

1. 프로젝트 개요

이번 미션의 목표는 **Streamlit 기반 웹 서비스**를 구현하여, 사용자가 직접 그린 숫자를 ONNX 모델(MNIST 분류기)에 입력하고 **0~9까지의 숫자를 예측** 하는 시스템을 만드는 것이다.

이를 통해 **모델 추론 서비스화(Serving)**, 웹 인터페이스 구성, 처리 파이프라인 설계를 경험한다.

2. 시스템 구조

(1) Streamlit 인터페이스

- 웹 페이지 실행 시 좌측에는 **모델 설정 옵션**(GPU/CPU 선택),
 - 본문에는 ① 입력 캔버스 → ② 전처리 이미지 → ③ 추론 결과(막대 차트) → ④ 이미지 저장소
- 의 4단계 UI가 제공된다.

(2) ONNX 모델 로드

- `onnxruntime.InferenceSession` 을 사용해 MNIST ONNX 모델(`mnist-12.onnx`)을 불러옴
- `@st.cache_resource` 를 사용하여 모델 세션을 캐싱, 불필요한 재로드를 방지한다.

(3) 입력 & 전처리

- `streamlit-drawable-canvas` 라이브러리를 사용하여 마우스로 숫자를 직접 그릴 수 있는 영역 제공
- 사용자 입력은 RGBA 이미지 형태로 `canvas_result` 객체에 저장됨
- 전처리 파이프라인:
 1. `preprocess_canvas_to_mnist()` : RGBA 이미지를 흑백 변환 → 블러링 → 대비 조정
 2. `autocrop_pil()` : 실제 숫자 영역만 crop + padding
 3. `to_mnist_28x28()` : MNIST 입력 규격(28×28, 중앙 정렬, 반전 처리)으로 변환

(4) 모델 추론 결과

- `predict_mnist()` : 전처리된 이미지를 ONNX 모델에 입력하여 0~9 클래스 확률 예측
- `_softmax()` : 모델 출력 logits을 확률 벡터로 변환

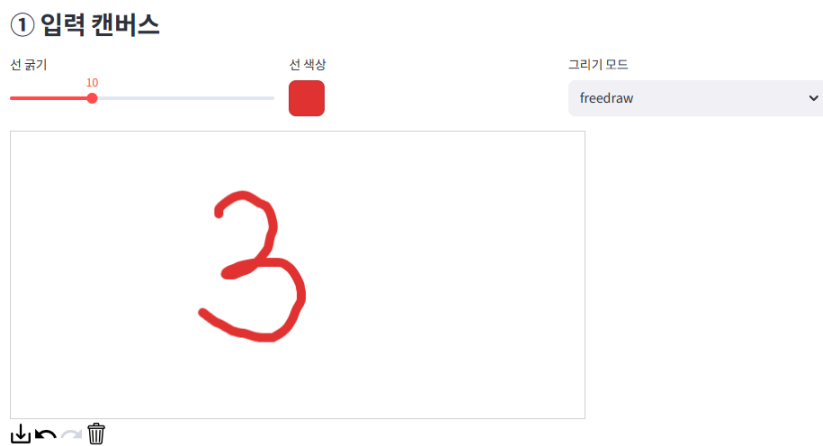
- Streamlit 막대 차트(`st.bar_chart`)로 시각화, Top-3 예측 결과도 표시

(5) 이미지 저장소 (갤러리)

- `pil_to_png_bytes()` : PIL 이미지를 PNG 바이트로 변환하여 저장
 - 세션 상태(`st.session_state.gallery`)에 예측된 이미지와 확률 기록
 - UI에 갤러리 영역을 만들어 과거 입력 및 예측 결과를 확인 가능
-

3. 실행 화면

- **입력 캔버스:** 사용자가 숫자 "3"을 그림

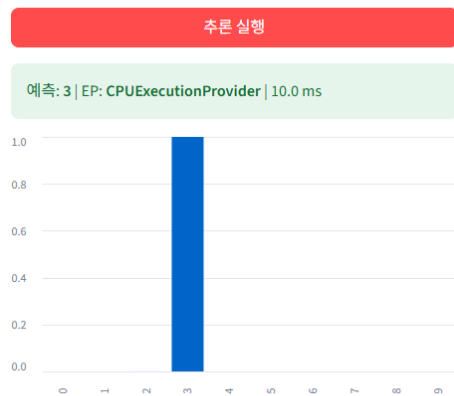


- **전처리 이미지:** Grayscale, Crop, Resize (28×28)



- 예측 결과: 예측 = 3 , Top-3 {3:0.95, 5:0.03, 8:0.01}

예측 확률 (0~9)



Top-3:

```
{
  "2" : 0.000009519277227809653
  "3" : 0.9999840259552002
  "5" : 0.00000605923924013041
}
```

- 갤러리: 이전 예측 기록을 모아볼 수 있음

④ 이미지 저장소 (예측 라벨/확률 포함)

갤러리 비우
기



Pred: 3

Top-3 → 3:1.00, 2:0.00, 8:0.00

4. 결론 및 학습 포인트

모델 캐싱을 통한 효율적 추론 관리

- ONNX Runtime 세션은 한 번 생성하면 메모리에 유지되는 객체로, 재사용이 가능하다.
- Streamlit에서는 `@st.cache_resource` 데코레이터를 사용하여 세션 객체를 캐싱함으로써,
 - 앱을 새로 고치거나 다른 동작을 해도 모델을 다시 불러오지 않음
 - 불필요한 디스크 접근/메모리 로드 비용 절감
 - 추론 버튼 클릭 시마다 즉시 응답 가능

5. Docker Hub URL

<https://hub.docker.com/repository/docker/daeseok2233/mnist-streamlit/general>