ISTANBUL TECHNICAL UNIVERSITY
ELECTRICAL-ELECTRONICS FACULTY

BLINK DETECTION WITH IN-CAR CAMERA ON NVIDIA JETSON TX2 BY NEURAL NETWORK

SENIOR DESIGN PROJECT

Mehmet Furkan BAĞCI

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT

JUNE 2019

ISTANBUL TECHNICAL UNIVERSITY
ELECTRICAL-ELECTRONICS FACULTY

BLINK DETECTION WITH IN-CAR CAMERA ON NVIDIA JETSON TX2 BY NEURAL NETWORK

SENIOR DESIGN PROJECT

Mehmet Furkan BAĞCI
040140028

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT

Project Advisor: Assoc. Prof. Dr. Sıddıka Berna ÖRS YALÇIN

JUNE 2019

İSTANBUL TEKNİK ÜNİVERSİTESİ
ELEKTRİK-ELEKTRONİK FAKÜLTESİ

SÜRÜCÜNÜN DAVRANIŞLARINDAN DERİN ÖĞRENME İLE ANALİZ JETSON TX2 GÖMÜLÜ SİSTEMİNİN ÜSTÜNDE

LİSANS BİTİRME TASARIM PROJESİ

Mehmet Furkan BAĞCI
(040140028)

Proje Danışmanı: Doç. Dr. Sıddıka Berna ÖRS YALÇIN

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ

HAZİRAN, 2018

We are submitting the Senior Design Project Report entitled as “Blink Detection with in-Car Camera on Nvidia Jetson TX2 by Neural Networks”. The Senior Design Project Report has been prepared as to fulfill the relevant regulations of the Electronics and Communication Engineering Department of Istanbul Technical University. We hereby confirm that we have realized all stages of the Senior Design Project work by ourselves and we have abided by the ethical rules with respect to academic and professional integrity.

Mehmet Furkan BAĞCI ……………
(040140028)

Project Advisor: **Doç. Dr. Sıddıka Berna ÖRS YALÇIN** …………….

FOREWORD

First, I really want to give my endless thanks to my supervisor, Assoc. Prof. Dr. S. Berna ÖRS YALÇIN for giving advice, sparing her precious time and sharing her knowledge with me. In addition, I would like to express my gratitude to my co-workers Çağlar KILCIOĞLU and Okan ULUSOY for their supports and efforts. Finally, I really appreciate to my managers Uğur HALATOĞLU and Serkan ÖNCÜL who believed in me to achieve this project.

JUNE 2019 Mehmet Furkan BAĞCI

TABLE OF CONTENTS

ABBREVIATIONS

AF	: Activation Function
API	**:** Application Programming Interface
AI	**:** Artificial Intelligent
ANN	**:** Artificial Neural Network
BAIR	: Berkeley AI Research
BSD	**:** Berkeley Software Distribution
CPU	: Central Processing Unit
CSI	**:** Camera Serial Interface
ECG	**:** Electrocardiogram
FPS	**:** Frame per Second
GB	**:** Gigabyte
GPU	**:** Graphics Processing Unit
MIPI	**:** Mobile Industry Processor Interface
MP	**:** Mega Pixel
NHTSA	: National Highway Traffic Safety Administration
PC	**:** Performance Computer
RGB	**:** Red Green Blue
SOM	**:** System on module

LIST OF TABLES

LIST OF FIGURES

BLINK DETECTION WITH IN CAR CAMERA ON NVIDIA JETSON TX2 BY NEURAL NETWORKS

SUMMARY

Traffic accidents cause loss of life and property. Most of these accidents caused by human origin. Some of the accidents caused by driver’s sleepiness and distraction. Solving these human based problems will be a very important benefit for humanity. Many automobile companies implement their own solutions in their vehicles. Some of these products determine problem by comparing the data obtained from the driver's behaviour data beforehand, and some of them read the ECG data of the driver with the help of sensor and determine the fatigue.
In this thesis, fatigue and drowsiness situations are observable by real-time processing with the help of an in-vehicle camera placed in front of the driver. The system instantly captures the image of all individuals in the vehicle and detects the driver from these people. The driver’s image process with computer vision systems. Those systems extract specific points from the face in high FPS. The condition of the driver's eyelids have analyzed with the help of ANN technology. The model that used is especially created for this thesis and if required system alerts the driver. The dataset carefully selected and features expanded. Whole system runs on the Nvidia Jetson TX2 development kit. The performance of the kit provides to handle high-resolution images in 50 ms.

ARAÇ İÇİ KAMERA İLE SÜRÜCÜNÜN GÖZ KIRPMASININ TESPİTİNİN NVIDIA JETSON TX2 ÜZERİNDE GERÇEKLEŞTİRİLMESİ

ÖZET

Günümüzde trafik kazaları can ve mal kaybına sebep olmaktadır. Bu kazaların birçoğunun insan kaynaklı olduğu görülmektedir. Sürücülerin sebep olduğu kazaların bir kısmının sebebi uykusuzluk ve dikkat dağınıklığıdır. Bu insan kaynaklı sorunların çözülmesi insanlık yararına çok önemli bir kazanç olacaktır. Birçok otomobil firması kendi çözümlerini araçlarında uygulamktadır. Bu ürünlerin bir kısmı sürücünün daha önce sürüşlerinden elde edilen verilerin anlık olarak karşılaştırılamsı ile tespit etmekte, bir kısmı da sensör yardımı ile sürücünün EKG verilerini okuyarak yorgunluk ve uykusuzluğunu saptamaktadır.
Bu tez içeriğinde diğer sistemlerden farklı olarak araç içi kamera ile anlık olarak sürücünün göz kapaklarının durumunu analiz ederek yorgunluk ve uyuklama durumunu anında tespit ederek ve sürücüyü uyarır. Sistem araç içerisindeki herkesin görüntüsünü yüksek FPS'de alıp sürücüyü tespitedip, görüntüsü üzerinden yüzde bulunan belirli noktaları tespit eder. Göz kapaklarının pozisyonunu yapay sinir ağı teknolojisi ile analiz ederek sonuç vermektedir. Bu noktaların bir bu tez için geliştirilmiş ve eğitimiş bir YSA modeline giriş olarak verilmiştir. Projede Nvidia Jetson TX2 geliştirme Board'u kullanılmıştır. Sistemin sahip olduğu güçlü GPU sayesninde yüksek çözünürlükte görsellerin işlenmesine olanak sağlamıştır. Görüntüler 50 ms gibi kısa bir sürede işlenebilmektedir.

1. INTRODUCTION

A neural network is an interconnected assembly of simple processing elements, units or nodes, whose functionality is loosely based on the animal neuron. The processing ability of the network is stored in the inter unit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns [7]. Several open source models and environments help us to produce new implementations of artificial neural network.

Theory of neural networks discovered several decades ago but due to lack of computing power it could not used widely. Thanks to supreme speed of computers, scientists and engineers are helping the world become a better place now. It has countless implementation areas, such as marketing, defense, agriculture, biomedical, finance, entertainment and telecommunication and so on. One of the implementation of ANN is computer vision intelligent. In the thesis computer vision applications widely used.

According to police reports, the US National Highway Traffic Safety Administration (NHTSA) determined that driver drowsiness directly causes 100,000 vehicle crashes each year. These crashes resulted in approximately 1,550 deaths, 71,000 injuries and $12.5 billion in financial losses [15]. The situation in Turkey is not different form USA, every year many people are losing their lives due to traffic accidents.

We aimed to solve this problem with the help of artificial intelligence technology. Fatigue and sleepiness tried to be determine by the position of the eyelids. Open source Opencv and Dlib libraries used. Due to the specially developed artificial intelligence model, the driver's eyelids situation could be detect. Our artificial intelligence model performed on Nvidia Jetson TX2. Images taken from the developer board’s camera. Thanks to its powerful processing capability, successful and high-speed results are achievable.

In the second part of the thesis, mathematical expressions, hardware description and concepts are explained.The implementation and evaluation process have clearly explained in the third part. In the last section, results and general evaluation take place.

2. FOREKNOWLEDGE AND MATHEMATICAL BACKGROUND

2.1 Nvidia Jetson TX2

Jetson TX2 is a GPU added SOM device as seen on the figure 2.1. NVIDIA Corp. produces it in order to supply a need of high performance device that required on the edge implementations. It integrates: 256 core NVIDIA Pascal GPU, ARMv8 (64-bit) Multi-Processor CPU Complex Denver 2 (Dual-Core) CPU, 128-bit Memory Controller, 8GB LPDDR4 and 32 GBeMMC [Jetson TX2 data sheet]. The developer board includes 5 MP Fixed Focus MIPI CSI camera. The camera mostly used in mobile industry and supports high-resolution inputs. This SOM does not support every libraries we needed in the project so some additional libraries required.

Figure 2.1:Nvidia Jetson TX2 Development Kit

2.2 Artificial Neural Network

Various advances made in creating ANN, some inspired from natural neural systems. Specialists from numerous logical orders are structuring artificial neural systems to take care of an assortment of issues in example acknowledgment, expectation, improvement, cooperative memory, and control [8]. Since ANN started to be widely used, many development libraries developed. We have benefited from some libraries in our project.

2.2.1 Development Libraries

The library is a set of program codes and data that software developers use when developing a program. Software libraries, developers and compilers help in the development of executable programs. Software libraries usually contain pre-made codes, classes, procedures, scripts, configuration data.

2.2.1.1 Pytorch

Pytorch is a Python-based logical research platform focused on a profound learning research stage that gives greatest adaptability and speed [18]. It is also provides model that can be run on Opencv 3.4.6 but our embedded system foes not support Opencv's that version.

2.2.1.2 OpenCV Deep Neural Network

OpenCV (Open Source Computer Vision Library) is an open source computer vision and AI programming library. OpenCV was worked to give a typical framework to PC vision applications and to quicken the utilization of machine learning in the business items. Being a BSD-authorized item, OpenCV makes it simple for organizations to use and adjust the code [13]. OpenCV does have a great supporters and it provides strong structure and reliable functions. In addition, it develops itself continuosly. The part of OpenCV used in the project is its DNN (Deep Neural Network) Class. DNN class has functions that can read models in different formats else than OpenCV Original format.

2.2.1.3 Caffe

Caffe is a profound learning system made with indication, speed, and particularity as a primary concern. It is created by Berkeley AI Research (BAIR) and by network supporters. Yangqing Jia made the undertaking amid his PhD at UC Berkeley. Caffe is discharged under the BSD 2-Clause license [1]. In the thesis, Caffe used because it is a comperable old framework and previous versions of OpenCV DNN has support to the Caffe models.

2.2.2 Mathematical Explanations

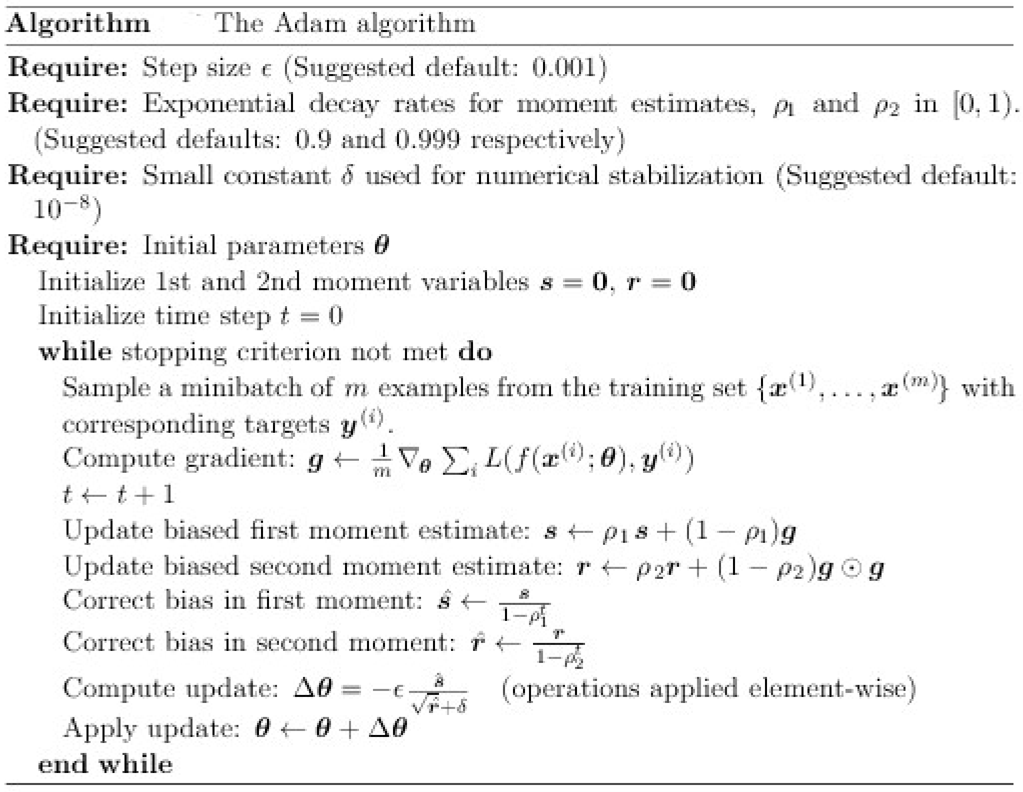
The functions and algorithms have been used in the project have serious mathematical background. However, there is no need to know all these information for using them. On the other hand, lack of knowledge can cause serious errors. For that, reason every possible method should worked on to achieve a good project.

2.2.2.1 Optimization Algorhtim

Optimization is one of the most essential ingredient in the recipe of AI algorithms. It begins with characterizing misfortune loss and cost capacity and closures with limiting the utilizing either improvement schedule. The decision of streamlining calculation can have any kind of effect between getting a succesful precision in serious amount of time. The utilization of enhancement is boundless and broadly examined theme in industry and the scholarly world.

2.2.2.1.1 Adam Optimization Algorithm

Adam is an adaptive learning rate advancement calculation and is exhibited in calculation (Figure 2.2). The name "Adam" gets from the expression "adaptive moments." with regards to the prior calculations, it is maybe best observed as a variation on the mix of RMSProp and energy with a couple of significant qualifications. In the first place, in Adam, force is joined legitimately as a gauge of the first-request snapshot of the angle. The most direct approach to add force to RMSProp is to apply force to the rescaled angles. The utilization of force in blend with rescaling does not have a reasonable hypothetical inspiration. Second, Adam incorporates inclination revisions to the evaluations of both the first-request minutes and the second-request minutes to represent their introduction at the cause (see figure 1). RMSProp likewise fuses a gauge of the second-request minute; nevertheless, it does not have the adjustment factor. Accordingly, not at all like in Adam, the RMSProp second-request minute gauge may have high inclination right off the bat in preparing. Adam is largely viewed as being genuinely vigorous to the decision of hyperparameters, however the learning rate here and there should be changed from the proposed default [6].



```
Algorithm   The Adam algorithm
Require: Step size $\epsilon$ (Suggested default: 0.001)
Require: Exponential decay rates for moment estimates, $\rho_1$ and $\rho_2$ in $[0, 1)$.
  (Suggested defaults: 0.9 and 0.999 respectively)
Require: Small constant $\delta$ used for numerical stabilization (Suggested default:
  $10^{-8}$)
Require: Initial parameters $\boldsymbol{\theta}$
  Initialize 1st and 2nd moment variables $\boldsymbol{s} = \boldsymbol{0}$, $\boldsymbol{r} = \boldsymbol{0}$
  Initialize time step $t = 0$
  while stopping criterion not met do
    Sample a minibatch of $m$ examples from the training set $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ with
    corresponding targets $\boldsymbol{y}^{(i)}$.
    Compute gradient: $\boldsymbol{g} \leftarrow \frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}), \boldsymbol{y}^{(i)})$
    $t \leftarrow t + 1$
    Update biased first moment estimate: $\boldsymbol{s} \leftarrow \rho_1 \boldsymbol{s} + (1 - \rho_1)\boldsymbol{g}$
    Update biased second moment estimate: $\boldsymbol{r} \leftarrow \rho_2 \boldsymbol{r} + (1 - \rho_2)\boldsymbol{g} \odot \boldsymbol{g}$
    Correct bias in first moment: $\hat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t}$
    Correct bias in second moment: $\hat{\boldsymbol{r}} \leftarrow \frac{\boldsymbol{r}}{1-\rho_2^t}$
    Compute update: $\Delta\boldsymbol{\theta} = -\epsilon \frac{\hat{\boldsymbol{s}}}{\sqrt{\hat{\boldsymbol{r}}}+\delta}$   (operations applied element-wise)
    Apply update: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$
  end while
```

Figure 2.2: The Adam Algorithm

2.2.2.2 Activation Function

Activation functions are mathematical functions utilized in neural systems to registers the weighted total of info and predispositions, of which is utilized to choose if a neuron can be fire or not. It controls the exhibited information through some slope preparing for the most part angle plunge and a short time later produce a yield for the neural system, that contains the parameters in the information [20]. The AF has great impact on the success of the neural network some good functions are not usable for digital base system due to their complex calculation processes.

2.2.2.2.1 Rectified Linear Unit (ReLU) Function

The rectified linear unit (ReLU) activation function was submited by Nair and Hinton 2010 [16]. ReLu is a widely used activation function due to its fast calculatiblty on algorithem base implementations. It discribed at (2.1).

$$\text{ReLU}(x) = \max(0, x) \tag{2.1}$$

It can be seen that it is turning negative input values to the zero and not changing the positive inputs. It is illustrated as below.

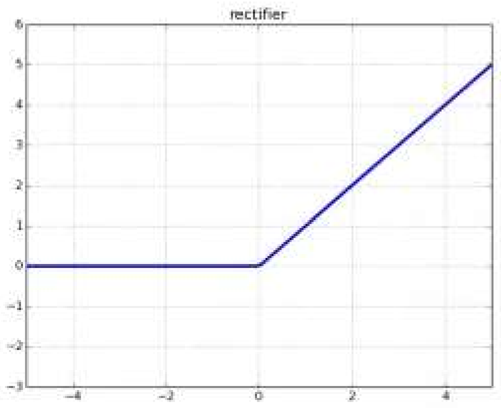


Figure 2.3 : ReLu Activation Function

2.2.2.3 Loss Function

Loss function is a significant part in ANN systems, which is utilized to gauge the irregularity between predicted value and actual label.

2.2.2.3.1 Cross Entropy Loss Function

The mathematical formula of cross entropy can be seen on figure 2.4.

$$\mathcal{L} = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

Figure 1.4:Cross Entropy Loss

- L: Loss

- M : number of classes
- y : binary indicator
- p : predicted probability observation

Cross entropy estimates the disparity between two likelihood appropriation, if the cross entropy is huge, which implies that the contrast between two conveyance is huge, while if the cross entropy is little, which implies that two dissemination is like one another [11].

2.3 Computer Vision

Computer vision is a field of software engineering that chips away at empowering computers to see, recognize and process pictures similarly that human vision does, and afterward give fitting yield. It resembles conferring human knowledge and senses to a machine. In actuality however, it is a troublesome undertaking to empower PCs to perceive pictures of various articles. Computer vision is firmly connected with produced consciousness, as the devices must decipher what it sees, and after that perform proper examination or act in like manner [17].

2.3.1 Dlib

Dlib is a C++ toolbox containing AI calculations and devices for making complex programming in C++ to tackle genuine issues. It is utilized in both industry and the scholarly community in a wide scope of spaces including mechanical autonomy, installed gadgets, cell phones, and huge superior registering situations. Dlib's open source permitting enables you to utilize it in any application, complimentary [2]. Dlib provides permission to commercial products. For that, reason the parts build with Dlib can be use in the mercantile projects.

2.3.1.1 Face Detection and Land Mark

Face detecting is the first step in the process, without a successful algorithm application could not reach success. In the Project Dlib Maximum-Margin Object

Detector (MMOD)[3] has been used. It is based on convolutional neural network (CNN) features. CNN is type of artificial neural network that used widely on the image-based problems with supervised learning. Structure of CNN does look like figure 2.5. First input passed through filters and that results made small by pooling after repeating the same procedure layers shift to the fully connected layers, at the end the out predictions take place for results. The MMOD is published in 2015 and it is successful on various of face positions. This method does not perform aby sub-sampling, but instead optimizes over all sub-windows. It also can be used with HoG (Histogram of Gradient) method to improve models.

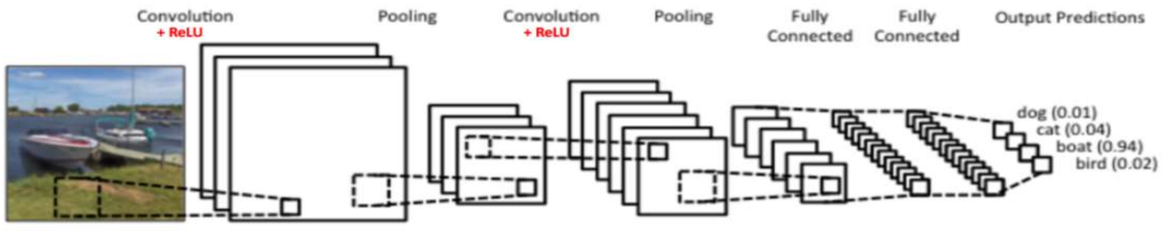


Figure 2.5: Convolutional ANN

After we detected the face 68 points on the face extracted by Dlib landmark detection function. These points are on the face such as the corners of the mouth, along the eyebrows, on the eyes, and so forth. This functions uses the classic Histogram of Oriented Gradients (HOG) feature combined with a linear classifier, an image pyramid and sliding window detection scheme. The pose estimator was created by using dlib's implementation of Kazemi's paper[10].

The reason both of the algorithm have been used is to increasing the speed of the system and ocuring more knowledge about the driver. The MMOD face detecter can obtain face even though they are not frontal but frontal face detector could not detect faces that are not frontal and it does give clue about the drivers' face position. If MMOD detects but Frontal cannot, that system assumes pace is not directed to the road. In addition frontal face detector is comperatively slower than MMOD face detector so The face cropped by MMOD given to the frontal face detector and this method provides speed to the system.

2.4 Blinks Dataset

This is a dataset for eye blink recognition made by W. J. Faithfull and meticulously hand-marked in 2015. It comprises of HD webcam film of the essences of 6 volunteers viewing a nature narrative. There are 5 minutes of film for each volunteer,

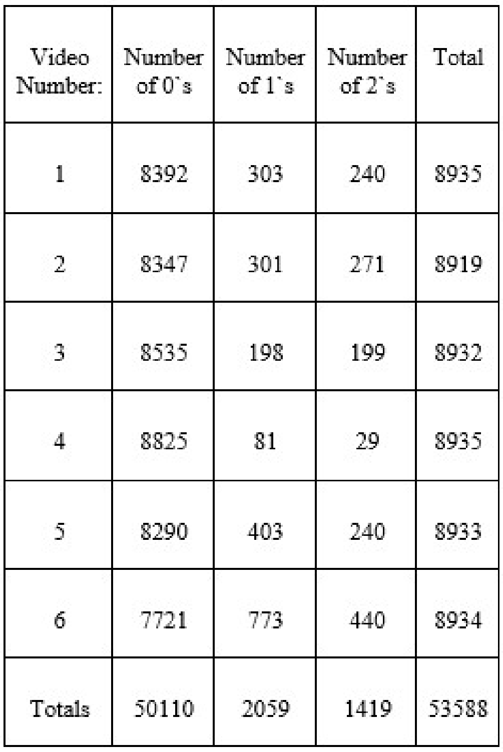
the recordings show an extensive spread of flickering rates, squint lengths, and different difficulties [5] .The states are:

0 The volunteer's eyes are open.

1 It is not evident whether the volunteer's eyes are open or shut.

2 The volunteer's eyes are shuted.

The recordings are in MPEG4 format. Videos recorded from direct view to the subjects. Total number and atenuations showed in the table 2.1.



Video Number:	Number of 0`s	Number of 1`s	Number of 2`s	Total
1	8392	303	240	8935
2	8347	301	271	8919
3	8535	198	199	8932
4	8825	81	29	8935
5	8290	403	240	8933
6	7721	773	440	8934
Totals	50110	2059	1419	53588

Table 2.1: Dataset Values Distribution

3. IMPLEMENTATION AND DEVELOPMENTS

3.1 Algorithm Structure

The projects pseudo code given in the figure 3.1. First values assigns out of the while loop because if the values assign in side of the loops they would allocate memory and it causes to memory storage it kills the program. The camera receives frame in

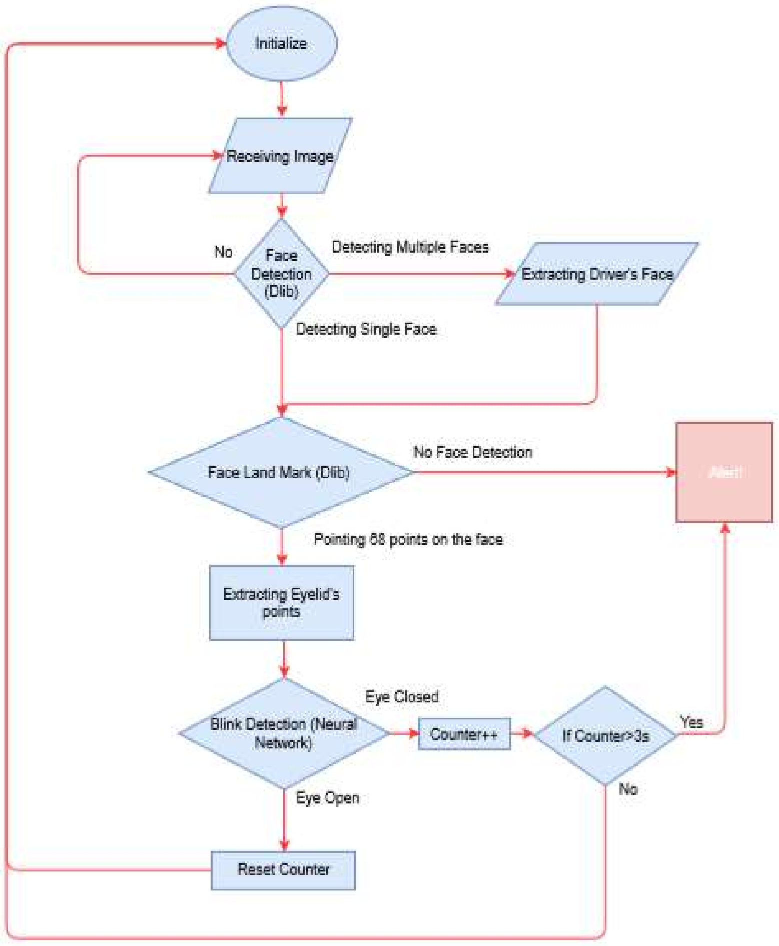


Figure 3.1 : Structure of The Code

the while loop. Then looks to image for face detection if it finds more than one, systems assign the face, which has biggest area on the screen, as the driver's face. The face-land mark function extracts the eyelids points and system normalize the these points to form that network can receive .If the function cannot find a face as in put it gives alert because it means driver is not looking to the road. If the results says the eyes are open loops starts from beginning, if the eyes are closed and it was been for a while system gives an alert to warn driver.

3.2 Installations of Libraries

3.2.1 Host PC

The host PC has i7 Intel processor and Nvidia Gtx 640. As operating system Ubuntu 16.04 prefered. Some of the installations completed for flashing jetson TX2 somes were for training the neural network.

Firstly, the Nvidia's jetpack 3.3 files downloaded from official website and instructions have completed [9]. The benefit of using jetpack is it does install cuDNN v7.1.5, Multimedia API v28.2, OpenCV 3.3.1 by itself.

Secondly, the Pytorch neural network training environment installed by Anaconda environment. Using Anaconda environment provides benefits for example if some libraries' different versions are needed them might break each other and changing versions most of the time not working properly. In addition, anaconda provides easier installations to its environments. Some libraries which need source installation or could not be installed by pip can be installed only a few commands. First updating the operatin system than installing curl to download Anaconda installing file.

```
$ sudo apt-get update
$ sudo apt-get install curl
```

Then downloading .sh file of installation.

```
$ curl -O https://repo.continuum.io/archive/Anaconda3-4.3.1-Linux-x86_64.sh
```

To verify the data integrity of the installer, we use a cryptographic hash algorithm called SHA-2 (Secure Hash Algorithm).

```
$ sha256sum Anaconda3-4.3.1-Linux-x86_64.sh
```

Than running .sh file is required.

```
$ bash Anaconda3-4.3.1-Linux-x86_64.sh
```

In order to activate installations bashrc file should source by typing following command.

```
$ source ~/.bashrc
```

After installation an environment should be created. Python3 based environment named myenv. With this command.

```
$ conda create -n myenv python=3 anaconda
```
 To activate environment:

```
$ conda activate myenv
```
To deactivate environment:

```
$ conda deactivate
```
Now environmnet is ready and needed libraries can be installed on it.

Pytorch:

```
$ conda install -c pytorch pytorch
```

Numpy:

```
$ conda install -c anaconda numpy
```
Matplotlib:

```
$ conda install -c conda-forge matplotlib
$ conda install -c conda-forge/label/testing matplotlib
$ conda install -c conda-forge/label/testing/gcc7 matplotlib
$ conda install -c conda-forge/label/gcc7 matplotlib
$ conda install -c conda-forge/label/broken matplotlib
$ conda install -c conda-forge/label/rc matplotlib
$ conda install -c conda-forge/label/cf201901 matplotlib
```

3.2.2 Jetson TX2

The Nvidia Jetson TX2 has processor has arm64 processor for that reason normal installations sometimes not properly working so they required source installation. Previously some OS libraries should be installed to the Jetson TX2:

```
$   sudo apt-get install build-essential cmake pkg-config
$   sudo apt-get install libx11-dev libatlas-base-dev
$   sudo apt-get install libgtk-3-dev libboost-python-dev
```

Opencv 3.4.0 installation**:**

First of all the Opencv3.3.1 has to be uninstall because it does notworking the development's boards camera properly. The couple of scripts should run on the device:
Removing old opencv files installed by Jetpack
```
$ sudo apt-get purge libopencv*
```

Removing other unused apt package
```
$ sudo apt autoremove
```

The script published at

https://github.com/AastaNV/JEP/blob/master/script/install_opencv3.4.0_TX2.sh

Used and successfully work on the device.

Dlib 19.17 installation:The latest version of Dlib library is 19.17 and can be installed from official website. However it does not required because it can be done by terminal.

First the ziped form should be download
```
$wget http://dlib.net/files/dlib-19.6.tar.bz2
```
Than it should be unzipped
```
$tar xvf dlib-19.17.tar.bz2
```
For installing some commands should be run:
```
$ cd dlib-19.6/
$ mkdir build
$ cd build
$ cmake ..
$ cmake --build . --config Release
$ sudo make install
$ sudo ldconfig
```

We can check the installation by calling the import dlib on the python

As:

```
$ Python
$ >> import dlib
```

If this commands does not give an error, the installation completed successfully.

Caffe:

Caffe has installed because Opencv 3.4.0 does not have support to the pytorch models but Caffe models. The trained models converted to the Caffe model from Pytorch model for that Caffe librarie required to install to the system. Firstly the source of caffe download to the home file

```
$ git clone https://github.com/BVLC/caffe.git
```

However continuing the installation is not possible without some changes on the files of Makefile.config.

First (Removed lines showed after – sign and new added lines showed with + sign):

```
-  LIBRARIES += glog gflags protobuf boost_system boost_filesystem m hdf5_hl
hdf5
+ LIBRARIES += glog gflags protobuf boost_system boost_filesystem m
hdf5_serial_hl hdf5_serial
-  INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include
+ INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include
/usr/include/hdf5/serial/
```

Than The Setup completed successfully.

3.3 Artificial Neural Network Development

The eye's situation determinate by the trained neural network. The network structure created and the input values normalized.

3.3.1 Creating and Training ANN Model with Pytorch

The training is the fundamental of the ANN models. First a structure should be created. In out neural network the layers' nodes fully connected, Relu activation function chosen and bias set. Several structures been tested and the final structure shaped as 15x105x105x100x100x100x100x10x2 in linear form shown in figure 3.2. While choosing final structure first its ability to over fitting was tested with a small data set.

```
class Net(nn.Module):

    def __init__(self):
        super(Net, self).__init__()

        self.fc1 = nn.Linear(15, 105)
        self.fc2 = nn.Linear(105, 105)
        self.fc3 = nn.Linear(105, 100)
        self.fc4 = nn.Linear(100, 100)
        self.fc5 = nn.Linear(100, 100)
        self.fc6 = nn.Linear(100, 100)
        self.fc7 = nn.Linear(100, 10)
        self.fc8 = nn.Linear(10, 2)
    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = F.relu(self.fc3(x))
        x = F.relu(self.fc4(x))
        x = F.relu(self.fc5(x))
        x = F.relu(self.fc6(x))
        x = F.relu(self.fc7(x))
        x = self.fc8(x)
        return x
```

Figure 3.2 : Network Structure

The Adam optimizer used to train model, the learning rate and momentum set to 0.0001 and Beta set to 0.9, 0.999 from default value.

```
learning_rate=0.0001
optimizer = optim.Adam(net.parameters(), lr = learning_rate)
```

Figure 3.3 : Optimization Function

The dataset split to 3 part: 80% given to the training-set, %10 given to the validation-set and 10% given to the test-set.

Cross entropy loss chosen as loss function.

```
criterion = nn.CrossEntropyLoss()
```

Figure 3.4 : Loss Function

Training batch size set to the 100. It provided more stable loss graph.

```
batch_size_train=100
trainloader = torch.utils.data.DataLoader(transformed_trainset,
batch_size=batch_size_train,shuffle=True, num_workers=2)
```

Figure 3.5 : Train Loader

The training started for 30000 epoch but if loss of validation-set does not less for 10 epoch system break the process.

```
number_of_epoch=30000
for epoch in range(number_of_epoch):
    running_loss=0.0
    for i, data in enumerate(trainloader, 0):
        result, dists = data
        #result ve dists seperated
        optimizer.zero_grad()
        #Before Training gradiants set to the 0 because it may
spoiled with previous results
        outputs = net(dists)
        #The outputs came from network assign to the variant
        loss = criterion(outputs, torch.max(result, 1)[1])
        #loss assign to variant
        loss.backward()
        #To take gradiant it taken to backward
        optimizer.step()
        #all weigths are reshaped with gradiant
        running_loss += loss.item()
        #Running loss calculated for displaying to terminal
        loss_val = criterion(outputs_valid, torch.max(result_val,
1)[1])
               towrite_loss_val = np.append(towrite_loss_val,
[running_loss_val / 10], axis=0)
       if((np.mean(yazilacak_loss_val[-10:-
1])>np.mean(yazilacak_loss_val[-20:-11])))
         break
```

Figure 3.6 : Training Script

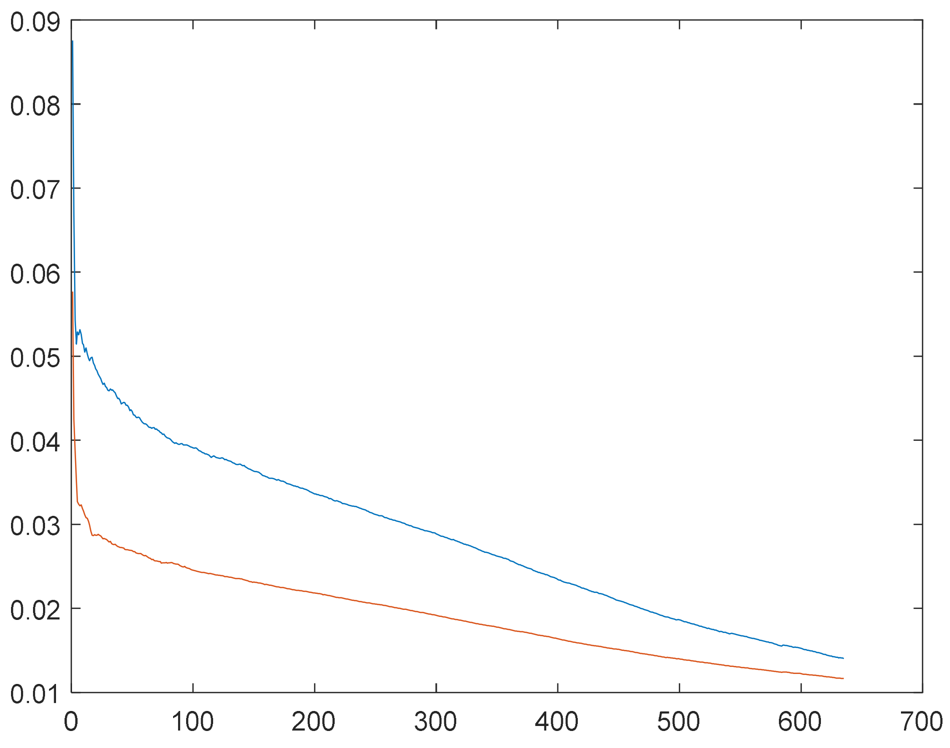


Figure 3.7 : Validation and Train Loss Plot

The Pytorch development environment have used and training done in Python programing language. The model achieved 90% at validation and %88 at test.

3.3.2 Transforming Pytorch Model to Caffe Model

The model trained in the Pytorch (Python) saved in its own format. The inference script writen in C++ and due to Jetson TX2 has ARMv8 processor and it does not support Pytorch C++(lib Torch). For that reason, the model transformed to the caffe model that has support by Opencv 3.4.0 by script available in appendex. The script taken from a GitHub account [19].

3.3.3 Inferencing ANN Model on Jetson TX2

The code have written in C++ because it is faster compare to python.The model embedded to the net:

```
netw = cv::dnn::readNetFromCaffe("model_val.prototxt", "model_val.caffemodel");
```

The input given to the model, C is the input array 15 valuable long:

```
cv::Mat C(1, 15, CV_32FC1, input);
```

```
netw.setInput(C);
```

The output variable type should be Mat. The output assigned to the cikti:

```
cv::Mat cikti=netw.forward();
```

The output normalized to 1-0 or 0-1:

```
minMaxLoc(cikti.reshape(1, 1), 0, &confidence, 0, &classIdPoint);
```

3.4 Computer Vision Implementations

3.4.1 Face Detection

Dlib's mmod face detection function used. It excracts faces and receives the biggest face as driver than gives the face to landmark excracter.

First, the mmod_human_face_detector model embedde to a network:

```
//Argv[1]-> mmod_human_face_detector.data

net_type net;

deserialize(argv[1]) >> net;
```

The receiving the outputs of face detector.

```
auto dets = net(img);

 for (auto&& d : dets

{ if (en_buyuk_yuz.rect.area()<d.rect.area())

en_buyuk_yuz=d; }
```

3.4.2 Land Mark extraction

The biggest face landmarks extracted with following command:

```
std::vector<full_object_detection> shapes;
shape_predictor pose_model;
```

The shape_predictor_68_face_landmarks.dat file is pretrained landmark model:

```
deserialize("shape_predictor_68_face_landmarks.dat") >> pose_model;
```

The pose_model gives 68 point of the face:

```
theShape = pose_model(croimg, faces[i]);
shapes.push_back(theShape);
```

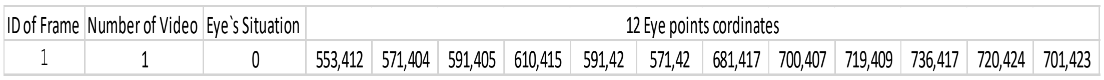
The points between 36 to 41 are eyelid’s points:

```
for (unsigned long i = 36; i <= 41; ++i)
{
draw_solid_circle (croimg,theShape.part(i),1.5,color);
dist[(i-36)*2]=theShape.part(i).x();
dist[1+(i-36)*2]=theShape.part(i).y();
}
```

After that the dist array will be given the to our blink detector model.

3.5 Data-set Improvement

The one of the most important step in the ANN training is the choosing correct training data. The Blinks dataset was barely satisfying project`s requirements. There was not enough information to train our network model so some extra information added to the dataset. First, specific ids have given to the each frame. Video numbers and 12 points that locations of the eyelids added to the attenuation list as showen on the figure 3.8. The 12 point extracted from the Dlib face landmark detection function, which previously explained in the thesis. The points labelled as half opened which labelled as 1 removed from the dataset to obtain sharper results. The final form of the data set mixed with all together.



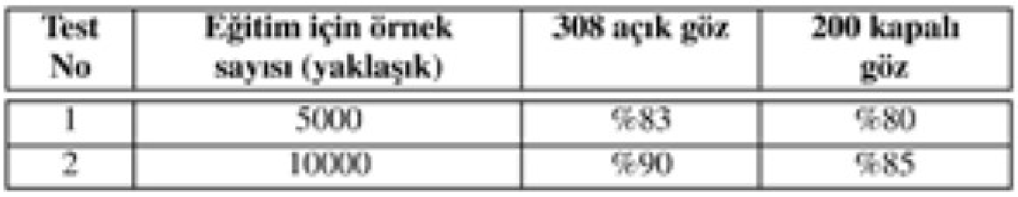
ID of Frame	Number of Video	Eye`s Situation	12 Eye points cordinates											
1	1	0	553,412	571,404	591,405	610,415	591,42	571,42	681,417	700,407	719,409	736,417	720,424	701,423

Figure 3.8 : Dataset Format

4. CONSTRAINTS AND CONCLUSIONS

4.1 Comparison

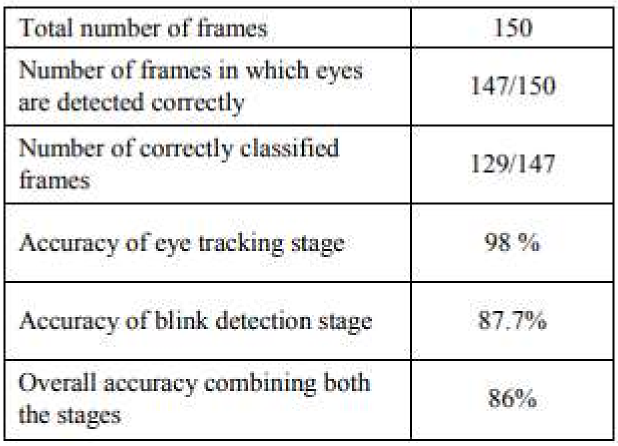
The blink detection have been a issue to solve for a long time and several methods produced.One of the solutions have created by ITU Computer Science department. They aim to warn computer users who forgets to blink in front of the computer. Their solution method is very similar to us: They are first detecting eye's location and than analyzing the situation [12]. Mr. Nusraddinov journals's results are:



Test No	Eğitim için örnek sayısı (yaklaşık)	308 açık göz	200 kapalı göz
1	5000	%83	%80
2	10000	%90	%85

Table 4.1 : Test Results of Nusraddinov' s System

Mr. Pauly also wrote a journal about blink detection. The histogram of oriented gradients (HOG) is used for blink detection in their paper [14]. The results are given in the table 4.2.



Total number of frames	150
Number of frames in which eyes are detected correctly	147/150
Number of correctly classified frames	129/147
Accuracy of eye tracking stage	98 %
Accuracy of blink detection stage	87.7%
Overall accuracy combining both the stages	86%

Table 4.2 : Test Results of Pauly's System

Results of thesis are given at table 4.3:

The situation of Eye	True	False	Over-all
Open	135	15	%90
Close	129	21	%86

Table 4.3 : Test Results

The given results are out networks test results but succes of face detection and landmark detection should be calculated to decide solid results. 3 ANN model have used and their mistakes should sum and it may causes more error than using one ANN model.

4.2 Realistic Constraints

4.2.1 Social, Environmental and Economic Impact

The projects' success would be very help full to the society. The number of the accidents that related with the fatigue would decrease. Thanks to that the less people would lost their lives.

4.2.2 Cost analysis

Our system has to work on the cars and they would not have internet acces all the time for that reason projects requires a strong edge device. The Nvidia Jetson TX2 is a powerful module that satisfy projects's needs. On the other hand, the development kit is unnecessary big for cars so module needs to entegrated to a carrier. The final cost would be similar to the table 4.4.

Equipment Name	Cost
NVIDIA® Jetson™ TX2 Module	519.00 USD
Orbitty Carrier for NVIDIA® Jetson™ TX2/TX2i	174.00 USD
Total:	693.00 USD

Table 4.4: Cost of Products

4.3 Future Work and Recommendations

Analyzing a face is a complicated task but it gives a lot of feature about the driver. For example, the system can detect the position of the driver's head and can give warning for other distractions such as if driver looking to a telephone. In addition, the car renter companies can grade their costumers driving skills and attention. They can gifted if costumer gets high scores with a discount for next renting process. Every day millions of people driving cars but not every car owner would want to have this sort of innovations but there would be companies who can be use this project for their needs.

REFERENCES

[1] **Caffe**. (n.d.). Retrieved May 18, 2019, from https://caffe.berkeleyvision.org/
[2] Dlib C Library. (n.d.). Retrieved May 17, 2019, from http://dlib.net/
[3] E., D. (2015, January 31). Max-Margin Object Detection. Retrieved May 17, 2019, from https://arxiv.org/abs/1502.00046
[4] Eye blink detection with OpenCV, Python, and dlib. (2017, April 13). Retrieved from https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/
[5] **Faithfull** W. J.,(2015). Blink Detection Dataset https://will.faithfull.me/blinks-dataset/ Bangor University, School of Computer Science
[6] **Goodfellow**, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Cambridge (EE. UU.): MIT Press. pp.305
[7]**Gurney**, K. (2014). *An Introduction to Neural Networks*. Hoboken: CRC Press.
[8] **Jain**, A. K., Jianchang Mao, & Mohiuddin, K. M. (1996). Artificial neural networks: a tutorial. Computer, 29(3), 31–44. doi:10.1109/2.485891
[9] JetPack 3.3 Release Notes. (2018, September 07). Retrieved May 18, 2019, from https://developer.nvidia.com/embedded/jetpack-3_3
[10] **Kazemi**, V., & Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. doi:10.1109/cvpr.2014.241
[11] Loss Functions. May 18, 2019 Retrieved from https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html
[12] **Nusraddinov**, T., & Ekenel, H. K. (2015). Eye blink based warning system for eye health while using computers. *2015 Medical Technologies National Conference (TIPTEKNO)*. doi:10.1109/tiptekno.2015.7374588
[13] **OpenCV.** (n.d.). Retrieved from https://opencv.org/about/ [14]**Pauly**, L., & Sankar, D. (2015). A novel method for eye tracking and blink detection in video frames. *2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)*. doi:10.1109/cgvis.2015.7449931
[15]**Rau** P. Drowsy Driver Detection and Warning System for Commercial Vehicle Drivers: Field Operational Test Design, Analysis, and Progress. National Highway Traffic Safety Administration; Washington, DC, USA: 2005.
[16] **V. Nair** and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Haifa, 2010, pp. 807–814. [Online]. Available: https://dl.acm.org/citation.cfm
[17] What is Computer Vision? - Definition from Techopedia. (n.d.). Retrieved from https://www.techopedia.com/definition/32309/computer-vision
[18] What is PyTorch. (n.d.). Retrieved from https://pytorch.org/tutorials/beginner/blitz/tensor_tutorial.html
[19] **Xxradon**. (2019, March 12). Xxradon/PytorchToCaffe. Retrieved from https://github.com/xxradon/PytorchToCaffe [20] *XXV International Mineral Processing Congress: IMPC 2010, "Smarter processing for the future": Brisbane, Australia 6-10 September 2010: Congress proceedings*. (2010). Carlton, Vic.: Australasian Institute of Mining and Metallurgy.

APPENDICES

APPENDIX A: CD

RESUME

Name Surname	**:** Mehmet Furkan BAĞCI
Place and Date of Birth	**:** Antalya, 1995
High School	**:** Mira Mesa High School 2010-2011
	Ankara Atatürk Lisesi 2009-2010, 2011-2014
Batchler Degre	**:** Electronics and Communication Engineering program, Istanbul Technical University
E-Mail	**:** mehmetfurkanbagci@gmail.com