**파이썬 과학 기초프로그래밍 수업 Term project**

스마트 iot

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명20205206

운석현

1.우선 프로젝트 코드를 짜기 위해 필요한 모듈을 불러왔다.

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

2.Data\_Read() 함수는 myInfo.txt 파일에서 데이터를 읽어와서 해당 데이터를 다른 입력 상자와 텍스트 상자에 표시하는 기능을 포함하고 있다.

1) myInfo.txt 파일 열기: 함수는 myInfo.txt 파일을 읽기 모드로 엽니다.

2) 데이터 읽기: 파일의 모든 내용을 한 줄씩 읽어와서 data 리스트에 저장합니다.

3) 파일 닫기: 파일을 읽은 후에는 파일을 닫습니다. (`f2.close()`)

4) 데이터 추출: data 리스트의 각 요소에서 개행 문자(`'\n'`)를 제거하고, 성명(Name), 학번(Number), 전공(Major), 학년(Grade) 변수에 저장합니다.

5) 입력 상자에 데이터 표시: e1, e2, e3, e4는 입력 상자로 가정하고, 이 상자들의 내용을 지우고 해당 변수들의 값을 입력 상자에 넣습니다

6) 텍스트 상자에 데이터 표시: t1은 텍스트 상자로 가정하고, t1의 내용을 모두 지우고 성명, 학번, 전공, 학년에 해당하는 데이터를 포맷에 맞게 텍스트 상자에 삽입합니다(insert()).

텍스트, 소프트웨어, 멀티미디어 소프트웨어, 스크린샷이(가) 표시된 사진

자동 생성된 설명

3. 이 함수는 입력박스에서 데이터를 불러와 해당 데이터를 myInfo.txt파일에 저장하는 역할을하게끔 만든 함수이다.

입력박스에서 성명,학번,전공,학년을 get()메소드를 사용해서 읽어오고 delete()를 통해 입력박스의 내용은 지워줍니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명파일에 입력하기 위해 myInfo.txt를 w모드로 열어주고 \n을 구분하여 write()메소드를 사용해서 파일에 입력을 해줍니다.

4.각 함수들은 파일로부터 데이터를 읽어와서 해당 데이터를 적정한 형태로 바꿔서 반환하는 역할을 하는 함수로써 만들어줬다. 파일을 읽기모드로 열고 빈리스트들을 생성해준다. 이 빈리스트들은 데이터들을 저장할 리스트이다. 파일을 한줄씩 읽으면서 각줄을 공백으로 분리하고 필요한 데이터를 추출하여 각 리스트에 저장해준다. 추출하는 과정은 그 전 과제 data plot과 동일하기에 설명은 생략하겠다. 이렇게 추출되고 가공된 데이터들은 파일이 닫힌 이후 반환되는 과정을 거치는데 이 반환되는 데이터는 뒤에 나오는 함수에서 사용 될 것이다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

5.이 함수는 GAN (Generative Adversarial Network)의 손실(loss) 값을 시각화하는 기능을 가지고 있습니다.

함수의 구조를 살펴보면 다음과 같습니다:

1. `Generate\_Data1()`과 `Generate\_Data2()` 함수를 통해 데이터를 생성합니다. 이 함수들은 각각 `epoch1`, `gan\_loss1`, `epoch2`, `gan\_loss2` 값을 반환합니다. 이 값들은 각각 데이터 1과 데이터 2의 학습 과정에서의 에포크(epoch)와 GAN 손실(loss) 값을 나타냅니다.

2. `plt.figure(2)`를 호출하여 새로운 Figure 객체를 생성합니다. 숫자 2는 Figure 객체의 번호를 나타냅니다.

3. `plt.subplot(2, 1, 1)`을 호출하여 Figure 객체를 2x1 크기의 서브플롯(subplot)으로 분할합니다. 이때, 현재 서브플롯 중 첫 번째를 선택합니다.

4. `plt.plot(epoch1, gan\_loss1)`을 호출하여 데이터 1의 GAN 손실(loss) 값을 선 그래프로 그립니다.

5. `plt.xlabel('epochs')`와 `plt.ylabel('GAN Loss')`를 호출하여 x축과 y축에 각각 'epochs'와 'GAN Loss'라는 레이블을 표시합니다.

6. `plt.title("data1")`을 호출하여 그래프의 제목을 'data1'로 설정합니다.

7. `plt.grid(True)`를 호출하여 그리드를 활성화합니다.

8. `plt.subplot(2, 1, 2)`을 호출하여 Figure 객체의 두 번째 서브플롯을 선택합니다.

9. `plt.plot(epoch2, gan\_loss2)`을 호출하여 데이터 2의 GAN 손실(loss) 값을 선 그래프로 그립니다.

10. `plt.xlabel('epochs')`와 `plt.ylabel('GAN Loss')`를 호출하여 x축과 y축에 각각 'epochs'와 'GAN Loss'라는 레이블을 표시합니다.

11. `plt.title("data2")`을 호출하여 그래프의 제목을 'data2'로 설정합니다.

12. `plt.grid(True)`를 호출하여 그리드를 활성화합니다.

13. `plt.show()`를 호출하여 모든 그래프를 보여줍니다.

이 함수를 호출하면 데이터 1과 데이터 2의 GAN 손실(loss) 값의 변화를 시각적으로 비교할 수 있습니다. 각각의 그래프는 에포크(epoch)에 따른 GAN 손실(loss)의 변화를 보여주며, x축은 에포크(epoch)를, y축은 GAN 손실(loss) 값을 나타냅니다.

텍스트, 스크린샷, 메뉴이(가) 표시된 사진

자동 생성된 설명

6.`Plot\_PSNR()`, `Plot\_Val\_MSE()`, `Plot\_MSE()` 함수들은 각각 PSNR (Peak Signal-to-Noise Ratio), Validation MSE (Mean Squared Error), MSE (Mean Squared Error) 값을 시각화하는 기능을 가지고 있습니다. 이 세 함수들은 `Generate\_Data1()`과 `Generate\_Data2()` 함수를 사용하여 데이터를 생성하고, 생성된 데이터에 대한 시각화를 수행합니다.

1. `Plot\_PSNR()` 함수:

- `Generate\_Data1()`과 `Generate\_Data2()` 함수를 통해 데이터를 생성합니다. 각각의 함수는 `epoch1`, `gan\_loss1`, `epoch2`, `gan\_loss2`, `psnr1`, `psnr2` 값을 반환합니다. `psnr1`과 `psnr2`는 데이터 1과 데이터 2의 PSNR 값을 나타냅니다.

- `plt.figure(3)`을 호출하여 새로운 Figure 객체를 생성합니다. 숫자 3은 Figure 객체의 번호를 나타냅니다.

- `plt.subplot(3, 1, 1)`을 호출하여 Figure 객체를 3x1 크기의 서브플롯으로 분할합니다. 이때, 현재 서브플롯 중 첫 번째를 선택합니다.

- `plt.plot(epoch1, psnr1)`을 호출하여 데이터 1의 PSNR 값을 선 그래프로 그립니다.

- `plt.xlabel('epochs')`와 `plt.ylabel('PSNR')`를 호출하여 x축과 y축에 각각 'epochs'와 'PSNR'라는 레이블을 표시합니다.

- `plt.title("data1")`을 호출하여 그래프의 제목을 'data1'로 설정합니다.

- `plt.grid(True)`를 호출하여 그리드를 활성화합니다.

- 위의 과정을 통해 데이터 1의 PSNR 값을 시각화합니다.

- 동일한 방식으로 데이터 2의 PSNR 값을 시각화하는 그래프를 그립니다.

2. `Plot\_Val\_MSE()` 함수:

- `Generate\_Data1()`과 `Generate\_Data2()` 함수를 통해 데이터를 생성합니다. 각각의 함수는 `epoch1`, `gan\_loss1`, `epoch2`, `gan\_loss2`, `val\_mse1`, `val\_mse2` 값을 반환합니다. `val\_mse1`과 `val\_mse2`는 데이터 1과 데이터 2의 Validation MSE 값을 나타냅니다.

- `plt.figure(4)`를 호출하여 새로운 Figure 객체를 생성합니다. 숫자 4는 Figure 객체의 번호를 나타냅니다.

- `plt.subplot(2, 1, 1)`을 호출하여 Figure 객체를 2x1 크기의 서브플롯으로 분할합니다. 이때, 현재 서브플롯 중 첫 번째를 선택합니다.

- `plt.plot(epoch1, val\_mse1)`을 호출하여 데이터 1의 Validation MSE 값을 선 그래프로 그립니다.

- `plt.xlabel('epochs')`와 `plt.ylabel('Validation MSE')`를 호출하여 x축과 y축에 각각 'epochs'와 'Validation MSE'라는 레이블을 표시합니다.

- `plt.title("data1")`을 호출하여 그래프의 제목을 'data1'로 설정합니다.

- `plt.grid(True)`를 호출하여 그리드를 활성화합니다.

- 위의 과정을 통해 데이터 1의 Validation MSE 값을 시각화합니다.

- 동일한 방식으로 데이터 2의 Validation MSE 값을 시각화하는 그래프를 그립니다.

3. `Plot\_MSE()` 함수:

- `Generate\_Data1()`과 `Generate\_Data2()` 함수를 통해 데이터를 생성합니다. 각각의 함수는 `epoch1`, `gan\_loss1`, `epoch2`, `gan\_loss2`, `mse1`, `mse2` 값을 반환합니다. `mse1`과 `mse2`는 데이터 1과 데이터 2의 MSE 값을 나타냅니다.

- `plt.figure(5)`를 호출하여 새로운 Figure 객체를 생성합니다. 숫자 5는 Figure 객체의 번호를 나타냅니다.

- `plt.subplot(2, 1, 1)`을 호출하여 Figure 객체를 2x1 크기의 서브플롯으로 분할합니다. 이때, 현재 서브플롯 중 첫 번째를 선택합니다.

- `plt.plot(epoch1, mse1)`을 호출하여 데이터 1의 MSE 값을 선 그래프로 그립니다.

- `plt.xlabel('epochs')`와 `plt.ylabel('MSE')`를 호출하여 x축과 y축에 각각 'epochs'와 'MSE'라는 레이블을 표시합니다.

- `plt.title("data1")`을 호출하여 그래프의 제목을 'data1'로 설정합니다.

- `plt.grid(True)`를 호출하여 그리드를 활성화합니다.

- 위의 과정을 통해 데이터 1의 MSE 값을 시각화합니다.

- 동일한 방식으로 데이터 2의 MSE 값을 시각화하는 그래프를 그립니다.

이 함수들을 호출하면 각각 PSNR, Validation MSE, MSE 값을 에포크(epoch)에 따라 시각화하여 비교할 수 있습니다. 각 그래프는 에포크(epoch)를 x축으로, 해당 지표(PSNR, Validation MSE, MSE) 값을 y축으로 나타내며, 데이터 1과 데이터 2에 대한 그래프를 분리하여 표시합니다.

텍스트, 스크린샷, 메뉴, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트, 메뉴이(가) 표시된 사진

자동 생성된 설명

7.`Calculate\_Statistics()` 함수는 "data1.txt"와 "data2.txt" 파일에서 데이터를 읽어와 각각의 데이터에 대한 최대값, 최소값, 평균값을 계산하고, 그 결과를 "DataProcessing.txt" 파일에 저장하는 함수입니다.

1. 데이터 1 처리:

- "data1.txt" 파일을 읽기 모드로 엽니다.

- `gan\_loss1`, `val\_mse1`, `psnr1`, `mse1`, `epoch1` 리스트를 초기화합니다.

- 파일의 각 줄을 읽어와 데이터를 분리하고, 해당 데이터를 각각의 리스트에 추가합니다.

- 파일을 닫습니다.

2. 데이터 2 처리:

- "data2.txt" 파일을 읽기 모드로 엽니다.

- `gan\_loss2`, `val\_mse2`, `psnr2`, `mse2`, `epoch2` 리스트를 초기화합니다.

- 파일의 각 줄을 읽어와 데이터를 분리하고, 해당 데이터를 각각의 리스트에 추가합니다.

- 파일을 닫습니다.

3. 데이터 2의 통계 계산:

- `gan\_loss2`, `val\_mse2`, `psnr2`, `mse2` 리스트에서 각각의 최소값, 최대값, 평균값을 계산합니다.

- 계산한 결과를 `gan\_loss2\_min`, `gan\_loss2\_max`, `gan\_loss2\_avg`, `val\_mse2\_min`, `val\_mse2\_max`, `val\_mse2\_avg`, `psnr2\_min`, `psnr2\_max`, `psnr2\_avg`, `mse2\_min`, `mse2\_max`, `mse2\_avg` 변수에 저장합니다.

4. 데이터 1의 통계 계산:

- `gan\_loss1`, `val\_mse1`, `psnr1`, `mse1` 리스트에서 각각의 최소값, 최대값, 평균값을 계산합니다.

- 계산한 결과를 `gan\_loss1\_min`, `gan\_loss1\_max`, `gan\_loss1\_avg`, `val\_mse1\_min`, `val\_mse1\_max`, `val\_mse1\_avg`, `psnr1\_min`, `psnr1\_max`, `psnr1\_avg`, `mse1\_min`, `mse1\_max`, `mse1\_avg` 변수에 저장합니다.

5. 결과 파일 작성:

- "DataProcessing.txt" 파일을 쓰기 모드로 엽니다.

- 계산한 통계 결과를 파일에 씁니다. 각 값은 포맷을 지정하여 최소값, 최대값, 평균값을 쓰고, 줄바꿈 문자('\n')로 구분합니다.

- 파일을 닫습니다.

6. 결과 파일 읽기:

- "DataProcessing.txt" 파일을 읽기 모드로 엽니다.

- 파일의 내용을 읽어와 `content` 변수에 저장합니다.

7. 텍스트 박스 업데이트:

- `t2`라는 이름의 텍스트 박스를 선택하여 내용을 업데이트합니다.

- 텍스트 박스의 내용을 지우고(`t2.delete(1.

0, END)`) 파일에서 읽어온 내용을 삽입합니다(`t2.insert(END, content)`).

이 함수를 호출하면 "data1.txt"와 "data2.txt" 파일에서 데이터를 읽어와 각 데이터에 대한 최대값, 최소값, 평균값을 계산하고, 결과를 "DataProcessing.txt" 파일에 저장한 후 텍스트 박스(`t2`)에 결과를 표시합니다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

8.1. Tkinter GUI 설정:

- `root.title(20 \* blank\_space + "Term Project")`를 사용하여 루트 윈도우의 제목을 "Term Project"로 설정합니다. `blank\_space`는 공백 문자열입니다.

- `Label` 위젯을 사용하여 "개인정보 입력"이라는 제목을 생성하고 `grid()` 메서드를 사용하여 해당 레이블을 그리드의 첫 번째 행에 배치합니다.

- `Label`과 `Entry` 위젯을 사용하여 "성명", "학번", "전공", "학년"에 대한 레이블과 텍스트 입력 상자를 생성합니다. 각각을 `grid()` 메서드를 사용하여 해당하는 위치에 배치합니다.

2. 개인정보 불러오기 및 저장 버튼:

- `Button` 위젯을 생성하여 "개인정보 불러오기"와 "개인정보 저장" 버튼을 만듭니다. `command` 매개변수를 사용하여 각 버튼에 해당하는 함수를 연결합니다.

- `place()` 메서드를 사용하여 각 버튼의 위치를 지정합니다. 버튼의 x, y 좌표를 설정하여 윈도우 내에서 버튼의 위치를 조정할 수 있습니다.

3. 플롯 그래프 및 데이터 처리 버튼:

- `Button` 위젯을 생성하여 "GAN Loss Plot", "PSNR Plot", "Validation MSE Plot", "MSE Plot", "Data Processing & Save", "Data1", "Data2" 버튼을 만듭니다. 각 버튼에 해당하는 함수를 `command` 매개변수를 통해 연결합니다.

- `place()` 메서드를 사용하여 각 버튼의 위치를 조정합니다.

4. 텍스트 상자:

- `Text` 위젯을 생성하여 개인정보와 데이터 처리 결과를 표시하는 상자를 생성합니다. `height`와 `width` 매개변수를 통해 상자의 크기를 조정할 수 있습니다.

- `place()` 메서드를 사용하여 각 텍스트 상자의 위치를 조정합니다.

5. Hallym 대학교 로고:

- `Image` 모듈을 사용하여 Hallym 대학교 로고 이미지를 불러옵니다.

- `Label` 위젯을 생성하여 이미지를 표시하고 `place()` 메서드를 사용하여 로고의 위치를 조정합니다.

6. Tkinter 이벤트 루프 실행:

- `root.mainloop()`를 호출하여 Tkinter 이벤트 루프를 실행합니다. 이 이벤트 루프는 사용자의 상호작용을 감지하고 이벤트에 따른 동작을 수행합니다. 따라서 GUI 창이 열리고 사용자가 버튼을 클릭하거나 입력을 할 때 이벤트가 처리됩니다.

**실행화면**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명-인터페이스 화면

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명-개인정보 저장 후 개인정보 불러오기 버튼 클릭시 화면

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명-개인정보 저장시 myinfo 파일에 저장된 모습

텍스트, 도표, 라인, 평행이(가) 표시된 사진

자동 생성된 설명

-각 plot버튼 클릭시 실행되는 화면

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명-Data Processing&Save버튼 클릭시 보여지는 화면 결과를 데이터 텍스트 박스에 출력되게끔 코드를 만들었다.

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

-또한 DataProcessing.txt파일에도 결과값이 쓰여짐과 동시에 저장될 수 있게 코드를 만들어놨다.