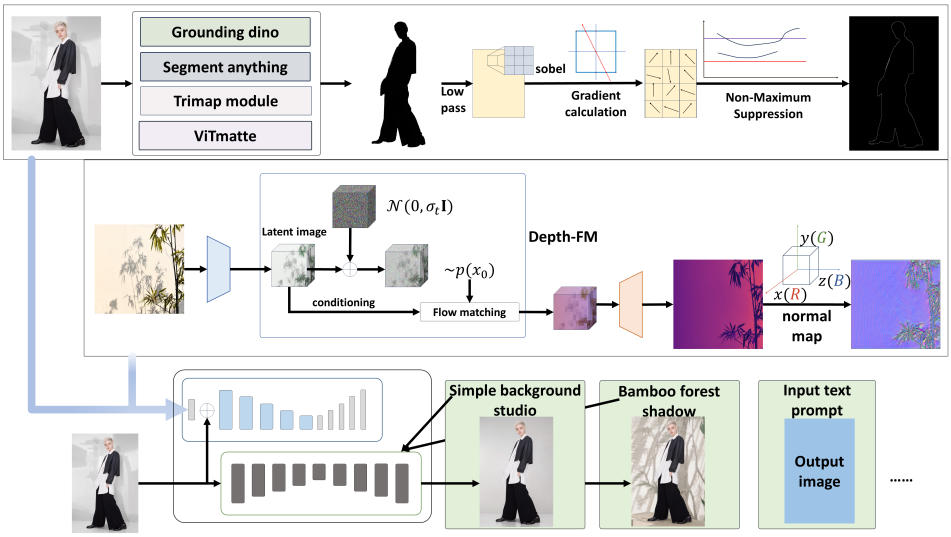


Figure 9. Workflow Diagram of the Background Generation Task.

4.3. Human Figure Generation Task

In the context of generating clothing product images, some e-commerce sellers use themselves as models to save costs. However, due to body shape or other issues, the quality of these product images may not be ideal, potentially affecting sales. This scenario has created a new demand: the simultaneous generation and replacement of both the background and the model. Therefore, this section proposes a novel algorithm to achieve the concurrent generation and replacement of backgrounds and human figures. The specific method involves first using Grounding DINO and Segment Anything for segmenting the clothing, followed by employing ControlNet to regenerate the background and the model, thus achieving simultaneous replacement.

4.3.1. Implementation Steps

- Clothing Segmentation:
  Initially, Grounding DINO and Segment Anything are used to perform instance segmentation of the clothing. After obtaining the mask for the clothing area, directly applying mask inpainting for human figure generation reveals limitations in the model's ability to generate content within the specified area. For instance, the generated content may not accurately match the original foreground area, the generated background may overflow the original area, and logical errors may occur between the foreground and background content. Hence, using these methods alone is insufficient to address all issues.
- Pose Estimation:

To control the pose and position of the generated human figure, the YOLOv8 model is used to generate a pose estimation map. However, while the pose estimation map can solve the issue of the human figure's generation position, it cannot effectively prevent background overflow, logical errors, and depth issues. Therefore, additional control mechanisms are required beyond pose estimation.

- Edge and Depth Control:
  Incorporating Canny edge detection helps in precise background edge control, ensuring that the generated background does not overflow into the foreground area. Additionally, a depth map (generated by the depth-FM model) is used to control the background's depth, ensuring consistency in depth between the generated background and the foreground.

4.3.2. Application Scenarios

In practical applications, e-commerce merchants aim to batch-produce product images with minimal manual intervention. Typically, they might use mannequins as input images. To transform these mannequin images into images with real models, this section's task is to process the input merchant-made product images. Whether the input is a real model or a mannequin, the goal is to regenerate the background and the human figure.

By combining advanced segmentation, pose estimation, and control techniques, this method provides a robust solution for e-commerce merchants to create high-quality product images that enhance the appeal and potential sales of their products.

The specific process is as follows:

- Utilize Grounding DINO and the Segment Anything model for clothing segmentation and obtain the edge detection map of its mask.
- Use the pose estimation map generated by the YOLOv8 model of the mannequin as input.
- Employ the depth-FM model to generate a depth map, which controls the background depth of field.
- Use these images as input for ControlNet, leveraging ControlNet to control the final generation of the background and the character.

This method effectively achieves the replacement of a real model with a mannequin while maintaining logical consistency and visual harmony between the foreground and background during the generation process. The specific implementation effect is shown in Figure 10.

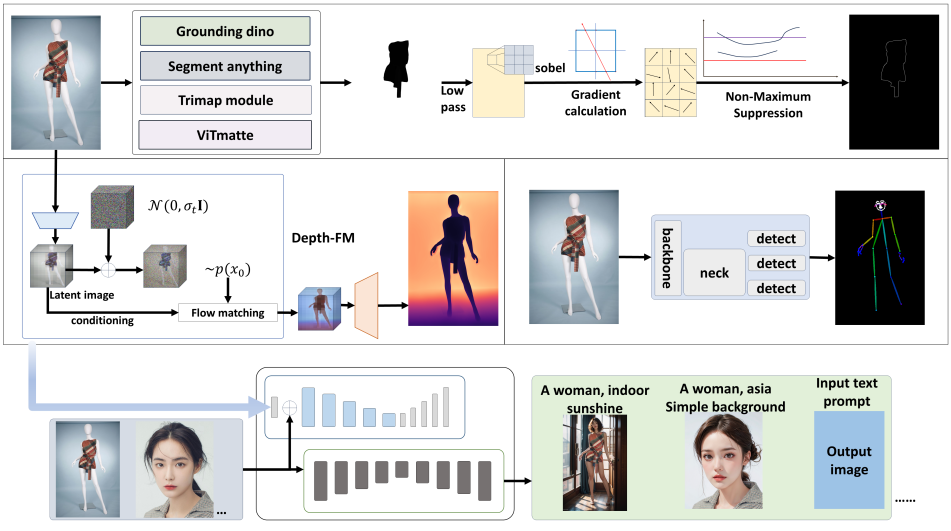


Figure 10. Schematic diagram of the character generation process.

4.4. Image Restoration

When accomplishing the task of character generation, especially when using generative models like Stable Diffusion, image restoration issues often arise. These issues mainly manifest in subpar generation quality in hand and face regions. During its early training stages, the Stable Diffusion model did not adequately cover hand details, leading to frequent quality issues when redrawing hands. Additionally, due to the small redraw regions of hands and faces during generation, these areas often exhibit poor quality, resulting in a deteriorated final image. To address these issues, this subsection proposes a local restoration method combining YOLOv8, Skin Detection, and ViTmatte algorithms to achieve high-resolution local image restoration.

The core idea of the local restoration algorithm is to focus on repairing critical areas of the image, such as the hands and face, thereby improving the overall image quality. The specific steps are as follows:

- Hand and Face Detection
  First, the YOLOv8 model is used to detect hands and faces. YOLOv8 efficiently detects the hands and faces in the image and provides the center point coordinates of these areas. This step lays the foundation for subsequent local restoration.
- Local Image Cropping
  Based on the center points detected by the YOLOv8 model, local images containing the hands and face are cropped. To ensure effective local image restoration, the cropping size is set to 512*512 pixels. This size is large enough to include sufficient detail for effective restoration.
- Skin Segmentation and Redrawing
  Skin segmentation algorithms, such as Skin Detection, are applied to the cropped local images. This step accurately segments the skin areas in the image and redraws these areas to enhance their detail and texture. The results of skin segmentation ensure natural transitions of skin texture during the restoration process.
- Image Blending
  The redrawn local images are blended with the original image using Poisson Image Editing. Poisson blending smoothly merges the restored areas with the original image, eliminating noticeable boundaries between the restored and surrounding regions. The goal of this step is to achieve a natural, seamless transition between the restored areas and the original image, ultimately enhancing the overall visual quality.

Through the aforementioned methods, the hand and face regions in Stable Diffusion-generated images can be effectively restored, addressing common generation quality issues. This local restoration method significantly improves image restoration quality and effectiveness through precise area detection, detailed skin segmentation and redrawing, and efficient image blending. Ultimately, the restored images not only exhibit enhanced detail but also achieve a more natural and harmonious transition with the original image. The image restoration process is illustrated in Figure 11.

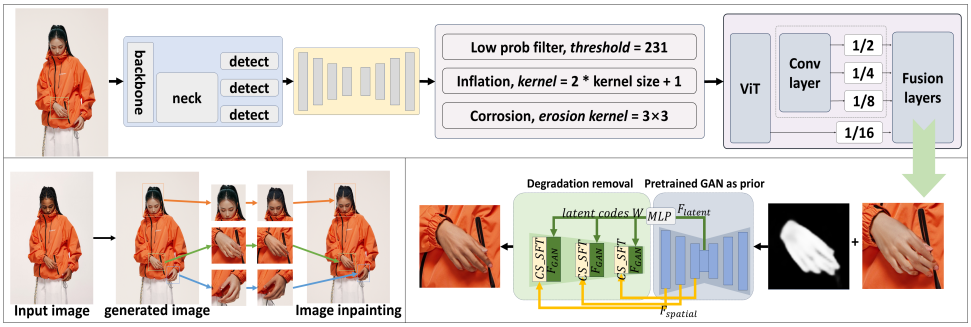


Figure 11. Schematic diagram of the image restoration process.

5. Experimental Results and Analysis

5.1. Preprocessing Experimental Results and Analysis

5.1.1. Evaluation Metrics

Algorithm evaluation is primarily divided into two aspects: computational cost and accuracy. Computational cost is measured using parameters such as the number of parameters (Params) and the number of gigaflops per second (GFLOPs). Generally, a smaller number of parameters and GFLOPs indicates lower demands on hardware computing resources and performance.

Accuracy evaluation involves the following metrics:

- Precision:

$$\text{Precision} = \frac{tp}{tp + fp}$$

where tp represents the number of true positive samples, and fp denotes the number of false positive samples.

- Recall:

$$\text{Recall} = \frac{tp}{tp + fn}$$

where fn represents the number of false negative samples.

- F1 Score:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1 Score is the harmonic mean of precision and recall, used to balance these two metrics.

- Average Precision (AP):

$$\text{AP} = \int_0^1 P(R)\, dR$$

AP measures the performance of the classifier by calculating the area under the precision-recall curve.

- Mean Average Precision (mAP):

$$\text{mAP} = \frac{\sum_{i=1}^{K} AP_i}{K}$$

where K is the total number of detected object categories and AP_i is the average precision for the i-th category.

These metrics collectively provide a comprehensive evaluation of model performance. By analyzing these accuracy metrics in conjunction with computational cost, one can determine the suitability of the model for different application scenarios. Higher accuracy metrics combined with lower computational costs are optimization goals, ensuring that the model maintains excellent performance even in resource-constrained environments.

5.1.2. Ablation Study Based on Improved YOLOv8

To systematically evaluate the effectiveness of the improvements made to the YOLOv8 backbone network, we designed a series of ablation experiments focusing on hand detection tasks. The goal was to empirically validate the specific benefits of the model optimizations through quantitative data. For this purpose, we selected the authoritative 11k Hands dataset[44] as the benchmark, known for its broad representativeness and challenge, making it an ideal platform for assessing hand detection algorithm performance.

In the experimental design, YOLOv8n was established as the baseline model framework to ensure consistency and comparability. We then constructed two variant models: one integrating the original VanillaNet as the backbone network within the YOLOv8 framework, referred to as YOLOv8-vanilla, as the pre-improvement control; the other

embedding our optimized backbone network within the YOLOv8 architecture, referred to as YOLOv8-ours, representing the direct manifestation of the optimization effects.

The experimental results indicate that the YOLOv8-ours model, which incorporates our improved backbone network, demonstrates significant advantages across several key metrics. Specifically, the model achieved notable reductions in the number of parameters and the gigaflops per second (GFLOPs) required, reflecting enhanced computational resource efficiency and facilitating lightweight deployment with faster response times. Most importantly, in terms of the core detection performance metric—mean Average Precision (mAP@0.5)—the YOLOv8-ours model showed a significant improvement over YOLOv8-vanilla. This finding directly validates the effectiveness of our backbone network optimization strategy and its positive impact on model performance enhancement.

Figure 12 visually illustrates the results of these experiments, providing a comparative analysis of different models across key dimensions such as parameter count, GFLOPs, and mAP@0.5. This comparison further solidifies our conclusions regarding the effectiveness of the model optimizations.

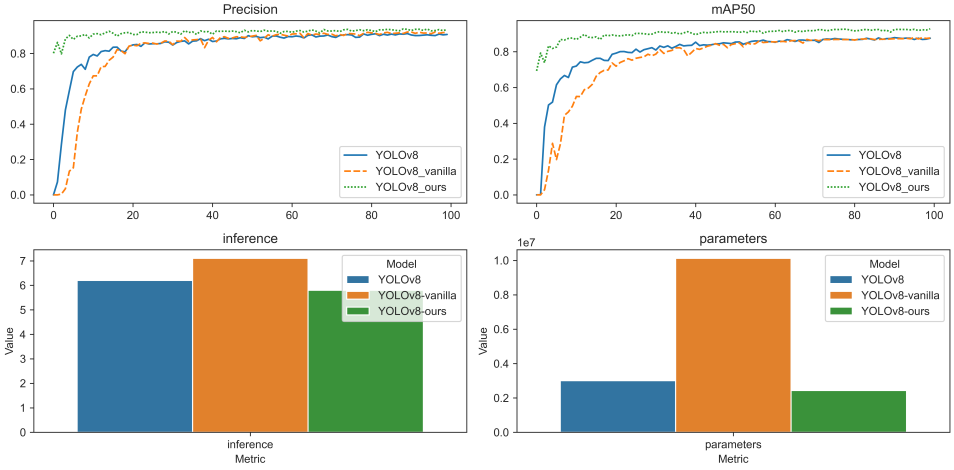


Figure 12. Ablation study results of the backbone network in object detection tasks.

In the complex and challenging task of pose estimation, this study further explores the potential application of the improved YOLOv8 model. By applying the optimized YOLOv8 model to the widely recognized and highly representative COCO2017 dataset, and conducting a meticulously designed 100-epoch training cycle, we obtained encouraging experimental results.

The experimental results reveal that the improved YOLOv8 model exhibits exceptional performance in pose estimation tasks. Specifically, the model demonstrated significant improvements in capturing human keypoints, estimating joint positions, and overall pose reconstruction. These enhancements are evident not only in the increased accuracy but also in the model's robustness to complex scenes and occlusions.

Figure 13 illustrates the experimental results, visually presenting the performance of the improved YOLOv8 model on the COCO2017 dataset. The figure includes metrics such as keypoint detection accuracy, mAP@0.5, model inference time, and model parameter count, which collectively validate the effectiveness of the model optimization strategy and its practical applicability.

Notably, after 100 training epochs, the model has thoroughly learned the rich pose variations and scene diversity present in the COCO2017 dataset, demonstrating stable performance on the test set. This achievement not only highlights the flexibility and scalability of the YOLOv8 model architecture but also establishes a solid foundation for future research and applications in the field of pose estimation.

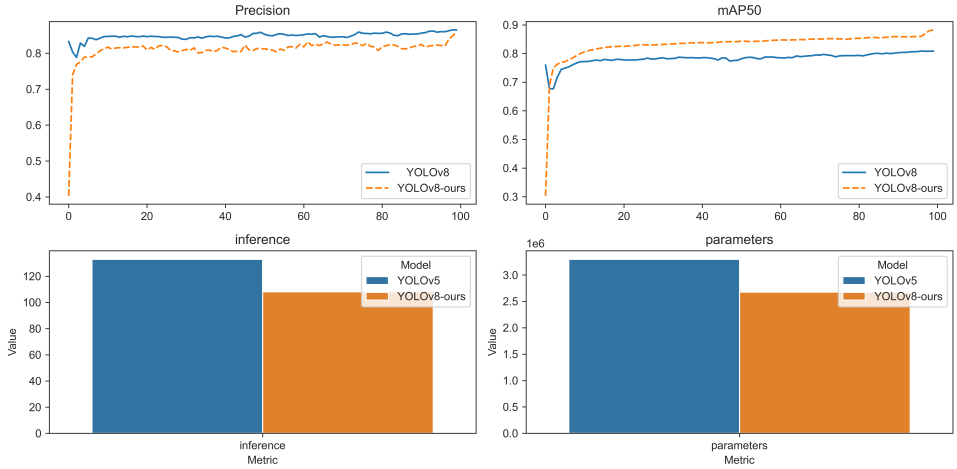


Figure 13. Ablation study results of the backbone network in pose estimation tasks.

5.1.3. Skin segmentation task

To comprehensively and scientifically evaluate the impact of the added ViTmatte module on model performance in the skin segmentation task, we designed a detailed validation process as follows:

- Dataset Preparation
  First, we used a self-constructed skin segmentation dataset as the baseline, which contains a diverse set of skin samples and their corresponding labels (Original dataset). This dataset serves as the reference for all subsequent training and validation steps.
- Baseline Model Inference
  We performed inference using a basic skin detection model (skin detection model) on the self-constructed dataset to generate a dataset without the ViTmatte module's processing results, referred to as old dataset. This step provides a reference for the model's performance before enhancement.
- Enhanced Model Inference
  Next, we incorporated the ViTmatte module to construct an enhanced skin detection and segmentation model (skin detection + ViTmatte). We performed inference on the same self-constructed dataset to generate a dataset with ViTmatte module optimizations, referred to as new dataset. The key here is to observe the improvements in model output quality due to the ViTmatte module.
- Model Training and Comparison
  1. Training and Evaluation: Train YOLOv5 separately on old dataset and new dataset. By comparing the training curves, convergence rates, and final accuracies of the two datasets, we can directly assess the performance improvement brought by the ViTmatte module on the YOLOv5 model in the skin segmentation task. To provide a more comprehensive view of the ViTmatte module's advantages, train YOLOv5 on the Original dataset and compare the results with those from the two previously mentioned datasets to ensure the generalizability and accuracy of the experimental results.
  2. Additional Verification with YOLOv8-van: To verify the performance of the YOLOv8-van model, train it on the Original dataset and compare the results with those from YOLOv5 on the same dataset. The focus here is on differences in model size, inference speed, and accuracy.
- Results Presentation and Analysis
  By plotting specific comparative curves, we visually present the differences in training performance across different datasets and models. This includes a detailed analysis of the ViTmatte module's impact on accuracy improvement and enhanced generalization capabilities.

Additionally, we compare the training results of YOLOv5 with YOLOv8-van to explore the strengths and weaknesses of different model architectures in the skin segmentation task.

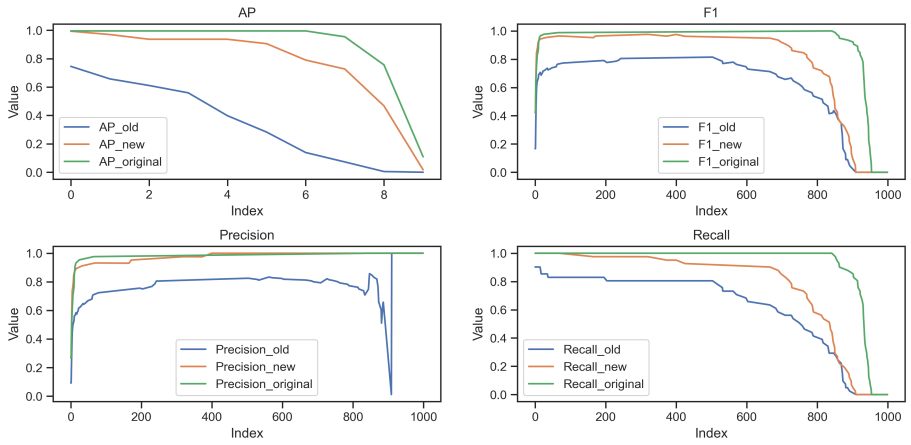


Figure 14. Ablation study results of the backbone network in the skin segmentation task.

From the analysis of Figure 14, it is evident that the introduction of the ViTmatte module has had a significant and positive impact on model performance. Specifically, the enhanced model exhibits considerable improvements across multiple key parameters compared to the original model. This finding not only reinforces the effectiveness of the ViTmatte module in improving model performance but also provides strong evidence for future related research.

Notably, the core evaluation metric, mean Average Precision (mAP@50), shows an impressive increase of 24.8%. This substantial improvement underscores the ViTmatte module's crucial role in optimizing segmentation accuracy and enhancing the model's ability to capture complex scenes and fine details. The rise in mAP@50 indicates a significant boost in the model's accuracy in detecting targets (in this case, skin regions), which is vital for enhancing the practicality and reliability of skin segmentation tasks.

In the comparative experiment using the Original dataset, we evaluated the performance differences between YOLOv5 and the improved YOLOv8-van model in skin segmentation tasks. The specific experimental results are shown in Figure 15.

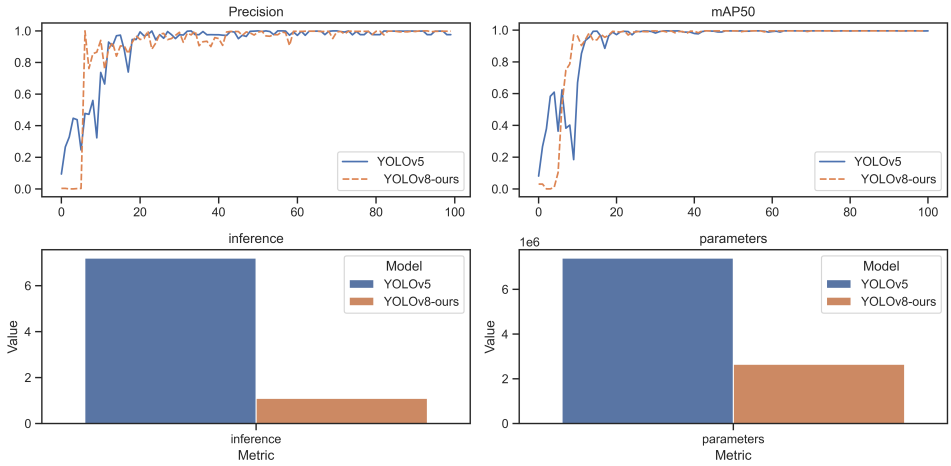


Figure 15. Parameter comparison results between YOLOv5 and the improved YOLOv8-van model.

From the analysis of the experimental results in Figure 15, it is evident that the improved YOLOv8-van model demonstrates significant advantages in the skin segmentation task, particularly regarding inference speed and parameter count.

Inference Speed Improvement: YOLOv8-van achieves a substantial increase in inference speed while maintaining nearly unchanged accuracy and mAP@50. This means that under the same hardware conditions, YOLOv8-van can process input images more rapidly and generate segmentation results faster.

Reduction in Model Parameters: Compared to YOLOv5, the YOLOv8-van model features a reduction of more than half in the number of parameters. This reduction not only makes the model more efficient in terms of storage and transmission but also decreases its computational resource requirements.

These improvements collectively highlight the exceptional results of YOLOv8-van in terms of architectural design and optimization. By introducing a more efficient network structure, YOLOv8-van achieves a more lightweight and faster inference while maintaining high performance.

Additionally, it is noteworthy that despite the significant reduction in the number of parameters, YOLOv8-van still retains comparable accuracy and mAP@50 to YOLOv5. This further validates its effectiveness and reliability in skin segmentation tasks.

5.2. Ablation Study on Image Generation

5.2.1. Evaluation Metrics

In the field of image generation, several metrics are commonly used to assess the quality and diversity of generated images. These include the Fréchet Inception Distance (FID)[45], Inception Score (IS)[46], and Kernel Inception Distance (KID)[47].

- Fréchet Inception Distance (FID)
  FID is a widely used metric for evaluating the discrepancy between the distributions of generated images and real images. Introduced by Martin Heusel et al. in 2017, FID measures this difference by calculating the Fréchet distance between two distributions. The Fréchet distance considers both the mean and covariance of the distributions, providing a comprehensive measure of how well the generated images match the real data distribution.
  To compute FID: 1. Extract a set of samples from both the real data distribution and the generated images. 2. Use a pre-trained Inception network to extract feature vectors from these samples. 3. Calculate the mean and covariance matrices of these feature vectors for both distributions. 4. Compute the Fréchet distance between the two distributions using the following formula:

$$\text{FID} = \|\mu_g - \mu_r\|_2^2 + \text{Tr}(\Sigma_g + \Sigma_r - 2(\Sigma_g\Sigma_r)^{1/2})$$

  where μ_g and μ_r are the mean vectors of the generated and real data distributions, respectively, and Σ_g and Σ_r are their covariance matrices. The lower the FID score, the closer the generated images are to the real data distribution, indicating higher image quality.
- Inception Score (IS)
  The Inception Score is based on the Inception Net-V3 model developed by Google. This model outputs a 1000-dimensional vector for each input image, where each dimension represents the probability of the image belonging to a particular class. The IS measures the quality of generated images based on two aspects: 1. Expectation of Image Quality: The score assesses how likely it is that the generated images belong to a distinct class. 2. Divergence of Class Distributions: It evaluates how different the class distributions of generated images are from the average class distribution across all images.
  The IS formula is given by:

$$\text{IS}(G) = \exp\left(\mathbb{E}_{x\sim p_g}\text{D}_{\text{KL}}(p(y|x)\|p(y))\right)$$

where G represents the generative model, D_{KL} is the Kullback-Leibler divergence, $p(y|x)$ is the conditional class distribution for a given image, and $p(y)$ is the marginal class distribution.

- Kernel Inception Distance (KID)
  KID is another metric used for assessing the quality of generated images. It is based on Maximum Mean Discrepancy (MMD) and does not require the inverse of the covariance matrix. KID calculates the distance between the distributions of image features from real and generated images using:

$$\mathrm{KID}(x, y) = \mathbb{E}[\log(p(x, y))] - \mathbb{E}[\log(p(x)p(y))]$$

  where x and y denote the pixel values of real and generated images, respectively. $p(x, y)$ represents the joint probability distribution, and $p(x)$ and $p(y)$ are the marginal distributions. KID measures the mutual information and KL divergence between the distributions, requiring substantial image data for accurate estimation.

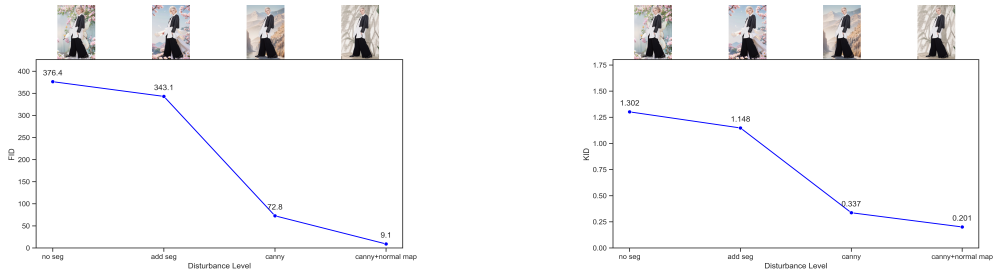
These metrics provide a comprehensive evaluation of the quality and diversity of generated images, each with its unique strengths and computational requirements.

5.2.2. Ablation Study on Background Generation Tasks

In the proposed background replacement task, we introduced improvements by incorporating an image segmentation model and the ControlNet model. The image segmentation model is used to obtain accurate masks, while the ControlNet model is employed to generate background content and control the depth of field. We tested four scenarios to evaluate the effectiveness of these enhancements:

- No Segmentation Model or ControlNet Model (no seg): This baseline scenario serves as a reference to assess the impact of incorporating segmentation and ControlNet features.
- Only Image Segmentation Model (add seg): This scenario evaluates the effect of using only the image segmentation model, without the ControlNet model.
- Using Canny Edge Detection (canny): This condition assesses the effectiveness of Canny edge detection for background generation, excluding the image segmentation model and ControlNet.
- Combination of Canny Edge Detection and Normal Map (canny+normal map): This scenario tests the impact of combining Canny edge detection with normal maps, providing both edge information and depth details for background generation.

The detailed experimental results are illustrated in Figure 16 and table 3.

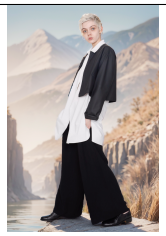


(a) FID variation curve Figure **(b)** KID variation curve Figure

Figure 16. Effectiveness Validation Results for Background Replacement Methods.

Based on Figure 16 (1), it can be observed that the FID score gradually decreases with the incremental addition of improvement measures. Without the use of image segmentation and ControlNet models, the FID score is 376.4. After incorporating the image segmentation model, the FID score drops to 343.1. With the application of Canny edge detection, the FID score significantly decreases to 22.8, and when combining edge detection with normal maps,

Table 3. IS scores under different models.

samples				
stage	No seg	Add seg	Canny	Canny + normal map
IS Score ± std.	1.01±0.002	1.1±0.03	1.62±0.101	1.86±0.032

the FID score further reduces to 9.1. This indicates that the quality of generated images significantly improves as the enhancement measures are progressively implemented.

Figure 16 (2) shows that the KID score also progressively decreases with the introduction of improvement measures. Initially, without image segmentation and ControlNet models, the KID score is 1.302. Following the addition of the image segmentation model, the KID score decreases to 1.148. After applying Canny edge detection, the KID score significantly drops to 0.337, and when combining edge detection with normal maps, the KID score further falls to 0.201.

Table 3 demonstrates that the IS score increases with the addition of improvement measures. Initially, without image segmentation and ControlNet models, the IS score is 1.01. After including the image segmentation model, the IS score rises to 1.1. With the use of Canny edge detection, the IS score increases to 1.62, and combining edge detection with normal maps further elevates the IS score to 1.86.

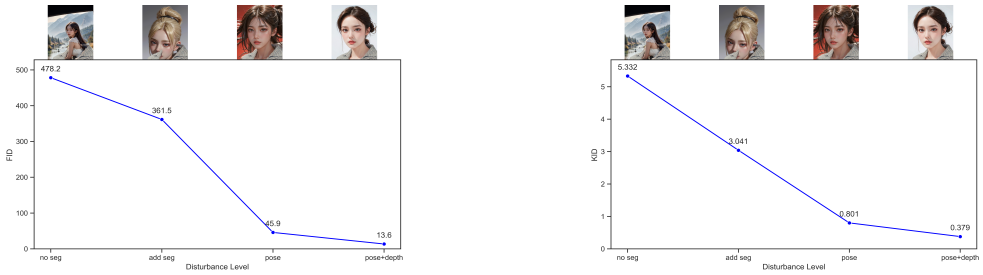
In summary, incorporating image segmentation and ControlNet models significantly enhances the performance of background replacement tasks. Specifically, the method combining Canny edge detection and normal maps shows the best performance in terms of FID and KID scores, indicating a substantial improvement in the quality of generated images compared to cases without these enhancements. This validates the effectiveness of the proposed methods.

These experimental results indicate that using image segmentation and ControlNet models significantly improves both the quality and diversity of the generated images. Notably, the approach that combines Canny edge detection and normal maps demonstrates superior performance across all metrics, further confirming the effectiveness and advantages of the proposed methods.

5.2.3. Character Generation Task

In this article, we propose two key improvements to accomplish the model replacement task. First, we introduce an image segmentation model to generate precise masks. Second, we incorporate a ControlNet model to regulate features such as image depth and character pose. These enhancements aim to improve the realism and consistency of the generated images.

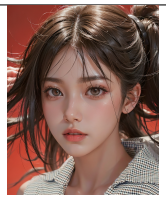
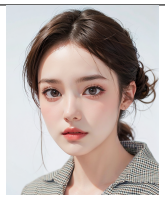
To validate the effectiveness of the proposed methods, we designed an experiment in which 200 images were generated both before and after incorporating the ControlNet model. A detailed comparative analysis of various parameters between these two datasets was conducted. The specific results are shown in Figure 17.



(a) FID variation curve Figure **(b)** KID variation curve Figure

Figure 17. Verification results of the effectiveness of the character generation task method.

Table 4. IS scores under different models.

samples				
stage	No seg	Add seg	Pose	Pose + depth
IS Score ± std.	1.12±0.18	1.71±0.06	2.21±0.11	3.31±0.03

Based on Figure 17 (1), it is evident that the FID score gradually decreases with the incremental incorporation of improvement measures. Without the use of image segmentation and ControlNet models, the FID score is 478.2. After incorporating the image segmentation model, the FID score decreases to 361.5. With the addition of pose estimation, the FID score significantly drops to 45.9, and further decreases to 13.6 when combining pose estimation with depth map estimation. This indicates a substantial improvement in the quality of the generated images with each added enhancement.

Figure 17 (2) illustrates that the KID score also decreases progressively with the implementation of the improvement measures. Initially, without the image segmentation and ControlNet models, the KID score is 5.332. After adding the image segmentation model, the KID score reduces to 3.041. Following the integration of pose estimation, the KID score significantly falls to 0.801, and with the combination of pose estimation and depth map estimation, it further decreases to 0.379.

Table 4 shows that the IS score increases with the addition of improvement measures. Initially, without image segmentation and ControlNet models, the IS score is 1.12. After incorporating the image segmentation model, the IS score rises to 1.71. With the addition of pose estimation, the IS score increases to 2.21, and further increases to 3.31 with the combination of pose estimation and depth map estimation.

The experimental results demonstrate that the proposed improvements significantly enhance the quality and controllability of the generated images, thereby validating the effectiveness of the methods presented in this study. These findings not only highlight the efficacy of the image segmentation and ControlNet models in model replacement tasks but also provide robust support and reference for future research in the field of image generation.

6. Conclusion and Future Work

In this study, we explored the application of the Stable Diffusion model in the field of clothing product image production. Through a series of experiments, we have verified the effectiveness and stability of the model at different stages of image processing. The following are the main findings and conclusions of this study:

- Model performance evaluation: In different stages of image processing, including no segmentation, added segmentation, Canny edge detection, and Canny edge detection additive line graph, the Stable Diffusion model exhibits good image generation ability. By measuring the IS Score ± std of the generated image We found that with the increase of processing stages, the image quality improved, especially in the Canny edge detection additive line graph stage, where the IS Score of the model reached 1.86 ± 0.032, showing significant improvement.
- Image quality improvement: After adding segmentation information, the details and contours of the generated image were significantly improved, indicating that segmentation information plays an important role in the image generation process. Canny edge detection further enhances the edge details of the image, making the generated image clearer and more realistic.
- Potential application of the model: The experimental results show that the Stable Diffusion model has great potential for application in clothing product image production. It can not only improve the quality of images, but also further optimize the generation effect through different image processing techniques.

This model can be used in various fields such as clothing product display, advertising design, and e-commerce platforms in commercial applications, and has broad application prospects. Although this study has achieved certain results, there are still some aspects that need further exploration and improvement:

- Algorithm optimization: Further optimize the algorithm of the Stable Diffusion model to improve the speed and quality of image generation.
  Explore the combination of more image processing techniques, such as super-resolution technology in deep learning, to enhance the detail representation of generated images.
- Large scale data training: Expand the size and diversity of the training dataset to improve the model's generalization ability and adaptability. Attempt to train and test the model using different types of clothing product images to verify its generality.
- Practical application verification: Apply the model to actual clothing product image production projects to verify its performance and effectiveness in real-world environments.

Collect user feedback and further improve the model and image generation process to meet the needs of different users.

Through this study, we have preliminarily validated the potential application of the Stable Diffusion model in the field of clothing product image production. In the future, we will continue to optimize and expand this model in order to achieve better results and value in practical applications.

References

1. Ho, J., & Jain, A. Denoising Diffusion Probabilistic Models. *arXiv preprint arXiv:2006.11239*, 2020.
2. Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. Generative Adversarial Nets. Département d'informatique et de recherche opérationnelle, Université de Montréal. *arXiv preprint arXiv:1406.2661v1 [stat.ML]*.
3. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. High-resolution image synthesis with latent diffusion models. 2021.
4. Luo, C. Understanding Diffusion Models: A Unified Perspective. *arXiv preprint arXiv:2208.11970v1 [cs.LG]*, 2022.
5. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhin, I. Attention is All You Need. In *Advances in Neural Information Processing Systems* (pp. 5998–6008).
6. Kingma, D. P., & Welling, M. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114v11 [stat.ML]*, 2013.
7. Zhang, L., & Agrawala, M. Adding Conditional Control to Text-to-Image Diffusion Models. *arXiv preprint arXiv:2302.05543v1 [cs.CV]*, 2023.
8. Liu, P., Jiang, Z., Yuan, W., Hayashi, H., Fu, J., & Neubig, G. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *arXiv preprint arXiv:2107.13586v1 [cs.CL]*, 2021.
9. Pan, S. J., & Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345–1359, 2010.
10. Karras, T., Aila, T., Laine, S., & Häivälä, J. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2018.

11. Karras, T., Laine, S., & Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4401–4410.
12. Batson, J., & Nowak, R. C. Noise2Noise: Learning Image Restoration without Clean Data. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2019.
13. He, K., Zhang, X., Ren, S., & Sun, J. Deep Residual Learning for Image Recognition. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
14. Esser, A., Rombach, R., & Ommer, B. Taming Transformers for High-Resolution Image Synthesis. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 12873–12883.
15. Carion, N., Misra, I., Goyal, A., Bojanowski, P., & Joulin, A. End-to-End Object Detection with Transformers. *arXiv preprint arXiv:2106.09777*, 2021.
16. Caron, M., Touvron, H., Misra, I., Jégou, H., Bojanowski, P., & Joulin, A. Emerging Properties in Self-Supervised Vision Transformers. *arXiv preprint arXiv:2104.14294*, 2021.
17. Lin, T., Maire, M., Belongie, S., et al. Microsoft COCO: Common Objects in Context. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2014, pp. 740–755.
18. Radford, A., Kim, J. W., Xiong, R. E., et al. Learning Transferable Visual Models From Natural Language Supervision. In *Proc. of the International Conference on Machine Learning (ICML)*, 2021, pp. 8748–8763.
19. Kirillov, A., Wu, Y., He, K., et al. Segment Anything. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 11271–11282.
20. Ravi, S., & Larochelle, H. Optimization as a Model for Few-Shot Learning. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2017.
21. Girshick, R., Donahue, J., Darrell, T., & Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.
22. Xu, N., Price, B., Cohen, S., & Huang, T. Deep Image Matting. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2970–2979.
23. Lu, K., Chen, Y., Lai, L., et al. IndexNet: A New Network Design for Image Matting. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 4987–4996.
24. Levin, A., Lischinski, D., & Weiss, Y. A Closed-Form Solution to Natural Image Matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2), 228–242, Feb. 2008.
25. Liu, T., Zhang, J., & Shen, C. ViTMatte: A Vision Transformer Approach for Transparent Object Matting. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 14966–14975.
26. Canny, J. F. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6), 679–698, Nov. 1986.
27. Sobel, I. An Isotropic 3x3 Image Gradient Operator. In *Presentation at the Stanford Artificial Intelligence Project (SAIL)*, 1968.
28. Duda, R. O., & Hart, P. E. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM*, 15(1), 11–15, Jan. 1972.
29. Glenn, J. YOLOv8: Ultralytics Official Implementation. Ultralytics, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics
30. Chen, H., Wang, Y., Guo, J., & Tao, D. VanillaNet: The Power of Minimalism in Deep Learning. *arXiv preprint arXiv:2305.12972*, 2023.
31. Li, C., Zhou, A., & Yao, A. Omni-Dimensional Dynamic Convolution. In *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=DmpCfq6Mg39
32. Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. *arXiv e-prints*, 2022. DOI: 10.48550/arXiv.2207.02696.
33. Gui, M., Fischer, J. S., Prestel, U., Ma, P., Kotovenko, D., Grebenkova, O., Baumann, S. A., Hu, V. T., & Ommer, B. DepthFM: Fast Monocular Depth Estimation with Flow Matching. *arXiv preprint arXiv:2403.13788*, 2024.
34. Cignoni, P., Montani, C., & Scopigno, R. A Comparison of Mesh Simplification Algorithms. *Computers & Graphics*, 22(1), 37–54, Feb. 1998.
35. Brennan, W. SemanticSegmentation. GitHub repository, 2024. [Online]. Available: https://github.com/WillBrennan/SemanticSegmentation
36. Long, J., Shelhamer, E., & Darrell, T. Fully Convolutional Networks for Semantic Segmentation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.
37. Yu, C., Wang, X., Peng, C., et al. BiSeNet V2: Bilateral Network with Guided Aggregation for Real-Time Semantic Segmentation. *International Journal of Computer Vision*, 129, 3051–3068, 2021.
38. Midjourney. Midjourney: AI Art Generator. [Online]. Available: https://www.midjourney.com/
39. Ramesh, A., Pavlov, M., Goh, G., et al. Zero-Shot Text-to-Image Generation. In *Proc. of the International Conference on Machine Learning (ICML)*, 2021, pp. 8821–8831.
40. Hugging Face. Diffusers: State-of-the-Art Diffusion Models. GitHub repository, 2023. [Online]. Available: https://github.com/huggingface/diffusers

41. ComfyUI. ComfyUI: A Powerful and Modular Stable Diffusion GUI and Backend. GitHub repository, 2023. [Online]. Available: https://github.com/comfyanonymous/ComfyUI
42. Hu, R., Zhang, H., Li, T., et al. LoRA: Low-Rank Adaptation of Large Language Models. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2023.
43. Song, J., Meng, S., & Cohen, J. E. Denoising Diffusion Implicit Models. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2021.
44. Afifi, M. 11K Hands: Gender Recognition and Biometric Identification Using a Large Dataset of Hand Images. *Multimedia Tools and Applications*, 78, 23223–23246, 2019. https://doi.org/10.1007/s11042-019-7424-8.
45. Heusel, M., Ramsauer, H., Unterthiner, T., et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Proc. of the Neural Information Processing Systems (NeurIPS)*, 2017, pp. 6626–6637.
46. Salimans, T., Goodfellow, I., Zaremba, W., et al. Improved Techniques for Training GANs. In *Proc. of the Neural Information Processing Systems (NeurIPS)*, 2016, pp. 2234–2242.
47. Borji, R. Pros and Cons of GAN Evaluation Measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11), 2551–2565, Nov. 2019.