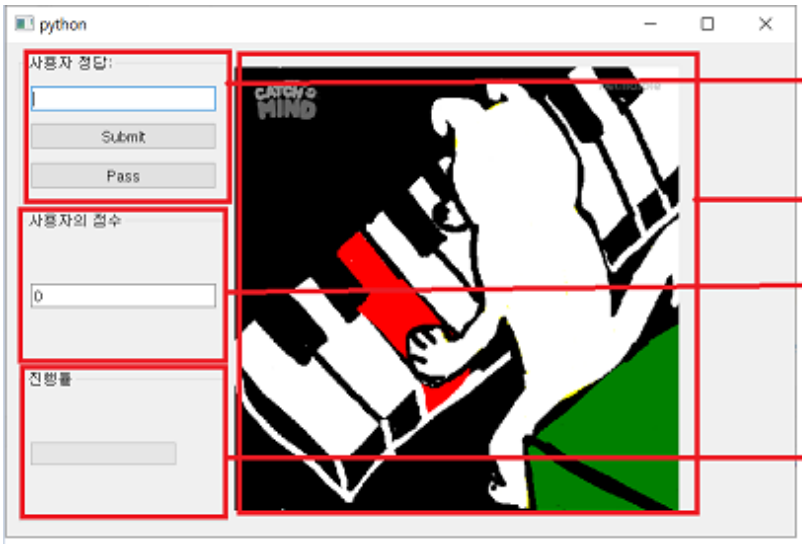
Software ProjectⅡ

20191622 양진우

20191640 이성연

목차

Ⅰ. 요구사항 분석(Requirement Analysis)

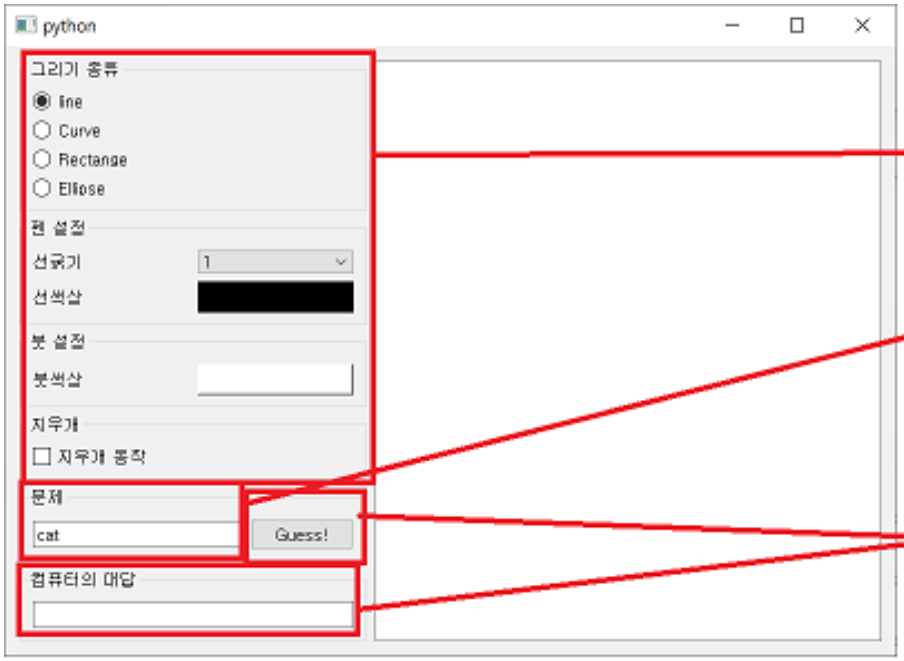


사용자의 정답을 입력받을 수 있어야함

사용자가 Submit을 눌렀고, 정답을 맞췄을 경우, Pass를 눌렀을 경우 그림이 바뀌어야 함

사용자가 Submit을 누르고, 정답을 맞췄을 경우, 점수 +1이 되어야 한다.

사용자가 Submt을 누르고, 정답을 맞췄을 경우, 또는 Pass를 눌렀을 경우, 진행률 +20을 해줘야 한다.



PyQt5를 이용하여 그림판의 기본적인 기능들을 만든다.

words에 저장을 한후, 랜덤으로 문제를 추출한다.

Guess를 누르면 사용자가 그린 그림을 QPixmap으로 잘라 컴퓨터가 예측을 할 수 있게 만든다.

Ⅱ. 소프트웨어 구조 설명

모듈	클래스	역할
start.py	MainWindow	시작화면을 만들어준다.
guess_main.py	CWidget	그림 맞추기의 기본적인 UI와 기능들을 제공해준다.
draw_main.py	CWidget	그림 그리기의 기본적인 UI와 기능들을 제공해준다.
	CView	그림판의 클래스.

클래스 및 모듈	메서드	기능
CWidget(guess_main)	submitClicked	만약 submit버튼이 클릭 되었다면, 사용자의 답변이 정답이랑 같은지를 확인 같으면, score을 올리고, progressBar을 증가시켜준다.
	passClicked	만약 pass버튼이 클릭된다면, 다음 그림으로 넘어가고, progressBar을 증가 시켜준다.
CWidget(draw_main)	guessClicked	guess버튼을 클릭하면, 그림판을 잘라 (./Images/)에 test.png의 이름으로 저장하고 CPU모듈의 start를 실행한다.
	radioClicked	어떤 버튼이 클릭되었는지 확인한다.
	showColorDlg	어떤 색깔이 클릭되었는지 확인한다.
CView(draw_main)	mousePressEvent	만약 마우스 왼쪽 버튼을 눌렀을 경우 시작 위치랑 끝 위치를 저장한다.
	mouseMoveEvent	마우스를 왼쪽 버튼을 누르고 움직이면 어떤 버튼을 눌렀는지 판단하고, 그에 맞추어 각각의 event를 시행한다.
	mouseReleaseEvent	Drawtype에 따라 다른 이벤트를 행한다.
CPU.py	start()	guessClicked버튼이 클릭되면 실행되는 메소드로 (./Images)에 저장된 test.png파일을 YOLO로 해석한다.

Ⅲ. 코드의 구현

1. YOLO를 이용한 CPU모듈의 start메소드

```
from darkflow.net.build import TFNet
import cv2

def start():
    options = {
        'model' : 'cfg/yolo.cfg',
        'load' : 'bin/yolo.weights',
        'threshold' : 0.1
    }

    tfnet = TFNet(options)
    print(2)
    imgcv = cv2.imread('./images/test.png')
    result = tfnet.return_predict(imgcv)
    print(1)
    max_confidence = 0
    result_label = ''
    for i in range(len(result)):
        if result[i]['confidence'] > max_confidence:
            max_confidence = result[i]['confidence']
            result_label = result[i]['label']
    if(result_label == ''):
        print("?????")
    else:
        print(result_label)
    return result_label
```

2. guess_main모듈의 guessClicked와 passClicked

```
85      def submitClicked(self):
86          if self.userAnswer.text() == self.Answer:
87              self.progress += 20
88              self.progressBar.setValue(self.progress)
89              if self.progress == 100:
90                  self.userScore.setText('당신의 최종 스코어 :'+str(self.score+1))
91                  return
92              #print(self.Images)
93              del self.Answers[0]
94              del self.Images[0]
95              self.Answer =self.Answers[0]
96              self.Image = self.Images[0]
97              #print(self.Image)
98              self.pixmap = QPixmap(self.Image)
99              self.pixmap = self.pixmap.scaledToHeight(450)
100             self.view.setPixmap(self.pixmap)
101
102             self.score+=1
103             self.userScore.setText(str(self.score))
104
105
106     def passClicked(self):
107         del self.Answers[0]
108         del self.Images[0]
109         self.Answer = self.Answers[0]
110         self.Image = self.Images[0]
111         # print(self.Image)
112         self.pixmap = QPixmap(self.Image)
113         self.pixmap = self.pixmap.scaledToHeight(450)
114         self.view.setPixmap(self.pixmap)
115
116         self.progress += 20
117         self.progressBar.setValue(self.progress)
118
```

IV. 한계점

- 한번 그림을 다른 것으로 착각하면 고쳐지지 않습니다.

- 상대적으로 처리 과정 속도가 빠른 YOLO를 사용했음에도 그림을 인식하기 위한 시간이 필요합니다.

- 개발자가 넣은 그림과 단어로만 게임이 진행되기 때문에 쉽게 지루함을 느낄 수 있고, 새로운 그림과 단어로 게임을 진행하고 싶을 경우 지속적인 추가가 필요합니다.

- 사람들이 흔히 하는 넌센스를 이해하지 못하는 현상.