Generate Like Experts: Multi-Stage Font Generation by Incorporating Font Transfer Process into Diffusion Models

Supplementary Materials

6. Proofs and Derivations

In this section, we provide the detailed proofs and derivations for our proposed model. Our derivations mainly follow the derivations of denoising diffusion probabilistic models in Ref. [1, 2], and people can refer to them for more details.

6.1. The Derivation of Eq. (9)

In this paper, our proposed model is constructed by utilizing a font transfer process to connect the traditional diffusion processes (DDPM) of the latent features z_t^c and z_t^g, where the latent feature z_t^g of the target font is gradually replaced by the latent feature z_t^c of the source font from $t = t_1$ to $t = t_2$. Therefore, in data sub-space, only z_t^g exists at $t = t_1$ while only z_t^c exists at $t = t_2$. The corresponding latent features $\tilde{z}_{t_1}$ and $\tilde{z}_{t_2}$ can be expressed as

$$\tilde{z}_{t_1} = z_{t_1}^g = \sqrt{\bar{\alpha}_{t_1}} z_0^g + \sqrt{1-\bar{\alpha}_{t_1}}\epsilon_0, \tag{24}$$

$$\tilde{z}_{t_2} = z_{t_2}^c = \sqrt{\bar{\alpha}_{t_2}} z_0^c + \sqrt{1-\bar{\alpha}_{t_2}}\epsilon_0. \tag{25}$$

As the latent feature z_t^g will be gradually replaced by the latent feature z_t^c in the font transfer region $t \in (t_1, t_2]$, we formulate the forward latent feature $\tilde{z}_t$ in this region as

$$\tilde{z}_t = \sqrt{\alpha_t}\tilde{z}_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1}^* - \Psi(z_0^g - z_0^c), \tag{26}$$

where $\{\epsilon_t^*, \epsilon_t\} \sim \mathcal{N}(0, I)$. Ψ is the coefficient for the term $z_0^g - z_0^c$, which will be determined in this section. With the reparameterization trick, the forward latent $\tilde{z}_t$ in region $(t_1, t_2]$ can be written as

$$\begin{aligned}
\tilde{z}_t &= \sqrt{\alpha_t}\tilde{z}_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1}^* - \Psi(z_0^g - z_0^c) && (27)\\
&= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\tilde{z}_{t-2} + \sqrt{1-\alpha_{t-1}}\epsilon_{t-2}^* - \Psi(z_0^g - z_0^c)\right) + \sqrt{1-\alpha_t}\epsilon_{t-1}^* - \Psi(z_0^g - z_0^c) && (28)\\
&= \sqrt{\alpha_t\alpha_{t-1}}\tilde{z}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\epsilon_{t-2} - (1+\sqrt{\alpha_t})\,\Psi(z_0^g - z_0^c) && (29)\\
&= \sqrt{\alpha_t\alpha_{t-1}\cdots\alpha_{t_1+1}}\tilde{z}_{t_1} + \sqrt{1-\alpha_t\alpha_{t-1}\cdots\alpha_{t_1+1}}\epsilon_{t_1} - \mathcal{H}_t\Psi(z_0^g - z_0^c), && (30)
\end{aligned}$$

and this equation can be further reduced by aligning the forward latent with the traditional diffusion process (DDPM) at $t = t_1$. Using Eq. (24), the latent feature $\tilde{z}_t$ in region $(t_1, t_2]$ can be expressed as

$$\begin{aligned}
\tilde{z}_t &= \sqrt{\alpha_t\alpha_{t-1}\cdots\alpha_{t_1+1}}\left(\sqrt{\bar{\alpha}_{t_1}}z_0^g + \sqrt{1-\bar{\alpha}_{t_1}}\epsilon_0^*\right) + \sqrt{1-\alpha_t\alpha_{t-1}\cdots\alpha_{t_1+1}}\epsilon_{t_1} - \mathcal{H}_t\Psi(z_0^g - z_0^c)\\
&= \sqrt{\alpha_t\alpha_{t-1}\cdots\alpha_1}z_0^g + \sqrt{1-\alpha_t\alpha_{t-1}\cdots\alpha_1}\epsilon_0 - \mathcal{H}_t\Psi(z_0^g - z_0^c)\\
&= \sqrt{\bar{\alpha}_t}z_0^g + \sqrt{1-\bar{\alpha}_t}\epsilon_0 - \mathcal{H}_t\Psi(z_0^g - z_0^c), && (31)
\end{aligned}$$

where we have defined $\mathcal{H}_t = 1 + \sqrt{\alpha_t} + \sqrt{\alpha_t\alpha_{t-1}} + \cdots + \sqrt{\alpha_t\alpha_{t-1}\cdots\alpha_{t_1+2}}$, and $\mathcal{H}_t$ has the following recursion formula,

$$\begin{aligned}
\mathcal{H}_t &= 1 + \sqrt{\alpha_t} + \sqrt{\alpha_t\alpha_{t-1}} + \cdots + \sqrt{\alpha_t\cdots\alpha_{t_1+2}}\\
&= 1 + \sqrt{\alpha_t}\left(1 + \sqrt{\alpha_{t-1}} + \sqrt{\alpha_{t-1}\alpha_{t-2}} + \cdots + \sqrt{\alpha_{t-1}\alpha_{t-2}\alpha_{t-3}\cdots\alpha_{t_1+2}}\right)\\
&= 1 + \sqrt{\alpha_t}\mathcal{H}_{t-1}, && (32)
\end{aligned}$$

which has been utilized in our implementation for calculating $\mathcal{H}_t$. The coefficient Ψ can be determined by aligning the forward latent with the traditional diffusion process at $t = t_2$,

$$\Psi = \frac{\sqrt{\bar{\alpha}_{t_2}}}{\mathcal{H}_{t_2}} = \frac{\sqrt{\tilde{\alpha}(1, t_2)}}{1 + \sum_{m=t_1+2}^{t_2}\sqrt{\tilde{\alpha}(m, t_2)}}, \tag{33}$$

where we have defined $\tilde{\alpha}(t_1, t_2) = \prod_{i=t_1}^{t_2} \alpha_i$. Note that $\mathcal{H}_t$ is a function with respect to time t, while Ψ is a constant once we have defined t_1 and t_2.

Therefore, with the above derivations, the forward latent $\tilde{z}_t$ in the region $(t_1, t_2]$ can be expressed as

$$\tilde{z}_t = \sqrt{\bar{\alpha}_t} z_0^g + \sqrt{1-\bar{\alpha}_t}\epsilon_0 - (z_0^g - z_0^c)\ \Psi\ \mathcal{H}_t, \tag{34}$$

which is the Eq. (9) in the main body.

6.2. The Derivation of Eq. (15)

Based on the variational bound on negative log-likelihood (Eq. (5) of DDPM [1]), we learn $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t)$ as an approximation to the ground-truth denoising transition step $q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right)$, where $\tilde{z}_0 = z_0^g$. Once we obtain the forward latent $\tilde{z}_t$ of our diffusion model, we can derive $q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right)$ in region $(t_1, t_2]$ by using Bayes' theorem:

$$q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right) \quad = \quad \frac{q\left(\tilde{z}_t|\tilde{z}_{t-1}, \tilde{z}_0\right) q\left(\tilde{z}_{t-1}|\tilde{z}_0\right)}{q\left(\tilde{z}_t|\tilde{z}_0\right)}. \tag{35}$$

With Eq. (27) and Eq. (34), this equation can be further reduced to

$$\begin{aligned}
q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right) &= \frac{\mathcal{N}\left(\tilde{z}_t; \sqrt{\alpha_t}\tilde{z}_{t-1} - \Psi(z_0^g - z_0^c), (1-\alpha_t)I\right)\mathcal{N}\left(\tilde{z}_{t-1}; \sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c), (1-\bar{\alpha}_{t-1})I\right)}{\mathcal{N}(\tilde{z}_t; \sqrt{\bar{\alpha}_t}z_0^g - \mathcal{H}_t\Psi(z_0^g - z_0^c), (1-\bar{\alpha}_t)I)} \\
&\propto \exp\left\{-\frac{1}{2}\left[\frac{\left(\tilde{z}_t - \left(\sqrt{\alpha_t}\tilde{z}_{t-1} - \Psi(z_0^g - z_0^c)\right)\right)^2}{1-\alpha_t} + \frac{\left(\tilde{z}_{t-1} - \left(\sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c)\right)\right)^2}{1-\bar{\alpha}_{t-1}}\right.\right. \\
&\qquad \left.\left. - \frac{\left(\tilde{z}_t - \left(\sqrt{\bar{\alpha}_t}z_0^g - \mathcal{H}_t\Psi(z_0^g - z_0^c)\right)\right)^2}{1-\bar{\alpha}_t}\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[\left(\frac{\alpha_t}{1-\alpha_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)\tilde{z}_{t-1}^2\right.\right. \\
&\qquad \left.\left. -2\left(\frac{\sqrt{\alpha_t}\tilde{z}_t + \sqrt{\alpha_t}\Psi(z_0^g - z_0^c)}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c)}{1-\bar{\alpha}_{t-1}}\right)\tilde{z}_{t-1}\right]\right\} \\
&= \exp\left\{-\frac{1}{2}\left[\left(\frac{\alpha_t\left(1-\bar{\alpha}_{t-1}\right) + 1 - \alpha_t}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right)\tilde{z}_{t-1}^2\right.\right. \\
&\qquad \left.\left. -2\left(\frac{\sqrt{\alpha_t}\tilde{z}_t + \sqrt{\alpha_t}\Psi(z_0^g - z_0^c)}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c)}{1-\bar{\alpha}_{t-1}}\right)\tilde{z}_{t-1}\right]\right\} \\
&= \exp\left\{-\frac{1}{2}\left(\frac{1-\bar{\alpha}_t}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right)\right. \\
&\qquad \left.\left[\tilde{z}_{t-1}^2 - 2\frac{\left(\frac{\sqrt{\alpha_t}\tilde{z}_t + \sqrt{\alpha_t}\Psi(z_0^g - z_0^c)}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c)}{1-\bar{\alpha}_{t-1}}\right)\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\tilde{z}_{t-1}\right]\right\} \\
&= \exp\left\{-\frac{1}{2}\left(\frac{1-\bar{\alpha}_t}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right)\right. \\
&\qquad \left.\left[\tilde{z}_{t-1}^2 - 2\frac{\sqrt{\alpha_t}\left(\tilde{z}_t + \Psi(z_0^g - z_0^c)\right)\left(1-\bar{\alpha}_{t-1}\right) + \left(\sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c)\right)\left(1-\alpha_t\right)}{1-\bar{\alpha}_t}\tilde{z}_{t-1}\right]\right\} \\
&\propto \mathcal{N}\left(\tilde{z}_{t-1}; \mu_q\left(\tilde{z}_t, z_0^g, t\right), \Sigma_q\left(t\right)\right),
\end{aligned} \tag{36}$$

where $\mu_q\left(\tilde{z}_t, z_0^g, t\right)$ and $\Sigma_q\left(t\right)$ are defined as

$$\mu_q\left(\tilde{z}_t, z_0^g, t\right) = \frac{\sqrt{\alpha_t}\left(\tilde{z}_t + \Psi(z_0^g - z_0^c)\right)\left(1-\bar{\alpha}_{t-1}\right) + \left(\sqrt{\bar{\alpha}_{t-1}}z_0^g - \mathcal{H}_{t-1}\Psi(z_0^g - z_0^c)\right)\left(1-\alpha_t\right)}{1-\bar{\alpha}_t}, \tag{37}$$

$$\Sigma_q\left(t\right) = \sigma_t^2 I = \frac{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t} I. \tag{38}$$

Note that the variance remains the same as DDPM, since we only incorporate font transfer in the data sub-space and keep the noise sub-space unchanged. We further rewrite the mean function $\mu_q\left(\tilde{z}_t, z_0^g, t\right)$ as

$$\begin{aligned}\mu_q\left(\tilde{z}_t, z_0^g, t\right) &= \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\tilde{z}_t + \frac{\mathcal{H}_{t-1}\left(1-\alpha_t\right)-\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\Psi z_0^c \\ &+\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)\Psi+\left(\sqrt{\bar{\alpha}_{t-1}}-\mathcal{H}_{t-1}\Psi\right)\left(1-\alpha_t\right)}{1-\bar{\alpha}_t}z_0^g. \end{aligned} \tag{39}$$

In order to match $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t)$ to the ground-truth denoising transition step $q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right)$, we model $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t)$ as a Gaussian with $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t) = \mathcal{N}(\tilde{z}_{t-1}|\mu_\theta(\tilde{z}_t, t), \Sigma_\theta(\tilde{z}_t, t))$. We first set the variances of $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t)$ and $q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right)$ to match exactly, thus

$$\Sigma_\theta(\tilde{z}_t, t) = \sigma_t^2 I = \frac{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}I. \tag{40}$$

Once the variances of the two Gaussian match exactly, Ref. [2] further proves that optimizing the KL-divergence between $q\left(\tilde{z}_{t-1}|\tilde{z}_t, \tilde{z}_0\right)$ and $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t)$ reduces to minimize the difference between the means of these distributions. Therefore, following the common practice [1, 2], we utilize the prediction network $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t)$ to predict z_0^g, and $\mu_\theta\left(\tilde{z}_t, t\right)$ can be expressed as

$$\begin{aligned}\mu_\theta\left(\tilde{z}_t, t\right) &= \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\tilde{z}_t + \frac{\mathcal{H}_{t-1}\left(1-\alpha_t\right)-\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\Psi z_0^c \\ &+\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)\Psi+\left(\sqrt{\bar{\alpha}_{t-1}}-\mathcal{H}_{t-1}\Psi\right)\left(1-\alpha_t\right)}{1-\bar{\alpha}_t}\tilde{z}_\theta^{(g)}(\tilde{z}_t, t), \end{aligned} \tag{41}$$

which is the Eq. (15) in the main body.

7. Implement Details

In this section, we provide more implementation details about our multi-stage font generation model.

7.1. Multi-Stage Font Generative Process

We provide the pseudo-code of multi-stage font generative process in Algorithm 1.

Algorithm 1: Multi-Stage Font Generative Process

Input: The coefficients α_t, $\bar{\alpha}_t$, σ_t, $\mathcal{H}_t$, and Ψ. The latent feature $\boldsymbol{z}_0^c$ of the source image. The conditions $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ for font transfer stage and font refinement stage, respectively.

Output: The generated font image $\boldsymbol{I}_G$.

// step 1: Structure Construction Stage

$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$;

$\tilde{\boldsymbol{z}}_{t_2} = \sqrt{\bar{\alpha}_{t_2}}\boldsymbol{z}_0^c + \sqrt{1-\bar{\alpha}_{t_2}}\boldsymbol{\epsilon}$;

// step 2: Font Transer Stage

for $t = t_2$ **to** $t = t_1 + 1$ **do**

  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$;

  $\boldsymbol{\mu}_\theta = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\tilde{\boldsymbol{z}}_t + \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\Psi + \frac{(\sqrt{\bar{\alpha}_{t-1}}-\mathcal{H}_{t-1}\Psi)(1-\alpha_t)}{1-\bar{\alpha}_t}\right)\tilde{\boldsymbol{z}}_{\theta_1}^{(g,1)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_1) + \frac{[\mathcal{H}_{t-1}(1-\alpha_t)-\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})]}{1-\bar{\alpha}_t}\Psi\boldsymbol{z}_0^c$;

  $\tilde{\boldsymbol{z}}_{t-1} = \boldsymbol{\mu}_\theta + \sigma_t\boldsymbol{\epsilon}$;

// step 3: Font Refinement Stage

for $t = t_1$ **to** $t = 1$ **do**

  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ if $t > 1$, else $\boldsymbol{\epsilon} = \mathbf{0}$;

  $\boldsymbol{\mu}_\theta = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\tilde{\boldsymbol{z}}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\tilde{\boldsymbol{z}}_{\theta_2}^{(g,2)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_2)$;

  $\tilde{\boldsymbol{z}}_{t-1} = \boldsymbol{\mu}_\theta + \sigma_t\boldsymbol{\epsilon}$;

$\boldsymbol{z}_0^g = \tilde{\boldsymbol{z}}_0$;

return $\boldsymbol{I}_G = \mathcal{D}(\boldsymbol{z}_0^g)$

7.2. Optimization Process

We use a two-stage optimization process to train our proposed model. The pseudo-code of the optimization process is provided in Algorithm 2.

Algorithm 2: Optimization Process

// Stage 1-1: Optimize E_c^1, E_s^1, and $\tilde{\boldsymbol{z}}_{\theta_1}^{(g,1)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_1)$ in Region $(t_1, t_2]$

while *True* **do**
- Loading training pairs (I_s, I_c, I_g) by picking I_s and I_c based on I_g;
- Calculating condition $y_1 = \mathcal{C}(\mathcal{F}(e_c^1), \mathcal{F}(e_s^1))$ by flattening and concating $e_c^1 = E_c^1(I_c)$ and $e_s^1 = E_s^1(I_s)$;
- Calculating $z_0^g = \mathcal{E}(I_g)$ and $z_0^c = \mathcal{E}(I_c)$;
- Sampling $t \in (t_1, t_2]$ and calculating $\tilde{z}_t$ by Eq. (9);
- Using conditional prediction network $\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_t, t, y_1)$ to predict z_0^g;
- Using loss function Eq. (22) to train E_c^1, E_s^1, $\tilde{\boldsymbol{z}}_{\theta_1}^{(g,1)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_1)$ in an end-to-end manner;
- **Until Converged**;

// Stage 1-2: Optimize E_c^2, E_s^2, and $\tilde{\boldsymbol{z}}_{\theta_2}^{(g,2)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_2)$ in Region $(0, t_1]$

while *True* **do**
- Loading training pairs (I_s, I_c, I_g) by picking I_s and I_c based on I_g;
- Calculating condition $y_2 = \mathcal{C}(\mathcal{F}(e_c^2), \mathcal{F}(e_s^2))$ by flattening and concating $e_c^2 = E_c^2(I_c)$ and $e_s^2 = E_s^2(I_s)$;
- Calculating $z_0^g = \mathcal{E}(I_g)$ and $z_0^c = \mathcal{E}(I_c)$;
- Sampling $t \in (0, t_1]$ and calculating $\tilde{z}_t$ by Eq. (8);
- Using conditional prediction network $\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_t, t, y_2)$ to predict z_0^g;
- Using loss function Eq. (23) to train E_c^2, E_s^2, $\tilde{\boldsymbol{z}}_{\theta_2}^{(g,2)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_2)$ in an end-to-end manner;
- **Until Converged**;

// Stage 2: Using $\tilde{\boldsymbol{z}}_{\theta_1}^{(g,1)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_1)$ to Fine-tune E_c^2, E_s^2, and $\tilde{\boldsymbol{z}}_{\theta_2}^{(g,2)}(\tilde{\boldsymbol{z}}_t, t, \boldsymbol{y}_2)$ in Region $(0, t_1]$

while *True* **do**
- Loading training pairs (I_s, I_c, I_g) by picking I_s and I_c based on I_g;
- Calculating condition $y_1 = \mathcal{C}(\mathcal{F}(e_c^1), \mathcal{F}(e_s^1))$ by flattening and concating $e_c^1 = E_c^1(I_c)$ and $e_s^1 = E_s^1(I_s)$;
- Calculating $z_0^g = \mathcal{E}(I_g)$ and $z_0^c = \mathcal{E}(I_c)$;
- Sampling $t' \in (t_1, t_2]$ and calculating $\tilde{z}_{t'}$ by Eq. (9);
- Using conditional prediction network $\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_{t'}, t', y_1)$ to predict z_0^g;
- Sampling $t'' \in (0, t_1]$ and using the predicted z_0^g to calculate $\tilde{z}_{t''}$ by Eq. (8);
- Calculating condition $y_2 = \mathcal{C}(\mathcal{F}(e_c^2), \mathcal{F}(e_s^2))$ by flattening and concating $e_c^2 = E_c^2(I_c)$ and $e_s^2 = E_s^2(I_s)$;
- Using conditional prediction network $\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_{t''}, t'', y_2)$ to predict z_0^g;
- Using loss function Eq. (23) to fine-tune E_c^2, E_s^2, $\tilde{\boldsymbol{z}}_{\theta_2}^{(g,2)}(\tilde{\boldsymbol{z}}_{t''}, t'', \boldsymbol{y}_2)$ in an end-to-end manner;
- **Until Converged**;

7.3. Additional Details of Implementation and Optimization

We utilize the widely-used Stable Diffusion platform [3] to construct our proposed diffusion process in latent space. In the training stage, we obtain the latent feature $z_0^c \in \mathcal{R}^{16\times16\times4}$ and $z_0^g \in \mathcal{R}^{16\times16\times4}$ by using the VAE encoder $\mathcal{E}$ to project the source image $I_c \in \mathcal{R}^{128\times128\times3}$ and the target image $I_g \in \mathcal{R}^{128\times128\times3}$ into latent space, respectively. In the inference stage, the prediction network $\tilde{z}_{\theta_i}^{(g,i)}(\tilde{z}_t, t, y_i)$ will generate $z_0^g \in \mathcal{R}^{16\times16\times4}$ by gradually denoising $\tilde{z}_t$, and the generated font image I_G can be obtained by using the VAE decoder $\mathcal{D}$ to project z_0^g back to the image space. Following common practice in FFG, we construct the content encoder E_c^i and the style encoder E_s^i to generate the condition y_i. The style encoder E_s^i and the content encoder E_c^i have the same architecture, including 16 convolution layers and 3 down-sampling layers. We utilize the content encoder E_c^i and the style encoder E_s^i to extract the content feature $e_c^i \in \mathcal{R}^{16\times16\times1024}$ from the source image I_c and the style feature $e_s^i \in \mathcal{R}^{16\times16\times1024}$ from the reference images I_s, respectively. Finally, the content and style features are flattened and concatenated to build the condition $y_i \in \mathcal{R}^{512\times1024}$. Since the dimension of the condition y_i is 1024, we

slightly modify the cross-attention module in $\tilde{z}_{\theta_i}^{(g,i)}(\tilde{z}_t, t, y_i)$ to perform attention operation on the dimension 1024.

In this paper, we keep the VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$ frozen and train the remaining networks. Specifically, we utilize the loss function L_g^i (in Eq. (22) and (23)) to optimize the prediction network $\tilde{z}_{\theta_i}^{(g,i)}(\tilde{z}_t, t, y_i)$ with the corresponding content encoder E_c^i and style encoder E_s^i. The prediction network $\tilde{z}_{\theta_i}^{(g,i)}(\tilde{z}_t, t, y_i)$ is initialized by the pre-trained Stable Diffusion models, while the style encoder E_s^i and content encoder E_c^i are initialized by Kaiming initialization. The AdamW optimizer with the learning rate $lr = 3.2 \times 10^{-6}$ and batch size 16 is used to train our model.

8. Additional Experimental Results

8.1. Ablation Study on the Multi-Stage Font Generative Process

The selections of the region for different stages in the multi-stage generative process are hyper-parameters in our model, which significantly impact image quality. Therefore, we conduct experiments on UFUC dataset to study the influence on model performance by selecting different t_1 and t_2. As shown in Tab. 3, the configuration $t_1 = 200$ and $t_2 = 800$ achieves the best record in terms of RMSE, PSNR and SSIM. Moreover, comparing the results in the first two rows (No. 1 and No. 2), the performance of $t_1 = 100$ is slightly lower than $t_1 = 200$, since the font refinement stage of $t_1 = 100$ is too short to modify the image appearances and local details. Furthermore, for the experiments with the fixed t_1 and different t_2 (e.g., No. 2, 4, 5), we find that the larger font transfer stage tends to generate better results, as the network is easy to model the slow transformation process. However, when t_2 is too large (e.g., $t_2 = 900$ in No. 6), the structure construction stage cannot provide enough global structure information for the generative process.

Table 3. Ablation study for the regions of different stages on UFUC dataset. The bold number indicates the best.

No.	t_1	t_2	RMSE↓	PSNR↑	SSIM↑	LPIPS↓
1	100	800	0.2488	12.42	0.7131	0.1553
2	200	800	**0.2475**	**12.45**	**0.7189**	0.1527
3	300	800	0.2493	12.39	0.7155	0.1530
4	200	600	0.2602	12.03	0.7035	0.1708
5	200	700	0.2521	12.29	0.7138	0.1544
6	200	900	0.2496	12.40	0.7175	**0.1524**

Therefore, the hyper-parameters t_1 and t_2 are trade-off selections, and we select the configuration $t_1 = 200$ and $t_2 = 800$ in the following experiments.

8.2. Additional Qualitative Results

In this section, we provide more qualitative results of our proposed model.

Figure 6. Qualitative comparisons of our proposed model with other state-of-the-art methods on cross-lingual (Chinese to Korean) generation. Note that we use eight reference images in inference, and we only display one reference image in this figure.

Figure 7. Additional qualitative results of our proposed model on UFUC dataset.

Figure 8. Qualitative comparisons of our proposed model with other state-of-the-art methods on UFSC dataset.

Figure 9. Additional qualitative results of our proposed model on UFSC dataset.

Figure 10. Additional qualitative results of our proposed model for handwritten fonts in UFUC setting.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020. 3, 5, 12.
[2] Calvin Luo. Understanding diffusion models: A unified perspective. arXiv preprint arXiv:2208.11970, 2022. 13.
[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022. 3, 4, 5, 6, 14