EUROGRAPHICS 2024 / A. Bermano and E. Kalogerakis
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 2

FontCLIP: A Semantic Typography Visual-Language Model for Multilingual Font Applications

Yuki Tatsukawa[1] I-Chao Shen[1] Anran Qi[1] Yuki Koyama[2] Takeo Igarashi[3] Ariel Shamir[4]

[1] {tatsukawa-yuki537, ichaoshen, annranqi1024}@g.ecc.u-tokyo.ac.jp , The University of Tokyo, Japan
[2] koyama.y@aist.go.jp, National Institute of Advanced Industrial Science and Technology (AIST), Japan
[3] takeo@acm.org, The University of Tokyo, Japan
[4] arik@runi.ac.il, Reichman University, Israel

Abstract

Acquiring the desired font for various design tasks can be challenging and requires professional typographic knowledge. While previous font retrieval or generation works have alleviated some of these difficulties, they often lack support for multiple languages and semantic attributes beyond the training data domains. To solve this problem, we present FontCLIP – a model that connects the semantic understanding of a large vision-language model with typographical knowledge. We integrate typography-specific knowledge into the comprehensive vision-language knowledge of a pretrained CLIP model through a novel finetuning approach. We propose to use a compound descriptive prompt that encapsulates adaptively sampled attributes from a font attribute dataset focusing on Roman alphabet characters. FontCLIP's semantic typographic latent space demonstrates two unprecedented generalization abilities. First, FontCLIP generalizes to different languages including Chinese, Japanese, and Korean (CJK), capturing the typographical features of fonts across different languages, even though it was only finetuned using fonts of Roman characters. Second, FontCLIP can recognize the semantic attributes that are not presented in the training data. FontCLIP's dual-modality and generalization abilities enable multilingual and cross-lingual font retrieval and letter shape optimization, reducing the burden of obtaining desired fonts.

1. Introduction

Acquiring a suitable font is a crucial step in many design workflow, especially when designing a poster or a banner with cross-lingual characteristics. While previous works have facilitated font retrieval [OLAH14, CWX*19] and generation [WGL20, WL21], they are often limited to the languages and attributes presented in the training data. The available datasets [OLAH14, CWX*19] only include Roman fonts and their associated attributes, which does not allow users to obtain fonts in other languages. Furthermore, the current datasets have limited annotated attributes, which prevents users from specifying their desired fonts freely.

In this paper, we tackle these deficiencies by defining a semantic latent space connecting language and visuals to *typography* to enable multilingual and cross-lingual font retrieval and editing tasks. We base our technique on modifying a pretrained vision-language model trained on large-scale natural image and text pair, and without requiring any additional data-gathering beyond what is already available. Models such as CLIP (Contrastive Language–Image Pre-training) [RKH*21], have demonstrated exceptional capabilities in learning aligned visual and language features. CLIP has exhibited remarkable zero-shot recognition capabilities, empowering a wide range of downstream visual recognition tasks [KCG*23, ZZL*22, LBW*22, ZLD22]. Additionally, the latent space of CLIP has been used for various content generation applications, including images [RDN*22], abstract sketches [VPB*22], 3D avatars [HZP*22], and artistic images [RBL*21]. These works collectively highlight that CLIP's latent space carries profound semantic understanding that can connect between language and visuals.

However, typography is a very specialized domain that is different from natural photographs, paintings, or sketches, which were originally used to train CLIP. Hence, simply using the original CLIP model cannot effectively recognize visual typographic characteristics and establish meaningful connections with language representations, as illustrated in Figure 2. The primary reason for this is the substantial domain disparity between the typographical image and those portraying natural scenes. Moreover, the language used to describe typographic data often diverges from the descriptions of natural scenes represented in different styles.

We present *FontCLIP*, a CLIP-based model specifically designed to learn a semantic typographic latent space that bridges language and visual typographic attributes, enabling various typographic applications. By finetuning a pretrained CLIP model on font data with attribute scores, we enhance its zero-shot recognition capability and enable it to generalize to the typography domain. The learned features of FontCLIP enable prediction of multilingual visual typographic attributes with the ability to generalize to *out-of-domain*

© 2024 Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd.

Authors' version

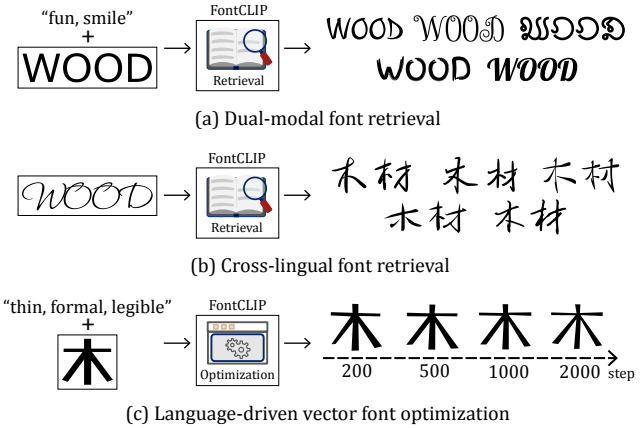


Figure 1: FontCLIP enables the following typography-specific applications. (a) **Dual-modal font retrieval**: our method retrieves results that preserve the style of the query font image (text with frame) while incorporating the desired attributes. (b) **Cross-lingual font retrieval**: our method retrieves results in other languages with a similar query font image style. (c) **Language-driven vector font optimization**: our method manipulates the shape of input letters aligning with a set of desired attributes.



Figure 2: Glyph images of Roman and Chinese letters sorted by (a) "complex, italic", (b) "strong", (c) "warm", and (d) "thin" attribute scores predicted by CLIP and FontCLIP. FontCLIP's sorting aligns more closely with human perception.

attributes. Remarkably, we achieve these generalizations by using an existing Roman character dataset without the need for collecting any new data.

Finetuning FontCLIP is accomplished using a novel compound descriptive prompt that encapsulates multiple attributes within a single prompt. To determine the attributes included in the compound descriptive prompt, we adaptively sample them to cover the distribution of attribute scores and convert continuous score into sampled text. In our finetuning process, we use a randomly generated compound descriptive prompt and a font image with a random augmentation transformation at each iteration, thereby significantly expanding the original font attributes dataset [OLAH14].

We evaluated the performance of FontCLIP through quantitative experiments that assessed the correlation between the predicted and manually annotated attribute scores. Our experiments reveal that the finetuned FontCLIP model, originally trained on Roman alphabet characters with 37 attributes, exhibits unprecedented generalization capabilities. First, FontCLIP is capable of generalizing to *out-of-domain* languages, which makes it possible to use it for multilingual and cross-lingual font-related tasks. Second, it can generalize to *out-of-domain* attributes, which means it can be used for font retrieval and editing using language descriptions beyond the original attribute set. Lastly, by leveraging the dual-modality of CLIP, FontCLIP allows users to obtain the desired fonts through both desired text attributes and font image examples.

We demonstrate FontCLIP in two major applications: (1) a novel dual-modal font retrieval interface that surpasses the traditional drop-down list interface in terms of user satisfaction and achieves similar performance without using vector-based typographical features extracted from all Roman characters and, (2) a novel optimization framework that utilizes FontCLIP latent space to manipulate vector letter shapes based on desired attributes or font image examples, thereby opening up exciting possibilities for font customization (see Figure 1).

To sum up, we make the following contributions:

- To the best of our knowledge, we present the first visual-language model that learns a semantic typographic latent space. Through experiments and user studies, we validate its generalization abilities over multilingual and *out-of-domain* attributes.
- We present a novel approach to finetune a vision-language model using font data with attribute scores.
- We present a dual-modal font retrieval application based on FontCLIP that uses visual examples and language descriptions to search for appropriate fonts across different languages.
- We present an optimization-based method to modify the shape of letters in vector representation to better match either a set of language descriptions or a visual input image sample.

2. Related Work

2.1. Vision-Language Representations and Applications

Traditional visual recognition models are often constrained in their practicality since they are trained to recognize a predetermined set of object categories. Hence, they require additional labeled data to generalize to new visual concepts and domains. However, recent advancements in large vision-language models pretrained on vast image-text pairs have demonstrated that such models can acquire rich image and object-level visual representations [RKH*21, JYX*21, LZZ*22]. These models are semantically rich because the paired texts contain a broader set of visual concepts than any pre-defined concept set. Thus, the learned representations can be directly used for downstream image recognition tasks such as image classification [RKH*21, JYX*21], object recognition [KCG*23], image segmentation [ZZL*22, LBW*22, ZLD22], and text-image retrieval [LROTG21] in zero-shot setting. Moreover, the learned representations are applied to 3D shape classification [ZGZ*22],

© 2024 Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd.

3D part segmentation [LZC*23], 3D avatar and shape generation [HZP*22, MBOL*22, MKXBP22, GAG*23, TGH*22] and NeRF generation and manipulation [JMB*22, WCH*22] tasks. To the best of our knowledge, we propose the first semantic typography visual-language model that connects language and visual typographic attributes. By doing so, FontCLIP enables font retrieval and manipulation tasks with a broader range of semantic concepts, expanding the possibilities for font customization and exploration.

2.2. Font Retrieval and Interface

Font selection is the process of selecting fonts from a set of fonts based on user-specified conditions across different formats. When the desired fonts are presented as images, traditional visual font recognition approaches identify the typeface, weight, and slope of text within them [WYJ*15, CYJ*14]. In the context of graphic design, users frequently aim to find a font that complements the overall design elements. As a result, they often rely on traditional font selection interface, such as a long list of font names, which is overwhelming to navigate and utilize. To address this issue, O'Donovan *et al.* [OLAH14] proposed selecting fonts using semantic attributes and collected a font attribute dataset. More recent advancements have introduced larger font attribute datasets and deep learning-based methods to improve the accuracy and efficiency of font retrieval [CWX*19, CMA19]. However, these previous attribute-based font selection methods only work on in-the-domain attributes and scripts. In contrast, using FontCLIP latent space enables font retrieval with out-of-domain attributes and scripts, thereby offering enhanced flexibility and efficiency in font selection.

2.3. Vector Font Generation

Example-based methods generate a complete character set of a font [SI10] or a personalized handwritten style [CLJ*15, LZCX18] from a single character. Parameterizing fonts is another method that allows users to create novel fonts by adjusting a set of parameters [SR98, Knu82]. Campell and Kautz [CK14] took a step further and presented the first generative model for fonts. By exploring the learned manifold, the model enables interpolation between existing fonts and the discovery of new fonts. Recently, various deep learning-based methods have been proposed for synthesizing vector glyphs [LHES19, CDAT20, WL21, RGLM21]. However, these methods often require users to provide sample characters of the desired font, making them challenging if they lack such resources. To address this issue, Wang *et al.* [WGL20] proposed Attribute2Font, which generates glyph images solely based on user-specified attributes. However, they can only generate bitmap glyph images for attributes and languages that are included in the training data. Thus, to generate bitmap glyph images for new attributes or languages, more training data is necessary. In contrast, our method can generate vector fonts that are easily manipulable. Moreover, our method can generate vector fonts for attributes and languages that are not part of the training data without requiring additional training data.

Our vector font optimization method is inspired by Word-As-Image [IVH*23], but with two significant differences. First, while Word-As-Image focuses on deforming a vector letter toward a conceptual visual representation (e.g., cat or dog), our optimization method using FontCLIP concentrates on capturing and reconstructing typographical features of each character (e.g., thin, italic, and serif). As we demonstrated in Figure 10, our method is more effective at capturing and reconstructing typographical features than Word-As-Image. Second, our optimization method utilizes FontCLIP's text and image encoder, which allows for dual-modal font optimization. In contrast, Word-As-Image only operates on text input. This makes our optimization method more practical and useful for capturing fonts in real-world scenarios, as shown in Figure 13.

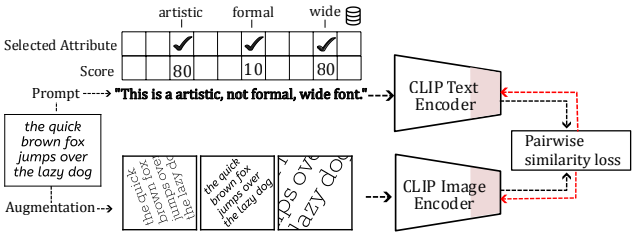


Figure 3: **Overview of FontCLIP finetuning.** During each finetuning iteration, we randomly select attributes based on their scores from an existing font-attribute dataset and create a compound descriptive prompt for each font. Simultaneously, we generate a font image and apply a random augmentation transformation to enhance variability. We finetuned the last three transformer blocks (highlighted in red) for both encoders using a pairwise similarity loss function.

3. CLIP Preliminaries

CLIP [RKH*21] is a vision-language model pretrained on a large number of image-text paired data and trains both an image encoder E_I and a text encoder E_T to a joint latent space. During training, CLIP uses a contrastive loss to learn a joint embedding for the two modalities. Specifically, for a mini-batch of image-text pairs, CLIP maximizes for each matching image-text pair the cosine similarity of their embeddings while minimizing the cosine similarities with all other unmatched texts/images. After training, the joint latent space of CLIP enables various downstream image processing and vision tasks in a zero-shot manner. For example, in image classification, given an input image I, its image embedding ($\mathbf{x} = E_I(I)$) is found using the image encoder, and a set of text embeddings ($\{\mathbf{w}_i = E_T(T_i)\}_{i=1}^K$) are found using the text encoder. In particular, each $\mathbf{w}_i$ is derived from a prompt T_i, such as "a photo of a {class}" where the "{class}" token is filled with the i-th class name. The prediction probability of class y is then defined as:

$$p(y|\mathbf{x}) = \frac{\exp(\text{sim}(\mathbf{x}, \mathbf{w}_y)/\tau)}{\sum_{i=1}^{K} \exp(\text{sim}(\mathbf{x}, \mathbf{w}_i)/\tau)} \quad (1)$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity and τ is a learned temperature parameter.

4. FontCLIP

Our goal is to learn a semantic typography latent space that can be effectively used for various typographic applications, including font retrieval and optimization-based font manipulation. To achieve

© 2024 Eurographics - The European Association
for Computer Graphics and John Wiley & Sons Ltd.

this goal, we focus on incorporating typography-specific knowledge into the existing large pretrained vision-language model CLIP. Our approach is to finetune a pretrained CLIP model using pairs of descriptive prompts and font image as inputs derived from a font-attribute dataset (Figure 3). In the following, we provide the technical details of our finetuning approach as well as the rationale behind its design.

4.1. Finetuning - Training Data

For finetuning, we use the dataset from [OLAH14]. This dataset consists of 200 Roman fonts, each annotated with 37 attribute scores. The attributes in the dataset can be broadly classified into two categories. Some of these attributes are related to the shape of the fonts, such as "serif", "italic", and "thin". The other attributes describe perceptual qualities such as "friendly", "warm", and "happy". Each font in the dataset is assigned scores ranging from 0 and 100 for each attribute. Among these attributes, there are some binary attributes: "capitals", "cursive", "display", "italic", "monospace" and "serif", meaning that they are assigned a score of either 0 or 100. To suit our finetuning requirement, we modify the original dataset and create a prompt-based dataset that aligns better with our objectives.

4.1.1. Compound Descriptive Prompt

Radford *et al.* [RKH*21] introduced a hand-crafted prompt: "a photo of a "{class}" for generic objects and scenes. However, in the case of our font dataset, each font is characterized by multiple attributes simultaneously with continuous scores, which differs from the original classification task that the original CLIP model was trained on. As a result, the above simple prompt is inadequate for accurately describing each font in our dataset. To overcome this challenge, we propose to use a compound descriptive prompt, combined with an adaptive sampling technique, to generate a more comprehensive and descriptive prompt for each font in our dataset.

During each finetuning iteration i, we generate the compound prompt T_i^F for a font F as

$$T_i^F = \text{"This is } [A]_1, [A]_2, ..., [A]_N \text{ font."}, \tag{2}$$

where each $[A]_n$ represents the n-th sampled attribute and N denotes the total number of attributes sampled. Throughout the finetuning process, we randomly set N to between 1 and 3 for each iteration. To determine the expression for each attribute, we consider whether its score is over or below 50. Specifically, if the score is over 50, we use the expression $[A]_n =$ "[attribute]". Conversely, if the score is below 50, we use $[A]_n =$ "not [attribute]". For example, if the score of attribute "happy" for font F exceeds 50, the corresponding expression in the compound prompt is set to "happy". Otherwise, it is set to "not happy". We randomly selected attributes for each font based on their attribute score distribution. Specifically, the probability of attribute a being selected is computed as

$$p(a) = \frac{\|S(a) - 50\|}{\sum_{i=1}^{37} \|S(a_i) - 50\|}, \tag{3}$$

where $S(a)$ represents the score of attribute a. We show how we generate a compound descriptive prompt in the top row of Figure 3.

4.1.2. Font Image

To generate the training images for each font in the dataset, we adopt the same approach used by O'Donovan *et al.* [OLAH14]. Specifically, we render an image of each font using the text "*The quick brown fox jumps over the lazy dog*", which is widely used for displaying font samples due to its inclusion of a diverse range of letters [OLAH14, KdM20]. To enhance the robustness of our finetuned model, we apply standard data augmentation techniques such as rotation, cropping, and scaling to the font image (refer to Figure 3).

4.2. Finetuning - Loss Function

Our finetuning method does not apply the CLIP-style contrastive learning [RKH*21] directly. Instead, it only requires positive pairs that include a compound descriptive prompt and a font image. The reason for this is that negative expressions like "not [attribute]" are already incorporated in our compound descriptive prompt. Thus, we can finetune the pretrained model to learn effective semantic typographic latent space solely by maximizing the cosine similarity between the embedded vectors of the compound descriptive prompts and those of their corresponding font images. Specifically, given a pair of a font image I^F and a compound descriptive prompt T^F, we define the *pairwise similarity* loss function as:

$$\mathcal{L}_{PS} = -\frac{1}{n} \sum_{q=1}^{n} \frac{E_I(I_q^F) \cdot E_T(T_q^F)}{\|E_I(I_q^F)\| \|E_T(T_q^F)\|}, \tag{4}$$

where n is the number of font descriptive prompt and font image pairs, E_I and E_T are the image and text encoder of the finetuned CLIP model, respectively.

4.3. Implementation Details

Our finetuning approach is based on the pretrained ViT CLIP model, which is publicly available on Hugging Face[†]. Throughout the finetuning process, we update the weights of the last three transformer block layers in both text and image encoders for $3,000$ epochs, while keeping the remaining weights frozen. We use the Adam optimizer [KB15] with a learning rate of 2×10^{-5}, which is halved every 500 epochs. The resolution of the font image used in finetuning is 214×214. In our setting, the finetuning process took around 12 hours on a machine equipped with an i7-12700K CPU with 32GB memory and RTX3080 GPU with 10GB memory.

5. Experiments

We conducted experiments using *in-domain* and *out-of-domain* attributes to evaluate the performance of FontCLIP. Specifically, we measured the correlations between the attribute score predicted using the FontCLIP features and the ground truth attribute score using the dataset from [OLAH14].

† https://huggingface.co/sentence-transformers/clip-ViT-B-32

© 2024 Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd.

Model	*In-domain* ↑	*Out-of-domain*↑
CLIP	0.159	0.159
FontCLIP (w/o CDP)	0.704	0.317
FontCLIP	**0.723**	**0.404**

Table 1: The average correlations for *in-domain* attributes and *out-of-domain* attributes of CLIP, FontCLIP trained without using compound descriptive prompts (w/o CDP), and FontCLIP trained with CDP (ours). By using CDP, FontCLIP can better generalize to *out-of-domain* attributes.

5.1. In-Domain Attributes

The first experiment aims to evaluate the consistency for *in-domain* attributes, wherein all attributes are used during the finetuning of each model. We used all 200 fonts from [OLAH14], which we randomly divided into 140 fonts for training, 30 fonts for validation, and 30 fonts for testing. We finetuned FontCLIP using the 140 fonts from the training set. For each font F in the testing dataset and for each attribute $[A]$, we calculated the similarity score between the font and the attribute in the following process. First, we obtained the visual embedding vector $E_I(I_F)$ of the font visual prompt I_F associated with F. Next, from the attribute $[A]$, we created a descriptive prompt T_A: "This is a $[A]$ font." and obtained the text embedding vector $E_T(T_A)$ for this prompt. Finally, we calculated the cosine similarity between $E_I(I_F)$ and $E_T(T_A)$. We considered a model's performance as the average correlation between predicted attribute scores and ground truth scores for all fonts in the testing set across all attributes. We compared three models: FontCLIP without using compound descriptive prompts (CDP), FontCLIP with CDP, and the baseline CLIP model.

As shown in the first row in Table 1, all variants of FontCLIP surpasses CLIP by a substantial margin. This observation suggests that FontCLIP has effectively learned the relationship between visual typographic attributes and the corresponding semantic attributes described by language, resulting in attribute ratings that are more aligned with human ratings. In addition, we provide a visualization of the correlation between predicted similarity scores and ground truth scores for the attributes "thin" and "playful" in Figure 4. This visualization allows us to observe the alignment between the predicted attribute scores and the ground truth scores. We also include correlation visualizations for other attributes in the supplemental material.

5.2. Generalization to Out-of-Domain Attributes

To evaluate the generalization capability of the FontCLIP latent space to *out-of-domain* attributes, meaning they are absent in the finetune training data, we conducted a leave-one-out experiment. During this experiment, we used all 200 fonts as training data for finetuning the model but excluded one attribute at a time during the finetuning process. Then, we calculated the average correlation between the predicted similarity score and ground truth attribute scores for all fonts, solely for the excluded attribute. This process was repeated for each attribute in the dataset from [OLAH14], resulting in a total N finetuning process, where N represents the number of attributes. The performance of each model was evaluated by computing the average correlation across all N attributes.

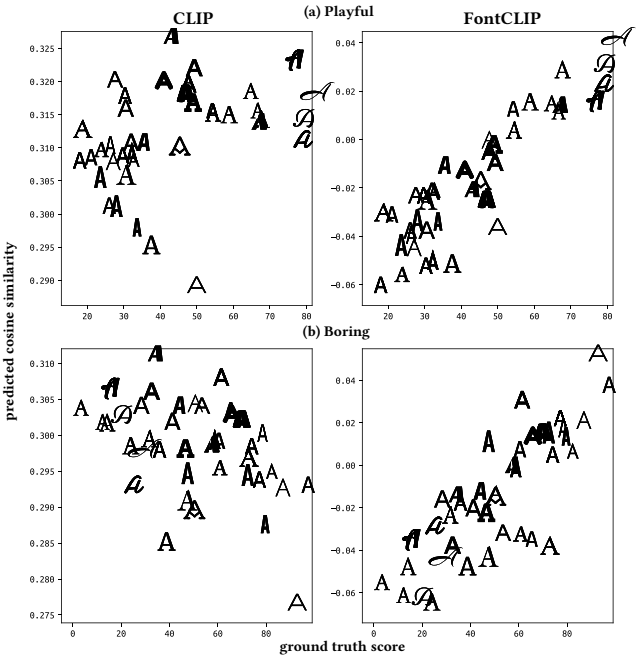


Figure 4: The visualization of the correlation between the predicted similarity scores from CLIP and FontCLIP, and the ground truth scores for (a) "playful" and (b) "boring" attributes.

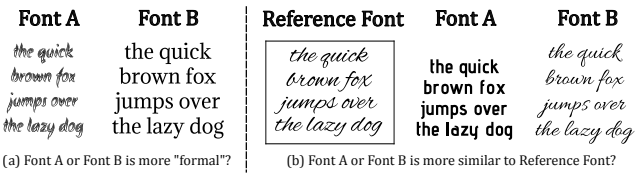


Figure 5: (a) In the pairwise attribute prediction task, we use a classifier to determine which font has a higher attribute score between two font options. (b) In the pairwise similarity prediction task, we use a classifier to determine which font is more similar to a reference font between two font options. In both tasks, the obtained results are compared with human judgments, and the prediction accuracy is calculated as the performance metric.

The results presented in the second row of Table 1 highlight that FontCLIP (w/o CDP) already performs better than CLIP. Moreover, the inclusion of CDP noticeably enhances the correlations. These findings indicate that FontCLIP can effectively generalize to *out-of-domain* attributes with straightforward finetuning and compound descriptive prompts. Additionally, we provide the correlation scores of all attributes in the supplemental material.

6. Dual-Modal Multilingual Font Retrieval

O'Donovan *et al.* [OLAH14] introduced attribute-based and similarity-based interfaces that enhance traditional dropdown menus with a list of font names. Inspired by their work, we propose

© 2024 Eurographics - The European Association
for Computer Graphics and John Wiley & Sons Ltd.

Model	Accuracy ↑
CLIP	51.87%
FontCLIP	65.32%
Feature-based	65.73%

(a): Pairwise attribute prediction.

Model	Accuracy ↑
CLIP	67.68%
FontCLIP	74.39%
Feature-based	75.95%

(b): Pairwise similarity prediction.

Model	*In-domain* accuracy	*Out-of-domain* accuracy
CLIP	54.92%	42.26%
FontCLIP	**64.14%**	**64.48%**
Feature-based	N/A	N/A

(c): CJK fonts pairwise attribute prediction.

Table 2: (a)(b) For both the pairwise attribute prediction task and pairwise similarity task, FontCLIP outperforms CLIP's performance and achieves similar performance to the best model that relies on geometric typographical features computed from the vector-based font file including all characters ("Feature-based") [OLAH14], while our FontCLIP-based method only requires a font image as input. (c) The FontCLIP latent space generalizes to *out-of-domain* attributes and to multiple languages. Note that previous feature-based methods require the vector-based font files, cannot recognize *out-of-domain* attributes, and might be able to support multi-lingual capabillities only if they had access to the multi-lingual fonts files (although this has never been tested).

leveraging the FontCLIP latent space for dual-modal font retrieval tasks. Besides the attribute-based interface, our interface facilitates image-based retrieval, which does not require the user to obtain the vector-based font files covering all characters, unlike O'Donovan *et al.* [OLAH14]. Moreover, our interface allows for any combination of the attributes and images. In addition, the FontCLIP latent space exhibits the capability to generalize beyond Roman fonts, allowing it to support multiple language settings. In the following sections, we present quantitative evaluations for attribute-based, image-based, and cross-lingual font retrieval, along with qualitative evaluation of multilingual and cross-lingual font retrieval using a combination of attribute and image inputs.

6.1. Quantitative Evaluation

To quantitatively evaluate the attribute-based and image-based font retrieval, we follow the experiment setup outlined in [OLAH14] and focus on *in-domain* attributes. For attribute-based retrieval, we conduct the pairwise attribute prediction task, while for image-based retrieval, we evaluate the pairwise font similarity prediction task. In both tasks, we use the complete 200 fonts and the 31 adjectives attributes in the dataset from [OLAH14]. Our main goal of FontCLIP is to enable font retrieval without the need to access the original vector-based font files. Therefore, we primarily focus on comparing the performance of FontCLIP and CLIP because both methods use font images as input instead of the vector-based font files. For reference, we provide a performance of the best machine learning model using vector-based typographical features [OLAH14].

Pairwise Attribute Prediction The goal of the pairwise attribute prediction task is to determine which font has a higher attribute score between two font options represented as font images (Figure 5(a)). We first evaluate this task for *in-domain* attributes.In total, we generated 198,400 pairwise comparison subtasks for evaluating this task. Each comparison was assessed by seven people, and we computed the accuracy through respective comparisons. As shown in Table 2(a), FontCLIP achieves better performance compared to CLIP. This suggests that the FontCLIP feature is more distinguishable regarding different attributes. Moreover, despite taking only a font image as input, FontCLIP achieves comparable performance to the best model that uses typographical features proposed by [OLAH14]. This result suggests that FontCLIP's image-based typographical features extracted only from target glyphs (i.e., not glyphs of all Roman characters) are representative and achieve similar retrieval performance as vector-based typographical features extracted from all Roman characters.

Furthermore, we conducted quantitative evaluations to assess FontCLIP's generalization capabilities on *out-of-domain* attributes and different languages. *Out-of-domain* attributes cannot be handled by the methods in [OLAH14] because their models need to be trained in an attribute-specific manner. In addition, their models use typographical features that are specifically designed for Roman characters. For the evaluations, we additionally collected pairwise attribute rating data for 50 CJK fonts with three participants recruited from our university. The collected dataset contains ratings for 5 *in-domain* attributes and 3 *out-of-domain* attributes specifically used for describing CJK fonts, including "traditional", "robust", and "Japanese style". In Table 2(c), we show the results of the attribute prediction task performed on CJK fonts. It can be observed that FontCLIP outperforms CLIP on both *in-domain* and *out-of-domain* attributes in this task, especially on *out-of-domain* attributes.

Pairwise Similarity Prediction The pairwise similarity prediction task involves selecting the font that is more similar to a given reference font image out of two font images (Figure 5(b)). This task serves as a means to assess whether the distances in the FontCLIP latent space accurately reflect the perceptual similarity between fonts. In total, we generated 35,387 comparisons for this task. Each pairwise comparison was voted by 10 to 15 individuals, and we computed the accuracy through respective comparisons. For the analysis, we excluded 52 comparisons with tie votes. As shown in Table 2(b), similar to the pairwise attribute prediction task, FontCLIP obtains better performance compared to CLIP and achieves comparable performance to the best model that uses vector-based typographical features extracted from all Roman characters [OLAH14]. The results of both tasks indicate that FontCLIP's image-based typographical features perform similarly to its vector-based counterpart without requiring vector-based font files for all Roman characters, thus significantly reducing the efforts of retrieving new fonts.

Cross-Lingual Pairwise Similarity Prediction Task We conduct two pairwise similarity prediction tasks to assess the cross-lingual retrieval ability of different methods, As shown in Figure 6, the first task is called "Roman-to-CJK", which involves selecting the CJK font that is more similar to a given Roman font. The second task is "CJK-to-Roman", which involves selecting the Roman font

© 2024 Eurographics - The European Association
for Computer Graphics and John Wiley & Sons Ltd.

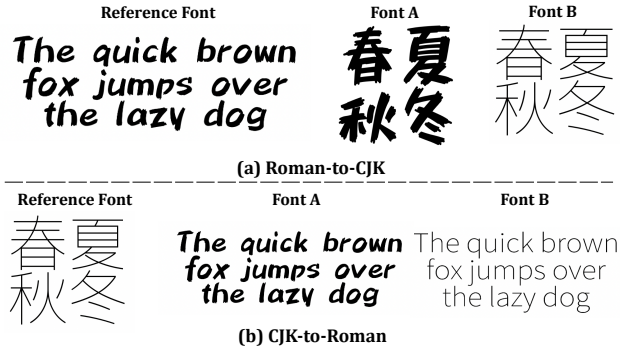


Figure 6: Cross-Lingual Pairwise Similarity Prediction. (a) For the "Roman-to-CJK" task, we use a classifier to determine which CJK font is more similar to a reference Roman font between two CJK font options. (b) Conversely, for the "CJK-to-Roman" task, we use a classifier to determine which Roman font is more similar to a reference CJK font between two Roman font options. In both tasks, we compare the results obtained using the classifiers with human judgments, and calculate the prediction accuracy as the performance metric.

Model	Roman-to-CJK ↑	CJK-to-Roman ↑
CLIP	57.4%	50.0%
FontCLIP	**67.2%**	**62.6%**

Table 3: For both cross-lingual pairwise similarity tasks, FontCLIP performed better than CLIP, suggesting that FontCLIP's prediction results are closer to human rating results.

that is more similar to the query CJK font. To evaluate these two tasks quantitatively, we collected 280 fonts that were not part of the training dataset. In total, we generated 100 pairwise comparison subtasks for "Roman-to-CJK" and "CJK-to-Roman" tasks and recruited five participants to rate these comparisons. In Table 3, we show the results for both "Roman-to-CJK" and "CJK-to-Roman" tasks. We can observe that FontCLIP's predictions are better aligned with human ratings compared to CLIP's predictions. This suggests that despite being finetuned solely on the Roman character dataset, the FontCLIP model can still learn general typographical features that achieve better cross-lingual font retrieval. We also observed that FontCLIP's performance of "Roman-to-CJK" was better than its "CJK-to-Roman" counterpart, which is in line with our expectation given that the FontCLIP model was trained only on the Roman character dataset.

6.2. Qualitative Evaluation

In our qualitative evaluation, we collected 1,169 Roman fonts and 293 CJK fonts in total. For each font, we generate its font image using the method described in Section 4.1.2 and extract its visual feature using FontCLIP visual encoder E_I. We denote the final font feature databases for Roman and for CJK as Ω_{Roman} and Ω_{CJK}.

Multilingual Font Retrieval First, we demonstrate FontCLIP's unprecedented generalization capability by showing multilingual font retrieval results using a combination of attributes and image inputs. Our goal aligns with [KdM20], where we aim to retrieve results that preserve the style of input font image while incorporating the desired attributes. Specifically, given a query font image I_{query} and a set of desired attributes $\mathbf{A} = \{a_1, a_2, ..., a_N\}$, we obtain the embedding vector of the desired font e_{desired} using the following formulation:

$$e_{\text{desired}} = E_I(I_{\text{query}}) + wE_T(T), \tag{5}$$

where E_I and E_T is the image and text encoder of FontCLIP, T is a text prompt containing all desired attribute $\mathbf{A}$, and $w \in [0,1]$ is a weight that controls the balance between preserving the original letter styles and incorporating the styles of the desired attributes. With the embedding vector e_{desired}, we obtain the top-K retrieved results by choosing the K closest fonts to e_{desired} in the corresponding font feature database (Ω_{Roman} or Ω_{CJK}).

In Figure 7, we compared the retrieved results obtained using FontCLIP and CLIP in different languages, considering both *in-domain* and *out-of-domain* attributes. In Figure 7, we found that the retrieved results using FontCLIP effectively incorporate the desired attributes while preserving the original style. For example, in Figure 7(a), more retrieved results are with serifs for all languages by using the FontCLIP feature than that of CLIP; in Figure 7(b), for *out-of-domain* attribute "traditional", the retrieved results by FontClip also better align with the human perception than that of CLIP. Besides, the CLIP latent space fails to interpret the "not" prompt, resulting in thicker retrieved results compared to the results obtained by the FontCLIP feature (Figure 7(a)).

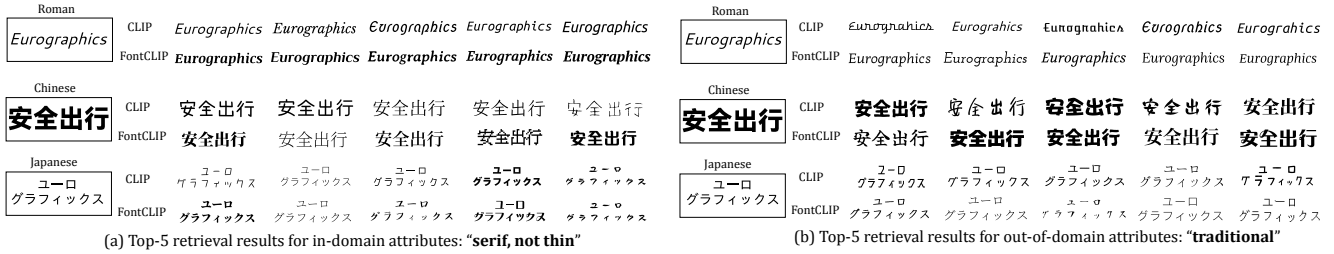
Cross-Lingual Font Retrieval Next, we demonstrate the cross-lingual font retrieval results using FontCLIP. Our goal is to retrieve fonts that have a similar style to the query font image I_{query} from other languages. To begin with, we calculate the visual feature of the query font image I_{query} as $E_I(I_{\text{query}})$. Following this, we search for the k nearest fonts to the visual feature in the font feature dataset of other languages. In the "Roman-to-CJK" results shown in Figure 8, we have found that the FontCLIP feature is better at retrieving CJK fonts that are more similar to the query Roman fonts, compared to the CLIP feature (Figure 8(a)). Meanwhile, we observed that the "CJK-to-Roman" retrieved results of CLIP deviate excessively from the style of the query font image (Figure 8(b)).

7. Dual-Modal Multilingual Vector Font Optimization

In this section, we describe another application of FontCLIP: an optimization-based method that modifies the letter shapes in vector fonts based either on text prompts or on image inputs in multiple languages. Figure 9 shows the overview of our vector font optimization method. Guided by the FontCLIP latent space, our optimization-based method supports both language-driven and image-driven font optimization in multiple languages.

The input to our method is a letter l from an existing font in vector format. Following [IVH*23], we represent l as a set of k control points, $P = \{p_j \in \mathbb{R}^2\}_{j=1}^{k}$, describing it's outline. P is obtained by a subdivision, which provides sufficient expressiveness even for letters

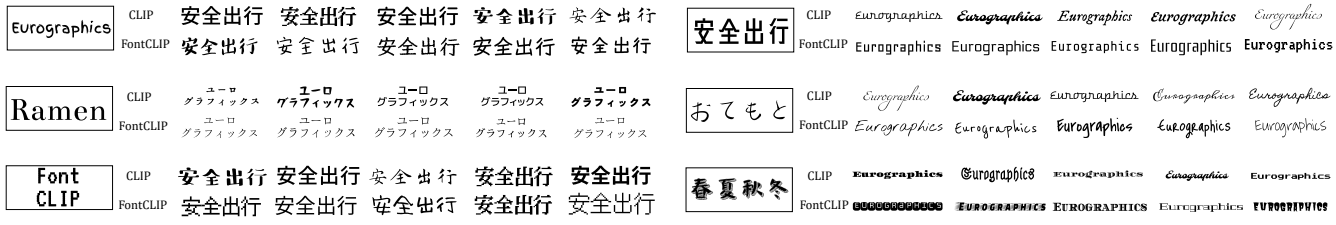
© 2024 Eurographics - The European Association
for Computer Graphics and John Wiley & Sons Ltd.



(a) Top-5 retrieval results for in-domain attributes: "**serif, not thin**"

(b) Top-5 retrieval results for out-of-domain attributes: "**traditional**"

Figure 7: The results of dual-modal font retrieval using FontCLIP latent space and CLIP latent space. The goal of this multi-modal retrieval is to preserve the style of input font image query (text with frame) while incorporating the desired attributes. (a) We show the top-5 retrieval results with *in-domain* attributes for Roman, Chinese, and Japanese characters. By using the FontCLIP feature, we can retrieve more fonts with serif for multiple languages. (b) We show the top-5 retrieval results with *out-of-domain* attributes for Roman, Chinese, and Japanese characters.



(a) Top-5 **Roman-to-CJK** retrieval results

(b) Top-5 **CJK-to-Roman** retrieval results

Figure 8: Cross-lingual font retrieval results using FontCLIP latent space and CLIP latent space. The goal of this cross-lingual font retrieval is to find fonts in other languages that match the style of the input font image query (text with frame). (a) We show the top-5 retrieval results for "Roman-to-CJK". (b) We show the top-5 retrieval results for "CJK-to-Roman".

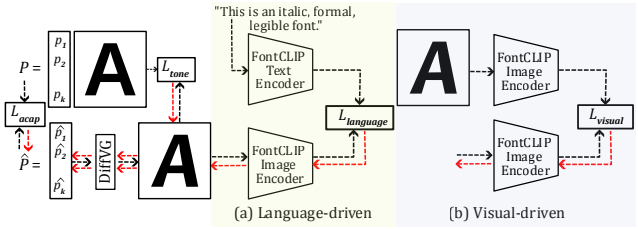


Figure 9: An overview of our multi-modal vector font optimization. Given an input letter l ("A" in this example) represented as a set of outline control points P, and either a language-driven descriptive prompt T_{user} (a), or a visual-driven reference font image I_{user} (b), we iteratively optimize the new positions of $\hat{P}$ creating the optimized letter shape $\hat{l}$. Inspired by [IVH*23], we first rasterize the deformed letter $\hat{l}$ by a differentiable rasterizer (DiffVG). To guide the optimization, we use a language loss L_{language} in (a) language-driven optimization, or a visual loss L_{visual} in (b) visual-driven optimization to ensure $\hat{l}$ aligns with desired attributes indicated by the descriptive prompt or the reference font image. Moreover, our objective function includes the tone preservation loss L_{tone} and an ACAP deformation loss L_{acap} similar to [IVH*23]. Black and red dashed arrows indicate forward and backward computation, respectively.

with few control points. The output of our pipeline is the same set of control points $\hat{P} = \{\hat{p}_j\}_{j=1}^{k}$ in different positions that represents the outline of the manipulated letter $\hat{l}$. The users define their goal either by providing a text prompt T_{user} of attributes or a reference font image I_{user} as additional input that drives the optimization (see Figure 9).

7.1. Language-Driven Font Optimization

Our specific goal here is to manipulate the original letter l into $\hat{l}$ by aligning it with desired attributes while preserving the original styles. To preserve the original styles of l, we begin by calculating similarity scores between its original shape $\mathcal{R}(P)$ and the 37 attributes. We then select the top-M (we set $M = 2$) attributes with the highest similarity scores as the attributes to be preserved. The user-specified prompt T_{user} is then combined with these M preserved attributes to form the final compound descriptive prompt T_{final}. To encourage the manipulated letter $\hat{l}$ to align with T_{final}, we define the following function using the FontCLIP visual encoder E_I and text encoder E_T:

$$L_{\text{language}}(\hat{P}, T_{\text{final}}) = \text{dist}(E_I(\mathcal{R}(\hat{P})), E_T(T_{\text{final}})), \tag{6}$$

where $\text{dist}(\mathbf{x}, \mathbf{y}) = 1.0 - \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}$ denotes the cosine distance between $\mathbf{x}$ and $\mathbf{y}$, and $\mathcal{R}$ is the differentiable rasterizer [LLGRK20]. However, we have noticed that using L_{language} alone can result in significant deviations from the initial letter geometry. Inspired by [IVH*23], we incorporate the ACAP deformation loss and tone preservation loss into our final objective function. The ACAP deformation loss minimizes the deviation of the final letter shape from its initial shape, while the tone preservation loss aims to preserve the font's style and letter structure. For detailed definitions of both losses, please refer to [IVH*23]. The objective function is defined as

$$L_{LD} = L_{\text{language}} + w_{\text{acap}} L_{\text{acap}} + w_{\text{tone}} L_{\text{tone}}, \tag{7}$$

© 2024 Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd.

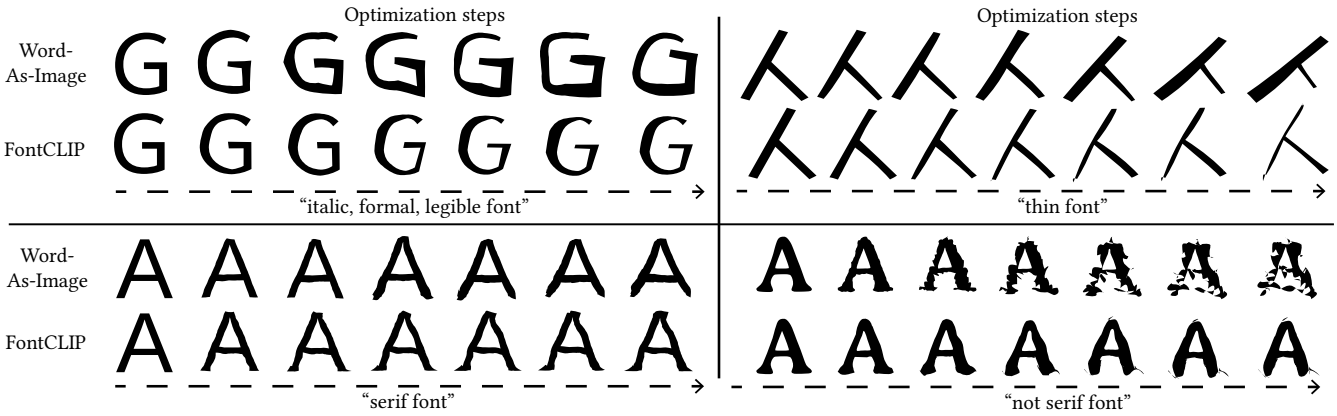


Figure 10: Visualization of the vector font optimization steps of the language-driven Roman and Chinese character optimization using FontCLIP. We compared the results obtained by Word-As-Image [IVH*23] and our method. Our method better captures and reconstructs each character's typographical features, including features such as serif.

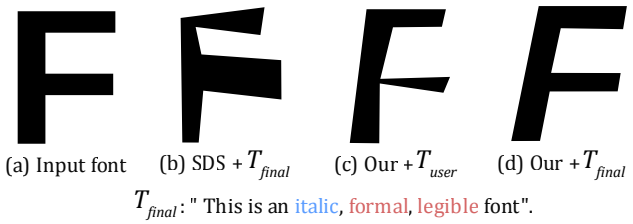


Figure 11: Ablation study on the language-driven font optimization. Given (a) an input font, we compare the results obtained by (b) replacing L_{language} into SDS loss, (c) our method using only T_{user}, and (d) our method using T_{final}. (The user specificed attributes are shown in blue and the attributes to be preserved are shown in red.)

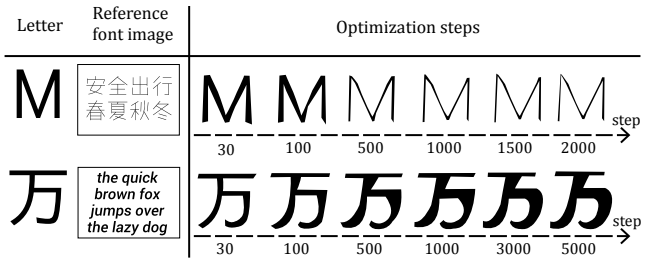


Figure 12: Visualization of the optimization steps of the cross-lingual image-driven Roman and Chinese character optimization.

where we set $w_{\text{acap}} = 0.2$ and $w_{\text{tone}} = 0.2$ throughout all examples shown in this paper. In Figure 1(b), we can observe the iterative optimization steps where the Chinese character gradually becomes thinner while maintaining its formal and legible appearance. In Figure 10, we compare the results obtained by our method and Word-As-Image [IVH*23] on Roman and Chinese characters. We can observe that our optimization method using FontCLIP feature captures and reconstructs typographical features better, even for features such as serif.

Ablation Study In Figure 11, we present a comparison of various formulations. This includes replacing the language loss L_{language} with the Stable Diffusion (SDS) loss, which was used in [IVH*23], and solely using T_{user} in L_{language} (i.e., excluding the attributes we aim to preserve). As can be seen, the result using SDS loss did not exhibit the desired attributes used in the text prompt T_{final} and severely deviated from the original letter shape. The result using T_{user} reflects the desired attribute ("italic") but fails to preserve the original styles of the input font.

7.2. Image-Driven Font Optimization

When a reference font image I_{user} is given, image-driven font optimization manipulates l into $\hat{l}$ while ensuring that $\hat{l}$ reflects the visual typographic attributes present in I_{user}. To achieve this, we define the following function using the FontCLIP visual encoder E_I and text encoder E_T:

$$L_{\text{image}}(\hat{P}, I_{\text{user}}) = \text{dist}(E_I(\mathcal{R}(\hat{P})), E_I(I_{\text{user}})), \tag{8}$$

and the overall objective function for image-driven font manipulation is defined as:

$$L_{VD} = L_{\text{image}} + w_{\text{acap}} L_{\text{acap}} + w_{\text{tone}} L_{\text{tone}}, \tag{9}$$

where we set $w_{\text{acap}} = 0.2$ and $w_{\text{tone}} = 0.2$. In Figure 12, we present cross-lingual optimization results on both Roman and CJK characters. We can observe the iteratiove optimization steps that gradually align the styles of the input Roman and Japanese letters with the style in the reference font images even from other languages. Finally, in Figure 13, we demonstrate the effectiveness of our font optimization method using a reference font image captured in real-world conditions. We extracted the letters from the captured image and used them as I_{user} to drive the optimization for the provided letters.

© 2024 Eurographics - The European Association
for Computer Graphics and John Wiley & Sons Ltd.

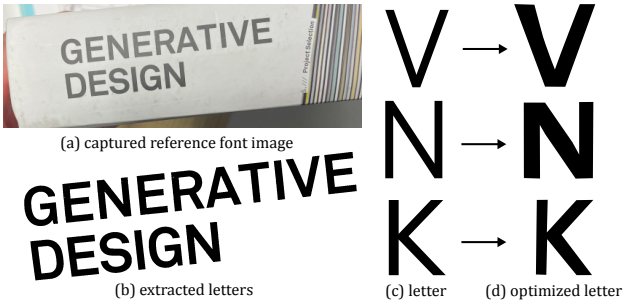


Figure 13: (a) Given a reference font image captured in real-world, our optimization method uses (b) the extracted letters to manipulate (c) the input letters. (d) The optimized letters exhibits a similar style to the fonts in the captured image.

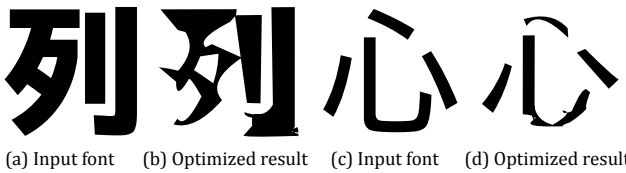


Figure 14: Our optimization method faces challenges in handling complex structures such as crossing and rounded strokes.

8. Limitations and Future Work

Attribute Entanglement Currently, FontCLIP latent space exhibits entanglement between different attributes. As shown in Figure 11(c), the optimized letter exhibits characteristics from attributes that are not specificed by the user. As a result, our method need to identify and preserve the most representative attributes of the font during language-driven font optimization (Figure 11(d)). In the future, a potential solution would be to explore contrastive finetuning, utilizing fonts with similar attribute scores but differing in only one attribute.

Vector Font Optimization on Complex Typographic Structures While our current character shape optimization method shows promising results, it faces challenges in handling complex typographic structures, such as crossing and rounded strokes (Figure 14). Future research could improve our optimization method by investigating more appropriate font parameterizations [HHH10] and incorporating more typographic-specific constraints. Nonetheless, our results validate the concept and suggest that our FontCLIP could be a foundation of future font optimization methods.

Generalization Enhancement Currently, FontCLIP is specifically finetuned using a dataset that exclusively contains Roman alphabet characters and commonly associated attributes. However, there is a possibility that cultural differences might affect how letter shapes are linked to attributes. To alleviate this issue, we plan to explore few-shot learning techniques for *out-of-domain* languages, which involve collecting small-scale datasets using the data collection process described in [OLAH14].

9. Conclusion

In this paper, we introduced FontCLIP – a model that bridges the semantic understanding of a large vision-language model with typographical knowledge. Our experiments demonstrated FontCLIP's two unprecedented generalization abilities. First, FontCLIP can generalize to multiple languages despite being finetuned only on a Roman character dataset. This ability enables multilingual and cross-lingual font retrieval and letter shape optimization. Second, FontCLIP can recognize *out-of-domain* semantic attributes, facilitating more diverse attribute-based font retrieval and letter shape optimization. Finally, FontCLIP's dual-modality allows unprecedented multilingual font applications through a unified space without extracting typographical features through vector-based font files. In summary, we believe FontCLIP can greatly simplify the process of obtaining desired fonts during the design process.

Acknowledgement

We thank the anonymous reviewers for their valuable feedback. This work was partially supported by JST AdCORP, Grant Number JPMJKB2302, JSPS Grant-in-Aid JP23K16921, Japan, and a collaboration with Dentsu Digital.

References

[CDAT20] CARLIER A., DANELLJAN M., ALAHI A., TIMOFTE R.: DeepSVG: A hierarchical generative network for vector graphics animation. In *Proc. NeurIPS* (2020), vol. 33, pp. 16351–16361. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/bcf9d6bd14a2095866ce8c950b702341-Paper.pdf. 3

[CK14] CAMPBELL N. D. F., KAUTZ J.: Learning a manifold of fonts. *ACM Trans. Graph. 33*, 4 (2014). doi:10.1145/2601097.2601212. 3

[CLJ*15] CHEN H.-I., LIN T.-J., JIAN X.-F., SHEN I.-C., CHEN B.-Y.: Data-driven handwriting synthesis in a conjoined manner. *Comput. Graph. Forum 34*, 7 (2015), 235–244. doi:10.1111/cgf.12762. 3

[CMA19] CHOI S., MATSUMURA S., AIZAWA K.: Assist users' interactions in font search with unexpected but useful concepts generated by multimodal learning. In *Proc. ICMR* (2019), pp. 235–243. doi:10.1145/3323873.3325037. 3

[CWX*19] CHEN T., WANG Z., XU N., JIN H., LUO J.: Large-scale tag-based font retrieval with generative feature learning. In *Proc. ICCV* (2019), pp. 9116–9125. doi:10.1109/ICCV.2019.00921. 1, 3

[CYJ*14] CHEN G., YANG J., JIN H., BRANDT J., SHECHTMAN E., AGARWALA A., HAN T. X.: Large-scale visual font recognition. In *Proc. CVPR* (2014), pp. 3598–3605. doi:10.1109/CVPR.2014.460. 3

[GAG*23] GAO W., AIGERMAN N., GROUEIX T., KIM V., HANOCKA R.: TextDeformer: Geometry manipulation using text guidance. In *Proc. SIGGRAPH* (2023). doi:10.1145/3588432.3591552. 3

[HHH10] HASSAN T., HU C., HERSCH R. D.: Next generation typeface representations: Revisiting parametric fonts. In *Proc. DocEng* (2010), pp. 181–184. doi:10.1145/1860559.1860596. 10

[HZP*22] HONG F., ZHANG M., PAN L., CAI Z., YANG L., LIU Z.: AvatarCLIP: Zero-shot text-driven generation and animation of 3D avatars. *ACM Trans. Graph. 41*, 4 (2022). doi:10.1145/3528223.3530094. 1, 3

[IVH*23] ILUZ S., VINKER Y., HERTZ A., BERIO D., COHEN-OR D., SHAMIR A.: Word-as-image for semantic typography. *ACM Trans. Graph. 42*, 4 (2023). doi:10.1145/3592123. 3, 7, 8, 9

© 2024 Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd.

[JMB*22] JAIN A., MILDENHALL B., BARRON J. T., ABBEEL P., POOLE B.: Zero-shot text-guided object generation with dream fields. In *Proc. CVPR* (2022), pp. 867–876. doi:10.1109/CVPR52688.2022.00094. 3

[JYX*21] JIA C., YANG Y., XIA Y., CHEN Y.-T., PAREKH Z., PHAM H., LE Q., SUNG Y.-H., LI Z., DUERIG T.: Scaling up visual and vision-language representation learning with noisy text supervision. In *Proc. ICML* (2021), pp. 4904–4916. URL: http://proceedings.mlr.press/v139/jia21b/jia21b.pdf. 2

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *Proc. ICLR* (2015). URL: https://arxiv.org/abs/1412.6980. 4

[KCG*23] KUO W., CUI Y., GU X., PIERGIOVANNI A., ANGELOVA A.: F-vlm:open-vocabulary object detection upon frozen vision and language models. In *Proc. ICLR* (2023). URL: https://openreview.net/pdf?id=MIMwy4kh9lf. 1, 2

[KdM20] KULAHCIOGLU T., DE MELO G.: Fonts like this but happier: A new way to discover fonts. In *Proc. MM* (2020), pp. 2973–2981. doi:10.1145/3394171.3413534. 4, 7

[Knu82] KNUTH D. E.: The concept of a meta-font. *Visible language 16*, 1 (1982), 3–27. 3

[LBW*22] LUO H., BAO J., WU Y., HE X., LI T.: SegCLIP: Patch aggregation with learnable centers for open-vocabulary semantic segmentation, 2022. arXiv:2211.14813. 1, 2

[LHES19] LOPES R. G., HA D., ECK D., SHLENS J.: A learned representation for scalable vector graphics. In *Proc. ICCV* (2019), pp. 7930–7939. doi:10.1109/ICCV.2019.00802. 3

[LLGRK20] LI T.-M., LUKÁČ M., GHARBI M., RAGAN-KELLEY J.: Differentiable vector graphics rasterization for editing and learning. *ACM Trans. Graph. 39*, 6 (2020). doi:10.1145/3414685.3417871. 8

[LROTG21] LIU Z., RODRIGUEZ-OPAZO C., TENEY D., GOULD S.: Image retrieval on real-life images with pre-trained vision-and-language models. In *Proc. ICCV* (2021), pp. 2125–2134. doi:10.1109/ICCV48922.2021.00213. 2

[LZC*23] LIU M., ZHU Y., CAI H., HAN S., LING Z., PORIKLI F., SU H.: PartSLIP: Low-shot part segmentation for 3d point clouds via pre-trained image-language models. In *Proc. CVPR* (June 2023), pp. 21736–21746. doi:10.1109/CVPR52729.2023.02082. 3

[LZCX18] LIAN Z., ZHAO B., CHEN X., XIAO J.: EasyFont: A style learning-based system to easily build your large-scale handwriting fonts. *ACM Trans. Graph. 38*, 1 (2018). doi:10.1145/3213767. 3

[LZZ*22] LI L. H., ZHANG P., ZHANG H., YANG J., LI C., ZHONG Y., WANG L., YUAN L., ZHANG L., HWANG J.-N., ET AL.: Grounded language-image pre-training. In *Proc. CVPR* (2022), pp. 10965–10975. doi:10.1109/CVPR52688.2022.01069. 2

[MBOL*22] MICHEL O., BAR-ON R., LIU R., BENAIM S., HANOCKA R.: Text2mesh: Text-driven neural stylization for meshes. In *Proc. CVPR* (2022), pp. 13492–13502. doi:10.1109/CVPR52688.2022.01313. 3

[MKXBP22] MOHAMMAD KHALID N., XIE T., BELILOVSKY E., POPA T.: CLIP-Mesh: Generating textured meshes from text using pretrained image-text models. In *Proc. SIGGRAPH Asia* (2022). doi:10.1145/3550469.3555392. 3

[OLAH14] O'DONOVAN P., LUNDEFINEDBEKS J., AGARWALA A., HERTZMANN A.: Exploratory font selection using crowdsourced attributes. *ACM Trans. Graph. 33*, 4 (2014). doi:10.1145/2601097.2601110. 1, 2, 3, 4, 5, 6, 10

[RBL*21] ROMBACH R., BLATTMANN A., LORENZ D., ESSER P., OMMER B.: High-resolution image synthesis with latent diffusion models, 2021. arXiv:2112.10752. 1

[RDN*22] RAMESH A., DHARIWAL P., NICHOL A., CHU C., CHEN M.: Hierarchical text-conditional image generation with clip latents, 2022. arXiv:2204.06125. 1

[RGLM21] REDDY P., GHARBI M., LUKAC M., MITRA N. J.: Im2Vec:synthesizing vector graphics without vector supervision. In *Proc. CVPR* (2021), pp. 7342–7351. doi:10.1109/CVPRW53098.2021.00241. 3

[RKH*21] RADFORD A., KIM J. W., HALLACY C., RAMESH A., GOH G., AGARWAL S., SASTRY G., ASKELL A., MISHKIN P., CLARK J., ET AL.: Learning transferable visual models from natural language supervision. In *Proc. ICML* (2021), pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a/radford21a.pdf. 1, 2, 3, 4

[SI10] SUVEERANONT R., IGARASHI T.: Example-based automatic font generation. In *Proc. Smart Graphics* (2010), pp. 127–138. doi:10.5555/1894345.1894361. 3

[SR98] SHAMIR A., RAPPOPORT A.: Feature-based design of fonts using constraints. In *Electronic Publishing, Artistic Imaging, and Digital Typography* (Berlin, Heidelberg, 1998), Hersch R. D., André J., Brown H., (Eds.), Springer Berlin Heidelberg, pp. 93–108. doi:10.1007/BFb0053265. 3

[TGH*22] TEVET G., GORDON B., HERTZ A., BERMANO A. H., COHEN-OR D.: MotionCLIP: Exposing human motion generation to clip space. In *Proc. ECCV* (2022), pp. 358–374. doi:10.1007/978-3-031-20047-2_21. 3

[VPB*22] VINKER Y., PAJOUHESHGAR E., BO J. Y., BACHMANN R. C., BERMANO A. H., COHEN-OR D., ZAMIR A., SHAMIR A.: CLIPasso: Semantically-aware object sketching. *ACM Trans. Graph. 41*, 4 (2022). doi:10.1145/3528223.3530068. 1

[WCH*22] WANG C., CHAI M., HE M., CHEN D., LIAO J.: Clip-NeRF: Text-and-image driven manipulation of neural radiance fields. In *Proc. CVPR* (2022), pp. 3835–3844. doi:10.1109/CVPR52688.2022.00381. 3

[WGL20] WANG Y., GAO Y., LIAN Z.: Attribute2Font: Creating fonts you want from attributes. *ACM Trans. Graph. 39*, 4 (2020). doi:10.1145/3386569.3392456. 1, 3

[WL21] WANG Y., LIAN Z.: DeepVecFont: Synthesizing high-quality vector fonts via dual-modality learning. *ACM Trans. Graph. 40*, 6 (dec 2021). doi:10.1145/3478513.3480488. 1, 3

[WYJ*15] WANG Z., YANG J., JIN H., SHECHTMAN E., AGARWALA A., BRANDT J., HUANG T. S.: DeepFont: Identify your font from an image. In *Proc. ICMR* (2015), pp. 451–459. doi:10.1145/2733373.2806219. 3

[ZGZ*22] ZHANG R., GUO Z., ZHANG W., LI K., MIAO X., CUI B., QIAO Y., GAO P., LI H.: PointCLIP: Point cloud understanding by CLIP. In *Proc. CVPR* (2022), pp. 8552–8562. doi:10.1109/CVPR52688.2022.00836. 2

[ZLD22] ZHOU C., LOY C. C., DAI B.: Extract free dense labels from CLIP. In *Proc. ECCV* (2022), pp. 696–712. doi:10.1007/978-3-031-19815-1_40. 1, 2

[ZZL*22] ZHOU Z., ZHANG B., LEI Y., LIU L., LIU Y.: ZegCLIP: Towards adapting clip for zero-shot semantic segmentation, 2022. arXiv:2212.03588. 1, 2

© 2024 Eurographics - The European Association
for Computer Graphics and John Wiley & Sons Ltd.