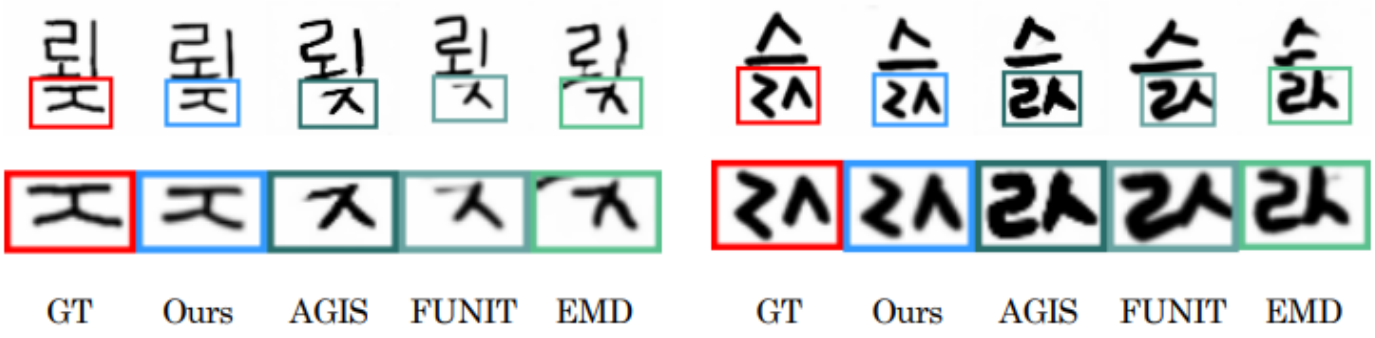
# Few-shot Compositional Font Generation with Dual Memory

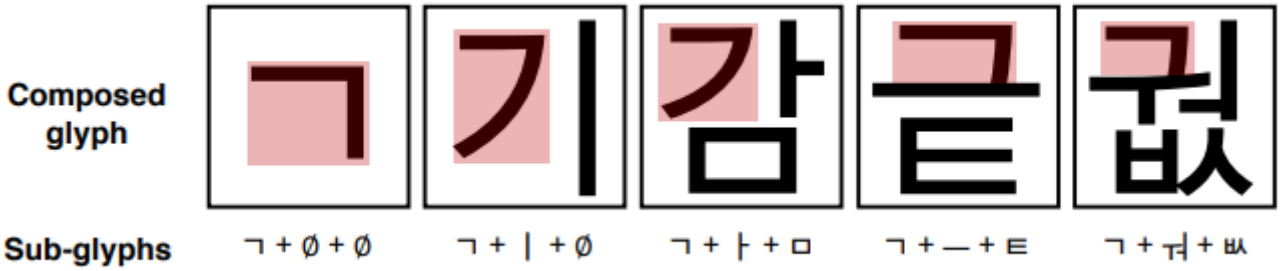



> Fig. 1: Few-shot font generation results. While previous few-shot font generation methods (AGIS, FUNIT, and EMD) are **failed to generate unseen font**, our model successfully transfer the font style and details.

## Abstract

- In this paper, we focus on **compositional scripts**, a widely used letter system in the world, where each glyph can be decomposed by several components.
- we propose a novel font generation framework, named **Dual Memory-augmented Font Generation Network (DM-Font)**, which enables us to generate a highquality font library with only a few samples.
- We employ **memory components** and **global-context** awareness in the generator to take advantage of the compositionality.

## Preliminary: Complete Compositional Scripts

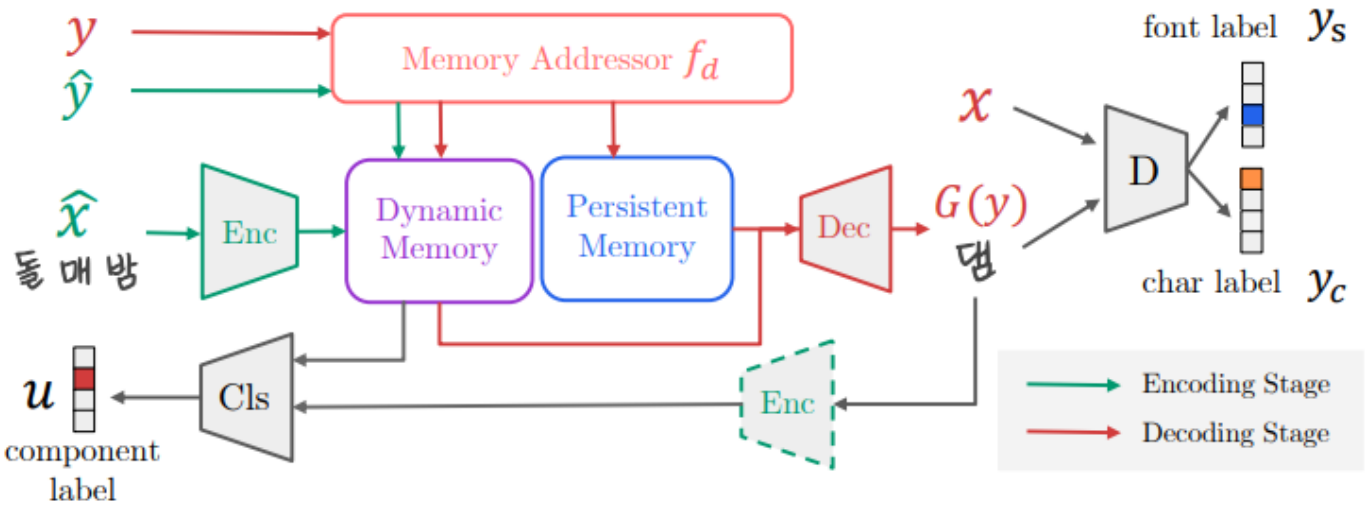


> Fig. 2: Examples of compositionality of Korean script. Even if we choose the same sub-glyph, e.g., "ㄱ", the shape and position of each sub-glyph are varying depending on the combination, as shown in red boxes.

# Dual Memory-augmented Font Generation Network

- DM-Font disentangles global **local styles** and **composition information**, and writes them into **dynamic** and **persistent** memory, respectively.

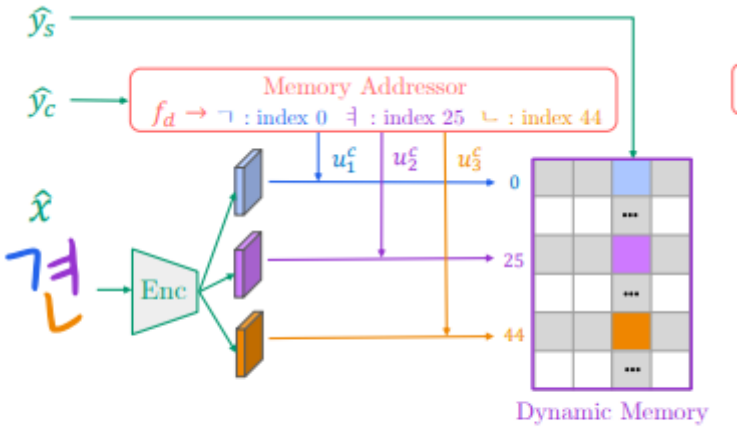
## Architecture overview




(a) Architecture overview.

- **encoding** stage, the reference style glyphs are encoded to the component features and stored into the dynamic memory.
- After the encoding, the **decoder** fetches the component features and generates the target glyph according to the target character label.

### Encoder




(b) Encoding phase detail.

> The encoder extracts the component-wise features and stores them into the dynamic memory using the component label $\hat{u}_c^i$ and the style label $\hat{y}_s$.

- Enc disassembles a source glyph into the several component features using the pre-defined decomposition function.
- We adopt multi-head structure (Thai: four & Korean: three ), one head per one component type.
- The encoded component-wise features are written into the dynamic memory

**dynamic memory**

- dynamic memory (DM) stores **encoded component features** of the given reference glyphs.
- encoded features in DM learn unique **local styles** depending on each font.

**persistent memory**

- persistent memory (PM) is a component-wise learned embedding that represents the **intrinsic shape** of each component and the **global information** of the script such as the compositionality.
- PM captures the **global information** of sub-glyphs independent to each font style.

Note that **DM simply stores and retrieves** the encoded features, but **PM is learned embedding** trained from the data. Therefore, DM is adaptive to the reference input style samples, while PM is fixed after training.

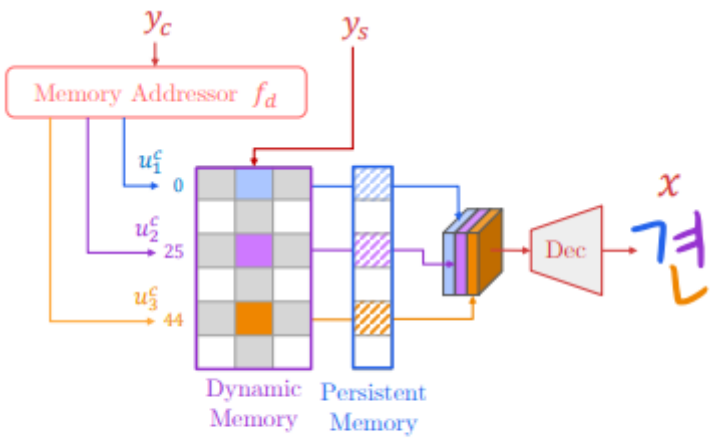
**Memory addressor**

- Memory addressor provides the access address of both dynamic and persistent memory based on the given character label $y_c$. We use pre-defined decomposition function
  $f_d : y_c \mapsto \{u_i^c \mid i = 1 \dots M_c\}$
  .

e.g.

“한” by $f_d(\text{“한”}) = \{\text{“ㅎ”}, \text{“ㅏ”}, \text{“ㄴ”}\}$

**Decoder**




(c) Decoding phase detail.

> The memory addressor loads the component features by the character label $y_c$ and feeds them to the decoder.
> In the decoding stage, decoder Dec generates a target glyph with the target character $y_c$ and the reference style $y_s$ using the component-wise features stored into the dynamic memory DM and the persistent memory PM.

**discriminator**

- For discriminator D, we use a multitask discriminator with the **font condition** and the **character condition**.
- The multitask discriminator has independent branches for each target class and each branch performs binary classification.
- Considering two types of conditions, we use two multitask discriminator, one for character classes and the other for font classes, with a shared backbone.

**component classifie**

We further use component classifier Cls to ensure the model to fully utilize the compositionality

**compositional generator**

Moreover, we introduce the global-context awareness and local-style preservation to the generator, called compositional generator

DM-Font learns the compositionality in the weakly-supervised manner; it does not require any exact component location, e.g., component-wise bounding boxes, but only component labels are required. Hence, DM-Font is not restricted to the font generation only, but can be applied to any generation task with compositionality, e.g., attribute conditioned generation tasks

**Experiments**

Pixel-level evaluation metrics assess the pixel structural similarity between the ground truth image and the generated image. We employ the **structural similarity index (SSIM)** and **multi-scale structural similarity index (MS-SSIM)**.

We report the **top-1 accuracy**, **perceptual distance** (PD), and **mean FID** (mFID) using the classifiers. PD is computed by L2 distance of the features between generated glyph and GT glyph, and mFID is a conditional FID [16] by averaging FID for each target class.

Table 1: **Quantatitive evaluation on the Korean-handwriting dataset.** We evaluate the methods on the seen and unseen character sets. Higher is better, except perceptual distance (PD) and mFID.

| | Pixel-level | | Content-aware | | | Style-aware | | |
|---|---|---|---|---|---|---|---|---|
| | SSIM | MS-SSIM | Acc(%) | PD | mFID | Acc(%) | PD | mFID |
| Evaluation on the **seen** character set during training | | | | | | | | |
| EMD [46] | 0.691 | 0.361 | 80.4 | 0.084 | 138.2 | 5.1 | 0.089 | 134.4 |
| FUNIT [30] | 0.686 | 0.369 | 94.5 | 0.030 | 42.9 | 5.1 | 0.087 | 146.7 |
| AGIS-Net [10] | 0.694 | 0.399 | **98.7** | **0.018** | 23.9 | 8.2 | 0.088 | 141.1 |
| DM-Font (ours) | **0.704** | **0.457** | 98.1 | **0.018** | **22.1** | **64.1** | **0.038** | **34.6** |
| Evaluation on the **unseen** character set during training | | | | | | | | |
| EMD [46] | 0.696 | 0.362 | 76.4 | 0.095 | 155.3 | 5.2 | 0.089 | 139.6 |
| FUNIT [30] | 0.690 | 0.372 | 93.3 | 0.034 | 48.4 | 5.6 | 0.087 | 149.5 |
| AGIS-Net [10] | 0.699 | 0.398 | 98.3 | 0.019 | 25.9 | 7.5 | 0.089 | 146.1 |
| DM-Font (ours) | **0.707** | **0.455** | **98.5** | **0.018** | **20.8** | **62.6** | **0.039** | **40.5** |

Table 2: **Quantatitive evaluation on the Thai-printing dataset.** We evaluate the methods on the seen and unseen character sets. Higher is better, except perceptual distance (PD) and mFID.

| | Pixel-level | | Content-aware | | | Style-aware | | |
|---|---|---|---|---|---|---|---|---|
| | SSIM | MS-SSIM | Acc(%) | PD | mFID | Acc(%) | PD | mFID |
| Evaluation on the **seen** character set during training | | | | | | | | |
| EMD [46] | 0.773 | 0.640 | 86.3 | 0.115 | 215.4 | 3.2 | 0.087 | 172.0 |
| FUNIT [30] | 0.712 | 0.449 | 45.8 | 0.566 | 1133.8 | 4.6 | 0.084 | 167.9 |
| AGIS-Net [10] | 0.758 | 0.624 | **87.2** | **0.091** | **165.2** | 15.5 | 0.074 | 145.2 |
| DM-Font (ours) | **0.776** | **0.697** | 87.0 | 0.103 | 198.7 | **50.3** | **0.037** | **69.4** |
| Evaluation on the **unseen** character set during training | | | | | | | | |
| EMD [46] | 0.770 | 0.636 | 85.0 | 0.123 | 231.0 | 3.4 | 0.087 | 171.6 |
| FUNIT [30] | 0.708 | 0.442 | 45.0 | 0.574 | 1149.8 | 4.7 | 0.084 | 166.9 |
| AGIS-Net [10] | 0.755 | 0.618 | 85.4 | 0.103 | **188.4** | 15.8 | 0.074 | 145.1 |
| DM-Font (ours) | **0.773** | **0.693** | **87.2** | **0.101** | 195.9 | **50.6** | **0.037** | **69.6** |


(a) Seen character set during training.


(b) Unseen character set during training.

Fig. 4: **Qualitative comparison on the Korean-handwriting dataset.** Visualization of generated samples with seen and unseen characters. We show insets of baseline results (green box), ours (blue box) and ground truth (red box). Ours successfully transfers the detailed style of the target style, while baselines fail to generate glyphs with the detailed reference style.


(a) Seen character set during training.


(b) Unseen character set during training.

Fig. 5: **Qualitative comparison on the Thai-printing dataset.** Visualization of generated samples with seen and unseen characters. We show insets of baseline results (green box), ours (blue box) and ground truth (red box). Overall, ours faithfully transfer the target style, while other methods even often fail to preserve contents in unseen character sets.