글자 단위 텍스트 인식 기반의 이미지 내 한글 글꼴 분류 시스템 개발

유사라[1], 김윤주[2], 송지효[2], 이기용[2]
[1]숙명여자대학교 컴퓨터과학과
[2]숙명여자대학교 소프트웨어학부

{rrrr4ra, angelkim0409, chrisjihyo, kiyonglee}@sookmyung.ac.kr

Development of a Korean Font Classification System for Images Based on Syllable-Level Text Recognition

Sara Yu[1], Kim Yoon-Ju[2], Song Ji-Hyo[2], Ki Yong Lee[2]
[1]Department of Computer Science, Sookmyung Women's University
[2]Division of Computer Science, Sookmyung Women's University

요 약

이미지 내 글꼴을 파악하는 것은 디자인 자료 제작, 저작권 확인 등 다양한 곳에서 중요한 문제이다. 하지만 이미지 내 한글 글꼴을 자동으로 식별하는 시스템은 아직 존재하지 않으며, 수동으로 한글 글꼴을 파악하는 것은 시간과 정확도 측면에서 매우 비효율적이다. 따라서 본 논문에서는 이미지 내 한글 글꼴을 자동으로 인식하는 시스템을 개발한다. 본 논문에서 개발한 시스템은 크게 두 가지 기법을 사용한다: (1) 한글의 기하학적인 특성을 활용하여 글자 단위로 텍스트를 인식하며, (2) 단어가 아닌 글자 단위로 글꼴을 분류하고 각 글자에 대한 글꼴 분류 결과를 종합하여 최종적인 글꼴 분류 결과를 얻는다. 10가지 한글 글꼴이 나타나는 직접 제작한 이미지를 사용하여 시스템의 성능을 평가한 결과 제안 방법은 비교 방법에 비해 더욱 정확히 한글 글꼴을 분류함을 확인하였다.

1. 서론

글꼴이란 인쇄물 혹은 컴퓨터에서 사용되는 글자의 디자인을 말하는 것으로, 이미지 내 글꼴을 파악하는 것은 다자인 자료 제작, 저작권 확인, 문서 작업 등 다양한 분야에서 중요한 문제이다. 하지만 이미지 내 한글 글꼴을 자동으로 식별하는 시스템은 아직 존재하지 않는 것으로 파악되며, 수동으로 한글 글꼴을 파악하는 것은 시간과 정확도 측면 모두에서 매우 비효율적이다.

최근 컴퓨터 비전(computer vision) 분야의 발전과 함께 객체 탐지(object detection)의 정확도도 크게 향상되었다. 객체 탐지란 이미지 내 객체의 종류와 위치를 식별하는 작업을 말하며, 특히 이미지 내의 텍스트를 탐지하는 장면 텍스트 탐지(scene text detection) 연구가 활발히 진행되고 있다 [1]. 하지만 텍스트 탐지에 비해 글꼴 인식(font recognition)은 아직 연구가 상대적으로 부족한 상황이다. 더욱이 영어의 알파벳과 한자에 대해서는 이미 글꼴 인식이 일부 연구된 바가 있으나 [2], 한글에 대해서는 아직 글꼴 인식이 깊이 연구된 바가 없다 [3].

따라서 본 논문은 이미지 내 한글 글꼴을 자동으로 인식하는 시스템을 개발한다. 본 논문에서 개발한 시스템은 한글 글꼴을 좀 더 정확하게 인식하기 위하여 크게 다음 두 가지 기법을 사용한다.

(1) **글자 단위 텍스트 탐지:** 제안 방법은 단어가 아닌 글자 단위로 한글 텍스트를 탐지하고(예: '꿀벌' → '꿀', '벌'), 각 글자별로 글꼴을 분류한다. 이는 글꼴 분류 모델의 훈련을 더욱 쉽게 만드는 한편 글꼴을 좀 더 정확히 분류할 수 있도록 한다.

(2) **최종 글꼴 분류:** 제안 방법은 각 글자별로 글꼴을 분류한 후, 각 글자에 대한 글꼴 분류 결과를 종합하여 전체 텍스트의 글꼴을 결정한다. 따라서 일부 글자에 대한 글꼴 분류가 확실하지 않더라도 모든 글자에 대한 글꼴 분류 결과를 사용하여 좀 더 정확히 글꼴을 분류할 수 있다.

제안 방법의 성능을 10가지 종류의 한글 글꼴이 나타나는 이미지를 직접 제작하여 평가한 결과, 제안 방법은 비교 방법에 비해 정확도, 정밀도, 재현율, F1-점수 모든 기준에서 인식 성능을 높임을 확인하였다. 본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 간략히 설명하고, 3장에서는 제안 방법을 자세히 설명한다. 4장에서는 성능 평가 결과를 보이며, 5장에서는 결론을 맺고 마무리한다.

2. 관련 연구

2.1 한글 글꼴 분류

한글 글꼴 분류는 이미지 내 한글 텍스트에 사용된 글꼴을 판별하는 작업이다. 글꼴은 디자인, 간격, 크기, 모양 등 다양한 요소로 구성된다. [3]은 형태, 특성, 쓰임새, 창작 의도 등 다양한 기준을 사용하여 한글 글꼴에 대한 분류 체계를 수립하였지만, 이는 한글 글꼴을 자동으로 인식하는 방법에 관한 연구는 아니다. 한편 [2]는 합성곱 신경망(convolutional neural network, CNN)을 이용하여 2,383종의 글꼴을 인식하는 시스템인 DeepFont를 제안하였다. 하지만 이는 한글이 아닌 영어의 알파벳과 한자에 대한 글꼴 인식 연구로서 아직까지 한글에 대한 글꼴 인식 연구는 깊게 진행된 바가 없다.

2.2 장면 텍스트 탐지

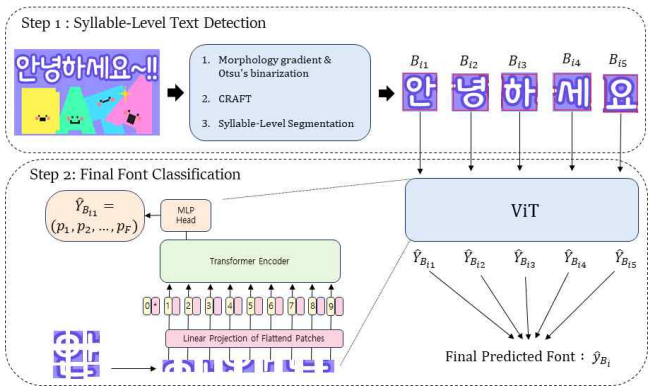
장면 텍스트 탐지(scene text detection)는 이미지 내에 존재하는 텍스트를 탐지하고 해당 영역을 검출하는 것을 목표로 한다. CRAFT[4]는 단어 수준(word-level)으로 경계 상자(bounding box)를 만들어 모델을 학습하던 05 방법의 취약점을 보완하여 개별 문자 수준(character-level)으로 모델을 학습시킬 수 있는 새로운 방법을 제안하였다. 이에 따라 CRAFT는 다양한 모양의 텍스트 영역도 비교적 정확히 검출할 수 있다. 본 논문에서는 이미지 내의 한글 영역을 검출하기 위해 CRAFT를 사용한다.

2.3 이미지 분류

이미지 분류(image classification)는 주어진 이미지를 사전에 정의된 클래스 중 하나로 분류하는 작업이다. 인공신경망을 활용한 이미지 분류는 전통적으로 CNN을 주로 사용했으나 최근에는 자연어 처리에서 우수한 성공을 거둔 트랜스포머(transformer) 모델을 컴퓨터 비전에 적용하고자 하는 노력이 계속되고 있다. 최근에는 트랜스포머를 이미지 분류에 적용한 비전 트랜스포머(vision transformer, ViT)가 제안되었다 [5]. ViT는 이미지를 작은 크기의 패치(patch)들로 조각낸 후, 이들을 시퀀스로 나타내어 트랜스포머 인코더의 입력으로 넣는다. ViT는 이들 패치 간의 관계를 어텐션 메커니즘(attention mechanism)을 사용하여 효과적으로 파악함으로써 이미지 분류에 뛰어난 성능을 내는 것으로 알려져 있다. 본 논문에서는 각 글자의 글꼴을 분류하기 위해 ViT를 사용한다.

3. 제안 방법

본 장에서는 이미지 내 한글 글꼴을 정확히 분류하기 위해 본 논문에서 개발한 시스템을 설명한다. (그림 1)은 제안 시스템의 전체 흐름도이다. 제안 시스템은 크게 (1) 글자 단위 텍스트 탐지 단계와 (2) 최종 글꼴 분류 단계로 구성된다. 아래에서는 이 두 단계를 각각 자세히 설명한다.



(그림 1) 제안 시스템의 전체 흐름도

3.1 글자 단위 텍스트 탐지

이 단계에서는 이미지 내 한글 텍스트 영역을 검출하고 해당 영역을 글자 단위로 분할한다. 제안 방법은 이미지 내 한글 텍스트 영역을 검출하기 전에 다음 두 종류의 전처리를 통해 한글 텍스트 영역이 좀 더 정확히 검출되도록 한다.

- 모폴로지 그래디언트(morphological gradient): 글자의 가장자리를 강조하고 배경과 글자 사이의 경계를 뚜렷하게 만든다. 이를 통해 추후 텍스트 영역을 글자 단위로 더 정확히 분할 할 수 있다.

- 오츠 이진화(Otsu's binarization): 이미지의 픽셀 값 분포를 기반으로 최적의 임곗값을 찾아 흑과 백의 이진 영상으로 변환한다. 이때 글자 영역은 검은색, 배경 영역은 흰색으로 표현된다.

상기 전처리가 완료되면 2.2절에서 설명한 CRAFT 모델[4]을 사용하여 이미지 내의 텍스트 영역을 검출한다. 이미지 내에서 검출된 N개의 텍스트 영역을 $B_1, B_2, ..., B_N$라 하자. 검출된 각 텍스트

영역 $B_i (i = 1, ..., N)$은 하나 또는 그 이상의 글자들이 하나의 가로줄 형태로 나열된 영역을 나타내며, 영역에 속한 모든 글자를 포함하는 경계 상자를 나타내기 위해 $B_i = (x_i, y_i, w_i, h_i)$로 표현한다. 여기서 (x_i, y_i)는 좌측 상단 모서리의 좌표이며, w_i는 너비, h_i는 높이를 나타낸다.

CRAFT에 의해 텍스트 영역 $B_1, B_2, ..., B_N$이 검출되면 후처리를 통해 각 $B_i (i = 1, ..., N)$를 글자 단위의 영역으로 분할한다. 한글의 기하학적인 특성상 각 글자의 너비와 높이는 대체로 1:1의 비율을 가진다. 이것은 한글의 각 글자는 초성, 중성, 종성이 조합되어 사각형의 형태를 가지기 때문이다. 이를 이용하여 본 논문에서는 각 B_i를 다음과 같이 분할한다. 먼저 B_i에 포함된 글자 개수 n_i를 다음 (식 1)으로 추정한다.

$$n_i = \left\lceil \frac{w_i}{h_i} \right\rceil \quad (1)$$

이후 B_i를 (그림 1)의 Step 1에서 나타낸 것처럼 n_i개의 영역 $B_{i1}, B_{i2}, ..., B_{in_i}$로 분할한다. 각 영역 $B_{ij} (j = 1, ..., n_i)$은 B_i가 포함하고 있는 텍스트 중 j번째 글자만 포함하고 있는 영역을 나타낸다. B_{ij}는 아래 (식 2)와 같이 정의된다.

$$B_{ij} = (x_i + (j-1)\frac{w_i}{n_i}, y_i, \frac{w_i}{n_i}, h_i) \; (j = 1, ..., n_i) \quad (2)$$

즉, B_{ij}는 좌측 상단 모서리의 좌표가 $(x_i + (j-1)(w_i/n_i), y_i)$이고 너비가 w_i/n_i이며 높이가 h_i인 경계 상자를 나타낸다. 이렇게 B_i를 $B_{i1}, B_{i2}, ..., B_{in_i}$로 분할한 후, 단일 모음, 단일 자음 또는 특수 문자를 포함하고 있는 영역은 한글 글꼴 분류 모델의 성능을 저하시키므로 제거한다.

3.2 최종 글꼴 분류

이미지에서 검출된 각 텍스트 영역 B_i를 한 글자씩만 포함하고 있는 영역 $B_{i1}, B_{i2}, ..., B_{in_i}$으로 분할하고 나면, 제안 방법은 각 B_i에 대해 그의 글꼴을 다음과 같이 분류한다.

제안 방법은 B_i를 분할한 $B_{i1}, B_{i2}, ..., B_{in_i}$을 각각 개별적으로 2.3절에서 설명한 ViT[5]의 입력으로 넣는다. ViT는 각 $B_{ij} (j = 1, ..., n_i)$에 대해 그의 글꼴을 분류한 결과를 출력한다. ViT는 내부적으로 B_{ij}를 크기가 $P \times P$인 동일한 크기의 패치들로 분할하여 총 $[(w_i/n_i)h_i]/P^2$개의 패치들을 얻는다. 그 후 이들을 차례대로 나열하여 시퀀스로 만든 후 트랜스포머 인코더의 입력으로 사용한다. 이때 시퀀스를 구성하는 각 패치에 대해 이미지 내에서의 위치 정보를 보존하기 위해 위치 임베딩을 추가한다. 이후 ViT는 셀프 어텐션 및 피드 포워드 네트워크(feed forward network)로 구성된 L개의 트랜스포머 층을 통해 B_{ij}의 특징을 추출한다. 이렇게 B_{ij}의 특징을 추출한 후 ViT는 최종적으로 Softmax 층을 통해 다음 (식 3) 형태의 예측 확률 벡터 $\hat{Y}_{B_{ij}}$를 출력한다.

$$\hat{Y}_{B_{ij}} = (p_1, p_2, ..., p_F) \quad (3)$$

여기서 F는 분류하고자 하는 글꼴의 개수를 나타내며, $p_k (k = 1, ..., F)$는 B_{ij}에 포함된 글자가 k번째 글꼴일 확률을 나타낸다. 이러한 과정을 통해 각 $B_{ij} (j = 1, ..., n_i)$에 대한 예측 확률 벡터 $\hat{Y}_{B_{ij}}$가 얻어지면 제안 방법은 아래 (식 4)를 사용하여 $\hat{Y}_{B_{ij}}$ $(j = 1, ..., n_i)$들의 평균 벡터 $\overline{Y}_{B_i}$를 구한다.

$$\overline{Y}_{B_i} = (\overline{p}_1, \overline{p}_2, ..., \overline{p}_F) = \frac{1}{n_i}\sum_{j=1}^{n_i} \hat{Y}_{B_{ij}} \quad (4)$$

$\overline{Y}_{B_i}$의 각 원소 $\overline{p}_k (k = 1, ..., F)$는 B_i에 포함된 글자들이 k번째 글꼴일 평균 확률을 나타낸다. 마지막으로 제안 방법은 (식 5)가 나타내는 바와 같이 $\overline{Y}_{B_i}$의 원소 중 값이 가장 큰 원소에 대응되는 클래스를 B_i의 글꼴 $\hat{y}_{B_i}$로 최종 분류한다.

$$\hat{y}_{B_i} = \arg\max_{k \in 1, ..., F} \overline{p}_k \quad (5)$$

제안 방법은 이러한 과정을 각 $B_i (i = 1, ..., N)$에 대해 반복하여 이미지 내 각 텍스트 영역의 글꼴을 분류한다.

4. 성능 평가

본 장에서는 본 논문에서 개발한 한글 글꼴 분류 시스템의 성능을 평가한다.

- **데이터셋**: 훈련 데이터로는 한글 무료 라이선스 글꼴 10가지(굴림, 궁서, 나눔고딕, 바탕, 돋움, 신명조, 함컴소망M, 한컴윤고딕760, 함초롱바탕, 휴먼둥근헤드라인)에 대해 각 글꼴별로 11,172자씩

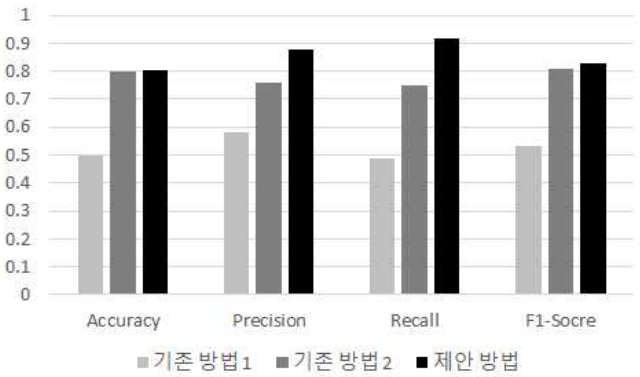
총 111,720개(11,172자×10가지)의 직접 제작한 이미지를 사용하였다. 실험에서는 데이터를 5:3:2로 분할하여 각각 훈련, 검증, 테스트 데이터로 사용하였다.

- **모델 훈련**: CRAFT 모델은 [6]에서 제공하는 사전 훈련된 모델을 사용하였으며, ViT 모델은 [7]에서 제공하는 모델을 받아와 앞서 설명한 데이터로 미세 조정(fine tuning)을 수행하였다. 이때 학습률은 0.001, 배치 사이즈는 256, 에포크는 100, 옵티마이저는 Adam을 사용하였다

- **비교 방법**: 본 실험에서는 제안 방법의 성능을 다음 2가지 방법과 비교하였다: (1) baseline 1: 3.1절과 3.2절에서 설명한 기법을 사용하지 않고 이미지 전체에 대해 ViT를 적용하여 한글 글꼴을 바로 분류하는 방법, (2) baseline 2: 3.1절에서 설명한 전처리 및 글자 단위 텍스트 탐지를 적용하지 않고 단어 단위 텍스트 탐지를 한 후 탐지된 영역을 바로 ViT에 넣어 분류하는 방법.

<표 1> 성능 평가 결과

Method	Performance			
	Accuracy	Precision	Recall	F1-score
baseline 1	0.500	0.582	0.487	0.530
baseline 2	0.797	0.757	0.748	0.808
proposed	**0.801**	**0.878**	**0.918**	**0.830**



(그림 2) 성능 평가 결과 시각화

<표 1>과 (그림 2)는 제안 방법과 2가지 비교 방법의 성능을 비교한 결과이다. 성능은 정확도(accuracy), 정밀도(precision), 재현율(recall), F1-점수(F1-score)로 평가하였으며, 제안 방법은 모든 척도에서 비교 방법에 비해 좋은 성능을 보인다. 따라서 본 논문에서 제안하는 글자 단위 텍스트 탐지 및 글자 단위 글꼴 분류에 기반한 기법은 한글 글꼴을 정확히 분류하는 데 효과적임을 알 수 있다. 이것은 단어 단위 이미지로 글꼴을 분류하는 것보다 글자 단위 이미지로 글꼴을 분류하는 것이 글꼴 분류 모델의 예측을 더 쉽게 만들기 때문이라 판단된다.

5. 결론

본 논문은 이미지 내 한글 글꼴을 더욱 정확히 분류하는 시스템을 개발하였다. 제안 시스템은 비교 방법과 달리 (1) 단어가 아닌 글자 단위로 텍스트 영역을 탐지하고, (2) 글자 단위로 글꼴을 분류한 뒤 이를 기반으로 전체 글꼴을 판단한다. 10가지 한글 글꼴이 포함된 실제 이미지 데이터로 제안 방법의 성능을 평가한 결과, 제안 방법은 비교 방법에 비해 글꼴 분류 정확도를 더욱 향상시킴을 확인하였다.

Acknowledgement

이 성과는 2022학년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF2022H1D8A303739411)

참고문헌

[1] F. Naiemi, V. Ghods, and H. Khalesi, "Scene text detection and recognition: a survey," Multimedia Tools and Applications, 81(14):20255-20290, 2022.

[2] Z. Wang, et al., "DeepFont: Identify Your Font from An Image." The 23rd ACM International Conference on Multimedia, New York, USA, 2015, pp. 451 - 459.

[3] J. Park, et al, "Hangeul Font Classification System Proposal of a Network Structure for Improved Font Search Result Similarity," Archives of Design Research, 36(2):145-169, 2023.

[4] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character Region Awareness for Text Detection", IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019, pp. 9365-9374.

[5] A. Dosovitskiy, L. Beyer, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," International Conference on Learning Representations, 2021.

[6] CRAFT, https://github.com/clovaai/CRAFT-pytorch.

[7] Vision Transformer, https://github.com/google-research/vision_transformer.