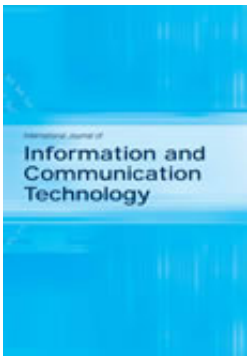
INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research



International Journal of Information and Communication Technology

ISSN online: 1741-8070 - ISSN print: 1466-6642
https://www.inderscience.com/ijict

Computer vision-driven automated generation and style simulation of calligraphic fonts

Kun Fu

DOI: 10.1504/IJICT.2025.10072940

Article History:

Received:	17 June 2025
Last revised:	08 July 2025
Accepted:	08 July 2025
Published online:	08 September 2025

Computer vision-driven automated generation and style simulation of calligraphic fonts

Kun Fu

School of Fine Arts and Design,
Changji University,
Changji 831100, China
Email: 13031742434@163.com

Abstract: As computer vision technology grows quickly, the automatic creation and stylistic replication of calligraphic fonts has become a major area of study in digital art and cultural heritage. This paper suggests a multi-task learning model that combines a convolutional neural network (CNN) and a generative adversarial network (GAN). The conditional generative adversarial network (cGAN) is used to do the style migration and to make high-quality calligraphic fonts automatically and to fine-tune the simulation of different styles. The method works better than several popular generation methods when it comes to image structure fidelity, style expressiveness, and perceptual consistency, showing that it has both artistic and practical significance. This paper's research gives new ideas and technical help for using computer vision to digitise traditional art.

Keywords: computer vision; calligraphic fonts; automated generation; style simulation.

Reference to this paper should be made as follows: Fu, K. (2025) 'Computer vision-driven automated generation and style simulation of calligraphic fonts', *Int. J. Information and Communication Technology*, Vol. 26, No. 32, pp.1–16.

Biographical notes: Kun Fu received his PhD from New Era University College, Malaysia in 2024. He is currently a full-time faculty member at the College of Fine Arts and Design, Changji University. His research interests include calligraphy culture, aesthetics and style simulation.

1 Introduction

As information technology grows quickly, traditional Chinese calligraphy, which is a cultural treasure, is facing new difficulties and chances. Calligraphy is more than just writing words; it is a deeply philosophical and artistic means to express oneself that shows the calligrapher's thoughts and feelings (Cao and Champadaeng, 2024). Calligraphy is a cultural language that goes beyond time and geography. It has a particular aesthetic value because of how the brush strokes, ink rhythm, structure, and other parts alter. But most individuals find it hard to master this art form because it takes a lot of time and practice to study and create calligraphy. At the same time, as society becomes more modern, it is becoming harder to promote and make traditional calligraphy popular. How to get around this problem so that more people may enjoy and appreciate the beauty of calligraphy is an important issue for the growth of art in modern society.

The main objective of this study is to not only come up with new ways to automatically create calligraphic typefaces, but also to help the process of making calligraphic art digital. With the help of computers, calligraphy can be passed on in the digital world and change to meet the needs of modern society. This opens more opportunities for calligraphy education, cultural exchange, and even business use.

The following are some ways in which this work is more original than previous research:

1 The proposal of multi-task learning model: this study creates a multitask learning model that can do both style simulation and automatic production of calligraphic typefaces at the same time. The model in this paper can not only automatically create beautiful calligraphic fonts based on what the user wants, but it can also imitate the styles of other calligraphers as it is making the fonts. This is how it achieves the integration function. This multi-task learning paradigm offers a novel way to make digital calligraphy.

2 Innovative application combining CNN and GAN: this study smartly blends CNN and GAN to make calligraphy fonts. CNN is in charge of getting deep-level features from calligraphy images so that the fonts it makes follow the structure and style rules of traditional calligraphy. GAN, on the other hand, keeps improving the results of the generation process by having the generator and the discriminator train each other in a way that makes the process more natural and beautiful. Not only does this combination improves the quality of the generation but also makes the model more durable.

3 Optimisation and innovation of style migration techniques: the classic style migration method can copy diverse art styles, but it does not always get the small changes between styles in the unique art form of calligraphy. This paper improves the style transformation process based on traditional CNN by introducing the optimised style migration technique. This makes the calligraphic works more delicate and natural in style transformation and better shows the unique qualities of calligraphy.

4 The realisation of cross-styled calligraphic font generation and style conversion: this paper's approach is different from many others because it looks at both the automated creation of calligraphic fonts and the modelling of styles at the same time. This new idea not only makes the model more flexible, but it also gives calligraphic art a wider range of ways to express itself.

The new ideas above give us a new way to think about how to digitally modify calligraphic art and open new possibilities for using computer vision technology in the art world.

2 Computer vision

Computer vision is a big part of AI that tries to let computers see and comprehend pictures and videos the same way that people do (Manovich, 2021). It comprises a number of activities, like recognising images, finding targets, segmenting images, generating images, and so on. Deep learning technology is growing quickly, and

computer vision research and use have made great strides. Deep learning approaches have shown to be very effective at tasks including picture classification, target recognition, and image synthesis. Computer vision is now used in a lot of different areas, such analysing medical images, driving cars automatically, and keeping an eye on security.

CNN is one of the most important technologies in the field of computer vision, and its use has helped computer vision make a lot of progress. By mimicking how the human visual system works, CNN automatically pulls information from incoming photos, processes them through a multi-layer network structure, and then sorts, finds, or creates them. The best thing about CNN is that it can automatically learn the local features of an image through convolutional processing, so you do not have to write complicated feature extraction techniques by hand (Hossain and Sajib, 2019).

A convolutional layer, a pooling layer, a fully connected layer, and an activation function are the most important parts of the CNN network construction. The convolutional layer uses convolutional procedures to find local features in the image, while the pooling layer uses downsampling to make the image smaller. This saves processing power and keeps important information in the image. The fully connected layer combines and sorts the features that were extracted. The activation function adds a nonlinear modification, which lets the network learn more complicated patterns. CNN's network structure not only makes image processing faster, but it also considerably increases the model's capacity to articulate itself. This allows deep learning to make big strides in computer vision.

CNN has been utilised for a lot of different vision jobs as computer vision technology has come a long way. AlexNet, VGGNet, and ResNet are examples of classical CNN models that have done quite well at tasks like image classification and target recognition (Wu, 2024). The advent of AlexNet, especially in the ImageNet picture classification challenge, signalled the start of the deep learning era and substantially sped up research in the field of computer vision. Faster R-CNN and other models can not only find objects in an image but also find their exact location. This is one of the basic technologies in the field of computer vision right now. CNN has also made great strides in the task of recognising faces.

In the realm of computer vision, GAN is also an essential approach, along with CNN. There are two elements to GAN, a generator and a discriminator. GAN is a popular tool in computer vision for things like making new images, changing the look of an image, and improving existing picture (Alqahtani et al., 2021). This technology is frequently utilised in art creation, image restoration, and data enhancement. The combination of CNN and GAN allows the generator to make images that are more detailed and realistic, which is a big step forward in the field of image production tasks.

CNNs are great at extracting and modelling visual features, whereas GANs are great at creating new data through adversarial training. Together, they work well. When you put the two together, you can solve some tough computer vision problems more easily. For instance, in image super-resolution reconstruction, GAN may make high-resolution images with a generator, and CNN can pull out the deep characteristics of the image to make sure the generated image is of good quality.

3 Calligraphy font generation and style simulation

In recent years, calligraphy font development and style simulation has become a more and more popular area of cutting-edge research that combines computer vision and artificial intelligence. The purpose of study in this area is not only to make typefaces that look good, but also to copy the styles of different calligraphers based on what is needed. This will help keep the art of calligraphy alive and new in the digital age. As deep learning and computer vision technology improve quickly, more and more algorithms have been used to create and style calligraphic fonts. This has led to many intriguing results (Al-Ayyoub et al., 2018).

First, most conventional approaches for making calligraphy fonts use rule-based generation models. By looking at and modelling existing calligraphic font examples, these technologies automatically make new fonts that follow specific guidelines. However, these rule-based models are generally not able to portray the deep artistic subtleties of calligraphic typefaces, and they are also not very flexible or creative, especially when it comes to showing the diverse styles of different calligraphers. So, in the last few years, more and more research has focused on learning-based generation methods, especially machine learning and deep learning, to make calligraphic typefaces that look good and can be customised.

Some old image processing technologies are also used to make calligraphic fonts and simulate different styles. For instance, image segmentation algorithms can cut apart different portions of calligraphic glyphs and make new fonts based on how these parts are put together (Miao et al., 2024). Image segmentation can help you get strokes and ink strokes out of a typeface so that you can better copy the writing style of a certain calligrapher in style simulation. Researchers can also utilise optimisation-based approaches to make calligraphic typefaces. For example, they can set optimisation goals like the balance and symmetry of the letters to make them seem better.

At the same time, several algorithms that use feature learning have also done well in this area. Deep learning and other machine learning methods can help feature learning algorithms automatically understand the qualities of calligraphy fonts, including the way the strokes curve and alter, and then use these features to make new fonts. According to this algorithm, creating calligraphy fonts can not only meet the requirements, but also show a unique artistic style with more creativity.

Still, there are certain problems with generating calligraphy fonts and simulating styles. First, the variety and intricacy of calligraphic typefaces make it hard for current generation models to capture all the artistic nuances, especially the smoothness of strokes and the way details are expressed. These models still cannot precisely imitate the artistic styles of calligraphers. Second, most of the current approaches for style migration and generation need a lot of training data, but making high-quality calligraphic font datasets is still quite hard (Iluz et al., 2023). So, how to make algorithms even more expressive and less reliant on data is still a very important issue in the field.

Even with these problems, using computer vision technology offers new ways to create calligraphic fonts and simulate styles. Computers can automatically pull out useful qualities from a lot of calligraphic font data to make better fonts that look good. These methods give new ways to make calligraphic art and give researchers a starting point for future work on making calligraphic fonts and simulating styles.

4 Model design for automated generation and style simulation of calligraphic fonts

4.1 Dataset and preprocessing

This study uses the CASIA-Online Chinese Handwriting Database, which is provided by the Institute of Automation of the Chinese Academy of Sciences (CASIA) and is all about studying Chinese handwriting fonts. The dataset has a lot of examples of calligraphic fonts and is good for making calligraphic fonts and simulating styles. Table 1 gives more information about this dataset:

Table 1 Information about CASIA-Online Chinese Handwriting Database

Dataset name	CASIA-Online Chinese Handwriting Database
Data source	Institute of Automation, Chinese Academy of Sciences (CASIA)
Dataset type	Handwritten font database
Number of samples	Over 10,00 Chinese character samples, including works from multiple calligraphers
Calligraphy styles	Includes regular script, running script, cursive script, clerical script, etc.
Data format	Image format (PNG, JPEG) with corresponding labels
Dataset features	Provides handwritten fonts in various calligraphy styles, suitable for font generation and style simulation research

Before training a model with the CASIA dataset, a number of exact preparation steps must be taken to make sure the data is of high quality and consistent. This will make model training more effective. Some of the specific procedures in preprocessing are scaling images, getting rid of noise, improving data, and processing labels.

The first step in preprocessing is to resize the image. All the photos in the dataset need to be the same size, which is 128 × 128 pixels (Ivanescu, 2022). This size keeps adequate details and makes sure that network processing is quick. We utilise the cv2.resize() method from the OpenCV package to make sure that all the input photos are the same size.

In image preprocessing, denoising is a key aspect. As calligraphy images may be affected by the shooting angle, scanning quality or writing irregularities, noise is often introduced. For this reason, we use Gaussian Blur to smooth the image, effectively removing high-frequency noise, preserving the font outline, and enhancing the learning effect of the subsequent model.

In order to enhance the generalisation ability of the model, we also performed data enhancement operations. The ImageDataGenerator in Keras is used to generate diverse training samples by rotating, cropping, scaling and flipping horizontally (Lalitha and Latha, 2022). This process not only enhances data diversity but also helps the model better adapt to different styles of calligraphy handwriting and reduces the risk of overfitting.

Finally, label processing is also a part that cannot be ignored. The character labels corresponding to each image are converted into one-hot encoding form so that the model can learn better classification. To do this, we make a dictionary that has all the possible characters for each character label. Then we use the np.zeros() and np.argmax() functions in NumPy to change the character labels into the right binary variables.

The preprocessed dataset gives the deep learning model high-quality inputs for training and makes the model better at generalising by adding more data and removing noise. This makes the task of generating calligraphic fonts and simulating styles go more smoothly.

4.2 Model design and implementation

To make sure that calligraphic fonts can be generated and styled automatically in a way that is both efficient and realistic, this model design has three main modules: feature extraction and modelling, generation and style simulation, and training and optimisation. The functions of each module are tightly related, and collectively they carry out the entire process, from entering data to creating the final typefaces.

4.2.1 Feature extraction and modelling

This module oversees getting useful feature information from the input handwritten font images so that the next steps in font development and style simulation can be based on it. The main way this module works is by using CNN, which can quickly pull out both local and global elements of the image, such the shape of the strokes, the thickness of the typeface, and the structure's regularity, from the original image through a multi-layer convolution operation (Li et al., 2024). The following equation can be used to define the convolution operation:

$$F_{out}(i,\ j) = \sigma\left(\sum_{m=1}^{k}\sum_{n=1}^{k} W_{m,n} \cdot X_{i+m,j+n} + b\right) \tag{1}$$

where $X_{i,j}$ is the pixel value of the input picture at position (i, j), $W_{m,n}$ is the weight of the convolution kernel, b is the bias term, σ is the activation function, and $F_{out}(i, j)$ is the feature map that comes out of the convolution operation. The formula demonstrates how the convolution kernel finds the convolution of the input picture in a certain area.

The pooling layer is also very critical for the feature extraction procedure, along with the convolution operation. The pooling layer makes the feature map smaller by downsampling it. It keeps the most important features in the image, cuts down on the amount of computation, and makes the model more stable. Maximum pooling is a standard way to accomplish pooling that picks the highest number in the pooling window. Its equation is:

$$P_{i,j} = \max_{m\in[i:i+k],n\in[j:j+k]}\left(F_{m,n}\right) \tag{2}$$

where $P_{i,j}$ is the feature map after pooling, $F_{m,n}$ is the value in the pooling window, and k is the size of the pooling window. The pooling process can efficiently lower the spatial dimension while keeping the most critical feature information which makes the network even more powerful (Akhtar and Ragavendran, 2020).

The feature extraction module is based on more than just convolution and pooling. It also incorporates batch normalisation and dropout to make the model train faster and better at generalising (Garbin et al., 2020). Batch normalisation guarantees that the input data has a constant distribution by normalising it at each layer. This stops the problem of gradient vanishing during training, as seen in equation:

$$\hat{X} = \frac{X - \mu}{\sigma} \cdot \gamma + \beta \tag{3}$$

where X is the input data, μ and σ are the mean and standard deviation of the input, γ and β are the learnable scaling and translation parameters, and $\hat{X}$ is the normalised output. Batch normalisation makes the network's training much faster and more stable.

Lastly, in the feature extraction stage, the final output feature maps are flattened and transmitted to the fully connected layer for more processing after the procedures of convolution, pooling, and normalisation. The fully connected layer's job is to turn the high-dimensional features taken from the image into feature variables that can be differentiated. This is the first step in the tasks of font synthesis and style simulation. The model can gradually get global information from the local details of the image and get a feature representation that can accurately show the calligraphic typefaces by designing these layers.

4.2.2 Font generation and style simulation

This module uses GAN to create font graphics that look like calligraphy and allow for migration between different styles. This module can make delicate and personalised calligraphy fonts by using GAN's generation and adversarial mechanisms and CNN's feature extraction. This adds to the variety of font styles and ways of expressing art.

The GAN has two parts: the generator and the discriminator (Zhang et al., 2021). The generator makes samples that look like real font pictures, and the discriminator checks to see if the created images are real. The generator and the discriminator are taught against each other to improve their tactics over time. We can use the following equation to show what the generator does:

$$G(z, y) = \text{Generator}\left(z, y; \theta_G\right) \tag{4}$$

where z is random noise or other possible variables, y is a style label that shows the intended style, $G(z, y)$ is the font image that the generator makes, and θ_G is a parameter of the generator.

The discriminator's job is to figure out if the input image is from a real dataset or not. It does this by giving an output value $D(x)$, which is the chance that the image is a real image. The following is the output formula for the discriminator:

$$D(x) = \sigma\left(W_D \cdot x + b_D\right) \tag{5}$$

where x is the input image, W_D and b_D are the weight and bias of the discriminator, respectively.

Here is the output expression of the conditional generator. The conditional generator's output expression is as follows:

$$\hat{x} = G\left([z, y]; \theta_G\right) \tag{6}$$

where y is a style tag that shows different styles, such regular script, running script, and so forth. Depending on the style tags, the generator creates the right font images. This design gives the generator more control over how font styles are made, so that the fonts it

makes not only follow the rules of calligraphy but can also work with a lot of diverse calligraphic styles.

The discriminator, on the other hand, makes the generator's output better by comparing the genuine image to the one that was made (Nie et al., 2018). Adversarial training helps the generator, and the discriminator work together to make realistic font images in a range of calligraphic styles. The goal of training the discriminator is to make the judgement of the created picture as accurate as possible.

4.2.3 Training and optimisation

The main purpose of adversarial training is to teach the generator how to make bogus images and the discriminator how to tell them apart. The two then play a game to help each other get better. This optimisation process's main goal function may be written as:

$$\min_{G}\max_{D} V(D, G) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{z,y}[\log(1 - D(G(z, y), y))] \tag{7}$$

where x is the genuine image, y is the style label, and z is a random noise variable taken from a Gaussian distribution. The generator's job is to lower the overall loss, while the discriminator's job is to raise the function (Pan et al., 2020). This creates a closed-loop structure for adversarial training.

A style consistency loss is also included in the training to make the style representation even more accurate. This loss helps the generator better capture style features while keeping the structure stable by looking at the discrepancies between the generated image and the target style samples in the deep feature space. Loss is defined as this:

$$L_{style} = \left\| \phi(G(z, y)) - \phi\left(x_{target}\right) \right\|_2^2 \tag{8}$$

where $\phi(\cdot)$ is the style feature mapping taken from the pre-trained CNN network, and x_{target} is the reference picture of the style that goes with it.

In short, this module effectively combines structural and stylistic aspects using an acceptable adversarial training mechanism and style guiding technique. This gives the final generation effect a strong training base. Algorithm 1 explains how to put the model into action:

Algorithm 1 Pseudo-code for calligraphy generation and style simulation model

```
begin
    Initialise generator G parameters θ_G, discriminator D parameters θ_D;
    Initialise optimiser with learning rate α;
    for t = 1 to N do
        Sample real images x, style labels y from dataset D;
        Sample noise vector z from standard Gaussian distribution;
        // Feature extraction phase
        RealStructFeat = CNN_encoder(x);
        StyleInfo = Style_embedding(y);
        // Generation phase
```

FakeImg = *G*(*z*, *StyleInfo*);
// Discrimination phase
D_real = *D*(*x*, *y*);
D_fake = *D*(*FakeImg*, *y*);
// Loss calculation
L_adv = log(*D_real*) + log(1 – *D_fake*);
L_gen = *L_adv*;
// Backpropagation and update for discriminator
Grad_D = compute gradients of *L_adv* w.r.t θ_D;
$\theta\_D \leftarrow \theta\_D - \alpha \cdot Grad\_D$;
// Backpropagation and update for generator
Grad_G = compute gradients of *L_gen* w.r.t θ_G;
$\theta\_G \leftarrow \theta\_G - \alpha \cdot Grad\_G$;
// Optional: learning rate scheduler
if *t* % 1,000 == 0 **then**
Adjust learning rate α;
end if
end for
return trained *G*(θ_G), trained *D*(θ_D);
end

5 Experiment and result analysis

5.1 Experimental setting and evaluation indicators

This paper creates a systematic experimental process to check how well the model works for automatically generating and simulating the style of calligraphic fonts. The process includes data partitioning, training configuration, comparison model selection, and a full performance evaluation. The experiments take place in Ubuntu 22.04, on an NVIDIA RTX 4090 GPU, using PyTorch 2.1.0 as the software framework (Nalini et al., 2025). The model parameters are updated in mini batches during training, with a batch size of 32 and an initial learning rate of 0.0002 for the Adam optimiser. The training process lasts for 200 rounds (Chen et al., 2019).

We split the dataset into training, validation, and test sets in the ratio of 8:1:1 to make sure that each style category was evenly distributed. We also normalised and centre-aligned each character sample.

To fully evaluate the quality and style simulation ability of the model-generated font images, this paper presents three evaluation indexes. These indexes are quantitatively analysed in terms of several dimensions, including image structural fidelity, stylistic expressiveness, and perceptual consistency.

The structural similarity index (SSIM) is a way to compare the brightness, contrast, and structure of the generated image to the target image. It is defined as:

$$SSIM(x,\hat{x}) = \frac{(2\mu_x\mu_{\hat{x}} + C_1)(2\sigma_{x\hat{x}} + C_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + C_1)(\mu_x^2 + \mu_{\hat{x}}^2 + C_2)} \tag{9}$$

where μ is the mean, σ is the variance, is the covariance, C_1 and C_2 are constants that make sure the denominator is never zero. The original SSIM value is between 0 and 1, and the closer it is to 1, the closer the structure is (Sara et al., 2019).

Fréchet inception distance (FID) finds the Fréchet distance between the generated image and the real image by taking the feature distribution of the inception network at a specific layer (Buzuti and Thomaz, 2023). This work changes FID into a uniform assessment direction to get rid of the problem of the numerical scale being too big:

$$FID = 1 - \frac{FID - FID_{\min}}{FID_{\max} - FID_{\min}} \tag{10}$$

where $FID_{\min}$ and $FID_{\max}$ are the lowest and highest FID values that were seen in the experiment. The bigger value of FID after processing means that the image that was made is more like the real image distribution.

Content perceived difference (CPD) looks at the difference in perception between the generated image and the target image (Kim et al., 2024). It does this by looking at the multilayer features derived from pre-trained CNNs, with a focus on the fine textures and local structures. CPD shows how the human eye prefers font style and stroke fineness over changes at the pixel level. This is what it means:

$$CPD(x,\hat{x}) = 1 - \frac{1}{L}\sum_{l=1}^{L} \frac{\left\|\phi_l(x) - \phi_l(\hat{x})\right\|_2}{\left\|\phi_l(x)\right\|_2 + \varepsilon} \tag{11}$$

where $\phi_l(\cdot)$ is the feature extractor for the l^{th} layer of the pre-trained CNN, L is the number of perceptual layers you want to use, and ε is a tiny constant that is included to prevent division by zero. This rating ranges from 0 to 1, and higher values mean that the style is better preserved, and the variations are less noticeable.

When put together, the metrics above can show how well the model does at generating fonts from different angles without needing to be scored by hand. SSIM and CPD work together to capture the connection between structural and stylistic characteristics, especially in the style simulation job. FID, on the other hand, gives a statistical picture of the overall quality.

5.2 Experimental process and results

Experiment 1 focuses on quantitatively comparing the fundamental parameters of structural integrity, textural details, and overall perceptual quality of character pictures in order to confirm the proposed model's basic generative capabilities in the calligraphic font production assignment. For comparison purposes, three typical generation models are chosen as baseline controls.

First, we employ the classic convolutional self-encoder (baseline-AE) as the basis generative framework. An encoder compresses the input image into low-dimensional features, and then a decoder reconstructs the image. This method is straightforward to set up and train, but it does not work as well for reducing complicated details or generating

high-quality images. For example, it is extremely hard to capture the rich stroke texture and structural characteristics of calligraphic typefaces.

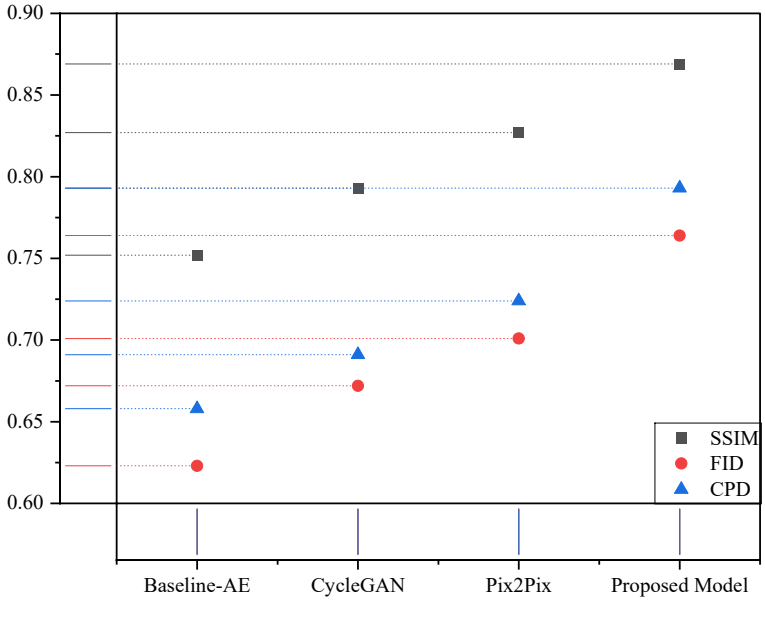
Second, CycleGAN is a method for changing the domain of images without supervision (Jiangsha et al., 2022). It uses generative adversarial networks to change the style of images by adding cyclic consistency loss. It does not need paired training data and works well for converting between styles, however for some font production jobs, the pictures it makes often have blurry borders and local deformation because there are not any strict structural limits.

Finally, Pix2Pix is a famous example of a conditional generative adversarial network (Rau et al., 2019). It uses paired picture data for end-to-end training and can better understand the relationship between input and output. For font generation, Pix2Pix can employ conditional information to make the generation more accurate and detailed. However, there is still a performance bottleneck when it comes to the wide range of style adjustments that can be made to calligraphic typefaces.

These three models show distinct technological pathways, such as classic encoder, unsupervised style migration, and conditional adversarial training. They give a full picture of the models' strengths in this study.

We used the same training set to train all the models and the same test set to test them. The obtained results are image centre-aligned and size-normalised so that the multiple model outputs can be compared. Figure 1 shows the average score of the four models in the test set based on the three metrics:

Figure 1 Performance comparison of different models in font generation task (see online version for colours)



The graphic shows that the proposed approach makes a big difference in all three measures. The SSIM score, which is 0.869, is about 12% higher than the baseline-AE score of 0.752. This shows that the model is quite good at keeping the font's structure

intact. This shows that the model can accurately capture and recreate the complicated stroke patterns of Chinese characters, making sure that the basic shape of the font it creates is clear and easy to understand.

The model also gets the best score of 0.764 on the FID metric, which is much higher than CycleGAN's 0.672 and Pix2Pix's 0.701. This shows that the generated fonts are considerably more like genuine data when it comes to feature distribution. This is another evidence that the model is better at restoring detail texture and overall style. The rise in the CPD index (0.793) shows that the generated fonts are more consistent in the multi-layer perceptual feature space. This means that the model can recover the subtle changes and artistic texture of the calligraphic strokes in a more detailed fashion.

The findings of experiment 1 showed that the model is not only good at modelling the structure of fonts, but also at restoring perceptual quality and stylistic detail. To fully test the proposed model's performance in the multi-style calligraphic font simulation task, experiment 2 uses a test set that includes several common calligraphic styles, like regular, running and cursive. This is done to see how well the model can convert between styles and restore details. The goal of the studies is to see if the model can capture and express complicated calligraphic styles. This will show that it has the best of both worlds: style migration and multi-task creation.

In this experiment, we compare seven well-known and widely applied multi-style image generation and transfer models.

One of them is CycleGAN, an unsupervised method that uses cycle consistency loss to convert images between two domains without requiring paired data. Another is StarGAN, which allows for image translation across multiple domains, making it easier to switch between different calligraphy styles. StarGAN v2 is an improved version that enhances style expression through additional style encoding and a more precise style control mechanism (Barzilay et al., 2021). Both multimodal unsupervised image-to-image translation (MUNIT) and diverse image-to-image translation (DRIT) may change styles clearly and evident. They do this by separating content and style in an implicit space. On the other hand, style-aware GAN adds a style-aware module to the model to help it capture more intricate textures and features. Finally, AttnGAN adds an attention mechanism that makes the model better at focusing on small details in an image and does well at generating fine-grained images. The models above show different technical approaches to current multi-style image generation. They include unsupervised learning, multi-domain transformation, style separation, and attention mechanism. They provide a full and strong reference for comparing the models' performance in this study.

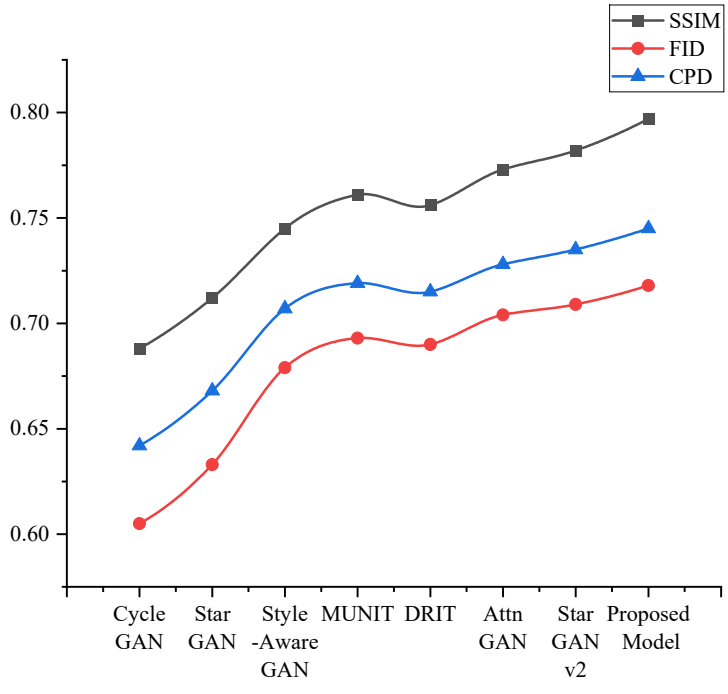
Figure 2 shows the outcomes of the experiment.

The figure shows that the model suggested in this study does the greatest job on all three evaluation metrics. This fully shows that it is the best at mimicking multi-style calligraphic typefaces. The current model has an SSIM score of 0.797, which is much higher than the second-ranked StarGAN v2 (0.782) and AttnGAN (0.773). This means that the current model successfully combines several calligraphic style elements while keeping the fonts' structure and clarity, and the typefaces it creates follow writing rules and have significant artistic expression.

Second, the FID shows how comparable the distribution of stylistic elements is between the created fonts and the real data. The proposed model gets the highest score of 0.718, which is better than all the other models that were compared. This means that the typefaces made by the model are more like real calligraphy samples in terms of how they are distributed generally, and they can change styles more naturally and smoothly.

CycleGAN, on the other hand, has the lowest FID value because it does not have a way to modify style, which makes the style profile of the converted fonts less clear.

Figure 2 Comparison of the performance of multi-style calligraphic font simulation models (see online version for colours)



The CPD metric backs up the model's claim that it is better at capturing calligraphic details. With a score of 0.745, this model is better than style-aware GAN (0.707) and other baseline models. This shows that it can better reproduce the fine details of stroke texture and style. This is very important for calligraphic fonts, which are an art style that needs a lot of detail to show.

Figure 3 Simulation generation of official script style (see online version for colours)

In short, the experimental results show that the current model is far better at simulating multi-style calligraphy fonts than other models. They also show that integrating multi-task learning, optimised style migration, and detail capturing techniques works.

To prove that the model can model dynamic strokes even more, we use the simulation of the clerical and cursive styles as an example (see Figurea 3 and 4). The model separates the temporal features of the two fonts into the probability of stroke distributions in the spatial domain using cGAN and multilayer convolutional feature extraction.

Figure 4 Simulation generation of cursive style (see online version for colours)

6 Conclusions

This study methodically designs and implements a multi-task learning model that combines CNN and GAN to solve the challenge of automatically generating and simulating the style of calligraphic typefaces using computer vision technologies. This paper combines the current state of research in related fields to show the problems with traditional methods for capturing style details and converting between styles. It then suggests a model architecture made up of three modules: feature extraction, generation and style simulation, and training optimisation. The model can not only make fonts that are clear and follow calligraphic rules, but it can also depict different styles and textures of strokes in multi-style simulation. In the experimental phase, two systematic tests are done to compare the most common generation models using three measures: SSIM, FID, and CPD. The results reveal that the model in this work has greatly improved the quality of the fonts and the performance of several styles. This proves that its technological route is both legitimate and useful.

Even if the model in this research has several good points, it still has some problems. First, the quality and variety of the dataset are very important for the training process. The current dataset, on the other hand, only covers a few styles and has a limited number of samples, which could make it harder for the model to generalise. Second, the model might still do better with complicated styles or details, and the restoration of some fine strokes and hyphenated styles is still not accurate enough. Also, the model needs a lot of computing power and takes a long time to train, which makes it hard to quickly use and promote in real-world situations.

To fix the problems listed above, future research can go in the following directions: first, to make calligraphic style datasets that are bigger and more varied, covering more

calligraphic genres and personalised styles. This will make the model more adaptable and expressive. Second, investigate lightweight network structures and efficient training methods to make the model less complicated and speed up the process of generating new styles and converting existing ones (Gendy et al., 2023). Third, the detailed expression mechanism of style migration technology is looked into in more depth, and the accuracy and naturalness of style conversion are improved even more by using new technologies. Finally, we create an intelligent calligraphy font generation system that gives real-time feedback and style regulation functions, considering user interaction and personalised customisation needs. This system will help computer vision technology become more widely used in the creation and preservation of digital calligraphy.

In short, this paper gives us a new way to do things and a way to test them out when it comes to automatically creating and simulating calligraphic fonts. This not only adds to the possible uses of computer vision, but it also sets the stage for the digital innovation of traditional art. We hope that future research will continue to push this field forward.

Declarations

The author declares that he has no conflicts of interest.

References

Akhtar, N. and Ragavendran, U. (2020) 'Interpretation of intelligence in CNN-pooling processes: a methodological survey', *Neural Computing and Applications*, Vol. 32, No. 3, pp.879–898.

Al-Ayyoub, M., Nuseir, A., Alsmearat, K., Jararweh, Y. and Gupta, B. (2018) 'Deep learning for Arabic NLP: a survey', *Journal of Computational Science*, Vol. 26, pp.522–531.

Alqahtani, H., Kavakli-Thorne, M. and Kumar, G. (2021) 'Applications of generative adversarial networks (GANs): an updated review', *Archives of Computational Methods in Engineering*, Vol. 28, pp.525–552.

Barzilay, N., Shalev, T.B. and Giryes, R. (2021) 'MISS GAN: a multi-illustrator style generative adversarial network for image to illustration translation', *Pattern Recognition Letters*, Vol. 151, pp.140–147.

Buzuti, L.F. and Thomaz, C.E. (2023) 'Fréchet autoencoder distance: a new approach for evaluation of generative adversarial networks', *Computer Vision and Image Understanding*, Vol. 235, p. 103768.

Cao, H. and Champadaeng, S. (2024) 'The art of Chinese calligraphy: educational protection and literacy study of cultural heritage', *International Journal of Education and Literacy Studies*, Vol. 12, No. 3, pp.160–171.

Chen, Y., Sun, X. and Jin, Y. (2019) 'Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation', *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 31, No. 10, pp.4229–4238.

Garbin, C., Zhu, X. and Marques, O. (2020) 'Dropout vs. batch normalization: an empirical study of their impact to deep learning', *Multimedia Tools and Applications*, Vol. 79, No. 19, pp.12777–12815.

Gendy, G., He, G. and Sabor, N. (2023) 'Lightweight image super-resolution based on deep learning: state-of-the-art and future directions', *Information Fusion*, Vol. 94, pp.284–310.

Hossain, M.A. and Sajib, M.S.A. (2019) 'Classification of image using convolutional neural network (CNN)', *Global Journal of Computer Science and Technology*, Vol. 19, No. 2, pp.13–14.

Iluz, S., Vinker, Y., Hertz, A., Berio, D., Cohen-Or, D. and Shamir, A. (2023) 'Word-as-image for semantic typography', *ACM Transactions on Graphics (TOG)*, Vol. 42, No. 4, pp.1–11.

Ivanescu, R.C. (2022) 'A statistical evaluation of the preprocessing medical images impact on a deep learning network's performance', *Annals of the University of Craiova-Mathematics and Computer Science Series*, Vol. 49, No. 2, pp.411–421.

Jiangsha, A., Tian, L., Bai, L. and Zhang, J. (2022) 'Data augmentation by a CycleGAN-based extra-supervised model for nondestructive testing', *Measurement Science and Technology*, Vol. 33, No. 4, p.045017.

Kim, D., Nam, S-W., Choi, S., Seo, J-M., Wetzstein, G. and Jeong, Y. (2024) 'Holographic parallax improves 3D perceptual realism', *ACM Transactions on Graphics (TOG)*, Vol. 43, No. 4, pp.1–13.

Lalitha, V. and Latha, B. (2022) 'A review on remote sensing imagery augmentation using deep learning', *Materials Today: Proceedings*, Vol. 62, pp.4772–4778.

Li, Z., Zheng, Y., Shan, D., Yang, S., Li, Q., Wang, B., Zhang, Y., Hong, Q. and Shen, D. (2024) 'Scribformer: transformer makes cnn work better for scribble-based medical image segmentation', *IEEE Transactions on Medical Imaging*, Vol. 43, No. 6, pp.2254–2265.

Manovich, L. (2021) 'Computer vision, human senses, and language of art', *AI & Society*, Vol. 36, No. 4, pp.1145–1152.

Miao, Y., Jia, H. and Tang, K. (2024) 'Artistic font generation network combining font style and glyph structure discriminators', *Multimedia Tools and Applications*, Vol. 83, No. 8, pp.21883–21903.

Nalini, J., Rekha, K.S., Triveni, K., Tanusri, K., Mounika, N. and Jyothsna, K. (2025) 'Temporal attention-augmented BiLSTM with meta-ensemble voting for predicting prescription toxicity in opioid medication', *Synthesis: A Multidisciplinary Research Journal*, Vol. 3, No. 1s, pp.49–59.

Nie, D., Trullo, R., Lian, J., Wang, L., Petitjean, C., Ruan, S., Wang, Q. and Shen, D. (2018) 'Medical image synthesis with deep convolutional adversarial networks', *IEEE Transactions on Biomedical Engineering*, Vol. 65, No. 12, pp.2720–2730.

Pan, Z., Yu, W., Wang, B., Xie, H., Sheng, V.S., Lei, J. and Kwong, S. (2020) 'Loss functions of generative adversarial networks (GANs): opportunities and challenges', *IEEE Transactions on Emerging Topics in Computational Intelligence*, Vol. 4, No. 4, pp.500–522.

Rau, A., Edwards, P.E., Ahmad, O.F., Riordan, P., Janatka, M., Lovat, L.B. and Stoyanov, D. (2019) 'Implicit domain adaptation with conditional generative adversarial networks for depth prediction in endoscopy', *International Journal of Computer Assisted Radiology and Surgery*, Vol. 14, pp.1167–1176.

Sara, U., Akter, M. and Uddin, M.S. (2019) 'Image quality assessment through FSIM, SSIM, MSE and PSNR – a comparative study', *Journal of Computer and Communications*, Vol. 7, No. 3, pp.8–18.

Wu, Z. (2024) 'Application of CNN classic model in modern image processing', *Journal of Advances in Engineering and Technology*, Vol. 1, No. 3, pp.1–6.

Zhang, H., Yuan, J., Tian, X. and Ma, J. (2021) 'GAN-FM: infrared and visible image fusion using GAN with full-scale skip connection and dual Markovian discriminators', *IEEE Transactions on Computational Imaging*, Vol. 7, pp.1134–1147..