

Few-shot Font Generation with Localized Style Representations and Factorization

2020/11/xx @dl-study

著者

Song Park^{1,*} Sanghyuk Chun^{2,*} Junbum Cha² Bado Lee² Hyunjung Shim^{1,†}

¹ School of Integrated Technology, Yonsei University, ² Clova AI Research, NAVER Corp.

- 延世大学校（韓国の私立大学）
- Clova AI Research, NAVER Corp.

概要

- Few-shot フォント生成
- **LF-Font**を提案
- 新しいフォントセットを少量のフォントから作成
- Fine-tuningを使わない
- 基本的に中国語を対象に書かれています

生成結果

Reference	郝	攀	碱	荷	癩	捐	霄	睛														
Source	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
EMD	申	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
AGIS-Net	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
FUNIT	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
DM-Font	审	弱	无	峰	潮	税	博	城	尊	睛	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
Ours	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
GT	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群

- Oursが提案手法, GTが正解
- Reference8例から学習するfew-shotフォント生成

生成結果

Reference	郝	攀	碱	荷	癥	捐	霄	晴	
Source	审	弱	和	峰	潮	税	博	城	尊
EMD	审	弱	和	峰	潮	税	博	城	尊
AGIS-Net	审	弱	和	峰	潮	税	博	城	尊
FUNIT	审	弱	和	峰	潮	税	博	城	尊
DM-Font	审	弱	无	峰	潮	税	博	城	尊
Ours	审	弱	和	峰	潮	税	博	城	尊
GT	审	弱	和	峰	潮	税	博	城	尊

郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴
郝	攀	碱	荷	癥	捐	霄	晴

呼	猗	屋	群
呼	猗	屋	群
呼	猗	屋	群
呼	猗	屋	群
呼	猗	屋	群
呼	猗	屋	群
呼	猗	屋	群
呼	猗	屋	群

- Oursが提案手法, GTが正解
- Reference8例から学習するfew-shotフォント生成

モデル定義

	Style s	Character c	Components U_c
夏	s_1	夏	{一, 目, 丶, 夂}
冬	s_1	冬	{夂, 冫}
冬	s_2	冬	{夂, 冫}

モデル定義

画像 x に3つのアノテーションを定義

- ① Style label : $s \in S$
- ② Character label : $c \in \mathcal{C}$
- ③ Component label : $U_c = [U_1^c \cdots U_m^c]$ ※ m は c 内の構成要素数

モデル定義

画像 x に3つのアノテーションを定義

- ① Style label : $s \in S$
- ② Character label : $c \in C$
- ③ Component label : $U_c = [U_1^c \cdots U_m^c]$ ※ m は c 内の構成要素数

- c は人が決めたルールに従って U_c に分解される
- $|S| = 482, |C| = 19,514, |U| = 371$

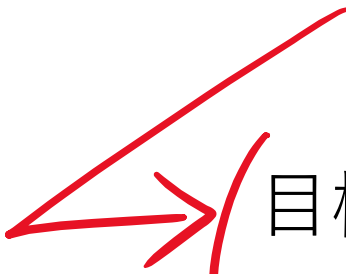
モデル定義

	Style s	Character c	Components U_c
夏	s_1	夏	{一, 目, 丶, 夂}
冬	s_1	冬	{夂, 冫}
𠂇	s_2	𠂇	{夂, 冫}

モデル概要

Few-shot font生成の目標は

$x_{\tilde{s}\tilde{c}} \in \chi_r$ (たとえば、 $|\chi_r| = 8$) から $x_{\tilde{s}c}$ をつくること。



目標スタイル \tilde{s}
 $c \in C$ すべての c に対して

モデル概要

これまでの研究は

Generator G

Encoders E_s, E_c として、以下の式で生成

$$x_{\tilde{s},c} = G(f_{\tilde{s}}, f_c),$$

$$f_{\tilde{s}} = E_s(\mathcal{X}_r) \text{ and } f_c = E_c(x_{s_0,c}),$$

↑
スタイルの
特徴

↑
8例

↑
文字の
特徴

↑
標準フォント
の「春」

↑
文字「春」

モデル概要

今回の研究では、**文字ごとの処理**→**構成要素ごとの処理に変更**
構成要素用スタイルエンコーダー $E_{s,u}$ として、
構成要素ごとのスタイルの特徴を算出

$$f_{s,u}(x, u) = E_{s,u}(x, u) \in \mathbb{R}^d$$

この $f_{s,u}(x, u)$ の sum が
文字単位のスタイルの特徴 $f_{s,c}$ になる

文字画像
↑
構成要素
ラベル
↑
 $u \in U_c$

モデル概要

$f_{\tilde{s}} \rightarrow f_{\tilde{s},c}$ に変化した

すると方程式は次のようになる

$$x(\tilde{s}, c) = G(f_{\tilde{s},c}, f_c),$$

$$f_c = E_c(x_{s_0,c}),$$

同じ

$$f_{\tilde{s},c} = \sum_{u \in U_c} f_{\tilde{s},u} = \sum_{u \in U_c} E_{s,u}(x_{\tilde{s},\tilde{c}_u}, u),$$

↑
文字単位の
スタイルの特徴

↑
構成要素単位の
スタイルの特徴

↑
構成要素 u を含む
 \tilde{c}_u という文字が描かれた
reference 内の画像

このように変化しています $f_{\tilde{s}} \rightarrow f_{\tilde{s},c}$ に変化

従来手法：

$$x_{\tilde{s},c} = G(f_{\tilde{s}}, f_c),$$
$$f_{\tilde{s}} = E_s(\mathcal{X}_r) \text{ and } f_c = E_c(x_{s_0,c}),$$

今回の手法：

$$x(\tilde{s}, c) = G(f_{\tilde{s},c}, f_c), \quad f_c = E_c(x_{s_0,c}),$$
$$f_{\tilde{s},c} = \sum_{u \in U_c} f_{\tilde{s},u} = \sum_{u \in U_c} E_{s,u}(x_{\tilde{s},\tilde{c}_u}, u),$$

ここまで

- ここまでで、構成要素ごとの処理ができるようになった
- でも、このままだと χ_r のサイズが大きくないといけない
→ 構成要素の数 $|U|$ が300ぐらいあるから。

→ χ_r に入っていない構成要素を補うことが必要だな～

モデル概要

$f_{s,u} \in \mathbb{R}^d$ を2要素に分解

→ (左はらい明朝体 に分解)

(Component factor $z_u \in \mathbb{R}^{k \times d}$
Style factor $z_s \in \mathbb{R}^{k \times d}$

※ k は factor の次元

このように分解

$$f_{s,u} = \mathbf{1}^\top (z_s \odot z_u),$$

↑
 $\mathbf{1} \in \mathbb{R}^k$
all-ones vector

↑
要素ごとの積

モデル概要

構成要素分解モジュール F_s, F_u を使って $f_{s,u}$ を2要素に分解

$$z_s = F_s(\underbrace{f_{s,u}}_{\text{変数}}; \underbrace{W, b}_{\text{パラメータ}}), \quad z_u = F_u(f_{s,u}; W, b).$$

z はどちらも次のように算出される $\left(\begin{array}{l} W = [w_1; \dots; w_k] \in \mathbb{R}^{k \times d} \\ b \in \mathbb{R}^k \end{array} \right)$

$$\underbrace{z}_{k \times d \text{ 次元}} = [\underbrace{w_1}_{d \text{ 次元}} \odot \underbrace{f_{s,u}}_{d \text{ 次元}} + b_1; \dots; w_k \odot f_{s,u} + b_k].$$

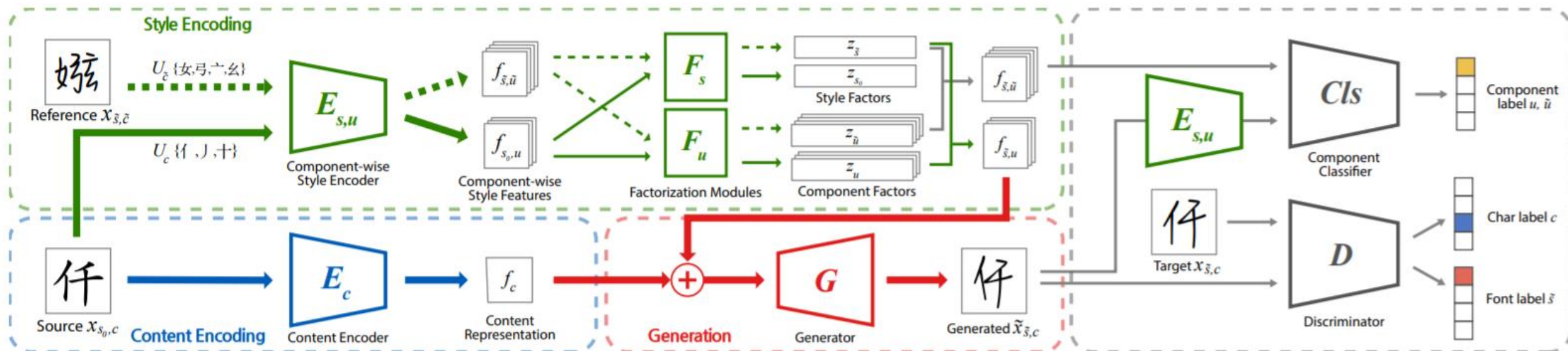
モデル概要

- z_s, z_u は構成要素の style factor, component factor
 - この算出では
 - 同じスタイルが同じ z_s にならない
 - 同じ構成要素が同じ z_u にならない という問題がある
- F_s と F_u を $\mathcal{L}_{consist}$ を最小化するように学習

$$\mathcal{L}_{consist} = \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \|F_s(f_{s,u}) - \mu_s\|_2^2 + \|F_u(f_{s,u}) - \mu_u\|_2^2,$$

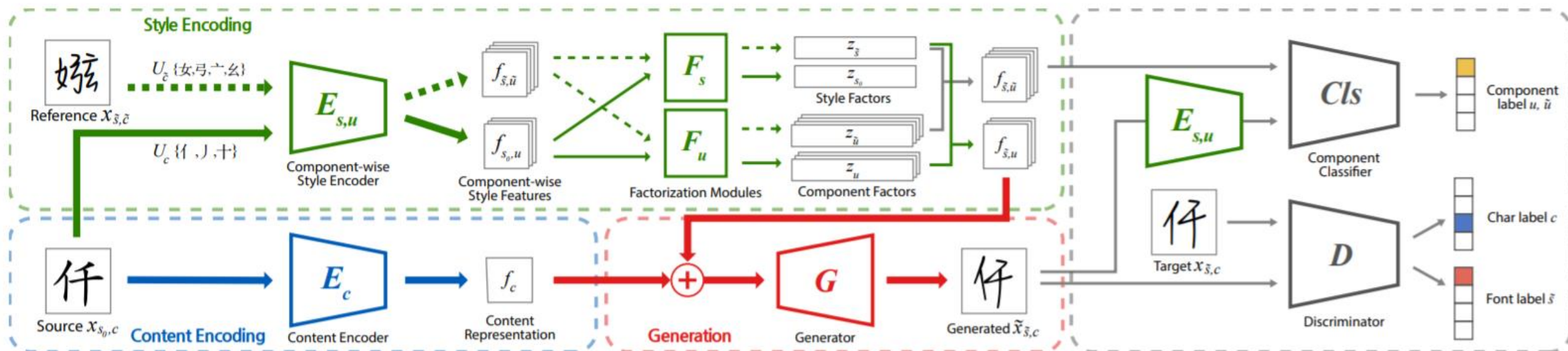
$$\mu_s = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} F_s(f_{s,u}), \quad \mu_u = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} F_u(f_{s,u}).$$

モデル概要



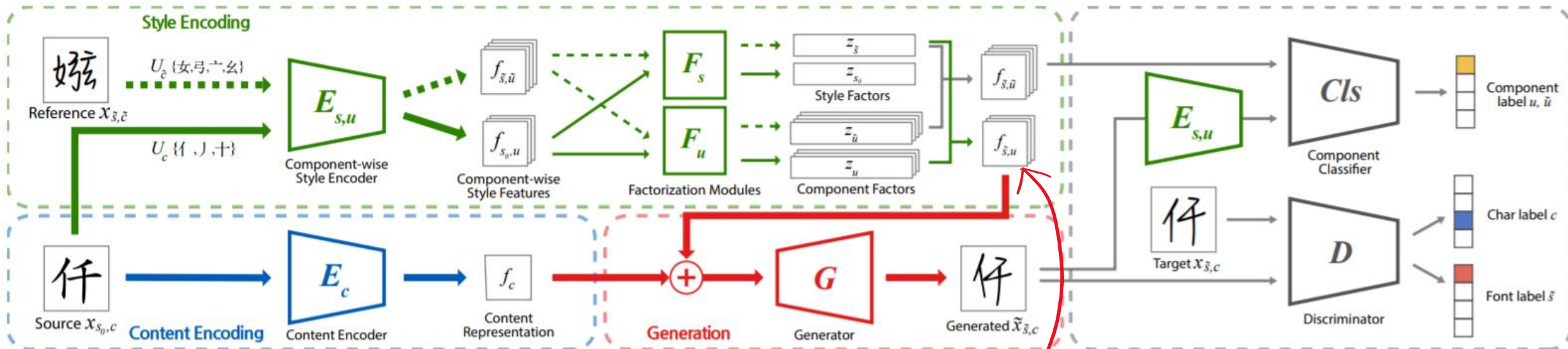
学習の時にはComponent ClassifierとDiscriminatorを使う
(どちらもlossを出して学習に貢献している)

モデル概要



学習の時にはComponent ClassifierとDiscriminatorを使う
(どちらもlossを出して学習に貢献している)

構成要素単位で「エンコード」→ 構成要素の「特徴量」→ それを (Z_s : style factor, Z_u : component factor) に分解 → 構成要素の「特徴量」を再構築

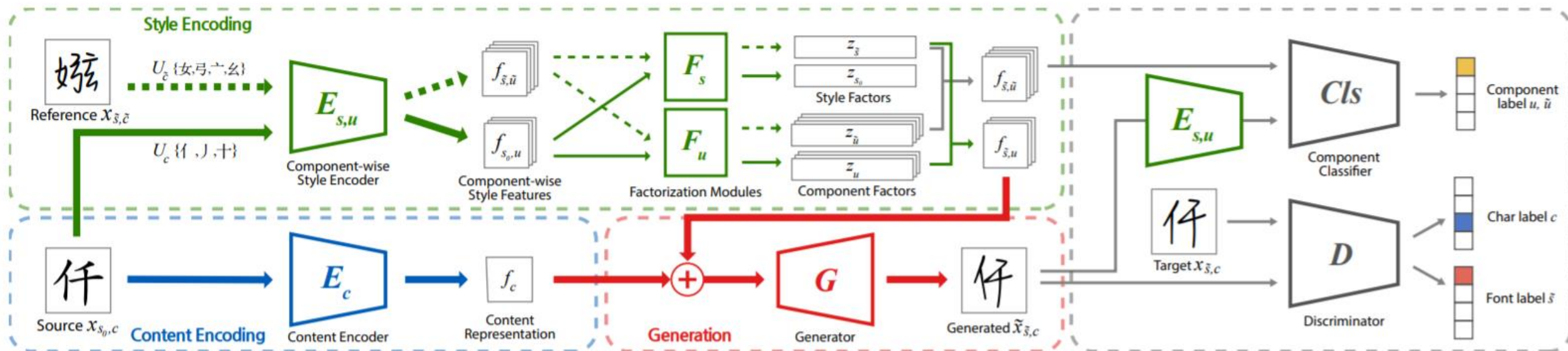


点線(好玄)の Z_s (style factor)
 実線(仟)の Z_u (component factor)
 から $f_{\tilde{s}, u}$ (構成要素の特徴量)
 を再構築したもの

補足

{口, 口, 人} → 员 呐 另 呗

- 同じ構成要素が少しずつ違う使われ方をする
- はねの有無とかまで反映されていてすごい
- このような構造的情報は、 E_c が担っている



LOSS

最適化対象

- スタイルエンコーダー $E_{s,u}$
- 文字エンコーダー E_c
- 構成要素(を)分解モジュール F_s, F_u
- Generator G

LOSS

DとGどちらかが一方的に強くなりすぎることを防いでくれる。

- Adversarial loss

Discriminator, Generatorの学習に使われる。 The hinge GAN Loss を使っている

$$\mathcal{L}_{adv}^D = -\mathbb{E}_{(x,s,c)\sim p_{data}} \max(0, -1 + D_{s,c}(x)) \\ - \mathbb{E}_{(\tilde{x},s,c)\sim p_{gen}} \max(0, -1 - D_{s,c}(\tilde{x}))$$

$$\mathcal{L}_{adv}^G = -\mathbb{E}_{(\tilde{x},s,c)\sim p_{gen}} D_{s,c}(\tilde{x}).$$

LOSS

- L1 loss and feature matching loss

pixel-level, feature-levelで生成画像が正解に近くなることを促進する

$$\mathcal{L}_{l1} = \mathbb{E}_{(x,s,c) \sim p_{data}} [\|x - \tilde{x}\|_1],$$

$$\mathcal{L}_{feat} = \mathbb{E}_{(x,s,c) \sim p_{data}} \left[\sum_{l=1}^L \|D_f^{(l)}(x) - D_f^{(l)}(\tilde{x})\|_1 \right]$$

↑
xの
特徴量
(正解画像)↑
x̃の
特徴量
(生成画像)

Loss

- Component-classification loss

構成要素の特徴量 $f_{s,u}$ を構成要素ラベル u に分類

$$\mathcal{L}_{cls} = \sum_{\tilde{u} \in U_{\tilde{c}}} \text{CE}(\text{Cls}(f_{s,\tilde{u}}), \tilde{u}) + \sum_{u \in U_c} \text{CE}(\text{Cls}(f_{s,u}), u),$$

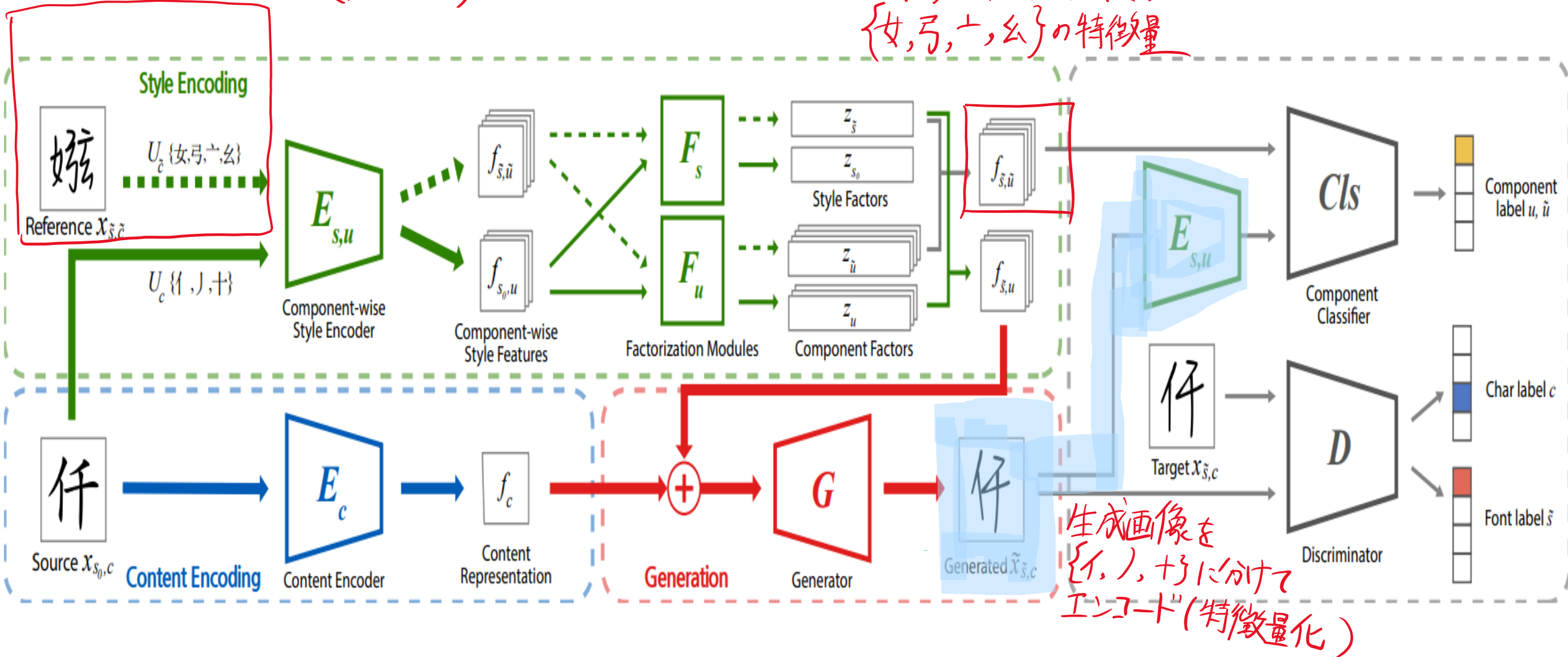
Reference 画像 Source 画像

CE : Cross Entropy

Cls : Classifier

Reference 画像 (点線)

い3い3計算した後の
{女, 弓, 十, 幺} の特徴量



LOSS

- Full objective

最終的なLossはこのようなになります

$$\min_{\substack{E_c, E_{s,u}, G, \\ F_s, F_u, Cls}} \max_D \mathcal{L}_{adv}(font) + \mathcal{L}_{adv}(char) + \lambda_{L1} \mathcal{L}_{L1} \\ + \lambda_{feat} \mathcal{L}_{feat} + \lambda_{cls} \mathcal{L}_{cls} + \lambda_{consist} \mathcal{L}_{consist},$$

- 今までのロスハイパーパラメータ λ を使って調整する

評価方法

- Visual Quality を評価

LPIPS : 2つの画像の類似度を perceptual similarity により計測

- Contentの保存, Style適応を評価

Style-aware、Content-awareな Classifierを特徴量抽出のために使い、その特徴量をFrechet inception distance(FID) で計測して評価

他モデルとの性能比較

		LPIPS ↓	Acc (S) ↑	Acc (C) ↑	Acc (Hmean) ↑	FID (S) ↓	FID (C) ↓	FID (Hmean) ↓
Seen chars	SA-VAE (IJCAI'18)	0.310	0.2	41.0	0.3	231.8	66.7	103.6
	EMD (CVPR'18)	0.248	11.9	63.7	20.1	148.1	25.7	43.8
	AGIS-Net (TOG'19)	0.182	34.0	99.8	50.7	79.8	4.0	7.7
	FUNIT (ICCV'19)	0.217	39.0	97.1	55.7	58.5	3.6	6.8
	DM-Font (ECCV'20)	0.275	10.2	72.4	17.9	151.8	8.0	15.2
	LF-Font (proposed)	0.169	75.6	96.6	84.8	40.4	2.6	4.9
Unseen chars	EMD (CVPR'18)	0.250	11.6	64.0	19.7	151.7	41.4	65.0
	AGIS-Net (TOG'19)	0.189	33.3	99.7	49.9	85.4	10.0	18.0
	FUNIT (ICCV'19)	0.216	38.0	96.8	54.5	63.2	12.3	20.6
	DM-Font (ECCV'20)	0.284	11.1	53.0	18.4	153.4	26.5	45.2
	LF-Font (proposed)	0.169	72.8	97.1	83.2	44.5	8.7	14.6

質的評価（というか、これまでのモデルの問題点）

Reference	郝	攀	碱	荷	癩	捐	霄	晴														
Source	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
EMD	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
AGIS-Net	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
FUNIT	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
DM-Font	审	弱	无	峰	潮	税	博	城	尊	晴	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
Ours	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群
GT	审	弱	和	峰	潮	税	博	城	尊	精	崗	棒	常	瑶	晁	省	城	荷	呼	猗	屋	群

青：ディテール損失

赤：正しくない文字

緑：ソースに頼りすぎ

黄：間違った構成要素

Ablation Study

Style representation f_s	Acc (S) ↑	Acc (C) ↑	Acc (Hmean) ↑
AGIS-Net	33.3	99.7	49.9
FUNIT	38.0	96.8	54.5
Universal without $E_{s,u}$	33.6	97.2	49.9
Universal with $E_{s,u}$	52.8	95.9	68.1
Localized with $E_{s,u}$	72.8	97.1	83.2

E_s
 $E_{s,u}$
 $E_{s,u} + f_s, f_u$

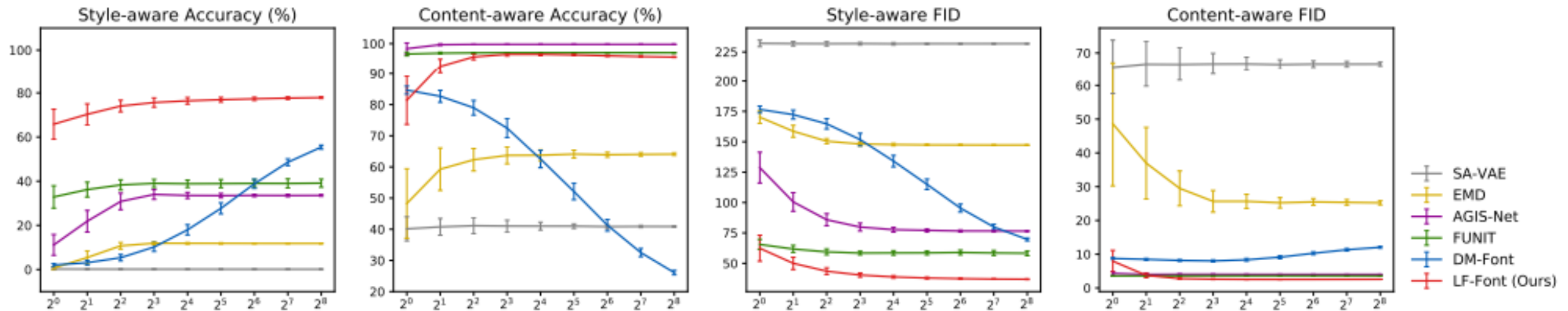
- 文字単位→構成要素単位の処理に切り替えたことは、**Styleの質に大幅に向上**させ**Contentの質を少しだけ低下**させていることがわかる

Ablation Study

Accuracies	Few-shot			Many-shot		
	S	C	H	S	C	H
DM-Font	11.1	53.0	18.4	51.8	15.0	23.2
LF-Font without E_c	36.3	15.4	21.7	37.8	5.1	8.9
LF-Font	72.8	97.1	83.2	74.7	96.5	84.2

- E_c がないと性能が落ちる（構成要素同士の位置関係が保てない）

Reference数 $|\chi_r|$



- いずれも $2^3 \sim 2^4$ ぐらいで十分そう
- Style より Content の方が $|\chi_r|$ が少なくてもよさそう ($2^1 \sim 2^2$)

Reference数 $|x_r|$

	潮 潮 \leftarrow target									城 城								
EMD																		
AGIS-Net																		
FUNIT																		
DM-Font																		
Ours																		
References	1	2	4	8	16	32	64	128	256	1	2	4	8	16	32	64	128	256

One-shot生成

さすがに
難しすぎて草

Source	弱	和	税	城	尊	棒	瑶	晁	城	呼
→	才	菊	和	城	尊	棒	瑶	名	中	呼
	元	弱	和	城	尊	棒	瑶	晁	城	呼
	項	弱	和	税	尊	棒	瑶	晁	城	呼
Target	弱	和	税	城	尊	棒	瑶	晁	城	呼

- （当たり前だけど）複雑な漢字を例に選んだ方がいいことがわかる
- 「弱」とか苦戦してそうですが、One-Shotとは思えない出来栄え

スタイル補間

推 推 推 推 推 推 推 推 推 推
度 度 度 度 度 度 度 度 度 度
弓 弓 弓 弓 弓 弓 弓 弓 弓 弓
定 定 定 定 定 定 定 定 定 定

- LF-FontのStyleの特徴量が意味深いことを見せるためにこの画像を示している（らしい）
- Styleの特徴量をいじれば2フォントを混ぜたり似てるフォントをソートできるとかそういうこと？

韓国語生成

Reference	탕년져은	탕년져은	탕년져은	탕년져은
AGIS-Net	죄룩맷몽	괴룩맷몽	섯웁연캐	죄뎡컨깁
DM-Font	죄룩맷몽	괴룩맷몽	섯웁연캐	죄뎡컨깁
LF-Font	죄룩맷몽	괴룩맷몽	섯웁연캐	죄뎡컨깁
GT	죄룩맷몽	괴룩맷몽	섯웁연캐	죄뎡컨깁

	LPIPS ↓	Acc (S) ↑	Acc (C) ↑	Acc (Hmean) ↑	FID (S) ↓	FID (C) ↓	FID (Hmean) ↓
AGIS-Net (TOG'19)	0.188	3.9	97.5	7.5	108.1	7.8	14.5
DM-Font (ECCV'20)	0.266	3.4	96.3	6.5	126.3	19.0	33.0
LF-Font (proposed)	0.145	41.6	98.4	58.5	47.2	4.9	8.9

まとめ

- Few-shotフォント生成モデル「LF-Font」を提案した。
- 構成要素単位の処理にすることでよりスタイルを反映させたフォントが作れるようになった。
- 構成要素を2ファクターに分解することによって未知の文字に対する性能も向上した。