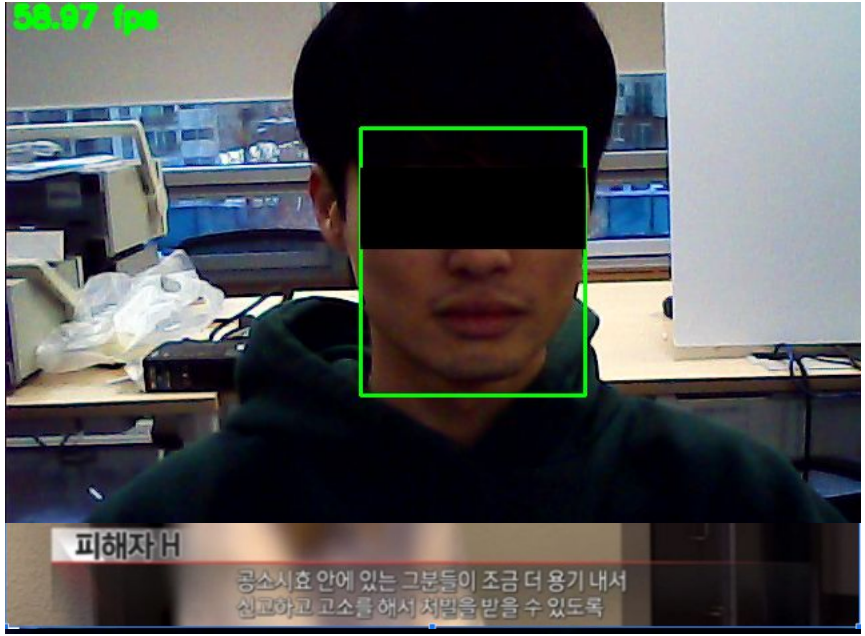
OV 수첩



개발동기

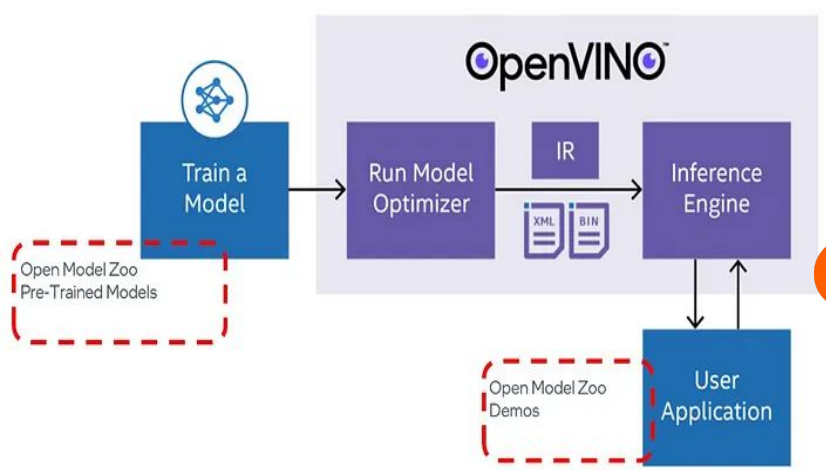
요즘 유행하는 틱톡, 스노우와 같은

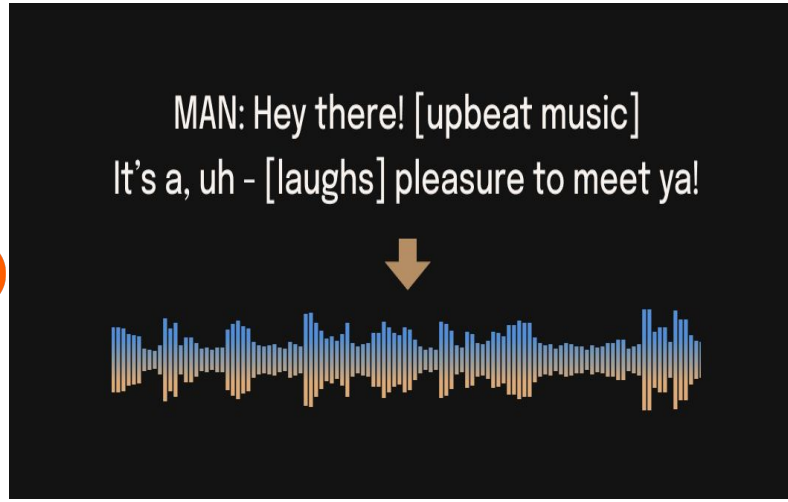
카메라 필터에 영감을 받아

사용자의 재미를 더해주기 위해

그것이 알고싶다 눈 모자이크 인터뷰를

모티브로 하여 만들기로 하였다.

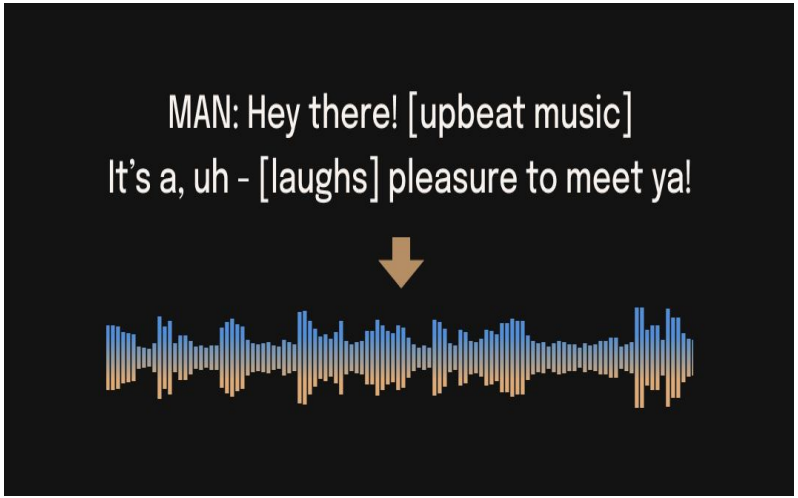




어떤 모델을 썼는지-? **BARK(SUNO AI)와 OpenVINO**

어디서 오픈소스를 가져왔는지? **OpenVINO model zoo**

face-detection-0205



256-bark-text-to-audio

모델 다운로드

```
21
22  # directory where model will be downloaded
23  base_model_dir = Path("./model")
24
25  # model name as named in Open Model Zoo
26  model_name = "face-detection-0205"
27  # model_name = "facial-landmarks-35-adas-0002"
28  # model_name = "person-detection-0202"
29  precision = "FP32"
30  model_path = (
31      f"model/intel/{model_name}/{precision}/{model_name}.xml"
32  )
33  download_command = f"omz_downloader " \
34                     f"--name {model_name} " \
35                     f"--precision {precision} " \
36                     f"--output_dir {base_model_dir} " \
37                     f"--cache_dir {base_model_dir}"
38
39  subprocess.run(download_command, shell=True)
40
```

```
-j N, --jobs N        how many downloads to perform concurrently

[6]: download_command = (
         f"omz_downloader --name {model_name} --output_dir {base_model_dir} --cache_dir {omz_cache_dir}"
     )
     display(Markdown(f"Download command: `{download_command}`"))
     display(Markdown(f"Downloading {model_name}..."))
     ! $download_command

Download command: omz_downloader --name mobilenet-v2-pytorch --output_dir model --cache_dir cache
```

face-detection-0205

Inputs

Image, name: `image`, shape: `1, 3, 416, 416` in the format `B, C, H, W`, where:

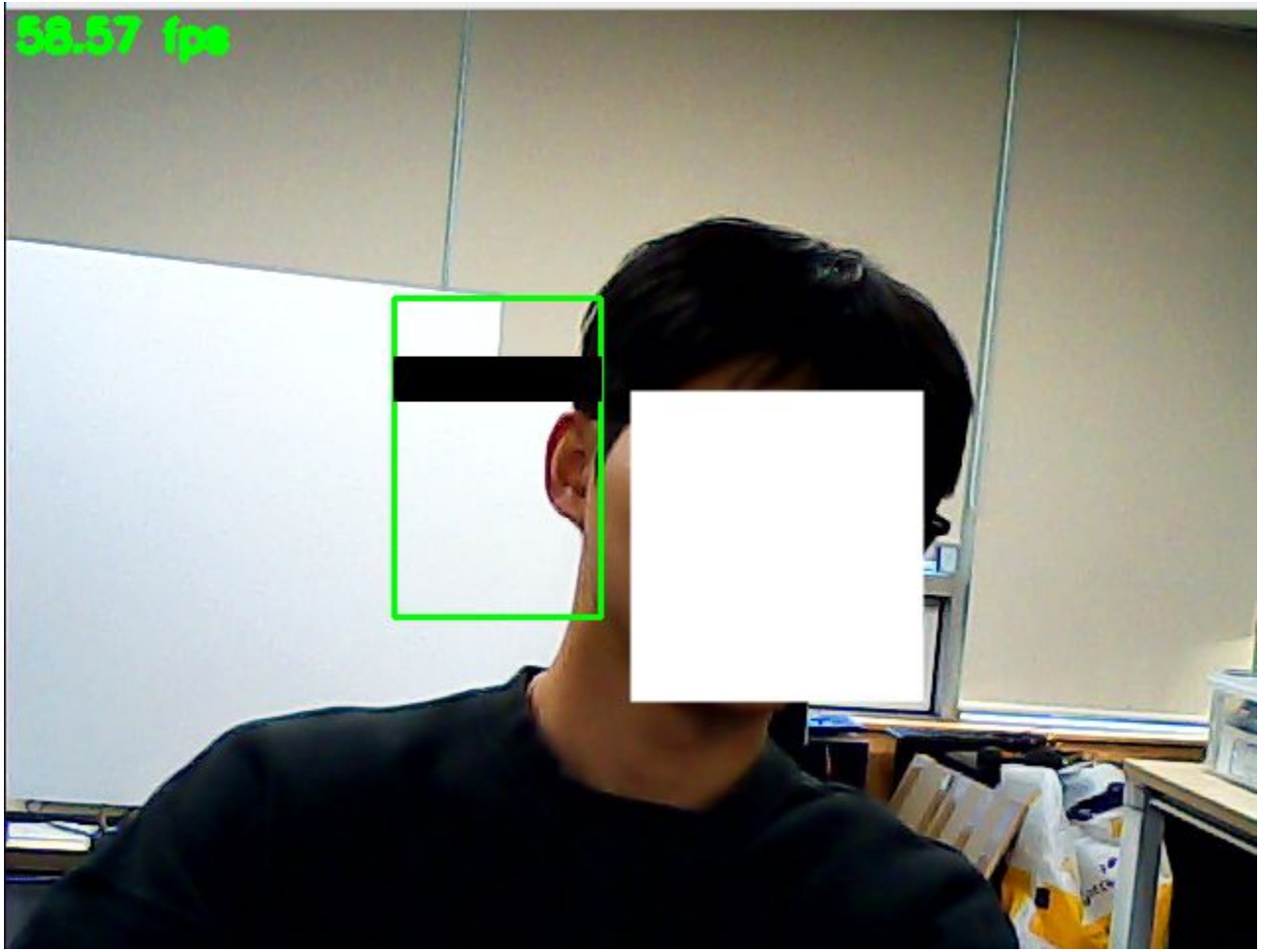
- `B` - batch size
- `C` - number of channels
- `H` - image height
- `W` - image width

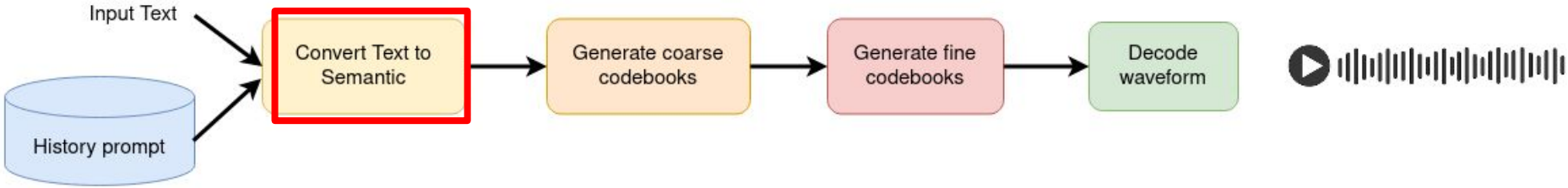
Expected color order: `BGR`.

Outputs

1. The `boxes` is a blob with the shape `200, 5` in the format `N, 5`, where `N` is the number of detected bounding boxes. For each detection, the description has the format [`x_min`, `y_min`, `x_max`, `y_max`, `conf`], where:
    - (`x_min`, `y_min`) - coordinates of the top left bounding box corner
    - (`x_max`, `y_max`) - coordinates of the bottom right bounding box corner
    - `conf` - confidence for the predicted class
2. The `labels` is a blob with the shape `200` in the format `N`, where `N` is the number of detected bounding boxes. It contains predicted class ID (0 - face) per each detected box.

```python
def postprocess(result, image, fps):
    """
    Define the postprocess function for output data

    :param: result: the inference results
            image: the orignal input frame
            fps: average throughput calculated for each frame
    :returns:
            image: the image with bounding box and fps message
    """

    detections = result.reshape(-1, 5)
    #print(detections)

    for i, detection in enumerate(detections):
        xmin, ymin, xmax, ymax, confidence = detection
        if confidence > 0.5:

            '''
            xmin = int(max((xmin * image.shape[1]), 10))
            ymin = int(max((ymin * image.shape[0]), 10))
            xmax = int(min((xmax * image.shape[1]), image.shape[1] - 10))
            ymax = int(min((ymax * image.shape[0]), image.shape[0] - 10))
            '''
            xmin = int(xmin + 100)
            ymin = int(ymin + 50)
            xmax = int(xmax + 150)
            ymax = int(ymax + 50)

            '''
```

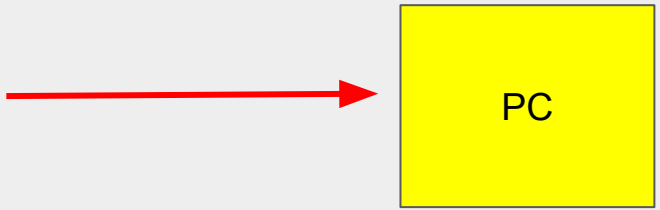

58.57 fps

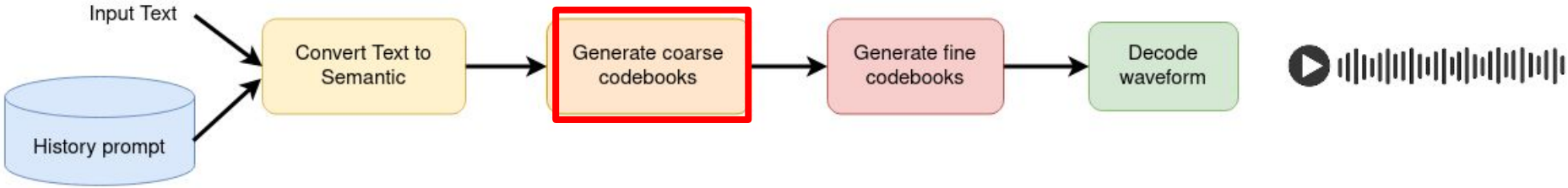


bark_text_encoder

텍스트를 PC가 알기 쉬운 언어로 변환

안녕하세요 김규원 입니다.

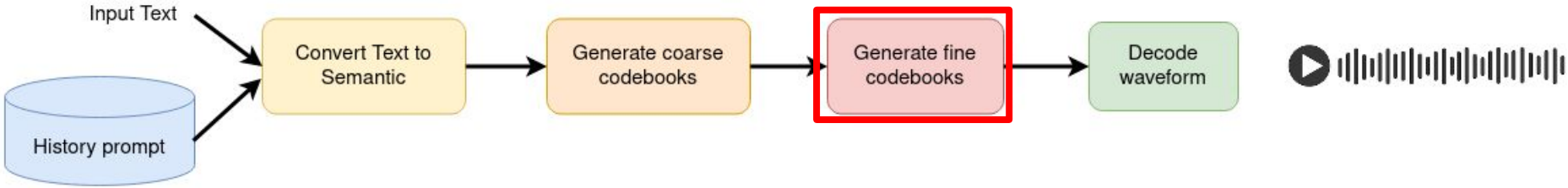




bark_coarse_encoder

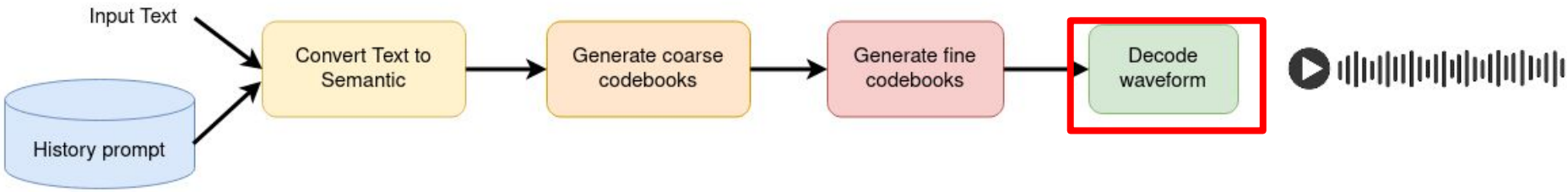
단어들을 그룹으로 나누는 작업

안녕하세요 김규원 입니다.

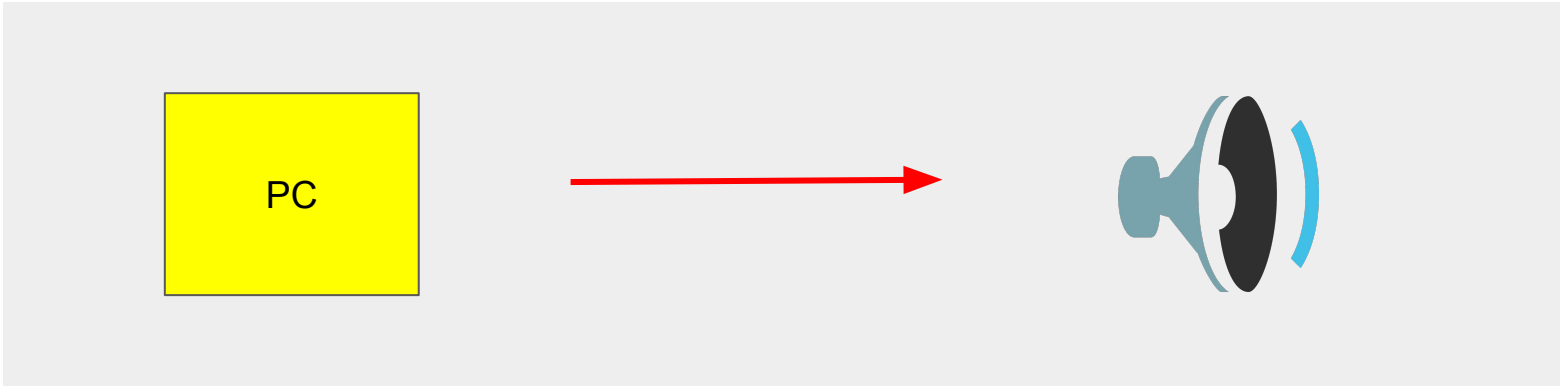


bark_fine_feature_extractor

그룹들을 더 작은 부분들로 쪼개는 작업



sound device(라이브러리)



한계점

CPU만을 사용함으로써 Text를 오디오로 변환하는 과정에서 너무 오래 걸린다

해결방안

1. 코렙을 사용하여 GPU를 이용해 속도를 증가시킨다.
2. wave 파일로 저장을 한 후, 불러오는 과정을 통해 가능하다.

   -> wave 파일이 저장이 되어있을 경우 속도가 향상된다.

동영상 및 스냅샷으로 결과물