Lab 11

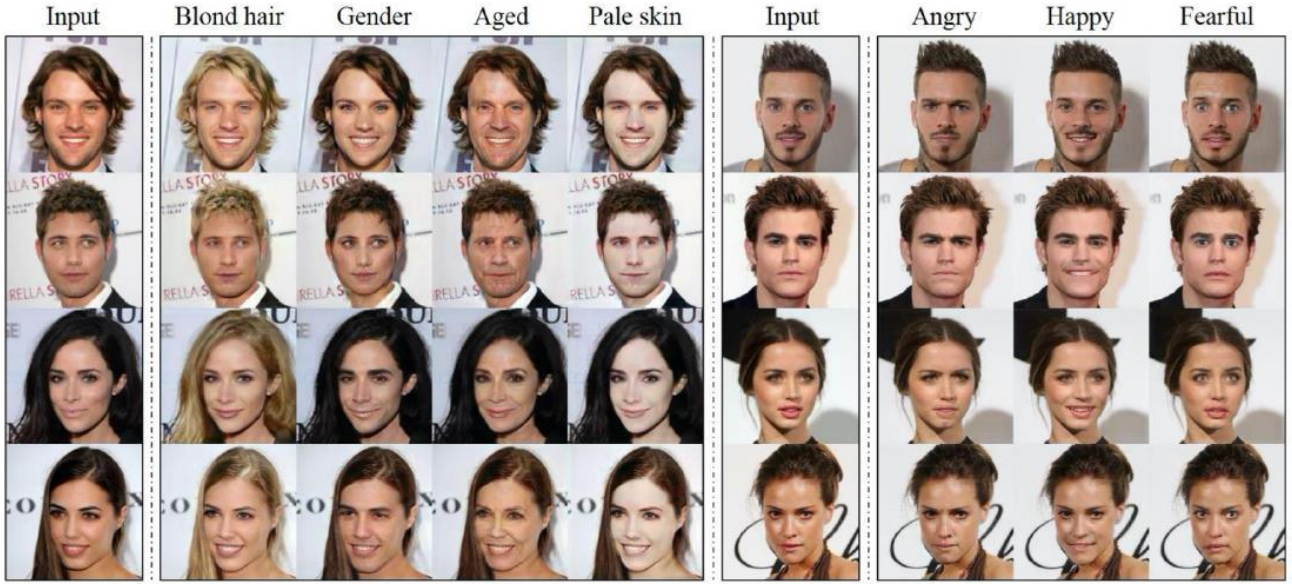
Weathermood: StarGAN

Software Studio DataLab, CS, NTHU

2023

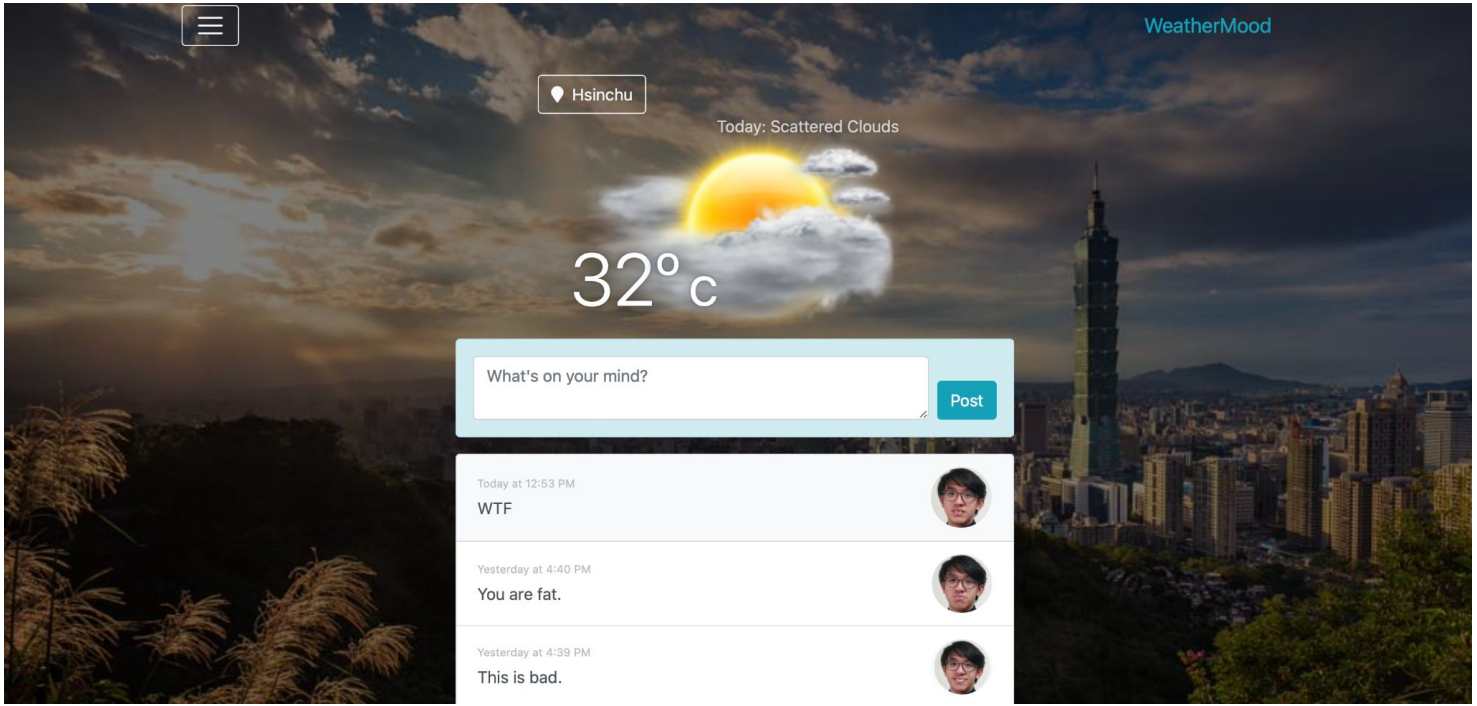
StarGAN

- Image-to-image cross domain translation.
  - Domain: A set of images sharing the same attribute value.



Weathermood-StarGAN

- The face will change based on the toxicity detection result.



Weathermood-StarGAN

- Self-hosted starGANmodel using tensorflow-serving.
  - https://www.tensorflow.org/tfx/serving/serving_basic

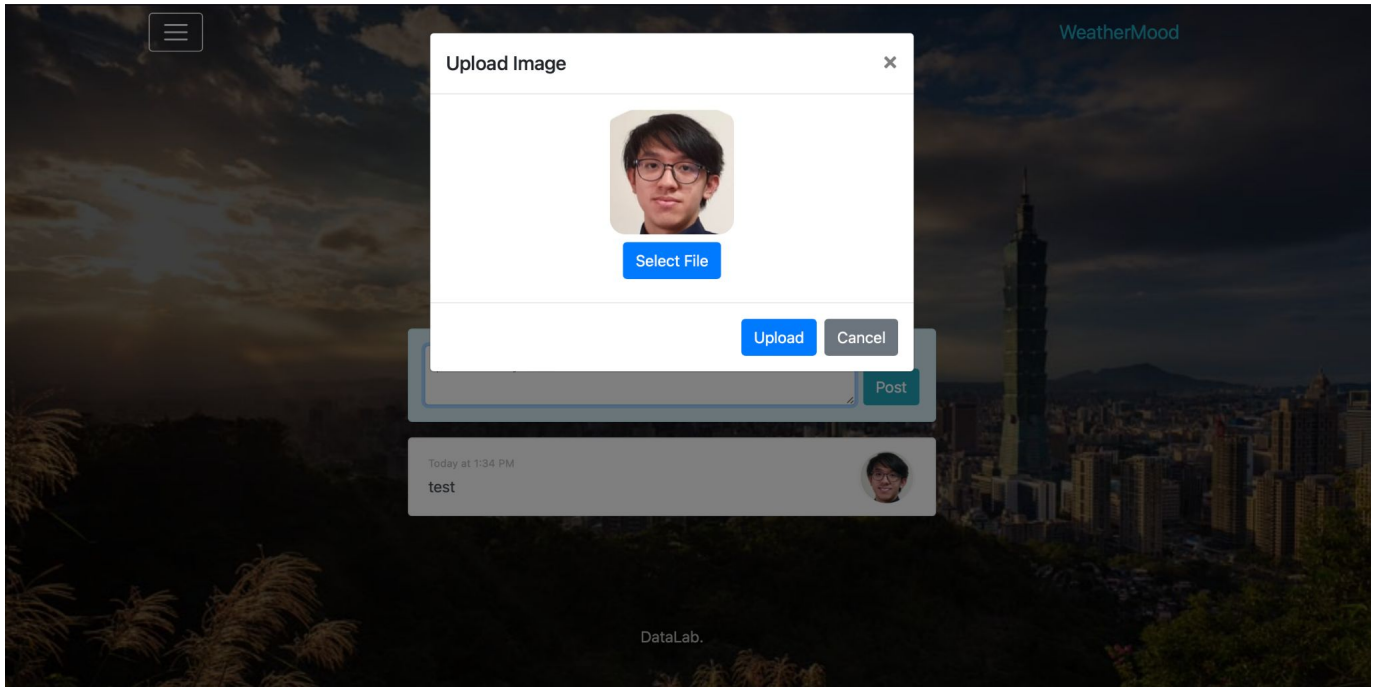
- Fetch the predicted result from a remote server or local model.

  The code is located in FaceSelectionModal.jsx.

```
// When you run python/python3 app.py on your computer, change to the local path.
// const url = "http://127.0.0.1:5000/model/predict/";
const url = "http://140.114.88.21:5000/model/predict/";
console.log(`Making request to: ${url}`);
this.props.setPredicting(true);
this.props.changeFaceFile();
fetch(url, {
    method: 'post',
    headers: {
        'Accept': 'application/json, text/plain, */*',
        'Content-Type': 'application/json'
    },
    body: JSON.stringify(data)
}).then(res=>res.json())
.then((res) => {
```

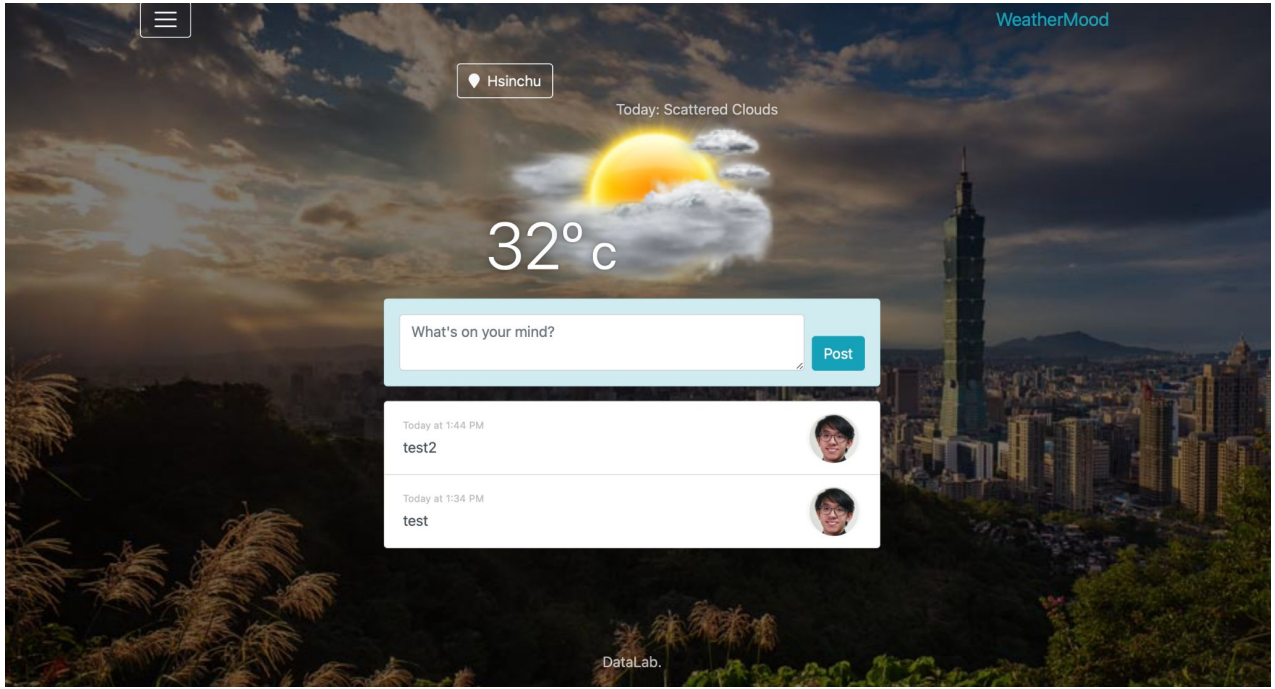
Weathermood-StarGAN

- You can upload your custom face images.



Weathermood-StarGAN

- And you will get the predicted results from the model.



Backend

- You can try to train a model by yourselves or use our pre-trained model.
  - https://github.com/Masao-Taketani/StarGAN-tf2
  - StarGan folder in gitlab.
- To run the model, you need a python + tensorflow environment.
  - Download Anaconda
  - Build an environment
  - Install tensorflow==2.3(w/o gpu) or tensorflow-gpu==2.3 in your environment.

Backend

- Run a small server to handle the request and return the predicted results to the client. Use python/python3 app.py in the StarGan folder.

```python
def getStarGanOutput(payload):

    img = np.float32(payload['inputs']['input_img'][0])
    cond = tf.constant(payload['inputs']['input_cond'])
    result = generate_html_image(gen, img, cond)

    return result * 255


@app.route('/model/predict/', methods=['POST'])
def predict():
    '''
        Note:
            1. The shape of ['inputs']['input_img'] is (1, 128, 128, 3), so please take the first element for the model input.
    '''
    payload = request.json
    outputs = []
    payload['inputs']['input_cond'] = [[0, 1, 0, 0]]
    outputs.append(getStarGanOutput(payload).tolist())

    payload['inputs']['input_cond'] = [[0, 0, 1, 0]]
    outputs.append(getStarGanOutput(payload).tolist())

    payload['inputs']['input_cond'] = [[0, 0, 0, 1]]
    outputs.append(getStarGanOutput(payload).tolist())


    content = {
        "outputs": outputs
    }
```