산학 캡스톤디자인 학생팀 과제 최종보고서

과 제 명	3D-Grad-CAM 기반 설명가능한 3D프린팅 제조시스템 운영성과 예측 프로세스				
팀 명	PSEL		메이커스페이스 사용여부		☐ 예 ☑ 아니오
특허출원여부	☐ 예 ☑ 아니오		출원내역		-
창업여부	☐ 예 ☑ 아니오		창업동아리여부		☐ 예 ☑ 아니오
분류	**구분**	**성명**	**소속학과**	**학년**	**학번**
참여 학생	팀장	김준우	산업경영공	4	201801386
	팀원	장부길	산업경영공	4	201801367
		김명준	산업경영공	4	201801377
		조수아	산업경영공	4	202001353
분류	**구분**	**성명**	**소속**		**직위**
지도교수	지도교수	박기정	산업경영공		부교수
멘토	산업체	성기석	현대자동차		대표이사
	대학원	문정윤	산업경영공		석사과정

본 보고서를 공학교육혁신센터 산학 캡스톤디자인 학생팀 과제 최종보고서로 제출합니다.

【붙임 : 최종 결과보고서 1부】

2023 년 06 월 20 일

대표학생 : 김 준 우 

인천대학교 공학교육혁신센터장 귀하

산학 캡스톤디자인 학생팀 과제 최종 결과보고서

Ⅰ. 과제 개발의 목적 및 필요성

1. 목적 및 필요성

3D프린팅 제조시스템은 3D 객체를 층층이 쌓아 올리는 제조 기술로 절삭 가공 기반의 전통적인 제조시스템에서 구현하기 어려웠던 복잡한 디자인을 구현할 수 있어 항공 우주, 자동차. 의료 분야 등 다양한 분야에서 각광받고 있다. 특히 3D프린팅 기술의 등장은 복잡한 형상의 설계 자유도 및 생산 가능성을 크게 향상시켰는데, 이에 3D프린팅을 효과적으로 이용하기 위한 실용적 설계 방법으로 3D프린팅 특화 설계가 대두되고 있다.

3D프린팅 특화설계란 설계 단계에서 3D프린팅을 이용한 설계의 제조 비용 및 품질 결함을 고려하여 제품을 설계하는 것을 말한다. 전통적인 제조 기술과 다르게 재료를 층층이 쌓아가며 제품을 제조하는 3D프린팅의 기술의 특성으로 인해 제품 설계 시 제품 디자이너는 3D프린팅의 다양한 설계 요소를 고려해야한다. 만약 3D프린팅 특화 설계 가이드라인을 따르지 않고 제품을 설계할 경우 제품의 기능성 저하와 부정적인 제조 성과를 초래하며 나아가 제품에 대한 설계 및 관련 프로세스를 변경할 경우 제품 개발 주기에서의 막대한 비용 증가로 이어질 수 있다. 따라서 출력물의 품질, 기능성 및 제조 성과를 만족시키기 위해 3D프린팅 특화 설계 가이드라인을 바탕으로 설계에 대한 요소를 검토하는 것은 3D프린팅을 이용한 설계에서 중요한 단계로 간주되어야 한다. 그러나 설계 단계에서 대부분의 제품 디자이너들은 본인의 경험적인 지식에 의존해 설계 요소를 확인하며, 3D프린팅 특화 설계 요소에 대한 지식이나 정보가 부족한 디자이너들은 3D프린팅 제조시스템 운영 성과에 부정적인 특화 설계 요소에 대한 파악이 어려운 실정이다.

3D프린팅을 이용한 설계 초기 단계에서 목표하는 운영성과 달성을 위해 설계에 대한 3D프린팅 적합성 파악이 요구됨에 따라 CAD 모델의 3D프린팅 제조시스템 운영성과 예측이 대두되고 있다. 이에 딥러닝 기반 CAD모델의 3D프린팅 운영 성과 예측에 대한 연구가 진행되었다. 그러나 딥러닝 모델은 우수한 성능을 지니고 있지만 블랙 박스 특성으로 인해 예측 근거에 대한 해석이 어렵다는 한계가 존재한다. 예측 모델에 대한 신뢰성을 보장하기 위해 예측 결과에 대한 근거 해석은 필수적이며 실질적인 운영성과 예측 및 설계 프로세스의 효율성 제고를 위해 3D프린팅 운영 성과 예측에 주요한 영향을 미친 설계 요소에 대한 직접적인 제시가 필요하다.

이러한 점에 착안하여 본 연구에서는 효율적인 3D프린팅을 이용한 제품 설계 프로세스를

위해 설명가능한 인공지능 기반의 3D프린팅 제조시스템 운영성과 예측 및 설계요소 시각화 프로세스를 제시하는 것에 목적을 둔다.

2. 활용성 및 기대효과

본 연구에서 제시되는 3D프린팅 제조시스템 운영성과 예측 및 설계 요소 시각화 프로세스를 활용하여 제품 디자이너는 3D프린팅을 이용한 제품 초기 설계 단계에서 운영 성과에 주요한 영향을 미치는 설계 요소를 파악할 수 있다. 이를 통해 디자이너는 제품에 대한 디자인의 3D프린팅 적합성 및 제조 운영 성과를 빠르게 평가할 수 있으며 나아가 목표하는 3D프린팅 제조시스템 운영성과 달성을 위해 제품을 수정할 수 있다. 또한 제안된 프로세스는 3D프린팅 특화 설계 관련 지식이나 정보가 부족한 디자이너의 설계 작업 효율성을 제고할 수 있을 것으로 기대된다.

Ⅱ. 과제내용 및 제작과정

1. 연구 방법

본 연구에서 제시되는 3D프린팅 운영성과 예측의 프로세스는 그림 1과 같으며, 각 단계별로 간략히 설명하면 다음과 같다.

• 1단계: 1,008개의 3D프린팅 부품에 대한 STL 모델, 설계 특징, 제조 운영 성과를 수집한다. 이후 수집된 데이터를 딥러닝의 입력으로 사용하기 위해 STL 모델은 복셀 형태로, 설계 특징과 운영성과는 로그 정규화하여 전처리를 진행한다.

• 2단계: 복셀과 설계 특징을 입력 변수로 운영 성과를 출력 변수로 하는 3D-CNN 모델을 구축하고 테스트 셋에 대한 예측 성능을 평가한다

• 3단계: 모델 평가에 사용된 데이터에 대해 3D-Grad-CAM을 적용하여 3D프린팅 제조시스템 운영성과 예측의 근거를 시각화한다.

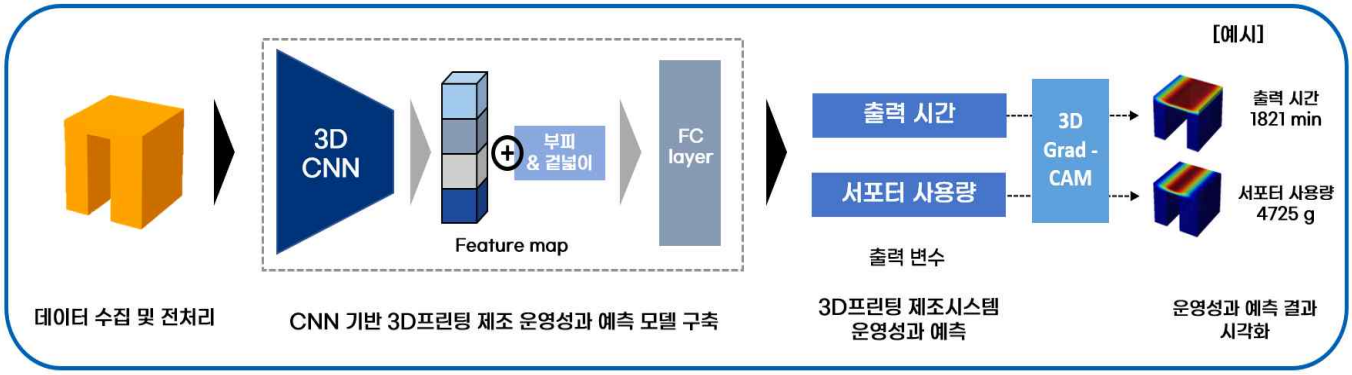


그림 1. 제안된 프로세스의 개략도

1-1. 데이터 수집

3D 데이터 공유 플랫폼 'GrabCAD'와 Thingiverse'를 통해 1008개의 3D프린팅 부품에 대한 STL 모델을 수집했다. STL은 3D Systems에서 만든 입체 CAD 소프트웨어 고유의 파일 형식으로 "표준 삼각형 언어"및 "표준 모자이크식 언어" 등으로 일컫는다. STL 형식은 다른 많은 소프트웨어 패키지에서 지원되어 빠른 프로토타이핑, 3D프린팅 및 컴퓨터 보조 제조에 널리 사용되고 있다.

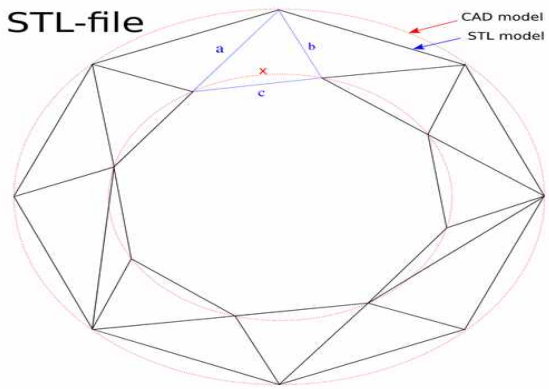


그림 2. STL 형식 예시

STL 모델 검색 시에는 산업용 부품(cylinder, engine, bracket, gear, hinge, cover, fitting, buckle 등)을 고려했다.

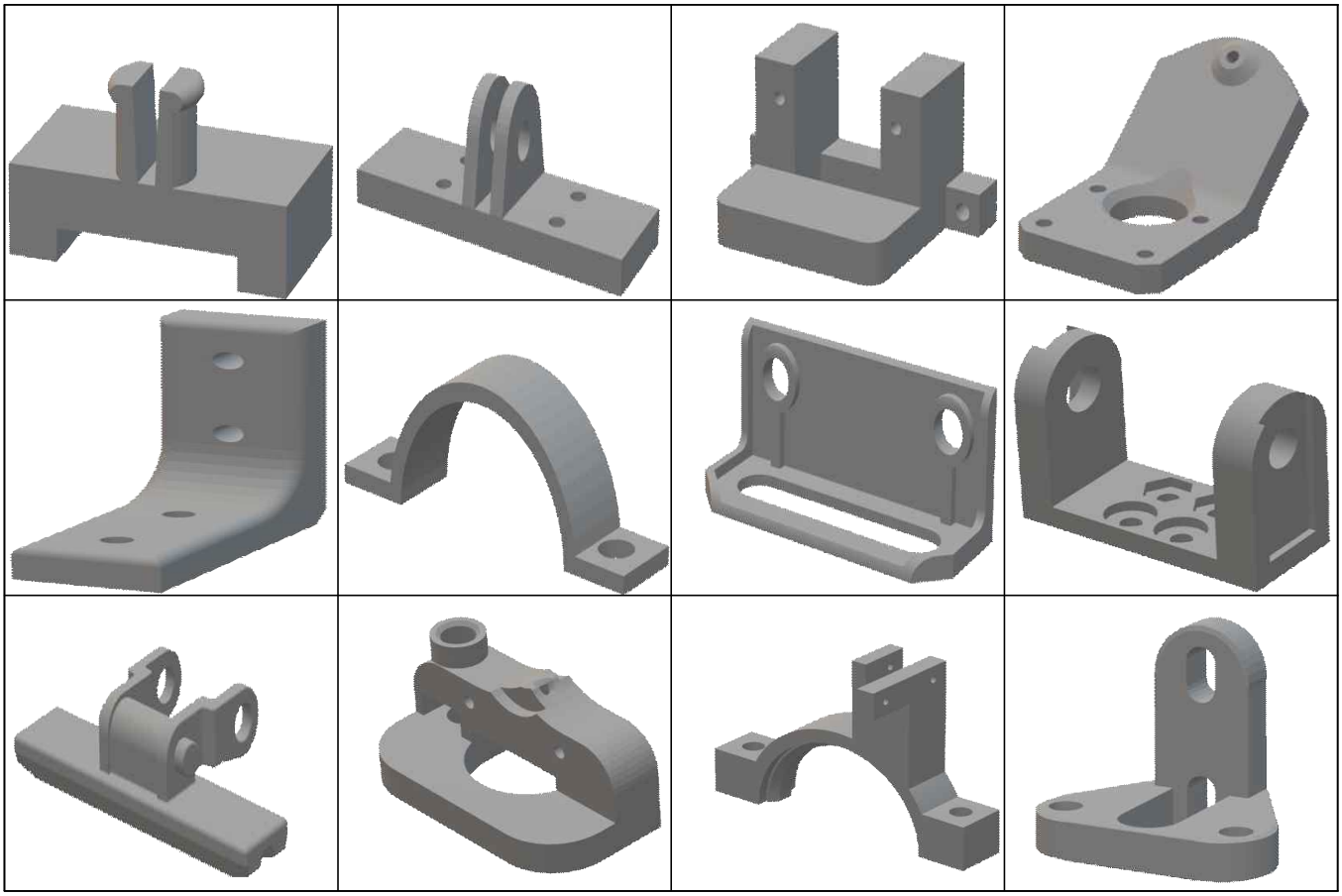


그림 3. STL 모델의 설계 특징 계산 과정

이후 Python numpy-stl 라이브러리를 활용하여 1008개의 STL 모델에 대한 설계 특징(부피, 표면적)을 수집하였다. 부피와 표면적은 수많은 삼각형으로 이루어진 STL 모델의 각 삼각형에 대한 평면과 법선 벡터를 바탕으로 각각 관련 수식으로 계산되었다.

```python
def stl_volume_calculate(file_name):

    your_mesh = mesh.Mesh.from_file(file_name)

    normals = your_mesh.normals

    # 각 삼각형 면적 계산하기
    a = your_mesh.vectors[:, 0, :]
    b = your_mesh.vectors[:, 1, :]
    c = your_mesh.vectors[:, 2, :]
    cross_product = np.cross(b - a, c - a)

    # 총 면적 계산하기
    volumes = np.abs(np.sum(a * cross_product, axis=0)) / 6.0

    # 총 부피 계산하기
    total_volume = np.sum(volumes)

    print('Volume:', total_volume)
```

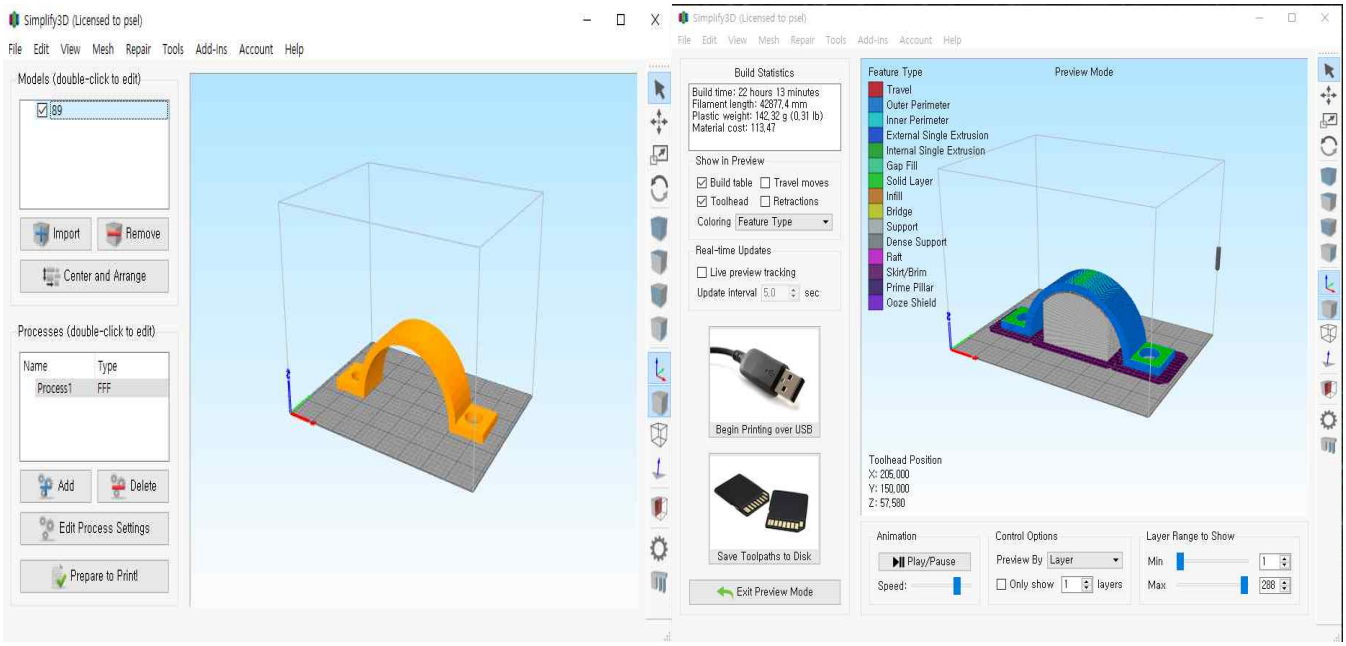
(가) 부피 계산 과정

```python
def stl_surface_area_calculate(file_name):

    your_mesh = mesh.Mesh.from_file(file_name)

    normals = your_mesh.normals

    a = your_mesh.vectors[:, 0, :]
    b = your_mesh.vectors[:, 1, :]
    c = your_mesh.vectors[:, 2, :]
    cross_product = np.cross(b - a, c - a)
    areas = 0.5 * np.sqrt(np.sum(cross_product ** 2, axis=1))

    total_area = np.sum(areas)
    print('Surface area:', total_area)
```

(나) 표면적 계산 과정

그림 4. STL 모델의 설계 특징 계산 과정

3D프린팅 운영성과 예측에 관련된 문헌 조사를 통해 가장 많이 고려된 운영 성과인 출력 시간, 서포트 사용량을 상용 슬라이싱 소프트웨어 'Simplify 3D'를 활용해 수집했다. 슬라이싱이란 STL 모델을 'G-code' 언어로 변환하는 과정으로 3D프린팅 출력을 위해 필수적인 과정이다.



(가) 슬라이싱 소프트웨어 초기 화면 (나) 슬라이싱 완료 후

그림 5. 슬라이싱 소프트웨어를 통한 운영 성과 수집

운영 성과 수집을 위해 슬라이싱 진행 시 고려한 3D프린터와 주요 매개변수는 표 1과 같다.

3D프린터	재료	출력 속도	층 두께	채움 정도
Apium P220	CFR-PEEK	1200mm/min	0.2mm	100%

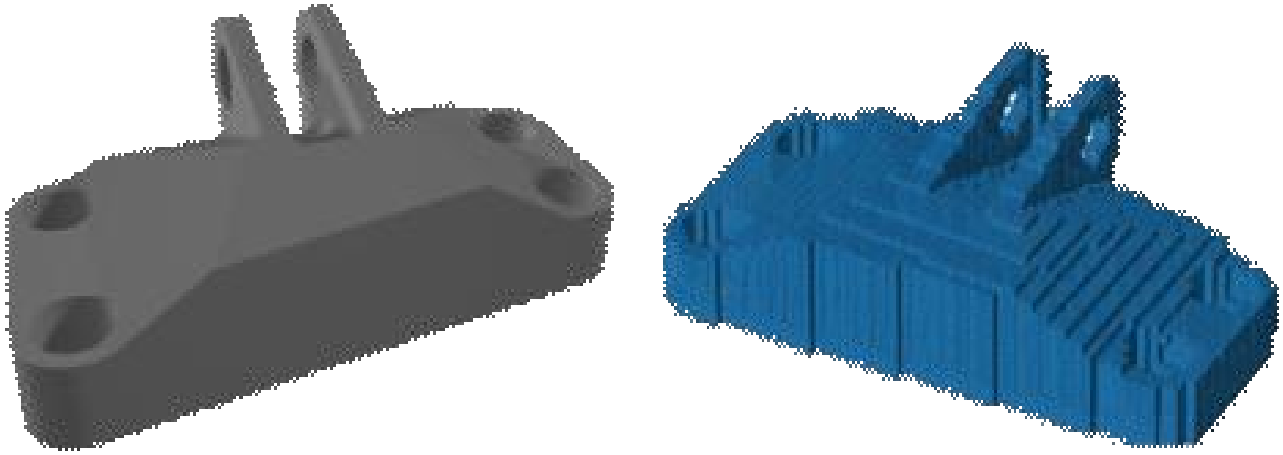
표1. 운영 성과 수집 시 고려된 슬라이싱 매개 변수

1-2. 데이터 전처리

데이터 전처리 과정은 크게 2단계로 나누어지며 1단계에서는 STL을 Voxel(복셀)로 변환하고 2단계에서는 설계 특징과 운영 성과를 로그 정규화하여 변환한다.

전처리 1단계에서는 STL 모델을 예측 모델의 입력데이터로 하기 위해 64*64*64 크기로 복셀화 (Voxelization) 했다. 복셀이란, volume과 pixel의 합성어로 부피를 갖는 픽셀을

의미하며 3D데이터를 행렬의 형태로 변환하는 과정이다.

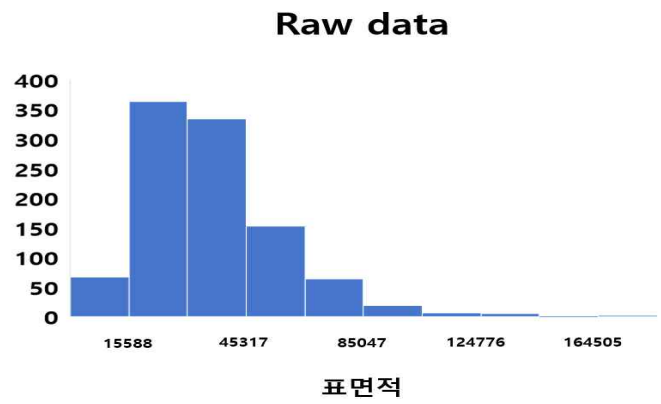


(가) 3D CAD 모델 (나) Voxel

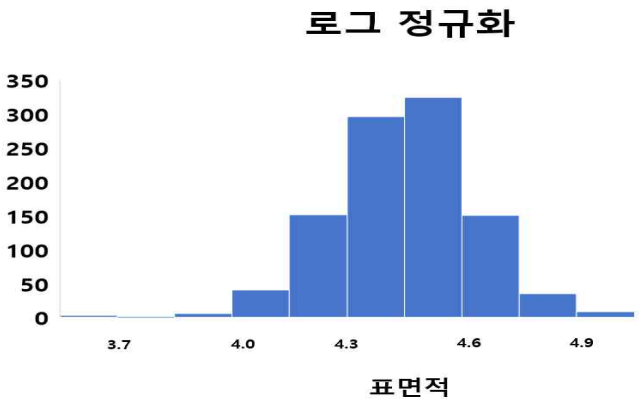
그림 6. 복셀 예시

다음으로 STL 모델의 설계 특징과 운영 성과에 대해 로그 정규화했다. 수집된 STL 파일의 설계 특징과 운영 성과는 그림 6과 같이 좌측으로 편향된 데이터 분포를 지니고 있다. 이 경우 데이터에 대한 정규화를 통해 편향된 데이터들을 고르게 분포시킬 수 있으며 Min-Max 정규화, 로그 정규화 등이 널리 사용된다. 본 연구에서 사용한 데이터에 해당 데이터셋에 Min-Max 정규화를 적용할 경우 데이터 분포가 변화하지 않기 때문에 그림 6과 같이 로그 정규화를 통해 데이터의 분포를 정규 분포에 근사시켰다.

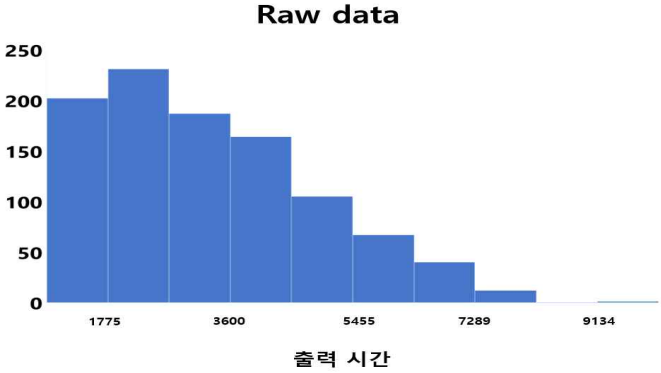
(가) 부피에 대한 원 데이터 분포 (나) 부피에 대해 로그 정규화된 분포



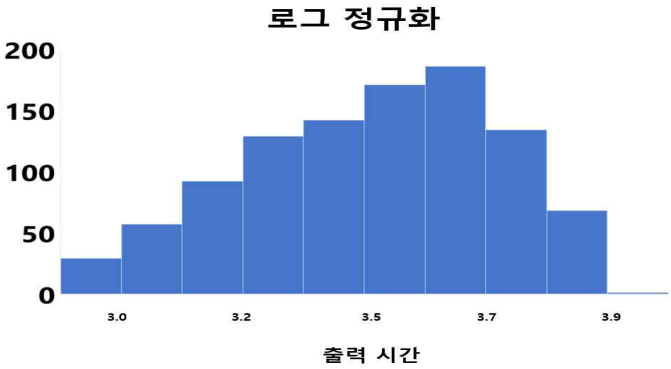
(다) 표면적에 대한 원 데이터 분포



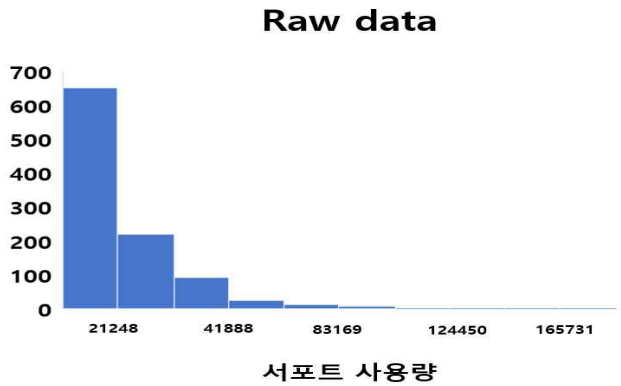
(라) 표면적에 대해 로그 정규화된 분포



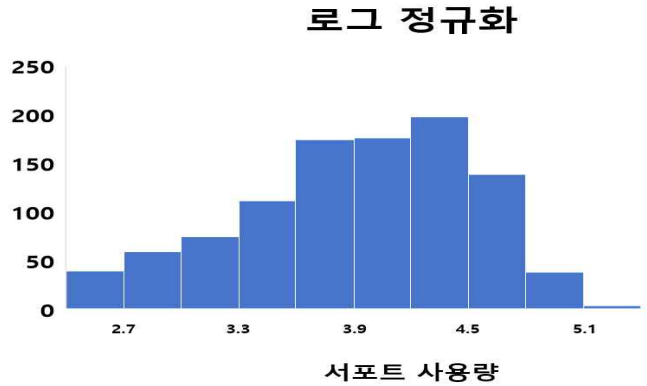
(마) 출력 시간에 대한 원 데이터 분포



(바) 출력 시간에 대해 로그 정규화된 분포



(사) 서포트 사용량에 대한 원 데이터 분포



(아) 서포트 사용량에 대해 로그 정규화된 분포

그림 7. 원 데이터와 로그 정규화된 데이터의 분포

1-3. 3D-CNN 기반 3D프린팅 제조시스템 운영성과 예측 모델 구축

딥러닝 기법 중 하나인 CNN은 비정형 데이터의 통계적 패턴을 인식하고 특징 추출을 통해 지도학습 기반으로 분류하는 방법론으로 STL의 디자인 정보를 효과적으로 추출할 수 있다.

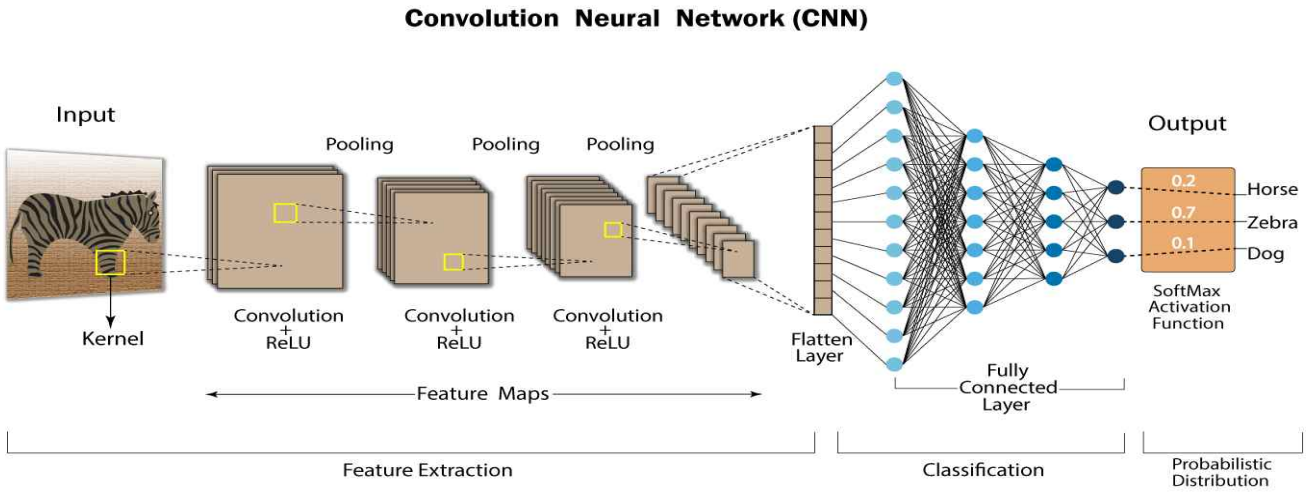


그림 8. CNN 구조 예시

이에 전처리된 데이터를 바탕으로 STL에 대한 복셀 데이터는 입력 변수, 운영 성과는 출력 변수로 하는 2개의 운영성과 예측 모델 (그림 8 참조) 을 구축했다. 이 때, 예측 성능 향상을 위해 압축된 디자인 정보(feature map)에 설계 특징을 추가적인 입력 변수로 설정하였다.

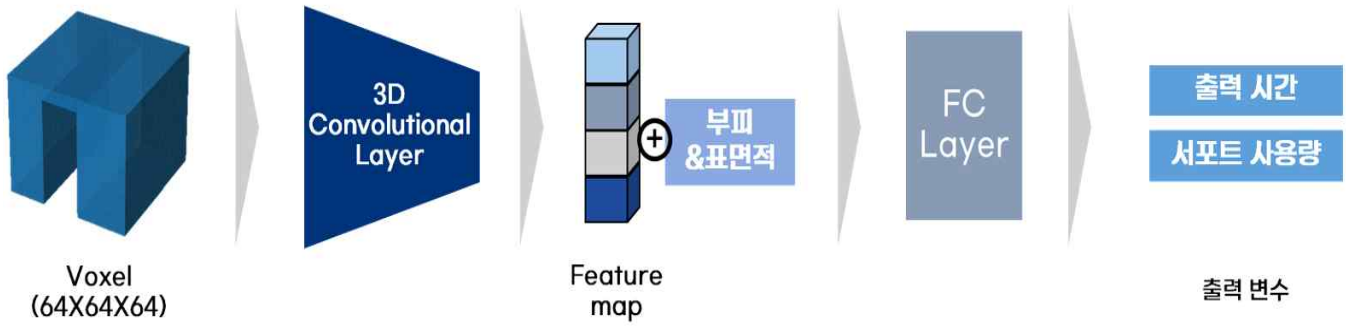


그림 9. 3D-CNN 기반 3D프린팅 제조시스템 운영성과 예측 모델의 구조

모델 구축 시에는 전체 데이터 중 80%는 훈련 데이터셋, 10%는 검증 데이터셋, 10%는 시험 데이터셋으로 나누어 훈련을 진행했다. 모델의 주요 매개 변수는 1000 epochs, 1e-5 learning rate, Adam Opimizer, 32 batch sizes를 사용했으며 과적합 방지를 위해 Early stopping을 적용하였다. 모델의 운영 성과 예측에 대한 정확도 평가를 위해 시험 데이터셋을 바탕으로 RMSE, MAE, MAPE 를 측정했다. (표 2 참조)

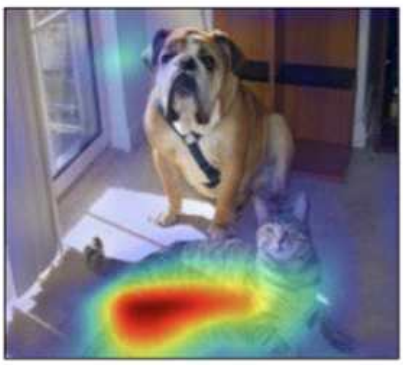
평가 지표	수식
평균 절대 오차 (*MAE*)	$\frac{\sum\lvert y_{ture} - y_{pred}\rvert}{n}$
평균 제곱근 오차 (*RMSE*)	$\sqrt{\frac{\sum(y_{ture} - y_{pred})^2}{n}}$
평균 절대 비율 오차 (*MAPE*)	$\frac{100}{n}\sum\left\lvert\frac{y_{ture} - y_{pred}}{y_{ture}}\right\rvert$

표2. 평가 지표 수식

1-4. 3D-Grad-CAM 기반 예측 결과에 대한 근거 시각화

딥러닝은 우수한 성능을 지니고 있지만 블랙 박스 특성으로 인해 예측 근거에 대한 해석이 어렵다는 한계가 존재한다. 특히 CNN의 경우 특징 추출 이후 공간 정보가 압축되어 예측 결과에 대한 해석이 더욱 어려울 수 있다. 예측 결과에 대한 해석 가능성은 예측 모델에 대한 신뢰성을 보장하기 위해 필수적이다.

Grad-CAM은 CNN의 예측 결과를 시각화하는 알고리즘으로 입력 변수의 어떤 특징이 예측에 큰 영향을 주었는지 파악할 수 있다.



(가) 고양이에 대한 Grad-CAM

(나) 원본 이미지

(다) 개에 대한 Grad-CAM

그림 10. Grad-CAM을 활용한 예측 결과 시각화 예시

이에 CNN의 Convolutional layer를 통해 추출된 복셀 데이터에 대한 디자인 압축 정보인 feature map의 가중치를 바탕으로 Grad-CAM을 활용하여 운영 성과에 대한 예측 결과를 시각화한다. 이를 통해 아래 그림과 같이 시각화된 결과를 통해 STL모델의 어떤 디자인 특징(설계 요소)이 운영 성과 예측에 있어 주요한 영향을 미쳤는지 확인할 수 있다.

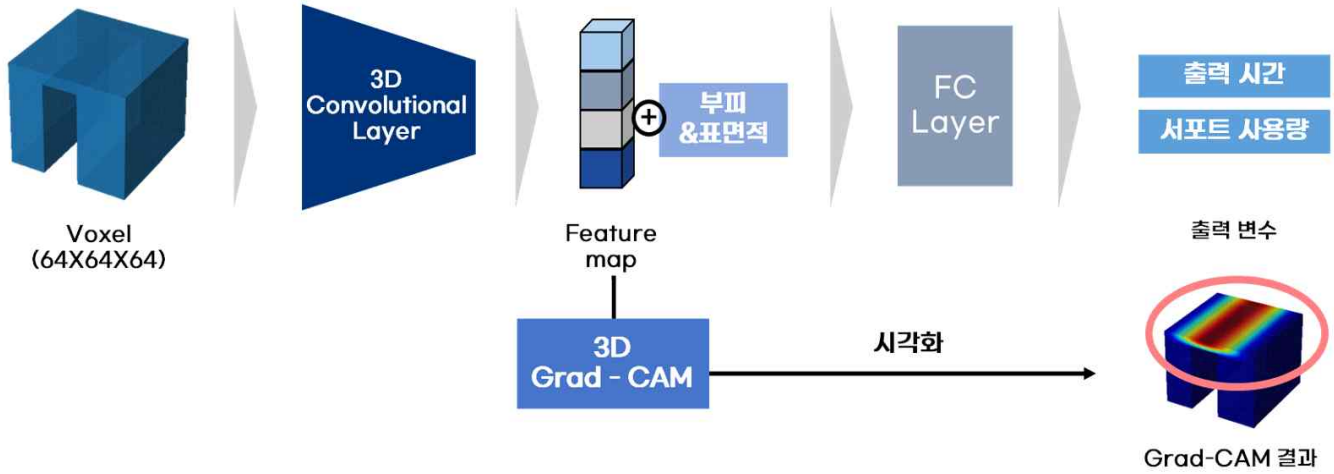


그림 11. 제안된 설명 가능한 3D프린팅 제조시스템 운영 성과 예측 프로세스

1-5. 제안된 운영 성과 산정 프로세스에 대한 사례 연구 : SimJEB(Simulated Jet Engine bracket) Dataset

제안된 운영 성과 산정 프로세스 및 예측 모델의 일반화 성능을 확인하기 위한 사례 연구를 수행하였으며 사례 연구 대상으로는 SimJEB 데이터셋을 선정하였다. SimJEB 데이터셋은 2013년 GrabCAD에서 주최한 “GE Jet Engine Bracket Challenge”에 참가한 설계 데이터 중 유한요소 시뮬레이션을 만족하는 데이터(381개)를 선별한 데이터셋이다. (그림 11 참조)

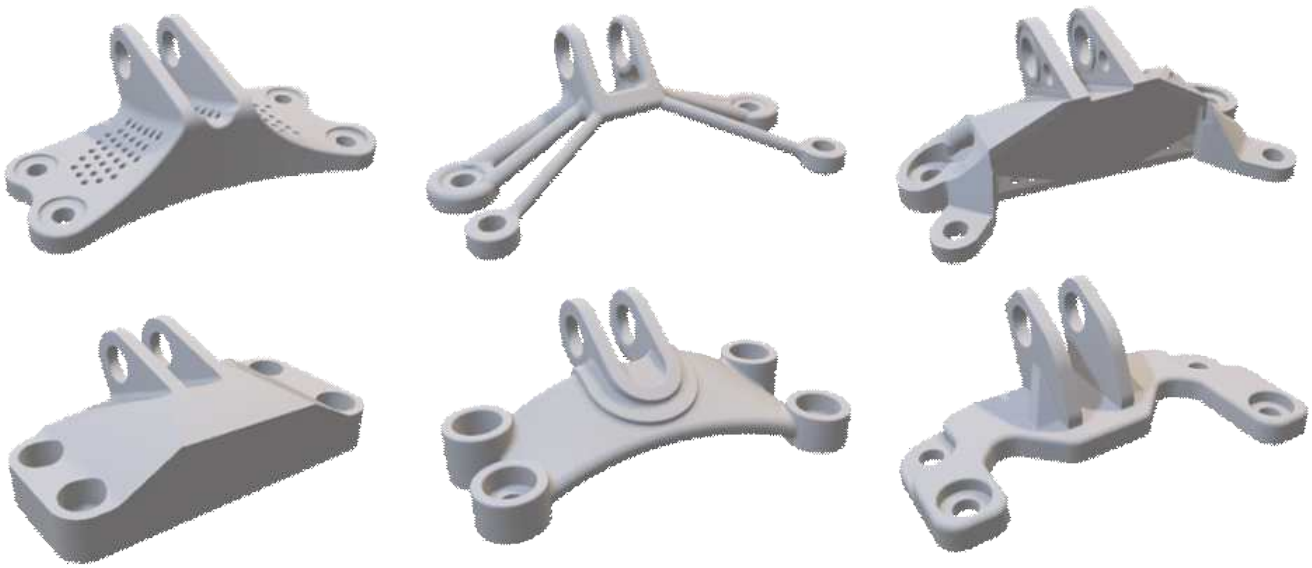


그림 12. SimJEB 데이터셋 예시

SimJEB 데이터셋에 대한 구축된 모델의 3D프린팅 운영 성과 예측 정확도를 평가하기 위해

RMSE, MAE, MAPE를 측정하였다. 이후 Grad-CAM을 활용하여 운영 성과 예측 결과의 산정 근거를 시각화하고 예측 결과에 주요한 영향을 미친 설계 요소를 식별하였다.

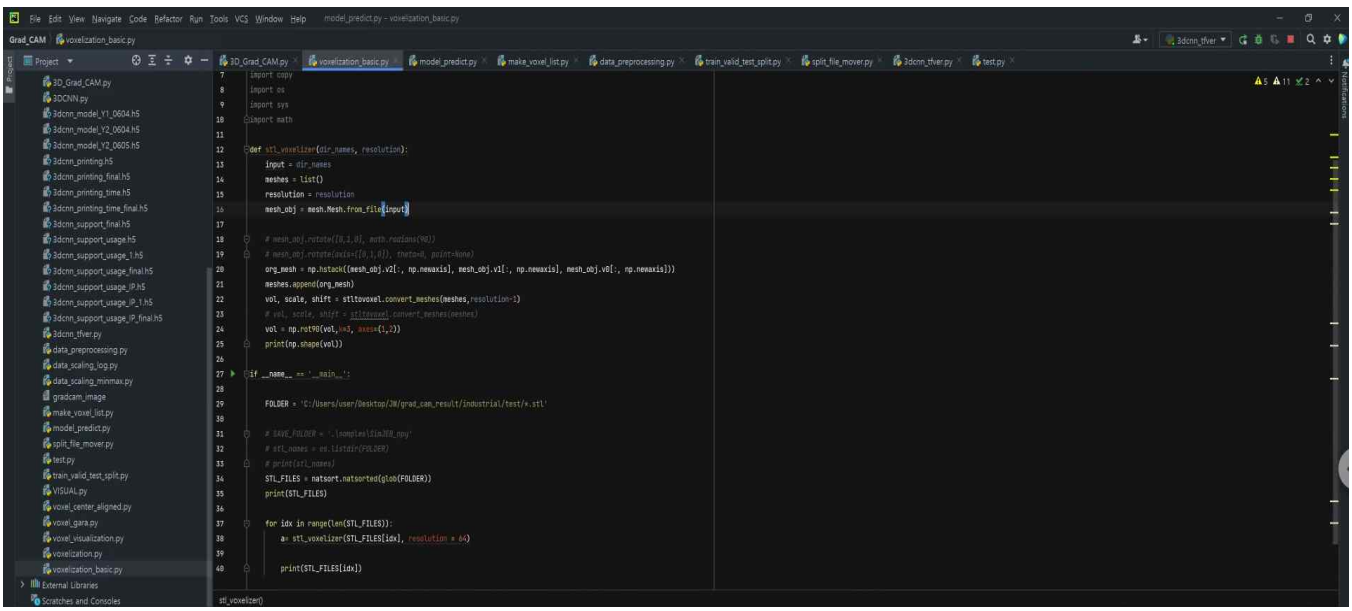
2. 제작과정

가. 제작일정

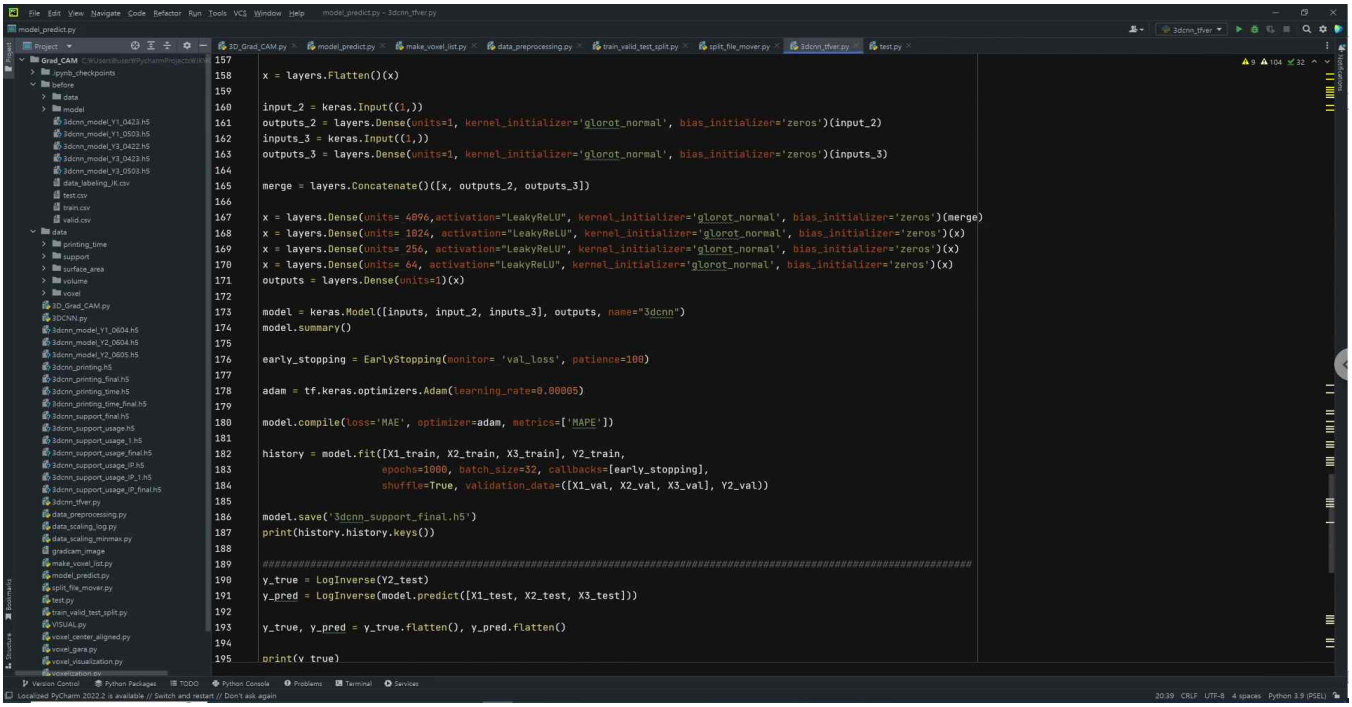
	일련번호	주요내용	제작 일정			기간(주)
			4 월	5 월	6 월	
일정표	1	3D프린팅 제조시스템 성과 예측 관련 문헌 조사				1주
	2	모델링을 위한 데이터 수집 및 전처리				2주
	3	3D CNN 모델 및 3D-Grad-CAM 알고리즘 설계				4주
	4	결과 분석				2주
	5	피드백 및 결과 검토				2주

나. 상세도면 및 과제사진 (설계도, 부품도, 제작과정 으로 자세히 기술)

1. STL 모델 복셀화‘

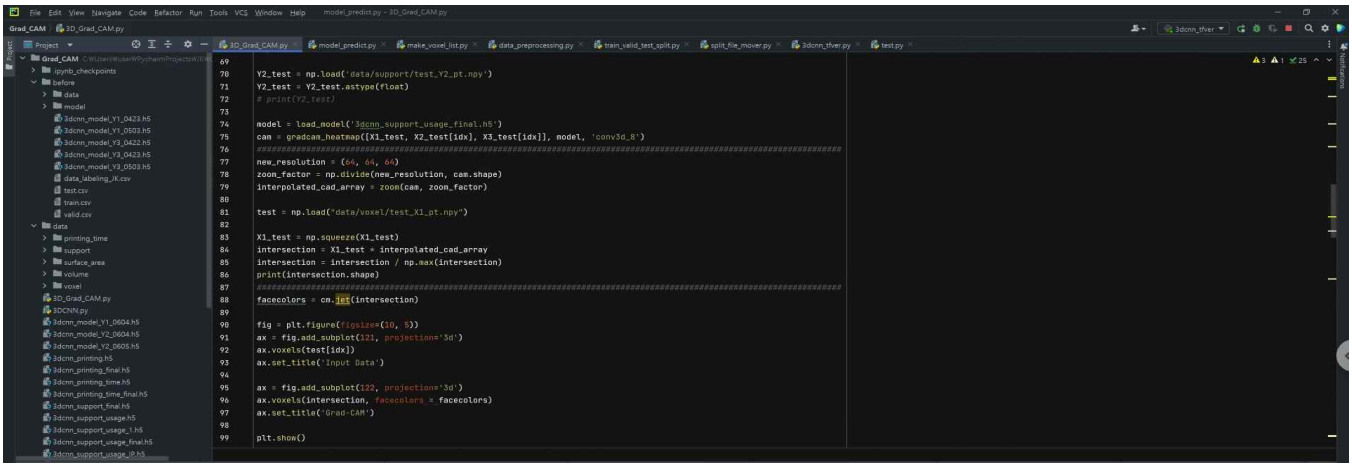


2. 3D-CNN 구축



```
x = layers.Flatten()(x)

input_2 = keras.Input((1,))
outputs_2 = layers.Dense(units=1, kernel_initializer='glorot_normal', bias_initializer='zeros')(input_2)
inputs_3 = keras.Input((1,))
outputs_3 = layers.Dense(units=1, kernel_initializer='glorot_normal', bias_initializer='zeros')(inputs_3)

merge = layers.Concatenate()([x, outputs_2, outputs_3])

x = layers.Dense(units= 4096,activation="LeakyReLU", kernel_initializer='glorot_normal', bias_initializer='zeros')(merge)
x = layers.Dense(units= 1024, activation="LeakyReLU", kernel_initializer='glorot_normal', bias_initializer='zeros')(x)
x = layers.Dense(units= 256, activation="LeakyReLU", kernel_initializer='glorot_normal', bias_initializer='zeros')(x)
x = layers.Dense(units= 64, activation="LeakyReLU", kernel_initializer='glorot_normal', bias_initializer='zeros')(x)
outputs = layers.Dense(units=1)(x)

model = keras.Model([inputs, input_2, inputs_3], outputs, name="3dcnn")
model.summary()

early_stopping = EarlyStopping(monitor= 'val_loss', patience=100)

adam = tf.keras.optimizers.Adam(learning_rate=0.00005)

model.compile(loss='MAE', optimizer=adam, metrics=['MAPE'])

history = model.fit([X1_train, X2_train, X3_train], Y2_train,
                    epochs=1000, batch_size=32, callbacks=[early_stopping],
                    shuffle=True, validation_data=([X1_val, X2_val, X3_val], Y2_val))

model.save('3dcnn_support_final.h5')
print(history.history.keys())

########################################################################################################################
y_true = LogInverse(Y2_test)
y_pred = LogInverse(model.predict([X1_test, X2_test, X3_test]))

y_true, y_pred = y_true.flatten(), y_pred.flatten()

print(y_true)
```

3. 3D-Grad-CAM



```
Y2_test = np.load('data/support/test_Y2_pt.npy')
Y2_test = Y2_test.astype(float)
# print(Y2_test)

model = load_model('3dcnn_support_usage_final.h5')
cam = gradcam_heatmap([X1_test, X2_test[idx], X3_test[idx]], model, 'conv3d_8')
########################################################################################################################
new_resolution = (64, 64, 64)
zoom_factor = np.divide(new_resolution, cam.shape)
interpolated_cad_array = zoom(cam, zoom_factor)

test = np.load("data/voxel/test_X1_pt.npy")

X1_test = np.squeeze(X1_test)
intersection = X1_test * interpolated_cad_array
intersection = intersection / np.max(intersection)
print(intersection.shape)
########################################################################################################################
facecolors = cm.jet(intersection)

fig = plt.figure(figsize=(10, 5))
ax = fig.add_subplot(121, projection='3d')
ax.voxels(test[idx])
ax.set_title('Input Data')

ax = fig.add_subplot(122, projection='3d')
ax.voxels(intersection, facecolors = facecolors)
ax.set_title('Grad-CAM')

plt.show()
```

4. 소프트웨어 구축

```python
def stl_open(self):

    stl_name, _ = QFileDialog.getOpenFileName(self, "Open STL file", "", "STL Files (*.stl)")

    if stl_name:

        # 파일 경로 업데이트
        self.label_file_path.setText(stl_name)

        stl_file = self.label_file_path.text()
        strings = stl_file.split('/')
        strings = strings[-1]
        string_s = strings.split('.')
        string_s = string_s[0]

        total_area = stl_surface_area_calculate(stl_name)
        total_area_scale = np.log(total_area)
        print(total_area_scale)
        np.save(f'{string_s}_surface.npy', total_area_scale)
        self.label_mesh.setText("{}".format(total_area))

        total_volume = stl_volume_calculate(stl_name)
        total_volume_scale = np.log(total_volume)
        print(total_volume_scale)
        np.save(f'{string_s}_volume.npy', total_volume_scale)
        self.label_volume.setText("{}".format(total_volume))

        self.btn.setEnabled(True)

    else:
        # 파일 선택 취소시 "No file selected" 표시
        self.label_file_path.setText("No file selected")
```

(가) STL 파일 삽입

```python
def voxel_start(self):

    stl_file = self.label_file_path.text()

    os.getcwd()

    voxel = []

    a = stl_voxelizer(stl_file, resolution=64)
    voxel.append(a)

    print(np.shape(a))

    if np.shape(a) != (64, 64, 64):
        print("Current index = ", stl_file, "voxel=", a)
    else:
        print('good', stl_file)

    voxel = np.array(voxel)
    print(np.shape(voxel))
    voxel = np.squeeze(voxel)
    print(np.shape(voxel))

    # voxel = np.rot90(voxel, 1, axes=(0, 2))

    strings = stl_file.split('/')
    strings = strings[-1]
    string_s = strings.split('.')
    string_s = string_s[0]
    print(string_s)

    np.save(f'{string_s}.npy', voxel)

    matplotlib.use('Qt5Agg')

    fig = plt.figure()
```

(나) 데이터 전처리

```python
def result_click(self):

    stl_file = self.label_file_path.text()

    strings = stl_file.split('/')
    strings = strings[-1]
    string_s = strings.split('.')
    string_s = string_s[0]
    # print(string_s)

    self.label_voxel_name.setText(string_s)
    font = self.label_voxel_name.font()
    font.setPointSize(15)
    self.label_voxel_name.setFont(font)

    # 3D CNN #
    # 복셀
    X1_test = np.load(f'{string_s}.npy')
    print(X1_test.shape)
    X1_test = X1_test.astype(float)
    X1_test = np.expand_dims(X1_test, axis=-1)
    X1_test = np.expand_dims(X1_test, axis=0)

    # 부피(스캔월렛)
    X2_test = np.load(f'{string_s}_volume.npy')
    X2_test = X2_test.astype(float)
    X2_test = np.expand_dims(X2_test, axis=-1)
    X2_test = np.expand_dims(X2_test, axis=0)

    # 표면적(스캔월렛)
    X3_test = np.load(f'{string_s}_surface.npy')
    X3_test = X3_test.astype(float)
    X3_test = np.expand_dims(X2_test, axis=-1)
    X3_test = np.expand_dims(X2_test, axis=0)

    model_PT = load_model('3dcnn_printing.h5')  # y1 -> printing time y3 -> support usage

    y_pred_PT = model_PT.predict([X1_test, X2_test, X3_test])
```

(다) 모델을 통한 운영성과 예측

```python
cam = gradcam_heatmap([X1_test, X2_test, X3_test], model_PT, 'conv3d_6')

new_resolution = (64, 64, 64)
zoom_factor = np.divide(new_resolution, cam.shape)
interpolated_cad_array = zoom(cam, zoom_factor)

test = np.load(f'{string_s}.npy')

X1_test = np.squeeze(X1_test)
intersection = X1_test * interpolated_cad_array
intersection = intersection / np.max(intersection)

print(intersection.shape)

facecolors = cm.jet(intersection)

matplotlib.use('Qt5Agg')

fig = plt.figure()

plt.ion()

ax = fig.add_subplot(122, projection='3d')
ax.voxels(intersection, facecolors=facecolors)
ax.set_title('Grad-CAM(Printing time)')
# plt.show()
ax.set_xlim(0, 64)
ax.set_ylim(0, 64)
ax.set_zlim(0, 64)

plt.ioff()

plt.savefig(f'{string_s}_cam1.png', bbox_inches='tight')

plt.close(fig)
```

(라) Grad=CAM

Ⅲ. 결론

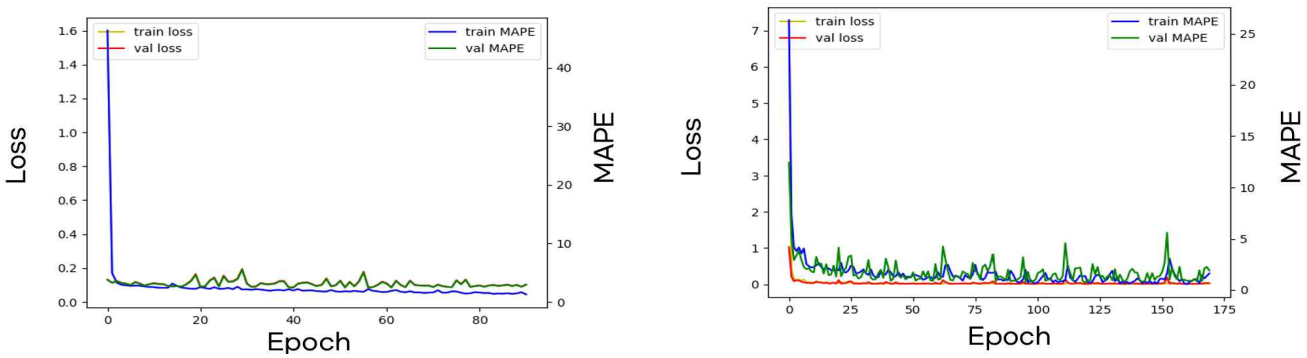
1. 과제개발성과

1-1. 시험 데이터셋에 대한 운영성과 예측 모델의 정확도

표 3은 시험 데이터셋에 대한 운영성과 예측 모델의 정확도를 나타낸다. 구축된 3D프린팅 운영성과 예측 모델은 낮은 오차율로 시험 데이터셋의 운영성과를 추정함을 보였다.

출력변수	RMSE	MAE	MAPE
출력 시간	343.52(min)	479.18(min)	1.4851(%)
서포트 사용량	1627.29(g)	1019.23(g)	3.5824(%)

표 3. 운영성과 예측 모델의 시험 데이터셋에 대한 예측 정확도



(a) 출력 시간에 대한 예측 모델의 훈련 결과 (b) 서포트 사용량에 대한 예측 모델의 훈련 결과

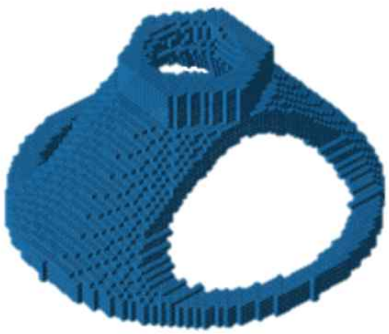
그림 13. 3D프린팅 제조시스템 운영성과 예측 모델의 훈련 결과

1-2. 시험 데이터셋에 대한 운영성과 예측 및 시각화 결과

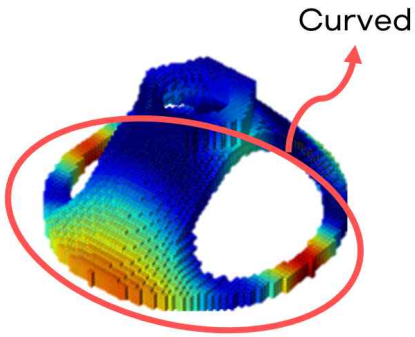
그림 14는 시험 데이터셋의 일부에 대한 3D프린팅 운영성과 예측 및 시각화 결과를 나타낸다. 그림 14-(가), (나)의 표는 각 STL 모델에 대한 3D프린팅 운영성과는 상용 소프트웨어 Simplify로 파악한 결과와 예측 모델의 운영 성과 예측값을 나타내며 상용 소프트웨어로 파악된 운영 성과와 예측된 운영성과 간 차이가 매우 적음을 확인할 수 있다.

또한 3D-Grad-CAM을 활용한 출력 시간에 대한 시각화 결과에서는 그림 14-(가)의 경우 밑면의 곡선 구조, (나)의 경우 밑면에서 출력 시간 예측에 큰 영향을 미친 것으로 확인되었다. 또한 서포트 사용량 예측에 대한 시각화 결과에서는 (가)와 (나) 모두 공중에

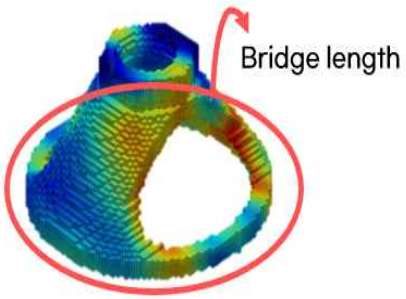
떠있는 구조를 지지하기 위한 설계 요소가 서포터 사용량 예측에 큰 영향을 미친 것으로 확인되었다.



Voxel



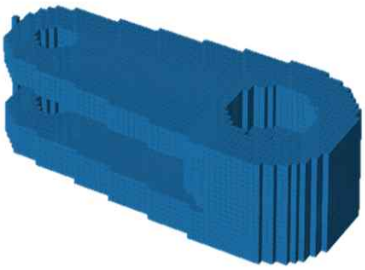
출력 시간에 대한 시각화



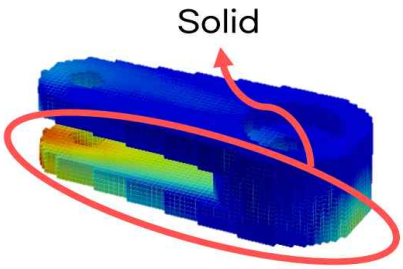
서포트 사용량에 대한 시각화

	출력 시간(min)	서포트 사용량(g)
실제(simplify 3d)	1745	8327
예측값	1788.2	8115.6

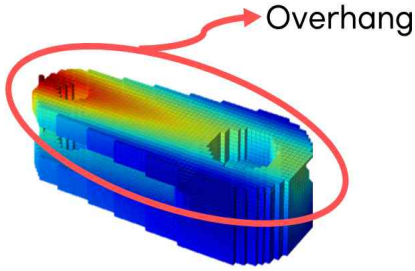
(가)



Voxel



출력 시간에 대한 시각화



서포트 사용량에 대한 시각화

	출력 시간(min)	서포트 사용량(g)
실제(simplify 3d)	3370	20044
예측값	3259.3	19808.8

(나)

그림 14. 시험 데이터셋에 대한 운영성과 예측값과 실제값 비교 및 시각화

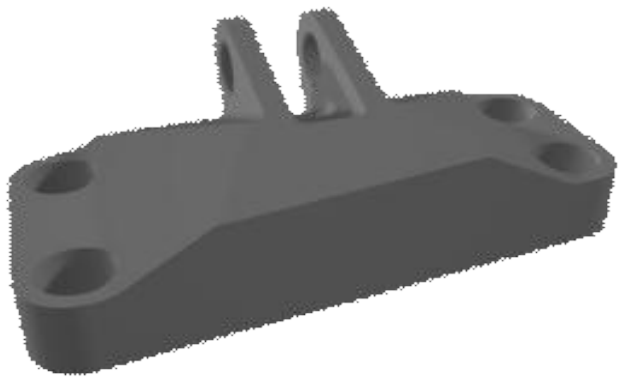
1-3. 사례 연구 : SimJEB 데이터셋에 대한 운영성과 예측 모델의 정확도

표 4는 구축된 운영성과 예측 모델의 SimJEB 데이터셋에 대한 정확도 평가 결과를 나타낸다. 예측 모델은 낮은 오차율로 SimJEB 데이터셋의 기대 출력시간 및 서포트 사용량을 추정함을 알 수 있다.

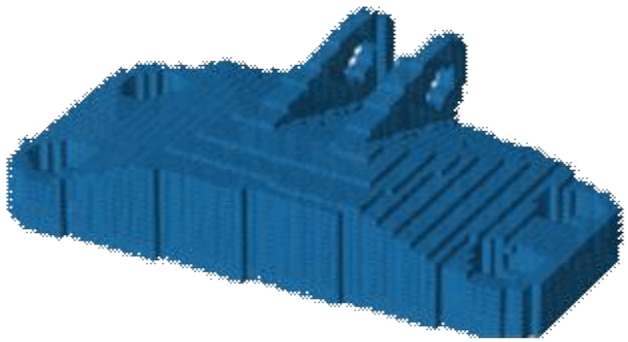
출력변수	RMSE	MAE	MAPE
출력 시간	598.7(min)	851.23(min)	2.067(%)
서포트 사용량	2980.74(g)	1502.78(g)	5.9241(%)

표 4. 운영성과 예측 모델의 SimJEB 데이터셋에 대한 예측 정확도

그림 15는 각 디자인에 대해 슬라이싱 소프트웨인 simplify 3d의 실제 출력 시간 및 서포트 사용량과 예측 모델의 예측 출력 시간과 서포트 사용량을 비교한 결과이다. 해당 결과를 통해 실제값과 예측값이 근소한 차이만을 보이는 것을 확인할 수 있다.



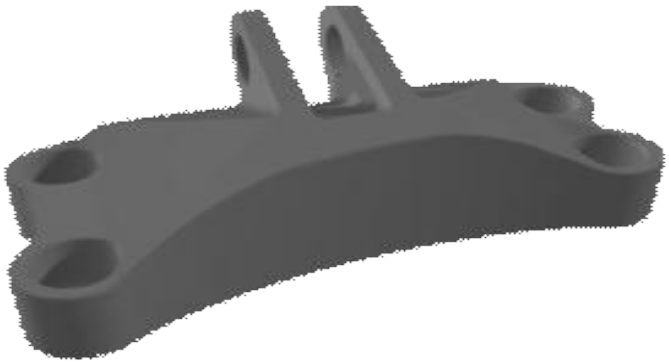
STL 원본



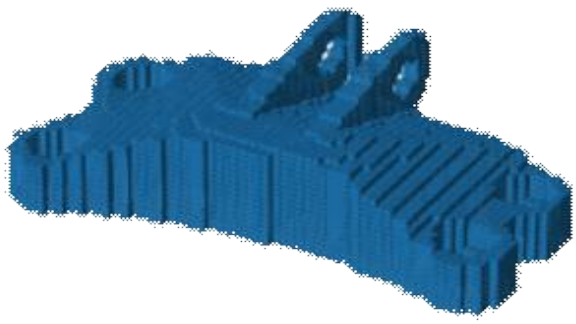
Voxel

	출력 시간(min)	서포트 사용량(g)
실제(simplify 3d)	5265	15535
예측값	5278.2	15789.58

(가)



STL 원본



Voxel

	출력 시간(min)	서포트 사용량(g)
실제(simplify 3d)	4065	23330.5
예측값	4052.8	23083.26

(나)

그림 15. SimJEB 데이터셋에 대한 운영성과 예측 결과

1-4. 사례 연구 : SimJEB 데이터셋에 대한 운영성과 예측 및 시각화 결과

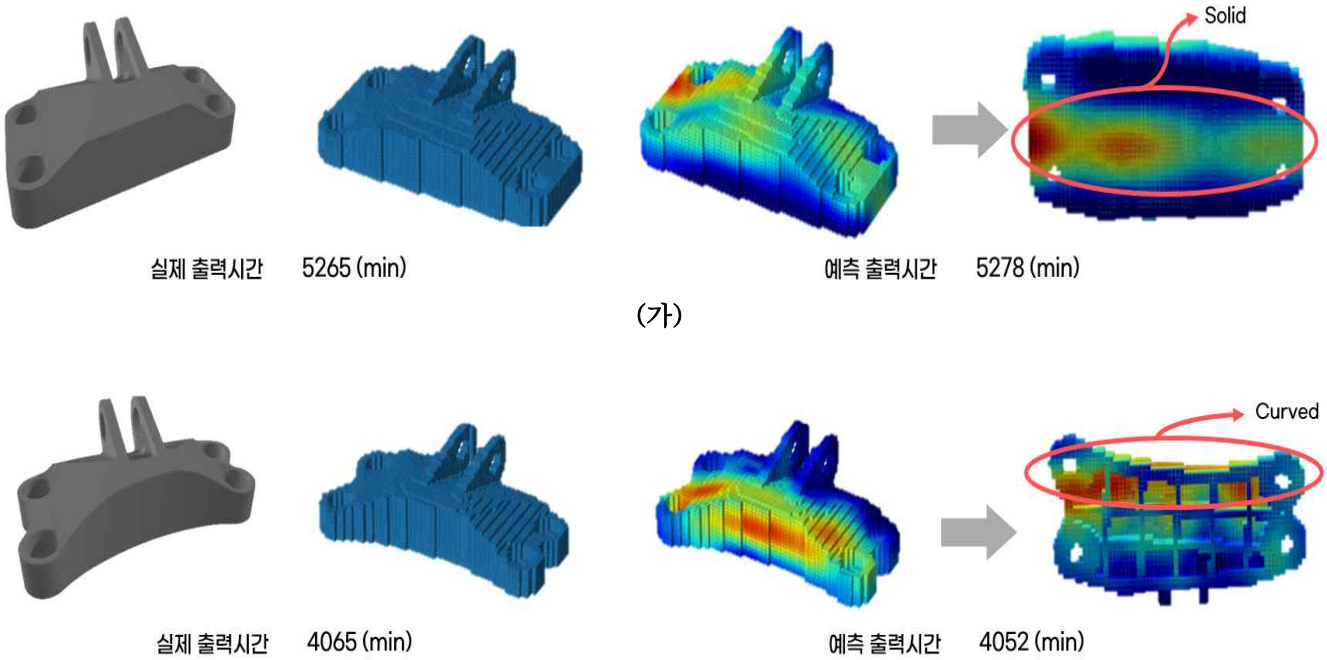
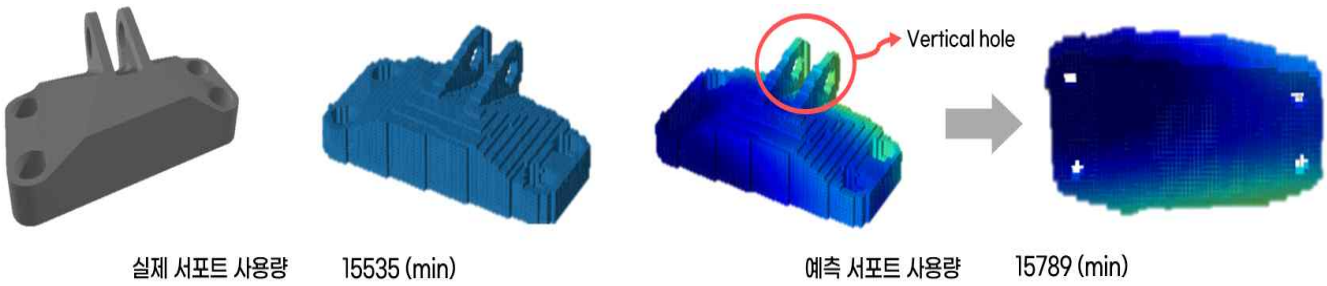
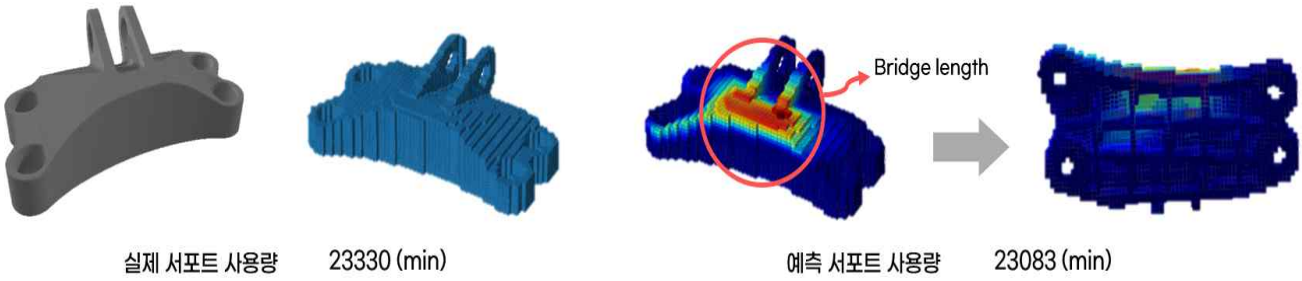


그림 16. Grad-CAM을 활용한 출력 시간 예측 결과에 대한 시각화

그림 16의 (가), (나) 는 유사한 형상임에도 출력 시간 예측 및 Grad-CAM 결과에서 차이가 존재함을 보였다. 이는 내부와 전면 구조에 의한 것으로 상대적으로 그림 16-(가)의 경우 밑면부, (나)의 경우 전면 곡선부에서 출력 시간 예측 결과에 더 큰 영향을 미치는 것으로 확인되었다.



(가)



(나)

그림 17. Grad-CAM을 활용한 서포트 사용량 예측 결과에 대한 시각화

서포트 사용량 또한 내부 구조에 의한 차이로 인해 예측 및 Grad-CAM 결과에서 차이를 보였다. 그림 17-(가)의 경우 3D프린팅 특화 설계 요소 중 vertical hole로 인해, (나)의 경우 비어있는 밑면을 지지하기 위한 윗면부에서 보와 보 사이의 거리를 나타내는 요소인 bridge length로 인해 서포트 사용량 예측에 큰 영향을 미치는 것을 확인하였다.

2. 제안된 프로세스의 일원화를 위한 소프트웨어 구축

그림 18. 제안된 프로세스에 대한 소프트웨어 구현 예시

본 연구에서는 Grad-CAM을 활용하여 STL 파일 형식의 3D CAD을 설명가능한 인공지능 기반으로 3D 프린팅 제조시스템 운영성과 예측 프로세스를 제안하였다. Python-PyQt5 라이브러리를 사용하여 데이터 입력부터 결과 시각화까지의 전 과정을 일원화하기 위한 소프트웨어를 구축하였다.

제안된 프로세스에 대한 소프트웨어 사용 시나리오는 아래와 같다.

1. STL 파일 형식의 3D CAD 디자인을 선정한다. ‘STL File Open’ 버튼을 클릭하여 STL 모델을 삽입한다. 파일 삽입 시 디자인에 대한 표면적과 부피가 계산되어 나타난다.

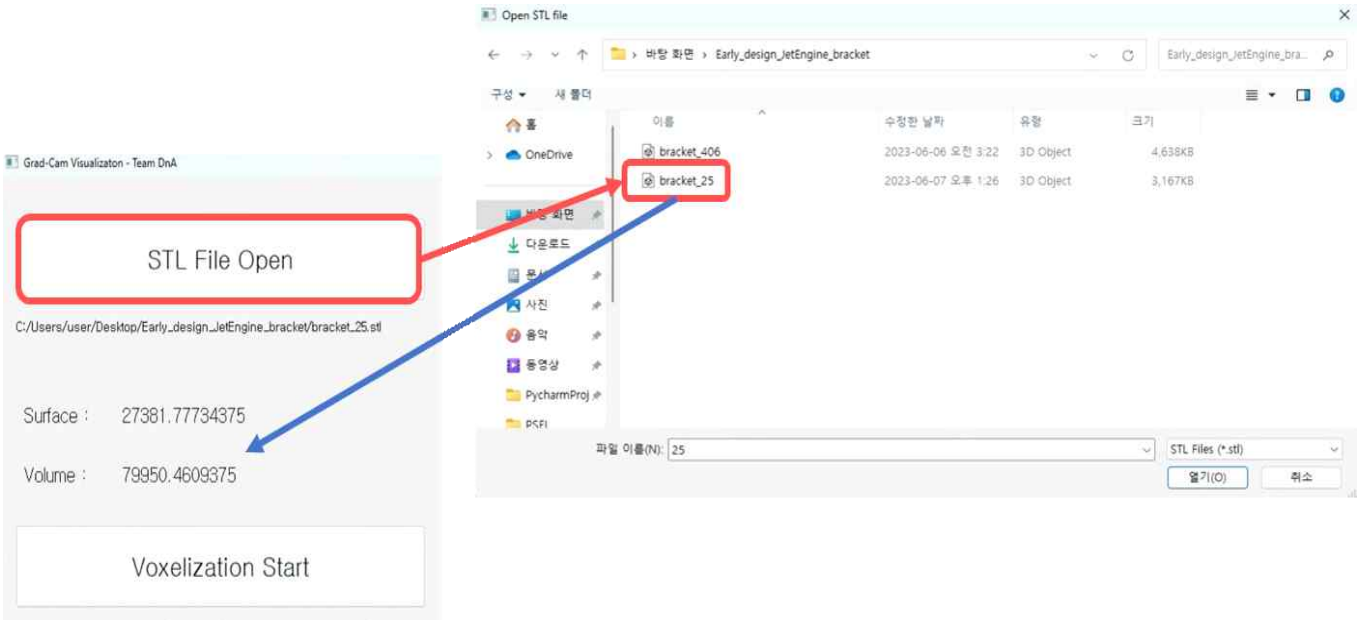


그림 19. STL 파일 삽입 예시

2. ‘Voxelization Start’ 버튼을 클릭 시 삽입된 디자인의 운영 성과 예측을 위한 데이터 입력 및 전처리 과정이 진행된다. 1번 과정에서 삽입된 디자인의 표면적과 부피 값이 로그 정규화되고 복셀화 된다. 복셀화된 데이터의 형상은 이미지로 저장되어 소프트웨어상에서 미리보기 창을 통해 확인할 수 있다.

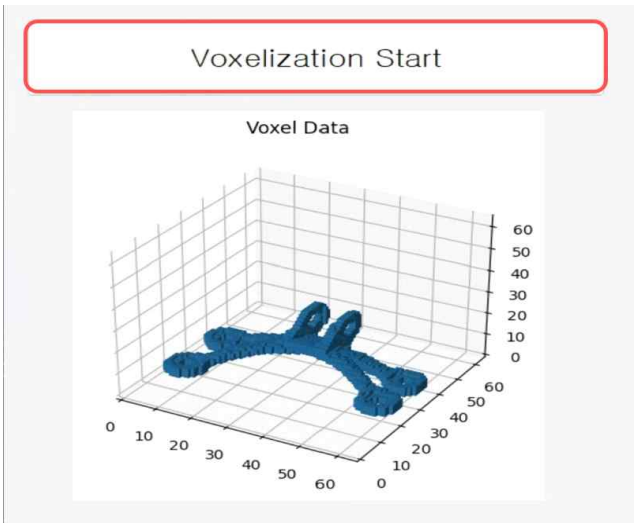


그림 20. Voxelization 시각화

3. 전처리된 데이터를 통해 3D프린팅 제조 운영성과를 예측하고 시각화하기 위해 ‘Result’ 버튼을 클릭 시 해당 데이터가 구축된 예측 모델의 입력값이 되어 서포트 사용량과 출력 시간이 예측된다. 또한 각 예측값에 대해 시각화된 Grad-CAM 결과가 2번 과정과 같이 이미지로 저장되어 소프트웨어상에서 확인할 수 있다.

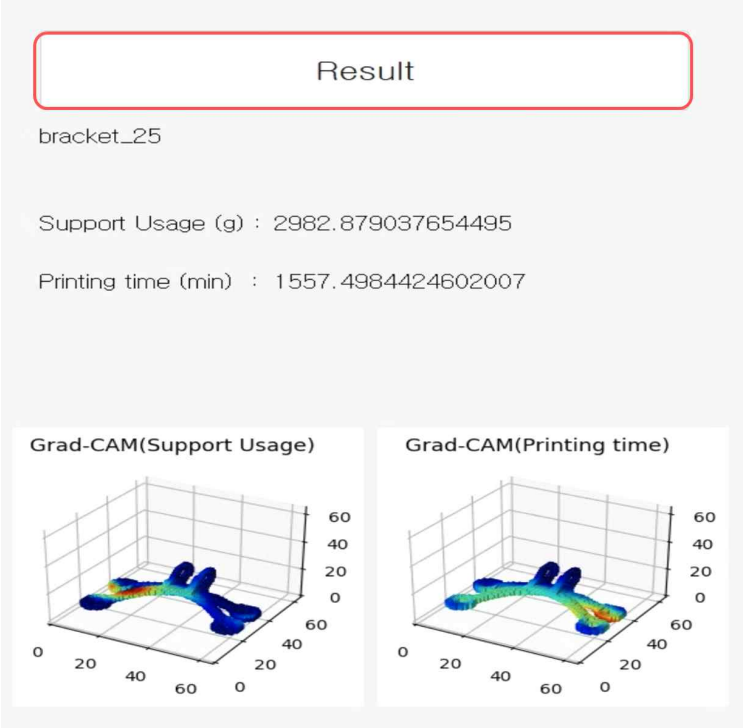


그림 21. 운영성과 예측 결과 및 Grad-CAM 시각화 예시

해당 소프트웨어를 통해 디자이너는 제품 설계 단계에서 운영성과에 주요한 영향을 미치는 설계 요소를 식별하고 설계를 수정할 수 있다. 본 연구에서 제시되는 3D프린팅 제조시스템 운영성과 예측 및 설계 요소 시각화 프로세스를 활용하여 제품 디자이너는 3D프린팅을 이용한 제품 초기 설계 단계에서 운영 성과에 주요한 영향을 미치는 설계 요소를 파악할 수 있다. 이를 통해 디자이너는 제품에 대한 디자인의 3D프린팅 적합성 및 제조 운영 성과를 빠르게 평가할 수 있으며 나아가 목표하는 3D프린팅 제조시스템 운영성과 달성을 위해 제품을 수정할 수 있다. 또한 제안된 프로세스는 3D프린팅 특화 설계 관련 지식이나 정보가 부족한 디자이너의 설계 작업 효율성을 제고할 수 있을 것으로 기대된다.

2. 문제점 및 개선방안

(작품에 대한 것뿐만 아니라, 과제수행의 전반적인 내용에 대하여 기술)

본 연구에서 구축된 프로세스에 활용되는 예측 모델은 딥러닝 기반의 모델이기 때문에 충분한 양의 데이터가 요구된다. 특히, 산업용 부품의 디자인은 각 분야, 용도 등에 따라 다르기 때문에 다양한 형태의 데이터 수집이 필수적이다. 해당 연구에서 수집한 데이터로 학습된 모델은 준수한 예측 성능을 보였지만 향후 연구에서 데이터를 추가적으로 수집하여 더 높은 정확도를 지닌 모델을 구축할 필요성이 있다. 또한, 추가적인 분석을 통해 시각적으로 식별된 특징(Grad-CAM의 결과)이 실제 3D프린팅 제조 운영성과에 있어 부정적인 영향을 미치는지 파악이 요구된다. 본 연구에서 예측 모델 구축 시 제조 성과 중 서포트 사용량, 출력 시간과 같아 운영적 측면에 초점을 두었지만, 추후 연구에서는 기계적 강도와 같은 기능적인 측면을 함께 고려하고자 한다.

Ⅳ. 참고문헌

[1] S. Bin Maidin, I. Campbell, E. Pei. (2012). Development of a design feature database to support design for additive manufacturing, Assembly Automation 32(3) 235-244
[2] Booth, J. W., Alperovich, J., Chawla, P., Ma, J., Reid, T. N., & Ramani, K. (2017). The design for additive manufacturing worksheet.journal of Mechanical Design, 139(10), 100904.
[3] S.C. Renjith, K. Park, G.E. Okudan Kremer. (2019). A design framework for additive manufacturing: Integration of additive manufacturing capabilities in the early design process, International Journal of Precision Engineering and Manufacturing 21(2) 329-345.
[4] Ning, F., Shi, Y., Cai, M., Xu, W., & Zhang, X. (2020). Manufacturing cost estimation based on a deep-learning method.journal of Manufacturing Systems, 54, 186-195.
[5] Williams, G., Meisel, N. A., Simpson, T. W., & McComb, C. (2019). Design repository effectiveness for 3D convolutional neural networks: Application to additive manufacturing.journal of Mechanical Design, 141(11).
[6] Yoo, S., & Kang, N. (2021). Explainable artificial intelligence for manufacturing cost estimation and machining feature visualization. Expert Systems with Applications, 183, 115430
[7] Smith, Z., & Pettis, B. Thingiverse, 2011. URL https://www. thingiverse. com.
[8] GrabCAD. (2020). GrabCAD Community Library.
[9] GrabCAD. (2013). GE jet engine bracket challenge. GrabCAD, 2013.
[10] Simplify3D, Simplify3D® Version 4.1, 2022. available at: https://www.simplify3d.com. (Accessed 14 July 2022).

부록 (지원재료/시작품사진, 경진대회참여 등 여러컷 및 아이디어봄 캡쳐사진 첨부)