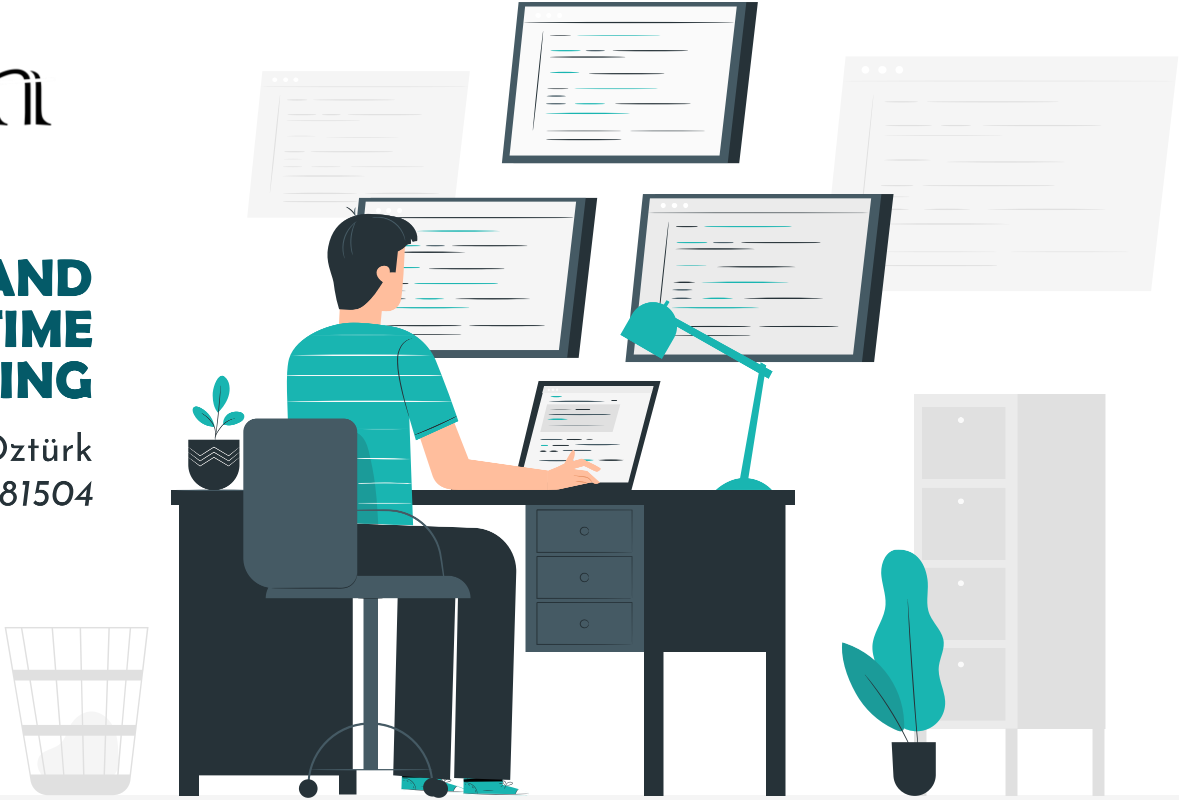
photon

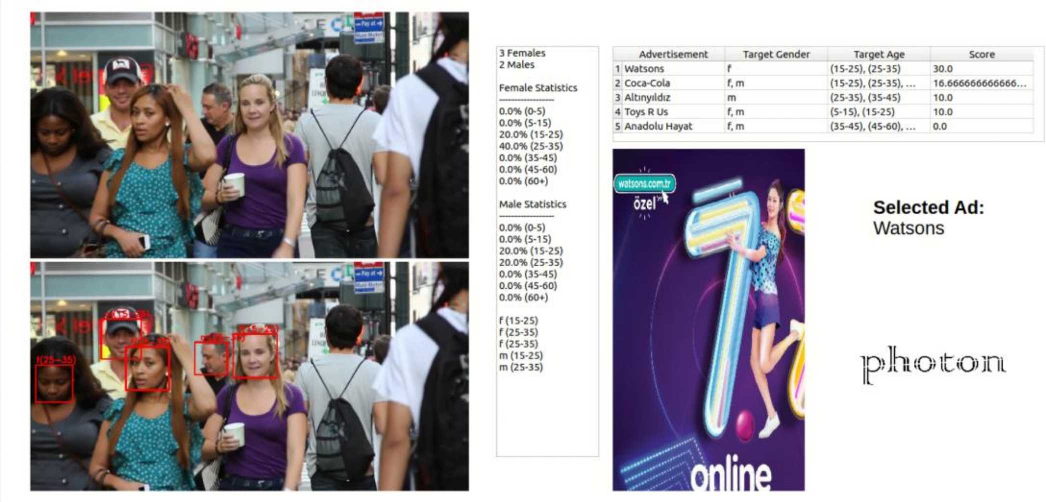
PREDICTING AGE AND GENDER IN REAL-TIME FOR SMART ADVERTISING

Anıl Öztürk
504181504



THE GOAL

The aim of the **Project Photon** is to develop a concept software that provides **the most relevant advertisement** according to the population distribution **in front of the billboard** for **that exact time**.



THE MOTIVATION

1. So much money gets wasted in the advertisement field
2. You rent a lot of billboards and don't even know how much potential customer have seen your ad
3. The companies would select the advertisement method that brings the most profit
4. **Photon targets the audience in a smart way**
5. It feeds back the results to company to give a observation for making better targeting

STEPS OF EXECUTION

1. Get raw video-feed
2. Process and find **faces** in it
3. Process faces and extract **age & gender estimations**
4. Select the **best-fit ad** for the estimated distribution

GETTING THE DATA

I wrote a little script to convert each of these datasets into a format that I can use easily. After converting each dataset, I **merged them into one file** to use them easily in the training phase.

Adience

around 19300 images

IMDB-Faces

around 460000 images

Wiki-Faces

around 63000 images

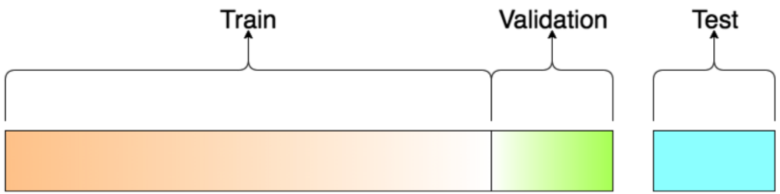
SETTING THE LABELS

I used **genders** and **ages** as labels. While I was **binarizing** gender values, I splitted the ages into **groups**. I decided to set the group count as 7. These age groups are;

- 0-5
- 5-15
- 15-25
- 25-35
- 35-45
- 45-60
- 60+

TRAINING PHASE

I have splitted my data into **training**, **validation**, **test** splits; respectively.



A visualisation of the splits

TRAINING PHASE

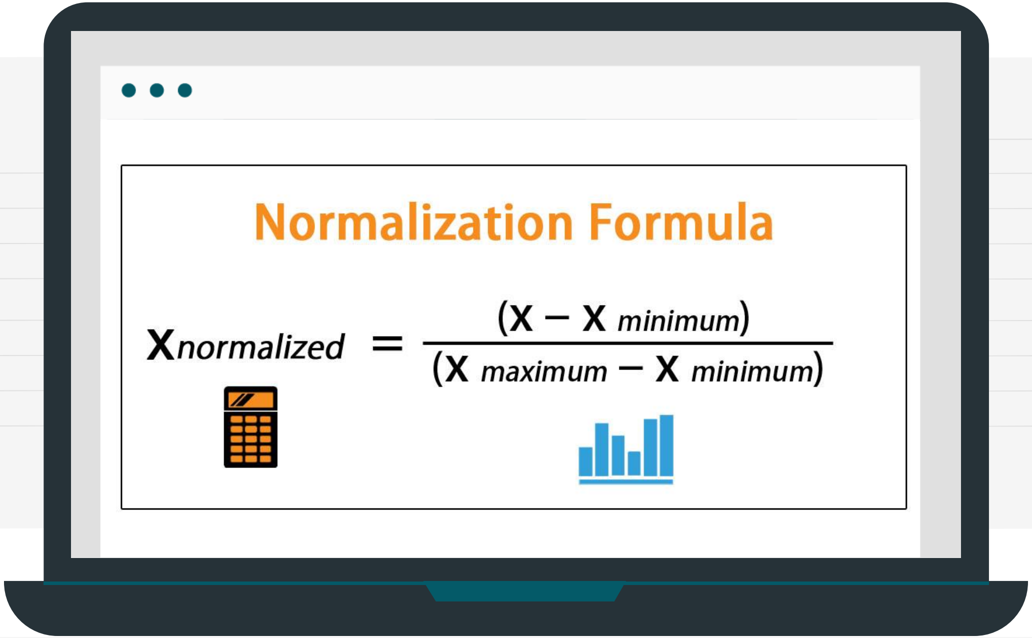
I **one-hot encoded** each type of output (*gender, age*)

Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

TRAINING PHASE

I have applied **normalization** on the images. But in my model, I implemented it as a **lambda layer**.



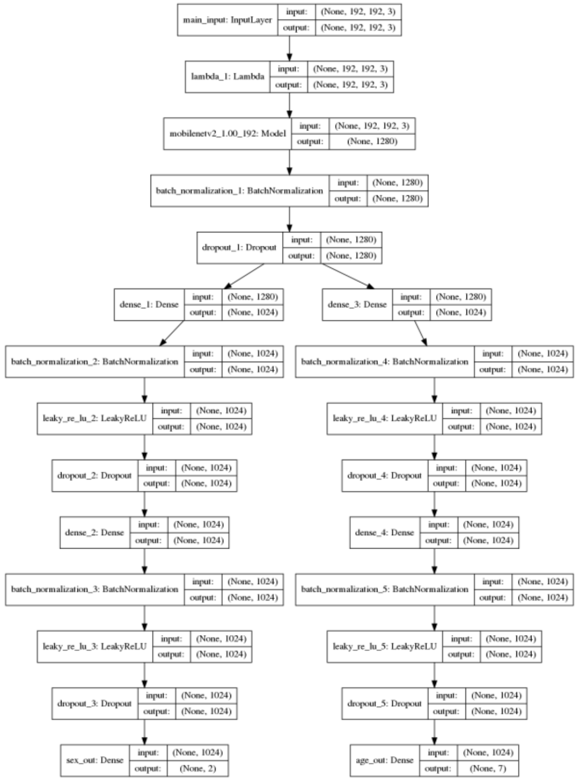
TRAINING PHASE

I have used the **MobileNet** architecture as **backbone** for my model.

Table 1. MobileNet Body Architecture

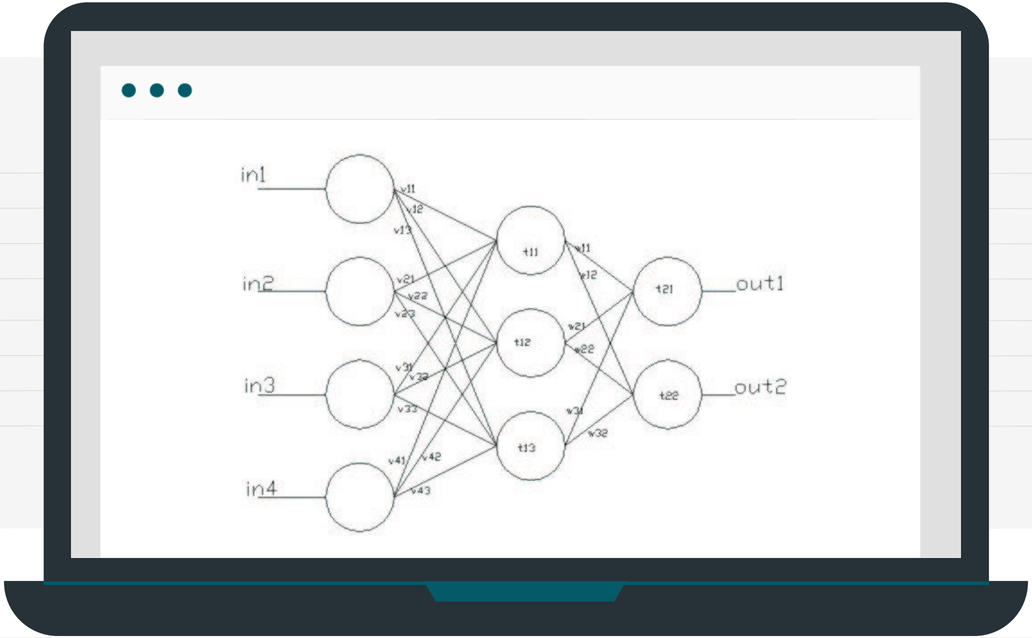
Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

MODEL ARCHITECTURE



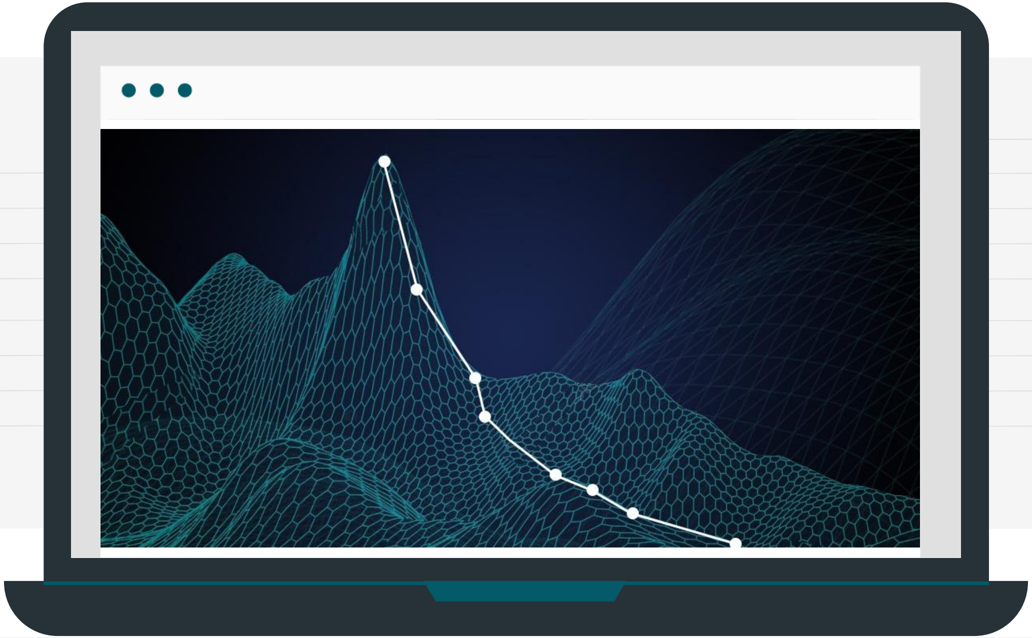
TRAINING PHASE

I have placed several **Fully Connected**, **Batch Normalization** and **Leaky ReLU Activation** layers after the backbone structure.

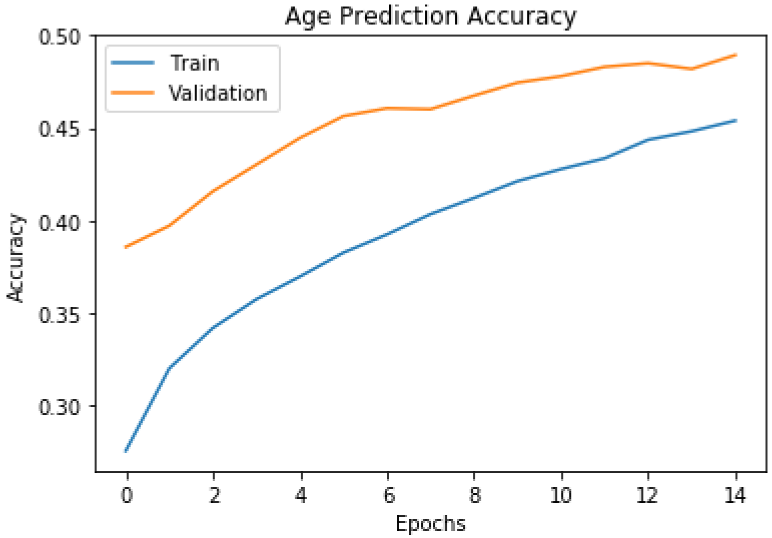


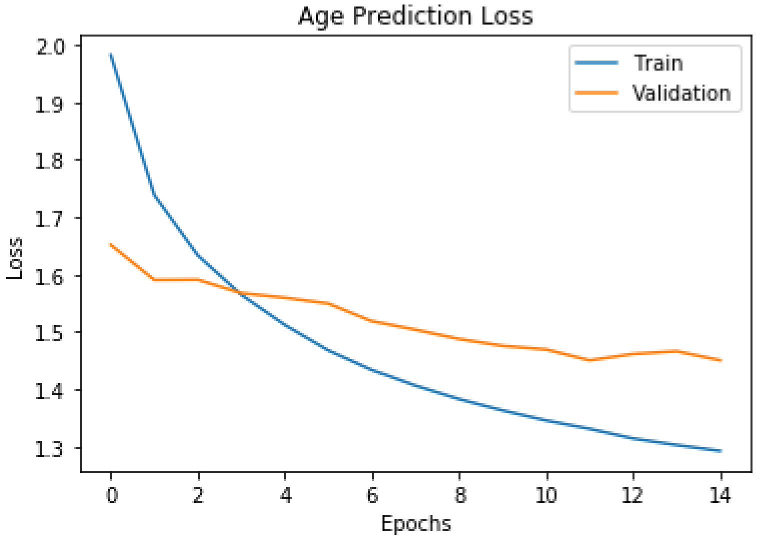
TRAINING PHASE

With **Adam** optimizer and **categorical-crossentropy**, I trained my model over **15** epochs.

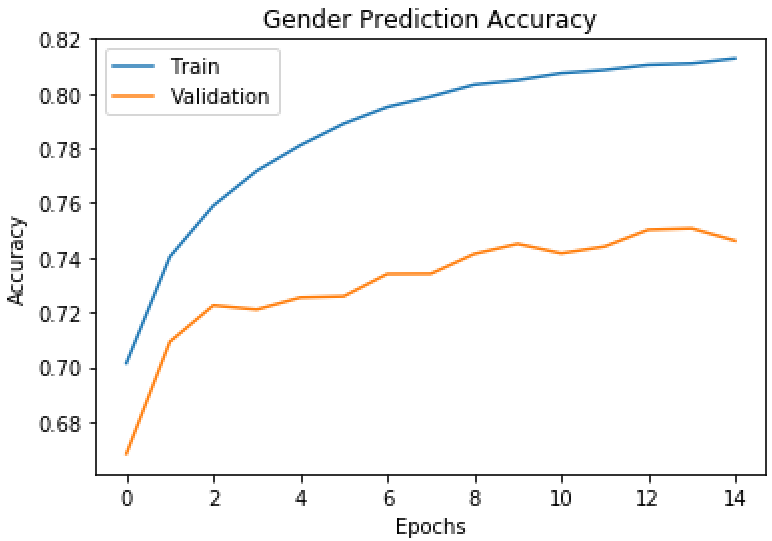


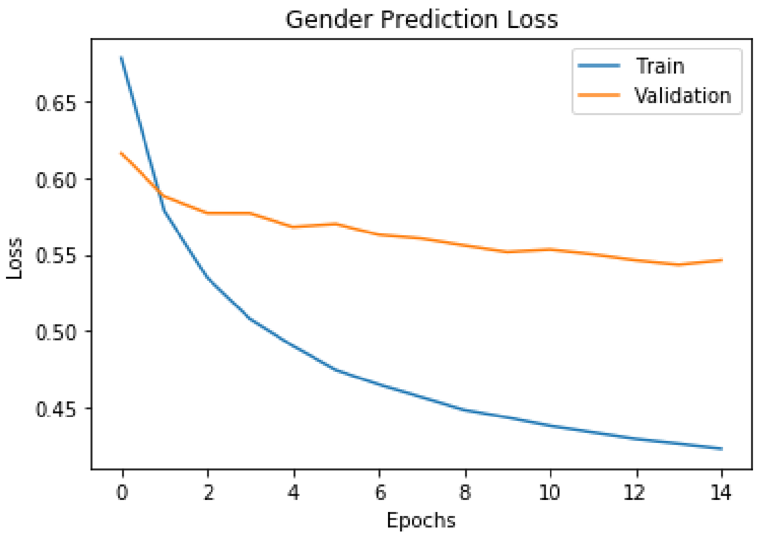
TRAINING RESULTS





TRAINING RESULTS





VALIDATION RESULTS

- **Gender Accuracy:** 74.62% (Over **2** groups)
- **Age Accuracy:** 48.94% (Over **7** groups)

USING READY-TO-USE FACE DETECTOR

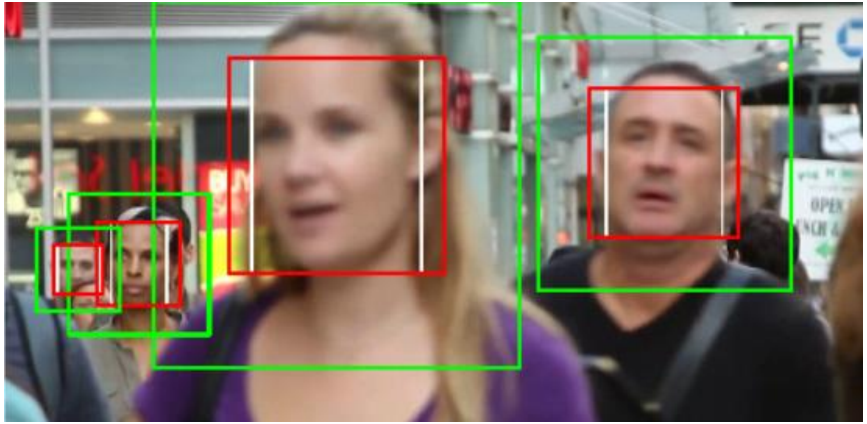
I have used Ultra-Light-Fast-Generic-Face-Detector as the face detector for my project. It's specs are below:

- **Inference Time:** 29ms
- **Model Size:** 1.04MB

INFERENCE PHASE IMAGE TRICK

Training data have been made up by images of faces with a **specific margin**. The detection model train easier when the cropping has been done like this, because they also learn to **adapt orientation changes of faces** and **the noise pattern around the face**.

But my ready-to-use face detector was giving me the bounding-box of the face. So I had to do some basic calculations and get that area to be wider.



SELECTING THE BEST-FIT ADVERTISEMENT

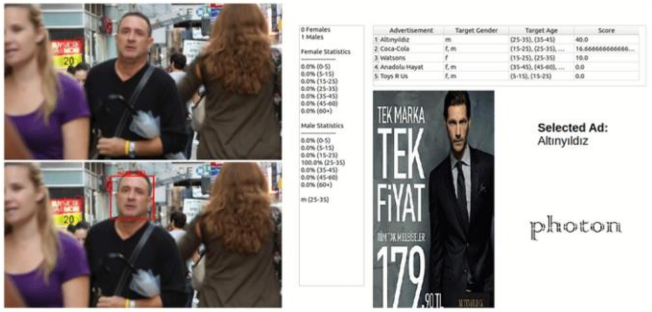
I set up an **advertisement desirability function** to select best-fit one for given crowd. It contains these variables:

- A_1^t: Age ratio of men in the whole crowd
- A_2^t: Age ratio of women in the whole crowd
- S_A: Age index selection of an advertisement
- S_G: Gender index selection of an advertisement
- C: Total display count of the advertisement
- T: Total age-gender combination of target selections of the advertisement
- E: Total advertisement count

THE BEST-FIT ADVERTISEMENT VER-1

The algorithm selects the best-fit ad **at a certain frequency**.

Watsons dominates female-dominant distributions, Altınyıldız dominates male-dominant distributions



THE BEST-FIT ADVERTISEMENT VER-2

The algorithm selects the best-fit ad **at a certain frequency**. Ad desirability **will decrease** as shown by the billboard.

	Advertisement	Target Gender	Target Age	Score
1	Coca-Cola	f, m	(15-25), (25-35), ...	13.89
2	Watsons	f	(15-25), (25-35)	13.108324606575...
3	Altınyıldız	m	(25-35), (35-45)	4.078603216636137
4	Anadolu Hayat	f, m	(35-45), (45-60), ...	2....
5	Toys R Us	f, m	(5-15), (15-25)	0.0

THE BEST-FIT ADVERTISEMENT VER-2

The algorithm selects the best-fit ad **at a certain frequency**. Ad desirability **will decrease** as shown by the billboard.

$$sc(x) = \frac{\sum_{i=s_G} \sum_{j=s_A} A_i^j}{\ln(CE + e)\, T}$$

```python
for adv_it in range(len(self.advertisements)):
    score = 0
    itt = 0
    for gender in self.advertisements[adv_it]["gender_target"]:
        for age in self.advertisements[adv_it]["age_target"]:
            score += self.running_stats[gender*7+age]
            itt += 1

    score /= np.log(np.e + len(self.advertisements)*self.advertisements[adv_it]["showed"])

    self.advertisements[adv_it]["score"] = score/itt
```

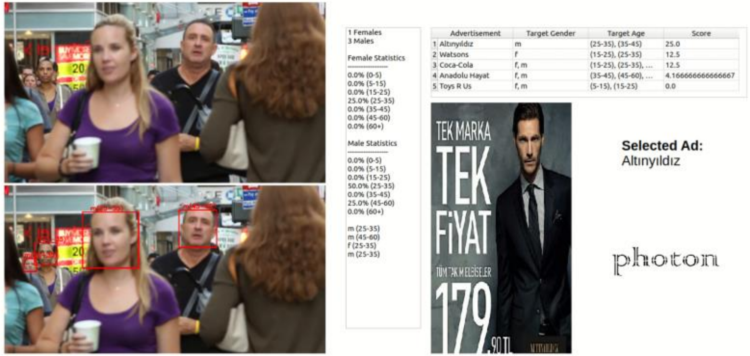
THE CRITIC

- The model runs at realtime on a CUDA based GPU but **it would lack on performance** on compact device like **Raspberry Pi**. It should be compressed or pruned into a smaller network.

- The model basically trained on **128x128** images. It **lacks of performance on small face detections** since they are **scaled up** to 128x128 and **it causes a blurry image**. The model should be trained on a **special small face dataset** and it would **increase** the accuracy of the model.

WRAP-UP

- We created a model to estimate age and gender of given face
- We ran a pre-trained face detector on a frame
- We ran our model on detected faces
- We statistically scored each advertisement w.r.t. our model's output
- We selected the ad with the highest score
- We displayed all the operations on our GUI live.



THANKS

Do you have any questions?