**Projeto de Classificação de Fadiga em Tempo Real usando Redes Neurais**

Bruno Porto e Rebeca Moreno

29/05/2020

**Resumo**

Este relatório descreve o desenvolvimento do hardware e do software de um Sensor de Fadiga em tempo real para motoristas de veículos, principalmente de grande porte. Primeiramente, é preciso obter e analisar um dataset. Após isso, com ferramentas de programação em Python, é preciso criar e treinar um modelo de Machine Learning que usa Redes Neurais para classificar e prever a fadiga do condutor ao longo do tempo. Usando o modelo treinado, pode-se aplicar o software a uma topologia de rede e, também, definir os equipamentos que constituem o hardware, de acordo com as especificações e requisitos de projeto.

1. **Introdução**

A sonolência do motorista é uma das principais causas de acidentes de automóvel. Segundo estudo do Departamento Intersindical de Estatística e Estudos Socioeconômicos (Dieese), aproximadamente 15% a 20% das colisões acontecem porque o condutor dormiu ao volante. [1]

Quando o caso é com caminhoneiros, a situação se agrava ainda mais, devido às características e ao porte desse tipo de veículo. A Confederação Nacional do Transporte (CNT) afirma que, a cada ano, são mais de 50 mil acidentes com vítimas, muitos deles com o envolvimento de caminhões. [2]

Esses acidentes podem ocorrer por diversas causas, mas a fadiga é a principal delas. Conforme levantamento feito pela Unilever junto a 100 transportadoras de carga, a fadiga provoca 76% dos acidentes com caminhões.Isso se dá por conta da alta carga horária de trabalho desses condutores. Apenas 6% deles conseguem trabalhar 8 horas por dia, segundo pesquisa da CNT. 51% são motoristas autônomos, que dirigem, em média, entre 13 a 19 horas diariamente. Outros 10% ficam mais de 20 horas no volante, enquanto 29% trabalham entre 9 e 12 horas todos os dias. O estudo ainda aponta que o risco de acidentes em vias aumenta a partir da nona hora de trabalho consecutiva. [3]

Além das condições rodoviárias terem essas preocupações, na área da mineração, as condições dos trechos são um pouco diferentes, pois há pouca iluminação e longos trechos perigosos (podendo chegar ao tamanho de cidades como Araxá) com vales e montanhas.

O trabalho consiste, basicamente, no transporte de carga de um local a outro dentro da mineração, basculando e carregando o veículo. A Fig. 1 mostra uma rotina de caminhoneiros nessa área.



Figura 1 – Mina Casa da Pedra, da CSN Mineração (fonte: FONTE)

Sendo assim, o cansaço, o sono e o desgaste físico ocasionam à perda de reflexo e, consequentemente, ao comprometimento da segurança do motorista, principalmente quando há excesso de velocidade. [2]

Diante dessas condições, investir em tecnologias de monitoramento automático em tempo real, como o sensor de fadiga, ajuda a evitar ocorrências de acidentes, aumentando a segurança do condutor e o desempenho do mesmo no trabalho. [2]

Como mostrado em pesquisas da SETCESP (Sindocato das Empresas de Transportes de Carga de São Paulo e Região), um sensor de fadiga tem como objetivo detectar o comportamento de um motorista, monitorando sua dirigibilidade, por meio da leitura de sinais, e registrando dados no sistema. [4]

Além dessas vantagens, também é interessante ressaltar que, em muitas minerações, há pouca ferramenta de monitoramento em tempo real. Atualmente, poucas empresas investem nesse tipo de tecnologia e, mesmo as grandes empresas que têm essa ferramenta, são monitoradas e acompanhadas por funcionários na sala de controle da mineração ou em empresas terceirizadas. Portanto, com a aplicação deste sensor de fadiga nos caminhões de uma grande mineiração, não há mais a necessidade de ficar, instantaneamente, atento quanto ao desempenho dos processos do trabalho ou quanto a segurança e, consequentemente, a produtividade dos caminhoneiros.

Este sensor funcionará por meio da leitura de expressões faciais. Sendo calibrado para cada pessoa, afinal, cada uma apresenta um comportamento diferente quanto ao olhos e boca quando está alerta.

Mais precisamente, o objetivo é desenvolver um projeto capaz de classificar e monitorar, em tempo real, o cansaço de um motorista. Ou seja, modelar tanto a parte de hardware do sensor de fadiga, quanto a parte do software, a qual irá não somente processar, mas também alertar o operador e a sala de controle ou transportadora quando necessário.

Para alcançar este *objetivo*, é necessário, primeiramente, obter um dataset de vídeos que será usado para treinar o modelo de Machine Learning. Com o dataset, pode-se usar ferramentas de programação em Python para detectar rostos, olhos e bocas presentes nos vídeos e, então, extrair informações numéricas que descrevem o comportamento das piscadas, por exemplo. Mas, como esta aquisição dos dados depende da qualidade do dataset e da biblioteca do Python, é preciso filtrar e limpar as anomalias e falhas da captação feita. Assim, depois do pré-processamento dos dados, pode-se criar o modelo de Machine Learning com Redes Neurais para classificar e prever a fadiga do operador.

O restante deste documento está organizado da seguinte forma: Seção 2 é dedicada à definição do Software. O Hardware e a topologia de rede que este sensor estará submetido serão discutidos na Seção 3. A Seção 4 fornecerá conclusões finais e sugestões para trabalho futuro.

1. **Software**

O software aqui descrito visa o desenvolvimento de um modelo de machine learning capaz de determinar o estado de fadiga de condutores de auto-veículos somente a partir da análise de imagens de vídeo.

* 1. **Dataset**

O Dataset que será utilizado para treinamento do modelo é *Open Source* e foi desenvolvido pela Universidade do Texas. [5]  Ele é constituído vídeos feitos por voluntários, sendo que cada um deles realizou 3 filmes de aproximadamente 10 minutos cada: um deles em estado alerta (0), outro em estado intermediário (5) e por fim um vídeo cansado (10). No total estão disponíveis 180 vídeos de 60 voluntários diferentes.



Figura 2 - Imagens exemplo dos vídeos disponíveis no dataset utilizado para realização do modelo. Na fileira superior: exemplos de vídeos classificados como alerta, nível de fadiga 0; na fileira do meio, vídeos classificados como intermediários, nível de fadiga 5; e na fileira de baixo exemplos de vídeos classificados como alto nível de fadiga, intensidade 10. (fonte: <https://sites.google.com/view/utarldd/home>)

* 1. **Planejamento do Modelo de Machine Learning**

O modelo de Machine Learning aqui realizado busca classificar o nível de fadiga de condutores apenas por meio da análise de imagens, portanto, as características (*features)* utilizadas no modelo devem ser apenas retiradas a partir de aspectos físicos físicos da pessoa (olhos, boca, nariz, expressões). Por uma limitação do dataset e com o objetivo de simplificar o produto final, não será incluído na análise aspectos fisiológicos do condutor (frequência cardíaca, pressão sanguínea e etc.) nem características de condução (perfil de condução, carga no acelerador e etc.).

Alguns estudos[6] indicam que o comportamento dos olhos e da boca são os principais indicadores de fadiga em pessoas. Sendo assim, as características que serão analisadas no modelo são:

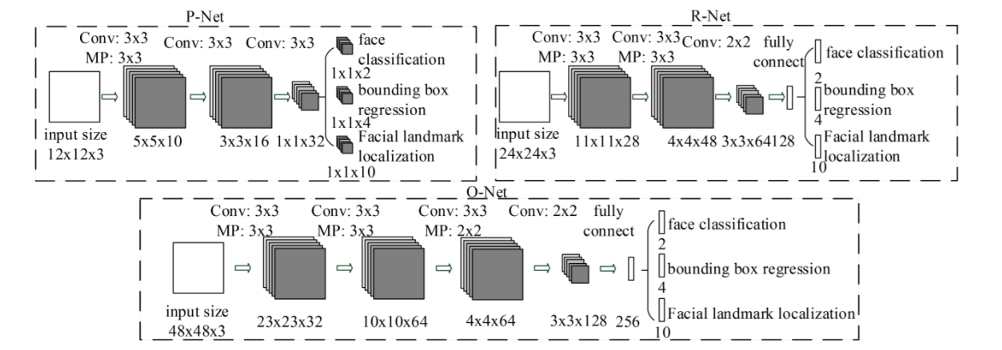
* **EAR (Eye Aspect Ratio) percentual**[7]: EAR é uma estimativa do nível de abertura dos olhos. Aqui será utilizada uma medida relativa percentual com base no valor base calculada para cada pessoa. Este valor base é retirado no início dos vídeos com classificação 0 (alerta) e estendido para os demais. Em anexo mais informações de como este índice é calculado.
* **Perclos:** Uma medida da porcentagem de tempo que o olho permanece fechado. Mais informações em anexo.
* **Taxa de piscadas:** a frequencia com que um indivíduo pisca. Teoricamente as pessoas piscam a uma taxa mais elevada quando cansados
* **Taxa de bocejos:** uma pessoa tende a bocejar quando cansada e com uma frequência ainda mais elevada quando a fadiga é extrema.

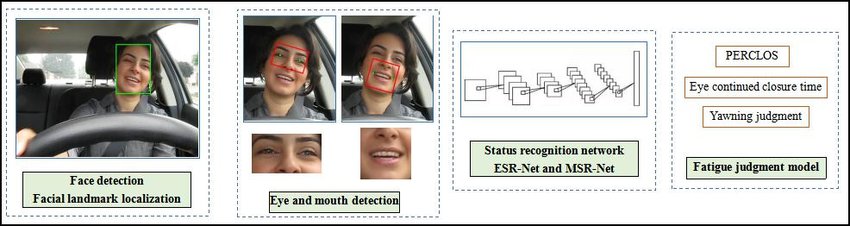
A arquitetura a ser utilizada no modelo não foi pré-estabelecida, porém pretende-se utilizar algumas técnicas clássicas de Machine Learning em comparação com técnicas de Redes Neurais Profundas a fim de determinar o mais promissor.

* 1. **Máscara de Detecção Facial**

Para realizar o processamento das imagens e extração das *features* discutidas acima, foram estudadas algumas bibliotecas do Python de visão computacional como: Dlib, OpenCV (CascadeClassifier como Haar Cascade), Cvlib, HOG, MTCNN (Multi-Task cascade Convolutional Neural Network)

* + 1. **Detecção facial com OpenCV e Dlib**
    2. **Detecção facial com MTCNN**





* 1. **Características de Entrada do Modelo**

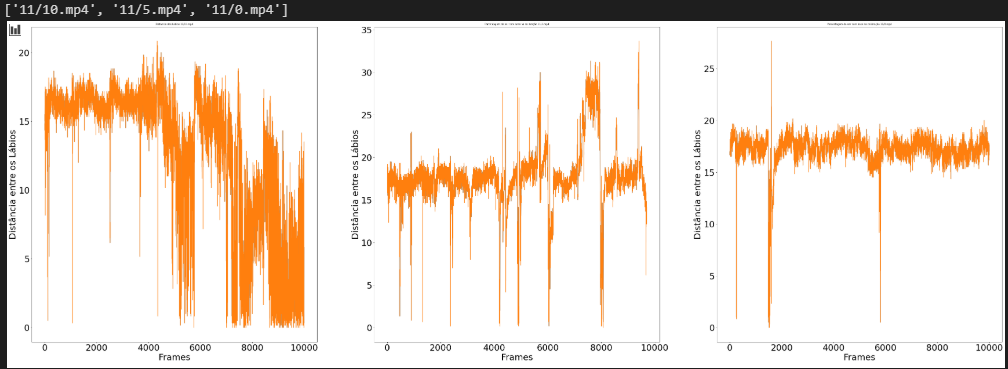
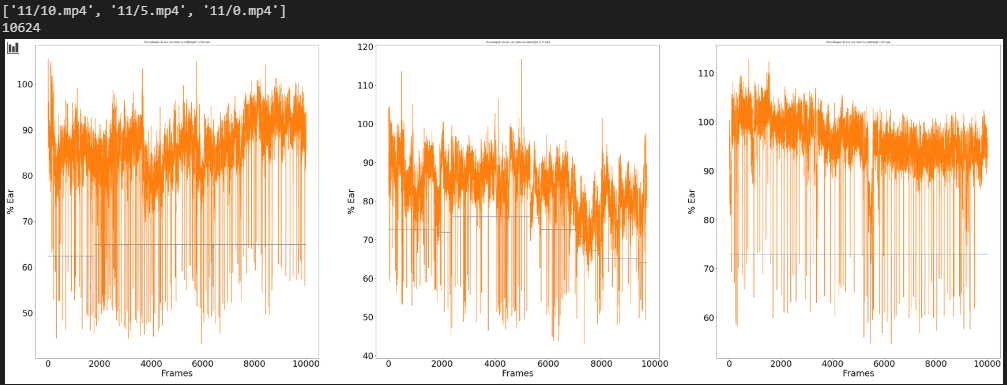
Para realizar o processamento das imagens e extração das *features* discutidas acima foi utilizada a biblioteca de visão computacional OpenCV programada em Python. Com o auxílio de uma máscara disponível na biblioteca dlib, foi possível capturar o nível de abertura dos olhos (EAR) e distância entre os lábios. Como auxílio da biblioteca de análise de dados, *pandas* foi possível extrair destes dois índices todas as demais features.



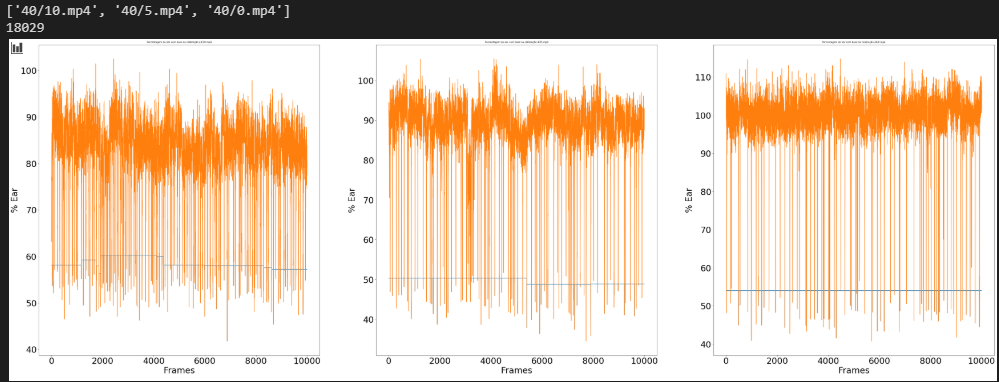
Figura 3 - . (fonte: <https://sites.google.com/view/utarldd/home>)

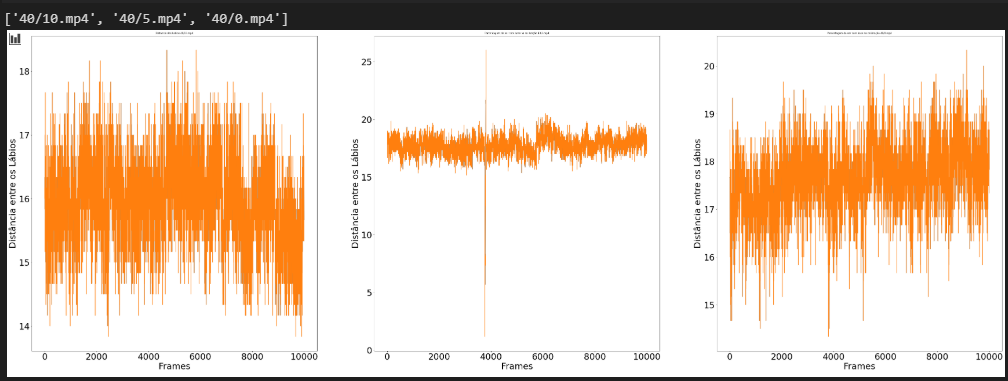
**Análise vídeos bons**

**Vídeo 11**

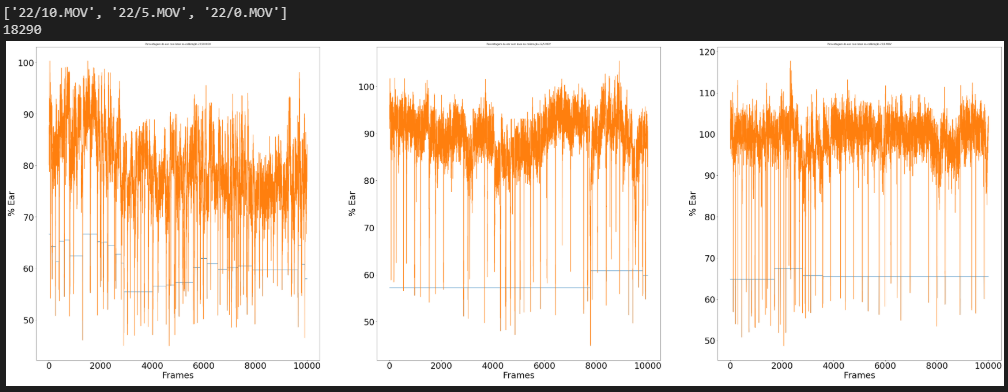


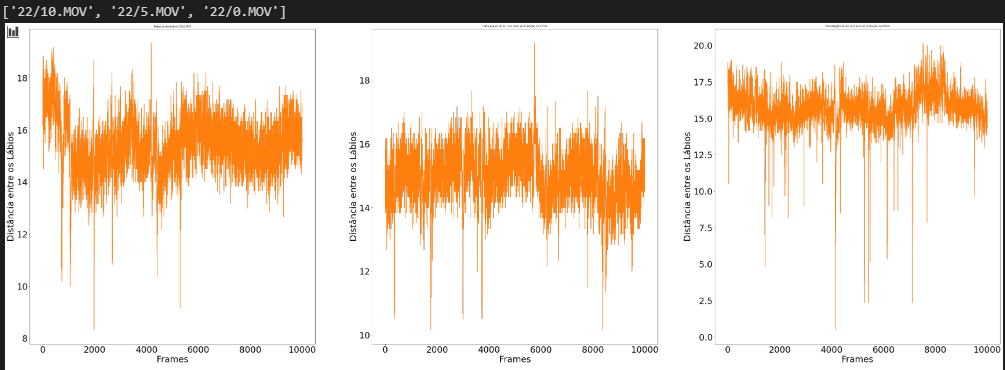
**Vídeo 40**

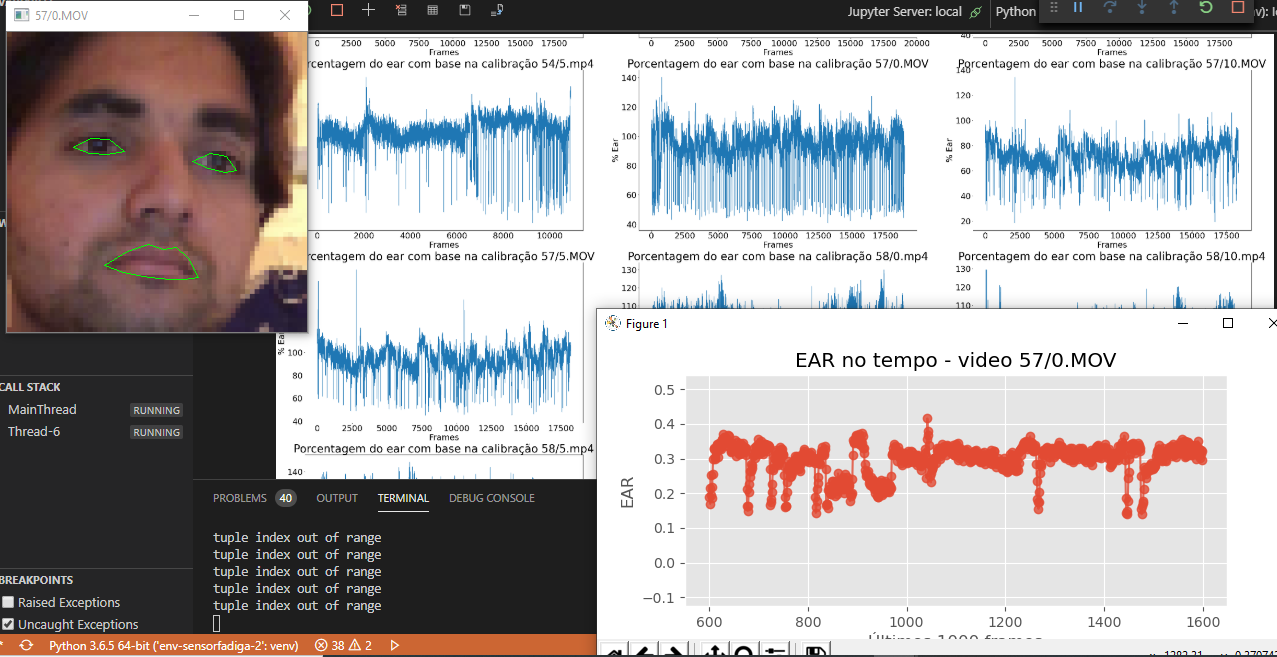




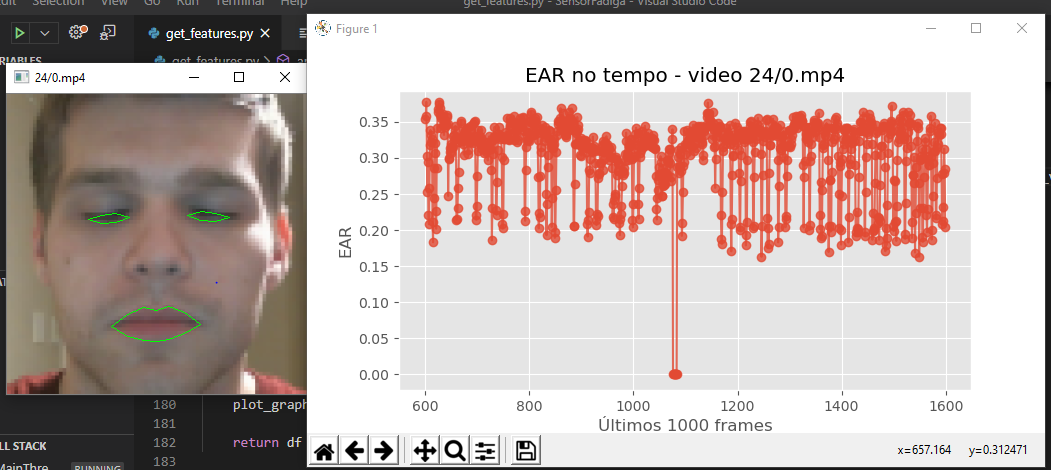
**Vídeo 22**

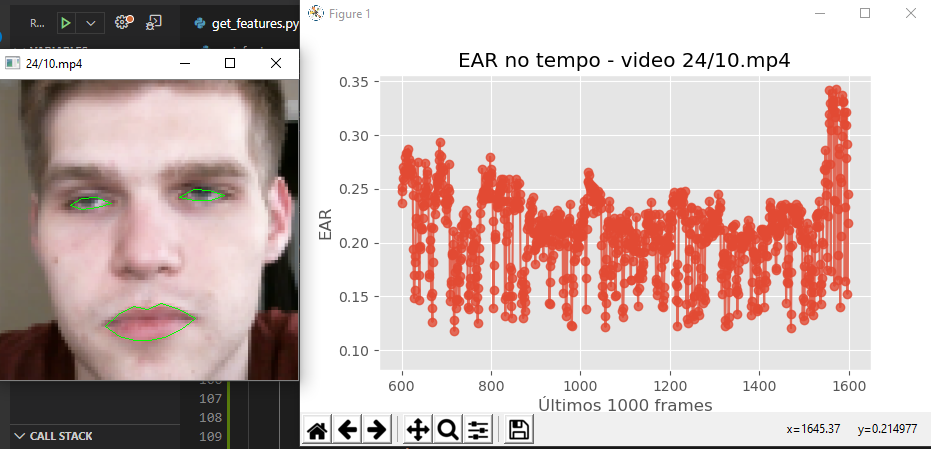


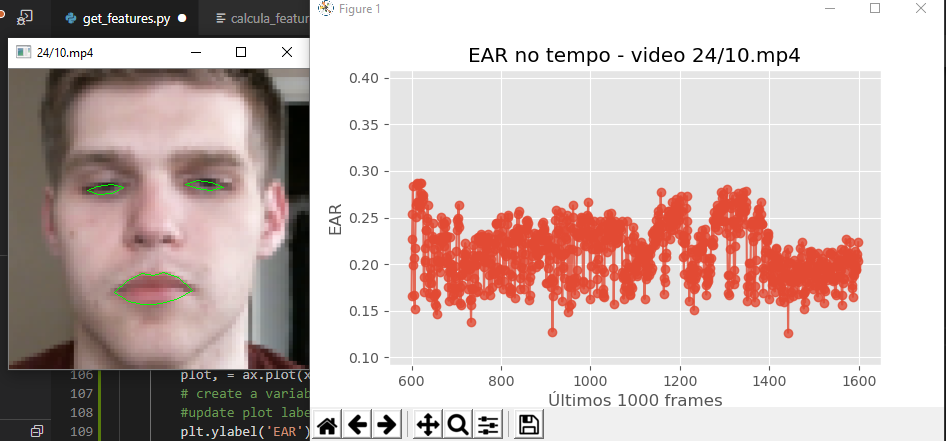


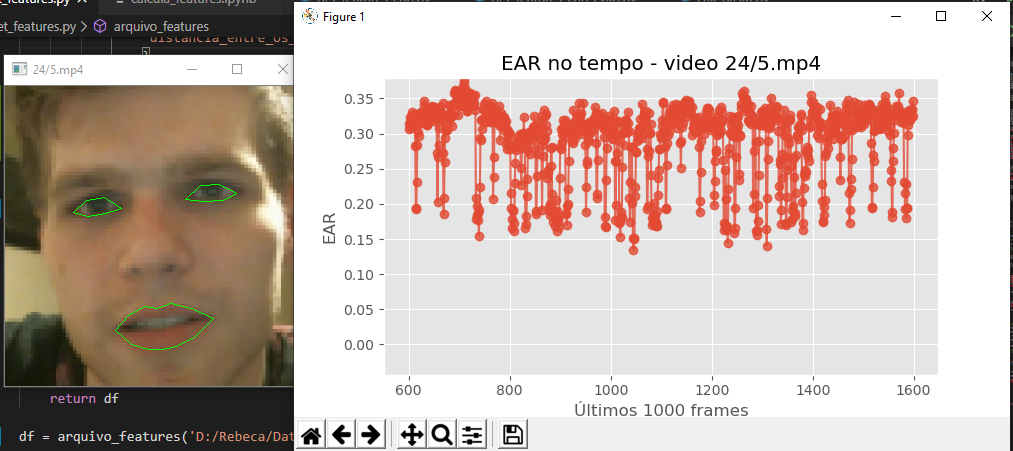


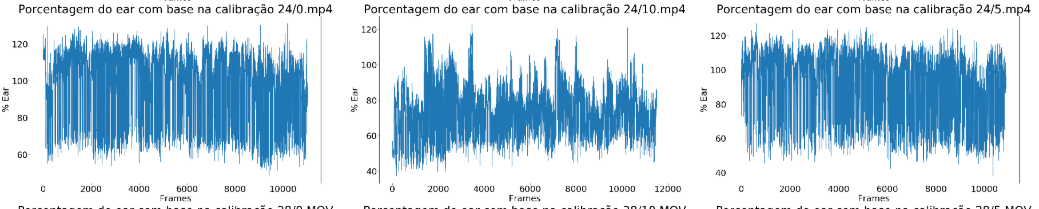




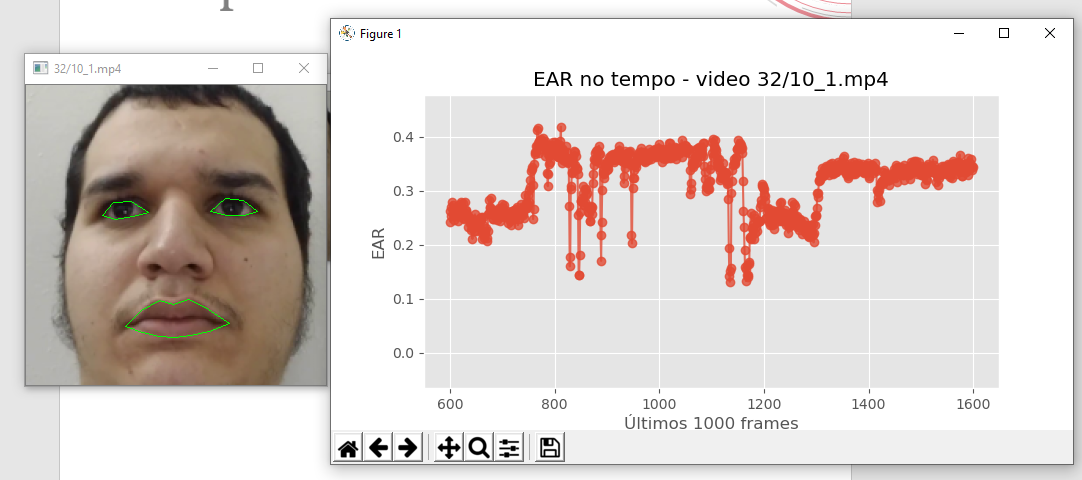




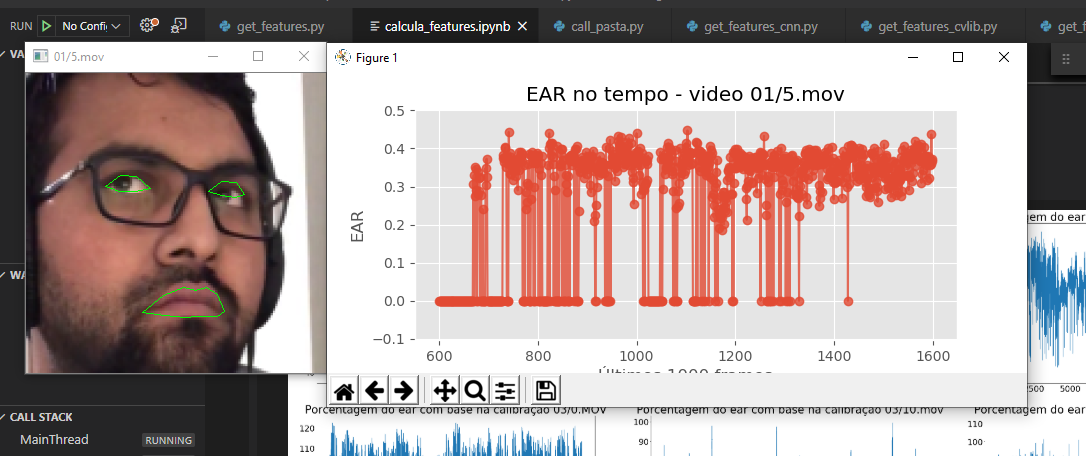


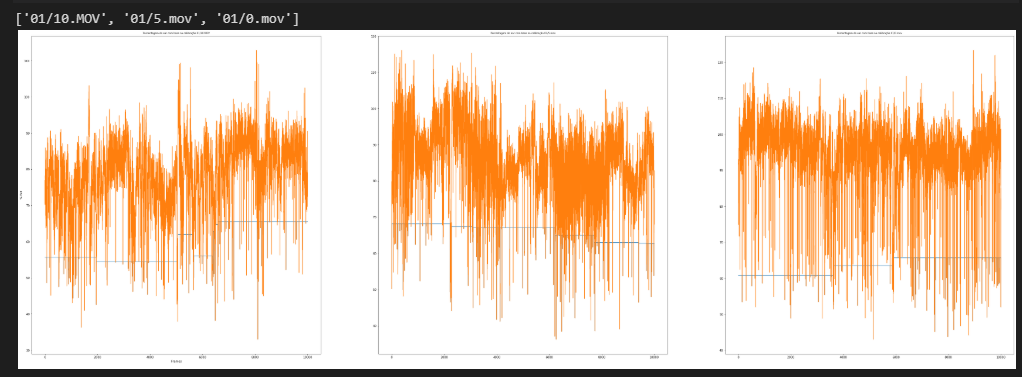


**Análise vídeos ruins**



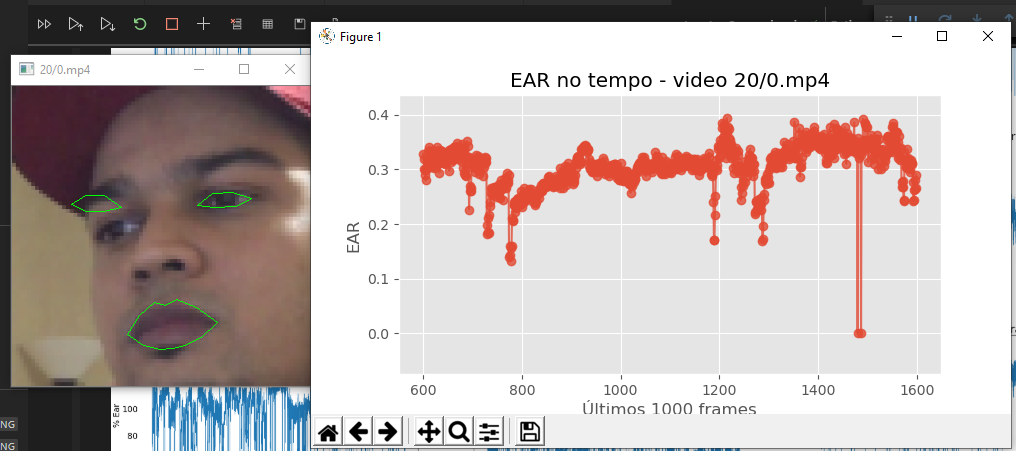
**Análise vídeos para limpar**

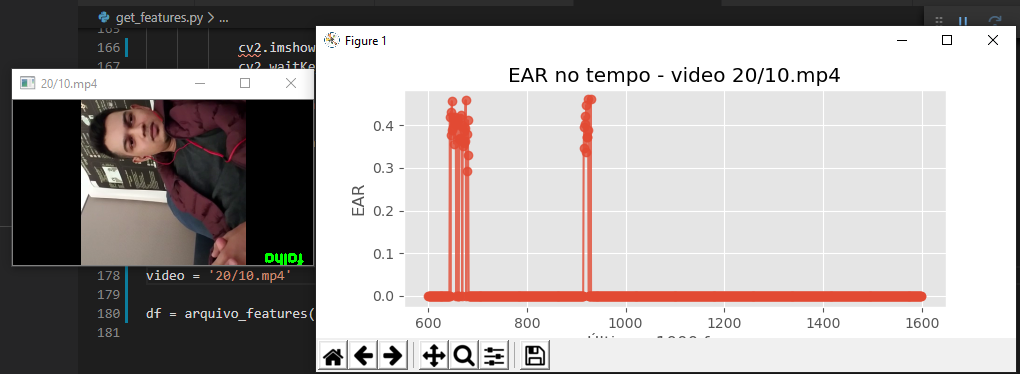


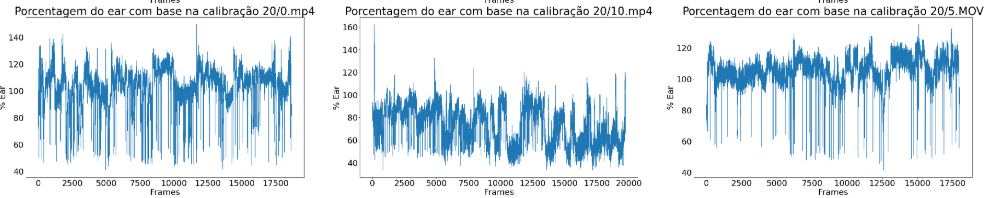


**Análise da biblioteca**









1. **Hardware**

Esta secção é dedicada à definição e explicação do Hardware deste projeto, identificando seus requisitos, os materiais necessários e a topologia de rede na qual o sensor estará submetido.

* 1. **Requisitos de Projeto**

- Alto FPS – maior que 100 FPS considerando o processamento no tablet

- Pouca iluminação – Noite e dia na mineração, infravermelho

- Custo Baixo – Ver concorrência

- Conexão por cabo no roteador

- Distância do objeto/pessoa ~ 90 cm – 70 a 110 cm

- Resolução alta – ROI no rosto para detecção de olhos e boca

- Dimensão do objeto/pessoa

* 1. **Topologia de Rede e Comunicação dos Dispositivos**

Esta secção visa definir e explicar a topologia de Rede

1. **Cronograma**

**08/06 – Segunda –** Texto de explicação e definição dos requisitos do hardware do projeto

**09/06 – Terça –** Implementar os vídeos bons no modelo de Machine Learning

**10/06 – Quarta –** Tabela comparando mapeamento de mercado

**11/06 – Quinta –** Conclusão e métricas de desempenho deste modelo de Machine Learning

**12/06 – Sexta –** Definição da câmera e da lente a ser utilizada

**15/06 – Segunda –** Texto de explicação do funcionamento da comunição dos dispositivos (Topologia de Rede)

**16/06 – Terça –** Implementar MTCNN e extrair dados acurados dos vídeos

**17/06 – Quarta –** Definição da câmera a ser utilizada

**18/06 – Quinta –** Implementar MTCNN e extrair dados acurados dos vídeos

**22/06 – Segunda –** Apresentação do projeto

**23/06 – Terça –** Implementar MTCNN e extrair dados acurados dos vídeos

1. **Referências**

[1] Macchiavello, Fiorella. Transoprte de Carga: Desafios para o desenvolvimento e para condições de trabalho dignas. Acessado em: 29/05/2020.

<https://cnttl.org.br/noticia/5329/cargas-estudo-aponta-que-fadiga-e-o-maior-fator-para-acidentes-nas-estradas-do-pais>

[2] Acidentes rodoviários: estatísticas envolvendo caminhões. – Brasília : CNT, 2019. Acessado em: 29/05/2020.

<https://cdn.cnt.org.br/diretorioVirtualPrd/34e78e55-5b3e-4355-9ebc-acf1b8e7b4a4.pdf>

[3] Revista Cobertura. Acessado em: 29/05/2020. <http://wwwold.revistacobertura.com.br/lermais_materias.php?cd_materias=20152&friurl=:-Fadiga-provoca-76-dos-acidentes-com-caminhoes->

[4] Asmontech. Notícias, Parceria SETCESP. Acessado em: 29/05/2020. <https://setcesp.org.br/noticias/sensor-de-fadiga-da-asmontech-o-que-e-como-funciona-e-para-que-serve/>

[5] UTA Real-Life Drowsiness Dataset. Acessado em: 02/06/2020.

<https://sites.google.com/view/utarldd/home>

[6] Paper sobre detecção de fadiga

[7] Eye Blink Detection With OpenCV, Pythin and dlib

<https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/>

[8] Brownlee, Jason. How to Perform Face Detection with Deep Learning. Acessado em: 08/06/2020.

<https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/>

**Anexo**