The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 20, No. 3, pp.89-95, Jun. 30, 2020. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2020.20.3.89
JIIBC 2020-3-13

심볼마커를 사용한 딥러닝 기반 모바일 응용 UI 요소 인식

UI Elements Identification for Mobile Applications based on Deep Learning using Symbol Marker

박지수*, 정진만, 은성배**, 윤영선****

Jisu Park*, Jinman Jung**, Seungbae Eun**, Young-Sun Yun**

요 약 최근 딥러닝을 사용하여 스케치이미지에 있는 GUI(Graphical User Interface) 요소를 인식하여 어플리케이션 구현에 필요한 코드를 자동 생성하는 연구 등이 있다. UI/UX 디자이너는 모바일 응용 프로그램 개발 시 스토리보드를 개발자와의 의사소통을 돕는 도구로 사용하나 모호한 위젯에 대해서는 UI/UX 디자이너의 의도와 다르게 구현되는 경우가 종종 발생한다. 본 논문에서는 DNN(Deep Neural Network) 기반의 GUI 요소 식별의 정확성을 높이기 위해 심볼마커를 사용하는 자동 GUI 요소 인식 기법을 제안한다. 심볼마커의 성능평가를 위해 심볼마커의 유무에 따라 실험을 진행하여 정확도를 평가하였고, 정확도 개선을 위해 원형과 괄호형으로 나누어 심볼마커 모양에 따른 결과를 분석하였다. 심볼마커를 사용한다면 개발자에게 정확한 의사 전달이 가능해져 피드백이 줄면서 시간과 비용이 감소하고 스케치이미지의 UI 요소 오탐률을 줄이고 정확성이 향상될 것으로 기대한다.

Abstract Recently, studies are being conducted to recognize a sketch image of a GUI (Graphical User Interface) based on a deep learning and to make it into a code implemented in an application. UI / UX designers can communicate with developers through storyboards when developing mobile applications. However, UI / UX designers can create different widgets for ambiguous widgets. In this paper, we propose an automatic UI detection method using symbol markers to improve the accuracy of DNN (Deep Neural Network) based UI identification. In order to evaluate the performance with or without the symbol markers, their accuracy is compared. In order to improve the accuracy according to of the symbol marker, the results are analyzed when the shape is a circle or a parenthesis. The use of symbol markers will reduce feedback between developer and designer, time and cost, and reduce sketch image UI false positives and improve accuracy.

Key Words : Deep Learning, Symbol Marker, Widget Recognition

Ⅰ. 서 론

모바일 응용 프로그램을 제작하는 과정에서 UI/UX 디자이너들은 Illustrator, Fireworks, InDesign, Keynote 그리고 Balsamiq 등 디자인 도구를 사용한다 [1]. 모바일 어플리케이션 개발을 위해 사용자의 요구에

*준회원, 한남대학교, 정보통신공학과
**정회원, 한남대학교, 정보통신공학과
접수일자 2020년 3월 13일, 수정완료 2020년 5월 17일
게재확정일자 2020년 6월 5일

Received: 13 March, 2020 / Revised: 17 May, 2020 / Accepted: 5 June, 2020
*Corresponding Author: ysyun@hnu.kr
Dept. of Information and Communication Engineering, Hannam University, Korea

맞춰 기능을 제작하고, 기능에 따른 UI 요소의 크기와 위치, 모양 등을 구상하여 한눈에 알아보기 쉽게 스케치한 것을 스토리보드라 한다. 그림 1은 칼로리 계산기 모바일 응용을 제작하기 위해 스케치한 스토리보드 예시이다. 제작된 스토리보드를 참고하여 개발자들은 XCode [2], Android Studio [3] 등을 통해 실제로 동작이 가능한 모바일 응용 프로그램을 개발하면서, 디자이너들과 개발자들은 피드백을 주고받으며 많은 시간과 비용을 들여 하나의 응용 프로그램을 완성한다. 이러한 작업과정을 단순화시키고 효율을 높이기 위하여 스케치이미지의 GUI 요소를 인식해 모바일 응용 프로그램에서 실제로 동작되는 코드로 변환해주는 연구가 진행되고 있다 [4-9].

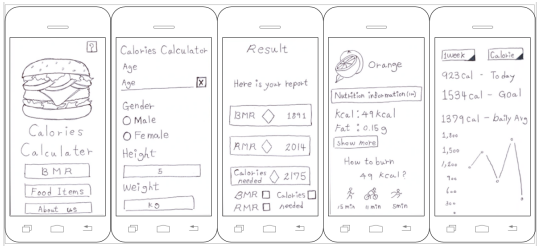

그림 1. 스토리보드 예시
Fig. 1. Storyboard example

본 논문에서는 UI 요소 인식 정확도 향상을 위해 심볼마커를 활용한 딥러닝 기반 UI 요소 자동인식 방법을 제안한다. 제안 기법의 평가를 위해 심볼마커가 없는 경우와 있는 경우에 대해 실험하고, 비교 분석하여 정확도를 높이기 위해 심볼마커의 모양을 다르게 하여 실험을 진행한다. UI 요소 자동인식으로 UI/UX 디자이너와 개발자 간의 시간과 비용을 줄이고 심볼마커를 사용하여 모호한 UI 요소 객체 식별을 줄이고 정확도 향상을 목표로 한다.

논문의 순서는 다음과 같다. 2장에서는 제안 기법의 관련 연구를 살펴보고 3장에서는 심볼마커를 사용한 딥러닝 기반 모바일 응용 UI 요소 인식 기법을 제안하고 4장에서는 제안 기법을 평가 후 마지막 5장에서는 본 연구의 결론을 내린다.

II. 관련 연구

대표적인 기법들로는 GUI 화면을 직관적으로 표현하거나 [4, 5] 컴퓨터 비전과 OCR(Optical Character Recognition) 기술을 사용하거나 [6], 딥러닝 기반으로 GUI 스크린샷을 DSL(도메인 특정 언어) 코드로 변환시키고 [7] 스케치이미지 또는 스크린샷의 GUI를 인식하여 XML 파일이나 HTML 파일로 생성하는 연구 등이 있다 [8, 9]. Parag et al [4]은 컴퓨터 비전 기반의 영상처리 기술(Edge Detection)을 활용하여 위젯들을 직관적으로 그래픽 특성에 따라 저장하고 비교하여 UI 위젯들을 검출하는 방법을 제안하였다. Pham et al [5]은 스크린샷의 GUI를 CannyEdges 방식을 사용하여 위젯들을 검출하는 UI 비주얼 로그에 중점을 두었다. Nguyen et al [6]은 컴퓨터 비전과 OCR 기술을 통해 UI를 인식하여 직관적으로 처리함과 동시에 UI 관련 규칙들을 작성하였다. Tony et al [7]은 Pix2Code 최초로 딥러닝을 사용해 UI를 식별하는 방법을 제안하여 GUI 스크린샷을 입력받아 CNN과 RNN을 통해 UI를 식별하고 DSL 코드로 변환하였다. Yun et al [8]은 딥러닝 기반으로 스케치이미지와 스크린샷의 GUI를 인식하고 XML 파일로 변환 및 저장하였다. Sketch2Code [9]는 Microsoft에서 개발한 것으로 사용자가 그린 스케치이미지의 GUI 위치와 위젯의 종류를 구별하고 인식하여 웹에서 사용하도록 HTML 코드로 변환하였다.

표 1. 관련 연구의 장단점
Table 1. Advantages and Disadvantages of Related Studies

관련연구	주요기술	취약점
Parag et al [4]	GUI를 Heuristic 제안함	위젯 추가 시 Rule도 추가되어 Rule Based 복잡해짐
Pham et al [5]		
Nauyen et al [6]	컴퓨터 비전 & OCR 사용함	
Tony et al [7]	딥러닝 기반으로 스크린샷을 코드로 변환 & UI 순서 식별 가능함	정확한 위치 인식 어려움
Yun et al [8]	이미지의 GUI 인식함 & XML 파일 자동생성	모호한 위젯 인식 어려움
Sketch2Code [9]	스케치이미지 GUI를 인식함 & HTML 파일을 자동생성	

표 1은 관련 연구들의 주요기술과 취약점을 정리한 것으로 기존 기법들의 문제점을 파악하고, 취약점을 해결하기 위하여 심볼마커 기반의 UI 요소 인식 방법을 제안한다. 제안된 방법은 기존 기법에서 UI 요소 인식 시 모호한 위젯에 대해서 명시적 표시인 심볼마커를 통해 정확도를 향상시키고자한다.

Ⅲ. 심볼마커를 활용한 딥러닝 기반 UI 요소 인식 기법

1. 딥러닝

딥러닝(Deep Learning)이란 Neural Network에서 Hidden layer가 1개 이상인 경우를 뜻하며 'Deep Neural Network' 라 불리기도 한다. 컴퓨터 비전의 발달로 딥러닝의 아키텍처와 구조가 발전되었으며, 이미지 분류 외에도 다양한 분야에서 사용되고 있다 [10-12]. 객체 인식(Object detection)은 이미지와 영상처리 부분에서 널리 사용되는 방법으로 컴퓨터 비전과 이미지 처리기술을 이용하여 사전에 정의된 클래스의 객체 인스턴스를 탐지한다 [13]. 객체 인식은 정보를 얻을 수 있는 영역을 선택하고 그 영역에 대한 특징을 추출하여 지정한 클래스에 따라서 객체를 분류한다 [14].

```
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   480 x 800 x   3   ->   480 x 800 x  32
    1 max          2 x 2 / 2   480 x 800 x  32   ->   240 x 400 x  32
    2 conv     64  3 x 3 / 1   240 x 400 x  32   ->   240 x 400 x  64
    3 max          2 x 2 / 2   240 x 400 x  64   ->   120 x 200 x  64
    4 conv    128  3 x 3 / 1   120 x 200 x  64   ->   120 x 200 x 128
    5 conv     64  1 x 1 / 1   120 x 200 x 128   ->   120 x 200 x  64
    6 conv    128  3 x 3 / 1   120 x 200 x  64   ->   120 x 200 x 128
    7 max          2 x 2 / 2   120 x 200 x 128   ->    60 x 100 x 128
    8 conv    256  3 x 3 / 1    60 x 100 x 128   ->    60 x 100 x 256
    9 conv    128  1 x 1 / 1    60 x 100 x 256   ->    60 x 100 x 128
   10 conv    256  3 x 3 / 1    60 x 100 x 128   ->    60 x 100 x 256
   11 max          2 x 2 / 2    60 x 100 x 256   ->    30 x  50 x 256
   12 conv    512  3 x 3 / 1    30 x  50 x 256   ->    30 x  50 x 512
   13 conv    256  1 x 1 / 1    30 x  50 x 512   ->    30 x  50 x 256
   14 conv    512  3 x 3 / 1    30 x  50 x 256   ->    30 x  50 x 512
   15 conv    256  1 x 1 / 1    30 x  50 x 512   ->    30 x  50 x 256
   16 conv    512  3 x 3 / 1    30 x  50 x 256   ->    30 x  50 x 512
   17 max          2 x 2 / 2    30 x  50 x 512   ->    15 x  25 x 512
   18 conv   1024  3 x 3 / 1    15 x  25 x 512   ->    15 x  25 x1024
   19 conv    512  1 x 1 / 1    15 x  25 x1024   ->    15 x  25 x 512
   20 conv   1024  3 x 3 / 1    15 x  25 x 512   ->    15 x  25 x1024
   21 conv    512  1 x 1 / 1    15 x  25 x1024   ->    15 x  25 x 512
   22 conv   1024  3 x 3 / 1    15 x  25 x 512   ->    15 x  25 x1024
   23 conv   1024  3 x 3 / 1    15 x  25 x1024   ->    15 x  25 x1024
   24 conv   1024  3 x 3 / 1    15 x  25 x1024   ->    15 x  25 x1024
   25 route  16
   26 conv     64  1 x 1 / 1    30 x  50 x 512   ->    30 x  50 x  64
   27 reorg              / 2    30 x  50 x  64   ->    15 x  25 x 256
   28 route  27 24
   29 conv   1024  3 x 3 / 1    15 x  25 x1280   ->    15 x  25 x1024
   30 conv     60  1 x 1 / 1    15 x  25 x1024   ->    15 x  25 x  60
   31 detection
```

그림 2. YOLO v2 계층구조
Fig. 2. YOLO v2 Layer Structure

YOLO는 실시간 객체 인식 시스템으로 학습 시 처음부터 끝까지 정확하게 구동되는 싱글 네트워크로 다른 객체 인식인 SSD321, DSSD321, R-FCN 등에 비해서 일반화되어 구조에 대한 크기 변경이 있더라도 추가적인 학습 없이 정확성과 속도가 높아 많은 연구에서 채택하고 있다. YOLO로 객체 인식을 할 때 Boundary box는 총 5개의 요소로 구성되어있다. x, y, width, height는 모든 cell의 0에서 1 사이로 구성이 되고, Confidence Score는 Boundary box의 정확도를 나타낸다. Darknet 기반으로 DarkNet 19는 YOLOv2에 Darknet 53은 YOLOv3에 사용되었다 [15, 16].

본 논문에서는 Darknet 19를 기반으로 한 YOLOv2를 사용하였으며 그림 2는 Darknet 19의 Layer 구조를 표시하였으며 Convolution, Maxpooling 등으로 총 30개의 Layer로 이루어져 있다.

2. 심볼마커와 딥러닝 기반 UI 요소 인식 기법

사전연구 결과 그림 3과 같이 정확성을 떨어뜨리는 요인 중 하나는 모호한 위젯이다. 본 논문에서는 모호한 위젯에 대하여 명시적 표시인 심볼마커를 통해 정확도 개선에 초점을 맞추어 딥러닝 기반의 심볼마커 기법을 제안한다.

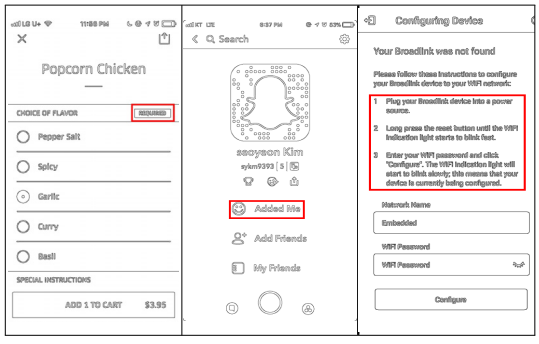

그림 3. 모호한 위젯들의 예시
Fig. 3. Examples of obscure widgets

그림 3은 구글스토어에 등록된 어플리케이션 중 모호한 위젯이 있는 어플리케이션의 스크린샷이다. 첫 번째는 배달 어플리케이션인 UberEats의 스크린샷으로 표시된 곳을 보면 네모박스 안에 텍스트로 구성되어 보통 버튼이라고 인식하지만, 텍스트 뷰이다. 두 번째는 소셜 어플리케이션인 Snapchat의 스크린샷으로 이미지 뷰와 텍스트 뷰로 인식되지만 하나의 버튼이고, 마지막 SURE 어플리케이션은 여러 개의 텍스트 뷰로 구성된 리스트 뷰로 인식하지만, 사용 설명을 써 놓은 텍스트 뷰이다. 예시와 같이 모호한 위젯이 존재하여 UI/UX 디자이너에게 전달받은 스토리보드를 개발자가 구현할 때 정확한 위젯 인식에 대한 어려움이 있고, 딥러닝으로 UI 요소 인식 시 오탐률이 증가하여 정확성이 떨어지므로 각 위젯의 앞부분에 심볼마커를 표시하여 오탐률을 줄이고 정확성을 높이는 방법을 제안하였다.

3. 데이터셋

본 실험의 데이터셋은 두 종류로 구성되어있다. 8가지 유형의 모바일 응용에 대한 각 7장의 스케치 이미지를

모바일환경에 익숙한 20대 대학생들이 그린 총 280개(8유형 * 7장 * 5명)의 스케치이미지와 구글 스토어 카테고리 중 다운로드 수가 높은 어플리케이션 10유형을 선정하여 주요기능 화면 5가지를 캡처한 총 50개(10유형 * 5장)의 스크린샷으로 총 330장의 데이터셋을 준비하였다. UI 객체는 총 7가지로 Button, CheckButton, EditText, ImageView, RadioButton, Spinner, TextView로 나누었고, UI 요소의 알파벳 앞글자 1-2 문자를 따서 심볼마커 모양 안에 작성하였다. 예를 들어, TextView는 T, Button은 B, CheckButton은 CB로 표시가 된다. 심볼마커의 모양은 UI 요소 인식 시 정확성에 영향을 주기 때문에 원형과 괄호형 두 가지 모양으로 나누어 실험을 진행하였다. 딥러닝 기반의 객체 인식을 위해 jpg 파일 형식으로, 사이즈는 500*800 pixel로 통일시켰으며 이미지의 위젯 정보를 담은 XML 파일도 같이 준비하였다.

표 2. 평가용 데이터셋 위젯의 개수
Table 2. Number of dataset widgets for evaluation

Widget	개수	Widget	개수
Button	214	ImageView	55
CheckButton	51	RadioButton	29
EdiText	20	Spinner	53
TextView	267	Total	719

데이터셋은 무작위로 선정하여 8:2 비율로 나누어 학습용 데이터셋 264개와 평가용 데이터셋 66개로 구성하였다. 표 2는 평가용 데이터셋의 7가지 위젯들의 분포를 보인다.

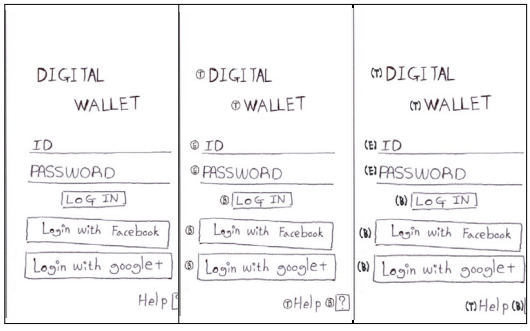

그림 4. 원형 심볼마커 및 괄호형 심볼마커 스케치이미지
Fig. 4. Circular Symbol Markers and Parenthesized Symbol Marker Sketch Images

제안 기법을 평가하기 위해 세 종류의 데이터셋을 준비하였다. 첫 번째 데이터셋은 기존 기법인 UI 요소만을 인식한 이미지이고 두 번째 데이터셋은 원형 심볼마커가 추가된 이미지이다. 마지막으로 세 번째 데이터셋은 괄호형의 심볼마커를 추가한 이미지이다. 그림 4는 실험에 사용된 모바일 응용을 제작하기 위해 스케치한 DigitalWallet의 이미지 중 하나인 로그인 화면을 작성한 예시이다.

Ⅳ. 실험 및 결과

1. 실험환경

실험은 Nvidia GTX 1070이 장착된 ubuntu 16.04 환경에서 pytorch-yolov2 [17]를 이용하여 진행되었다.

2. 실험절차

실험은 다음과 같이 진행되었다. 데이터셋의 각 UI 객체에 Bounding box를 지정한다. UI Bounding box와 Labels의 정보를 표기하기 위해 LabelImag Tool [18]을 사용하였다. LabelImg Tool은 PASCAL VOC format의 정보를 XML 파일로 저장한다. Class는 위젯 종류에 따라 7개로 작성되어 학습 값인 Epoch을 통해 500, 1000, 1500으로 총 세 번의 학습을 진행한다. 초기에는 Mirror (좌우반전) 데이터 처리를 하여 데이터 강화(data augmentation)를 진행하였으나, 심볼마커의 위치를 UI 왼쪽 앞으로 고정하여 Mirror 효과 대신 Flip(상하반전)과 Scale(크기) 효과를 이용하여 학습을 진행하였다. 사전에 ImageNet을 통하여 학습된 darknet19_448.weights 파일을 사용하여 학습용 데이터셋을 학습시키고, 학습한 Weights를 바탕으로 학습용 데이터셋에 적용한 결과를 기반으로 평가용 데이터셋의 각 위젯의 AP와 mAP를 확인하였다.

3. 실험결과

실험을 통하여 위젯들의 mAP와 위젯들의 전체 평균을 비교 및 분석하였다. 객체 인식에서 mAP (Mean Average Precision)은 PASCAL VOC 2012에서 정의한 것으로 각 클래스에 대하여 AP를 구하고 각각의 클래스에 대한 평균을 취하는 방식이다. AP (Average Precision)은 인식된 부분과 클래스를 낮은 순서로 정렬하여 전체 영역을 지정하고, 실제 객체와 예측한 영역 중 겹치는 부분과 실제 객체와 인식 객체 영역의 합집합의 비율인 IoU(Intersection over Union)가 0.5 이상 일치한 경우 검출된 객체의 정확성을 뜻한다 [19, 20].

표 3. 정밀도 및 재현율 계산
Table 3. Precision and recall rate calculation

	p (Predicted)	n (Predicted)
P (Actual)	True Positive	False Negative
N (Actual)	False Positive	True Negative

표 3은 IoU 비율을 기반으로 정밀도 / 재현율 곡선을 계산하는 것으로 정밀도는 “True Positive / (True Positive + False Positive)“ 정의되고, 재현율은 “True Positive / (True Positive + False Negative)” 로 정의된다 [21]. 재현율의 곡선을 정밀도에 따라 얻은 재현율로 감소시키면서 수치 적분을 통해 AP를 계산한 그 값들의 평균값을 mAP라 한다.

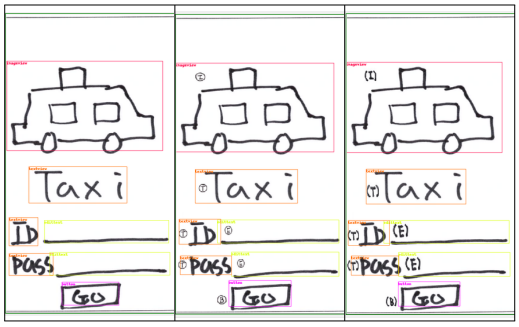

그림 5. 원형 및 괄호형 심볼마커 포함된 UI 인식결과이미지
Fig. 5. UI recognition result image with circular and parenthesized symbol marker

그림 5는 택시요금 계산 관련 어플리케이션을 스케치한 스케치이미지로 ImageView, TextView, Button, EditText가 인식됨을 확인하였다. 첫 번째 이미지는 기존기법인 심볼마커가 없는 이미지이고, 두 번째는 제안기법 원형 심볼마커를 표시한 이미지이고, 세 번째는 괄호형 심볼마커를 표시한 이미지이다. 데이터셋을 기반으로 한 데이터셋 당 학습횟수를 500, 1000, 1500으로 3번으로 나누어 총 9번의 실험을 진행하였다. Epoch은 전체 데이터에 대한 한 번의 학습으로 Epoch 500일 때는 약 2시간, 1000일 때는 약 4시간, 1500일 때는 약 7시간이 소요되었다.

각 데이터셋의 Epoch에 따른 7종류의 위젯에 대한 mAP 결과값을 표 4-6에 기술하였다. 표 4는 기존 기법인 심볼마커가 없는 위젯의 mAP 값을 나열하였으며 Epoch 1000인 경우 인식률이 높다는 것을 확인하였다.

표 4. 기존기법 (심볼마커가 없는) 각 위젯의 mAP
Table 4. mAP of each existing technique (without symbol marker)

No symbol marker	500	1000	1500
Button	81.7%	83.3%	82.4%
CheckButton	64.3%	66.3%	67.2%
EditText	74.9%	75.3%	70.4%
ImageView	71.0%	67.2%	63.1%
RadioButton	56.8%	69.4%	62.7%
Spinner	78.2%	79.2%	79.9%
TextView	70.8%	70.6%	70.6%

표 5은 원형 심볼마커 결과로 Epoch가 증가할수록 위젯들의 mAP가 증가하였고, 7종류의 위젯 중 RadioButton과 CheckButton의 인식률이 현저히 낮음을 알 수 있다.

표 5. 원형 심볼마커 데이터셋 각 위젯의 mAP
Table 5. Circular symbol marker data set mAP for each widget

Circular symbol marker	500	1000	1500
Button	87.5%	84.1%	91.2%
CheckButton	53.9%	57.1%	69.5%
EditText	82.5%	81.3%	83.4%
ImageView	69.8%	74.0%	88.9%
RadioButton	32.2%	58.2%	62.3%
Spinner	70.6%	84.3%	83.5%
TextView	82.1%	86.8%	88.7%

표 6. 괄호형 심볼마커 데이터셋 각 위젯의 mAP
Table 6. Parenthesized symbol marker data set mAP for each widget

Parenthesized symbol marker	500	1000	1500
Button	81.7%	83.3%	82.4%
CheckButton	64.3%	66.3%	67.2%
EditText	74.9%	75.3%	70.4%
ImageView	71.0%	67.2%	63.1%
RadioButton	56.8%	69.4%	62.7%
Spinner	78.2%	79.2%	79.9%
TextView	70.8%	70.6%	70.6%

표 6은 괄호형 심볼마커 결과로 Epoch에 따라 위젯들의 인식률이 증가하였다. Spinner와 TextView는 Epoch 1000에 인식률이 높았고, 평균적으로 위젯들의 인식률이 높았지만 RadioButton의 인식률은 다른 위젯에 비하여 낮았다. 그림 6은 Epoch 따른 데이터 값을 막대 그래프로 표시하였다. 심볼마커가 없는 데이터셋은 Epoch이 1500인 경우보다 1000일 때 2% 낮게 나왔고, 심볼마커 없는 데이터셋과 원형 심볼마커 데이터셋을 비교하면 0.1%, 2%, 10% 증가하였고, 원형 심볼마커와 괄

호형 심볼마커의 결과값이 Epoch에 따라서 7.4%, 8%, 5.7% 증가 되었다.

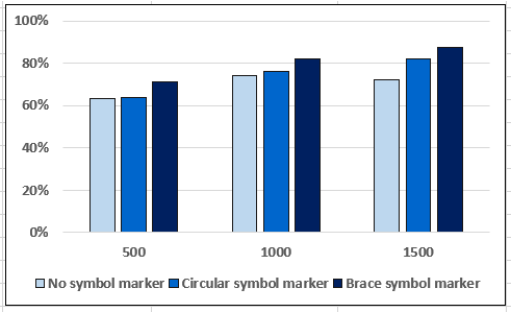

그림 6. Epoch에 따른 데이터셋의 총 mAP 결과
Fig. 6. Total mAP result of data set according to epoch

마지막으로 심볼마커가 없는 데이터셋과 괄호형 심볼마커의 데이터셋은 7.5%, 10%, 15.7% 차이를 보였다. 3개의 결과를 비교하였을 때, 인식률 증가가 가장 낮았던 것은 Epoch이 500일때에 심볼마커가 없는 데이터셋과 원형 심볼마커 데이터셋으로 0.1% 였고, 인식률 증가가 가장 높았던 것은 Epoch 1500일 때 심볼마커 없는 데이터셋과 괄호형 심볼마커 데이터셋이 15.7%로 가장 큰 차이를 보였다.

V. 결 론

최근 딥러닝은 여러 분야에 접목되어 사용되고 있으며 그 중 이미지 처리와 객체 인식에 대한 연구 결과로 GUI 요소를 인식하여 어플리케이션에 실제 구현되는 코드로 변환시켜주는 연구들이 제안되고 있다. 기존 연구기법으로는 컴퓨터 비전과 OCR을 사용한 영상처리 기반 기법과 딥러닝 기반의 GUI 요소 인식 기법이 연구되었으나 모호한 위젯에 대한 한계가 확인되었다.

본 논문에서는 딥러닝 기반 자동 UI 요소 인식과 모호한 위젯에 대한 정확성 향상을 위해 명시적 표시로 심볼마커를 사용하는 기법을 제안하였다. 기존 기법에서 심볼마커를 추가한 제안 기법을 증명하기 위해 실험을 진행하였고, 심볼마커의 모양에 따른 결과값의 차이를 비교분석하기 위해 원형 심볼마커와 괄호형 심볼마커로 나누어 실험을 진행하였다.

실험 진행 시 학습횟수를 다르게 하여 총 아홉 번의 실험을 진행하였으며 세 개의 데이터셋에 대한 결과값인 mAP를 확인하였다. 실험결과 심볼마커가 없을 때보다 괄호형 심볼마커가 있을 때 정확성이 15.7% 증가하였고, 원형의 심볼마커보다 괄호형 심볼마커의 결과값이 높게 나왔다. 원형 심볼마커는 라디오 버튼과 혼동되어 인식률이 떨어졌다고 판단한다. 따라서 심볼마커가 없는 데이터셋으로 UI 요소를 인식할 때 Epoch 1000 이상으로 설정하고, 심볼마커가 있는 데이터셋을 사용할 때는 괄호형 심볼마커 사용을 권장한다. UI 요소 자동인식을 통해 개발자의 편의성을 도모하고, 괄호형 심볼마커를 사용함으로써 UI/UX 디자이너와 개발자 간의 만족도를 충족시키고, 정확성이 향상되어 추후 GUI Code 개발에 도움이 될 것으로 기대한다.

References

[1] Joe Ligman, Marco Pistoia, Omer Tripp, Gegi Thomas, "Improving design validation of mobile application user interface implementation." In Proceedings of the International Conference on Mobile Software Engineering and Systems, ACM pp. 277-278, 2016. DOI: https://doi.org/10.1145/2897073.2897708

[2] https://developer.apple.com/kr/xcode/

[3] https://developer.android.com/

[4] Toufiq Parag, Claus Bahlmann, Vinay Shet, Maneesh Singh, "A grammar for hierarchical object descriptions in logic programs", Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on. Computer Vision and Pattern Recognition Workshops, pp. 33-38, 2012. DOI: https://doi.org/10.1109/CVPRW.2012.6239171

[5] Hung Pham, Tam Nguyen, Phong Vu, Tung Nguyen, "Toward mining visual log of software", arXiv preprint arXiv:1610.08911, 2016.

[6] Tuan Anh Nguyen, Christoph Csallner, "Reverse engineering mobile application user interfaces with REMAUI(T)" Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on. IEEE, pp. 248-259, 2015. DOI: https://doi.org/10.1109/ASE.2015.32

[7] Tony Beltramelli, "pix2code: Generating code from a graphical user interface screenshot", Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems. ACM, pp. 1-6, 2018. DOI: https://doi.org/10.1145/3220134.3220135

[8] Young-Sun Yun, Jisu Park, Jinman Jung, Seungbae Eun, Sin Cha, So-Sun Sup, "Automatic Mobile Screen Translation Using Object Detection Approach Based on Deep Neural Networks", Journal of Korea Multimedia Society, Vol. 21, No. 11, pp. 1305-1316, 2018. DOI: https://doi.org/10.9717/kmms .2018.21.11.1305

[9] https://github.com/Microsoft/ailab/tree/master/Sketch2Code.

[10] Ye-Seul Lee, Hyunl-Jae Choi, Dong-Myung Shin, Jung-Jae Lee, "Deep Learning based User Anomaly Detection Performance Evaluation to prevent Ransomware", Journal of Korea S/W Assessment and Valuation Society, Vol. 15, No.2, pp. 43-50, 2019.
DOI: https://doi.org/10.29056/jsav.2019.12.06

[11] Dong-Hoon Lee, Lee Bong Kyu, "A Study on the Analysis of Jeju Island Precipitation Patterns using the Convolution Neural Network", Journal of Korea S/W Assessment and Valuation Society, Vol. 15, No. 2, pp. 59-66, 2019.
DOI: https://doi.org/10.29056/jsav.2019.12.08

[12] Dong-Hee Yun, Young-Ung Kim, "Design and Implementation of Mobile Communication System for Hearing- impaired Person", The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 13, No. 6, pp. 47-54, 2013.
DOI: https://doi.org/10.7236/JIIBC.2013.13.6.47

[13] https://en.wikipedia.org/wiki/Object_detection

[14] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu, "Object detection with deep learning: a review", IEEE transactions on neural networks and learning systems, arXiv:1807.05511, Vol. 30, No. 11, pp. 3212-3232, 2019.
DOI: https://doi.org/10.1109/TNNLS.2018.2876865

[15] https://pjreddie.com/darknet/yolo/511.

[16] Joseph Redmon, Ali Farhadi, "YOLO9000: better, faster, stronger", In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263-7271, 2017.
DOI: https://doi.org/10.1109/CVPR.2017.690

[17] https://github.com/andy-yun/pytorch-0.4-yolov3

[18] Tzutalin, LabelImg, Git code 2015.
https://github.com/tzutalin/label Img

[19] https://github.com/rafaelpadilla/Object-Detection-Metrics

[20] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, Andrew Zisserman, "The pascal visual object classes (voc) challenge", International journal of computer vision, Vol, 88, No. 2, pp. 303-338, 2010.
DOI: https://doi.org/10.1007/s11263-009-0275-4

[21] https://en.wikipedia.org/wiki/Precision_and_recall

저 자 소 개

박 지 수(준회원)

- 2017년: 건양대학교 의료IT 공학과 졸업(학사)
- 2019년: 한남대학교 정보통신공학과 졸업(석사)
- 2019년~한남대학교 정보통신공학과 재학(박사)
- 관심분야: 음성처리, 패턴인식, 의료정보 시스템, 임베디드 시스템

정 진 만(정회원)

- 2008년: 서울대학교 컴퓨터공학과 졸업(학사)
- 2014년: 서울대학교 전기컴퓨터공학과 졸업(박사)
- 2014년~현재: 한남대학교 정보통신공학과 교수
- 관심분야: 운영체제, 임베디드 시스템, IoT, 시스템 보안

은 성 배(정회원)

- 1985년: 서울대학교 컴퓨터 공학과 학사
- 1987년: KAIST 전산학전공 (석사)
- 1987~1990년: 한국전자통신연구원 TDX개발단 연구원
- 1995년: KAIST 전산학전공(박사)
- 1995년~현재: 한남대학교 정보통신공학과 교수
- 관심분야: 실시간 시스템, 임베디드 시스템 등

윤 영 선(정회원)

- 2001년: KAIST 전산학전공 (박사)
- 2001년~현재: 한남대학교 정보통신공학과 교수
- 2006년: 한국전자통신연구원 초빙연구원
- 2012년: University of Washington 방문학자
- 2004년~현재: Interspeech Scientific Reviewer
- 관심분야: 음성인식, 음성처리, 웹 접근성, 내장형 시스템 등

※ 본 논문은 박지수의 2019년도 석사 학위 논문에서 발췌 정리하였음