Detecção de Motorista Desatento com a Rede Neural MobileVGG e Base de Dados FFD

Lucas Luppi Amorim
Universidade Federal do Espírito Santo
Vitória, Brasil
lucasluppiam@gmail.com

Marcelo Bringuenti Pedro
Universidade Federal do Espírito Santo
Vitória, Brasil
marcelo.bp01@gmail.com

***Abstract*— De acordo com a Organização Mundial da Saúde (OMS), o número de mortes decorrentes de acidentes no trânsito tem aumentado continuamente nos últimos anos. Muitos desses acidentes decorrem de momentos de distração do motorista, que acaba se distraindo com o celular, olhando no retrovisor sem necessidade, ocupando suas mãos com algum objeto, ou alguma outra atividade, enquanto deveria ter sua atenção voltada exclusivamente à direção do veículo. Neste contexto, técnicas de visão computacional poderiam ser empregadas para avaliar se o motorista encontra-se distraído ou concentrado ao volante. Neste trabalho, utilizamos a rede neural MobileVGG para classificar dez ações diferentes do motorista, sendo uma delas o motorista concentrado, e as outras nove ações do motorista distraído. Para treinar a rede neural, utilizamos três datasets diferentes, sendo dois datasets utilizados por trabalhos relacionados e disponibilizados publicamente, e um dataset criado por nós a partir de filmagens feitas com motoristas colaboradores simulando momentos de distração ao volante.**

Keywords—Redes Neurais Convolucionais, Detecção de Motorista Desatento, MobileVGG.

I. INTRODUÇÃO

Segundo relatório publicado pela Organização Mundial da Saúde (OMS) [1], mais de 40 mil pessoas perdem a vida no trânsito no Brasil todos os anos. Estudos indicam que até 80% do volume dos acidentes de trânsito podem ser correlacionados com distrações do condutor [2]. Outros estudos identificam o uso do celular ao volante como responsável pelo aumento em até 4 vezes da probabilidade do condutor se envolver em um acidente grave [3].

Neste contexto, as técnicas de Visão Computacional [4] poderiam ser empregadas para, mediante análise de imagens de um condutor, avaliar se o mesmo encontra-se distraído ou concentrado ao volante. A partir da percepção de métricas acerca dos episódios de distração ocorridos, um condutor poderia se conscientizar sobre suas práticas ao volante e eventualmente tomar medidas no sentido de minimizar a quantidade de distrações ocorridas, quer ocorram voluntariamente ou não.

Neste trabalho, para classificar de forma automática se um motorista está distraído ou não, foi utilizada uma rede neural convolucional MobileVGG [5]. Avaliamos sua eficácia em 3 bases de dados diferentes, sendo duas bases utilizadas por trabalhos relacionados na literatura e disponibilizadas publicamente, e uma base de dados construída por nós.

Este trabalho está organizado nas seguintes seções: Trabalhos Relacionados, em que descrevemos trabalhos na literatura no domínio de detectar motoristas distraídos feitos anteriorment. A rede neural MobileVGG, em que descrevemos a rede convolucional utilizada neste trabalho. Metodologia, em que descrevemos os métodos utilizados para elaboração deste trabalho, incluindo os datasets e experimentos. Resultados e Discussões, em que descrevemos os resultados obtidos nos experimentos realizados. E finalmente, Conclusão, em que apresentamos a conclusão deste trabalho.

II. TRABALHOS RELACIONADOS

Nesta seção, apresentamos uma revisão de trabalhos relevantes na literatura sobre monitoramento das ações do motorista utilizando redes neurais convolucionais.

Bahetti *et al*. [5] apresentam a rede neural convolucional MobileVGG, como uma arquitetura modificada a partir da VGG16 [6]. Os autores tem como objetivo detectar e classificar ações de distração do motorista. A arquitetura é avaliada em dois diferentes datasets, e os resultados são comparados com arquiteturas CNN no estado-da-arte da literatura. Resultados obtidos indicam que a MobileVGG atinge valores de acurácia de 95.24% e 99.75% nos datasets utilizados, e requer menos complexidade computacional e memória, sendo mais eficiente que outras arquiteturas.

Abouelnaga *et al*. [7] criaram e disponibilizaram publicamente o dataset American University in Cairo (AUC) com dez classes de ações de motoristas para classificação de motorista desatento. Adicionalmente, os autores propuseram um sistema que alcança 95.98% de acurácia na classificação da postura de motoristas. O sistema consiste de uma rede neural convolucional treinada com um algoritmo genético. Entretanto, o sistema é muito ineficiente para aplicações em tempo real. Os autores também apresentam uma versão simplificada da rede que é capaz de operar em tempo real e atinge valor de acurácia de 94.29%.

Behera *et al.* [8] utilizaram o dataset AUC em uma proposta baseada em DenseNet que combina ideias de trabalhos de detecção de pose humana e transferência de conhecimento para reconhecimento visual, atingindo uma acurácia 94.2% na classificação.

III. A REDE NEURAL MOBILEVGG

Neste trabalho utilizamos a rede neural convolucional MobileVGG, apresentada em [5] para avaliar o desempenho da rede na tarefa de detectar motoristas distraídos. Informações detalhadas da arquitetura da MobileVGG podem ser visualizadas na Tabela I.

A MobileVGG possui 32 camadas. Ela não usa operações de maxpooling explicitamente para downsampling. Entretanto, são realizadas operações de convolução depthwise separable com stride 2. Todas as camadas de convolução depthwise e pointwise são seguidas por normalização em batch. Todas as camadas exceto a última são seguidas por uma função de ativação LeakyRelu após a normalização em batch, e a última camada de pooling é seguida de uma camada de softmax para classificação [5].

TABELA I
ARQUITETURA DA REDE MOBILEVGG

Nome da Camada	Formato do Filtro	Formato de Saída
Input Image	-	224 × 224 × 3
Conv_0	3 × 3 × 3 × 64	224 × 224 × 64
Depthwise_Conv_1	3 × 3 × 64 (stride 2)	112 × 112 × 64
Pointwise_Conv_1	1 × 1 × 64 × 64	112 × 112 × 64
Depthwise_Conv_2	3 × 3 ×64	112 × 112 × 64
Pointwise_Conv_2	1 ×1 ×128 × 128	112 × 112 × 128
Depthwise_Conv_3	3 × 3 ×128 (stride 2)	56 × 56 × 128
Pointwise_Conv_3	1 × 1 × 128 × 128	56 × 56 × 128
Depthwise_Conv_4	3 ×3 × 128	56 × 56 × 128
Pointwise_Conv_4	1 ×1 × 128 × 256	56 × 56 × 256
Depthwise_Conv_5	3 × 3 × 256	56 × 56 × 256
Pointwise_Conv_5	1 × 1 × 256 ×256	56 × 56 × 256
Depthwise_Conv_6	3 × 3 × 256 (stride 2)	28 × 28 × 256
Pointwise_Conv_6	1 × 1 × 256 × 256	28 × 28 × 256
Depthwise_Conv_7	3 × 3 × 256	28 × 28 × 256
Pointwise_Conv_7	1 × 1 × 256 × 512	28 × 28 × 512
Depthwise_Conv_8	3 × 3 × 512	28 × 28 × 512
Pointwise_Conv_8	1 × 1 × 512 × 512	28 × 28 × 512
Depthwise_Conv_9	3 × 3 × 512 (stride 2)	14 × 14 × 512
Pointwise_Conv_9	1 × 1 × 512 × 512	14 × 14 × 512
Depthwise_Conv_10	3 × 3 × 512	14 × 14 × 512
Pointwise_Conv_10	1 × 1 × 512 × 512	14 × 14 × 512
Depthwise_Conv_11	3 × 3 × 512	14 × 14 × 512
Pointwise_Conv_11	1 × 1 × 512 × 512	14 × 14 × 512
Depthwise_Conv_12	3 × 3 × 512 (stride 2)	7 × 7 × 512
Pointwise_Conv_12	1 × 1 × 512 × 512	7 × 7 × 512
Depthwise_Conv_13	7 × 7 × 512	1 × 1 × 512
Pointwise_Conv_13	1 × 1 × 512 × 512	1 × 1 × 512
Conv_14	1 ×1 × 512 × 512	1 × 1 × 512
Global Average Pooling	1 × 1	512
Softmax	Classifier	10

A MobileVGG é uma rede modificada a partir da VGG16 [6]. As modificações foram motivadas pelo fato da VGG16 não conseguir generalizar nos dados de teste desconhecidos para a tarefa de detecção de motorista distraídos. As modificações que foram feitas são descritas a seguir:

A. *Função de ativação LeakyReLU*

A função de ativação da unidade linear retificada (ReLU) tornou-se muito popular nos últimos anos devido a sua eficiência e convergência mais rápida. Mas, como a função ReLU define o valor de saída para zero para todas as entradas menores que zero, os pesos de alguns neurônios podem nunca ser atualizados e pode resultar em neurônios mortos. Para a MobileVGG a função de ativação foi alterada para LeakyReLU, devido esta superar o problema descrito anteriormente, introduzindo uma pequena inclinação na região negativa para manter as atualizações vivas [5].

B. *Dropout*

À MobileVGG foram adicionandas camadas do tipo dropout, a fim de evitar o problema de overfitting, reduzindo o aprendizado interdependente entre os neurônios. Esta camada faz isso ignorando alguns neurônios durante a fase de treino, por exemplo [5].

C. *L2 Regularização dos Pesos*

A regularização de pesos, também chamada de redução de pesos, baseia-se fortemente na suposição implícita de que um modelo com pesos menores é de alguma forma mais simples do que uma rede com pesos grandes [9]. É implementado pela penalização da magnitude quadrada de todos os parâmetros diretamente na função de custo. À MobileVGG foi adicionado o termo $0.5*\lambda*w^2$ para a função de custo considerando cada peso w da rede, onde λ é a força de regularização. A escolha de λ é um hiperparâmetro e é definida como 0.001 [6].

D. *Normalização em lote*

A normalização em lote ajuda a melhorar o desempenho e a estabilidade das redes neurais, forçando explicitamente as ativações através de uma camada de rede a seguir uma distribuição gaussiana [10]. Ele reduz a forte dependência da inicialização do peso, melhora o fluxo do gradiente através da rede e também permite taxas de aprendizagem mais altas. Em nosso trabalho, as ativações de todas as camadas convolucionais e totalmente conectadas são normalizadas [6].

E. *Substituição de camadas*

A principal desvantagem do VGG-16 é o número total de parâmetros, que chega a quase 140 milhões. Camadas totalmente conectadas são computacionalmente caras demais e também consomem a maioria desses parâmetros. Além disso, a camada de rede totalmente conectada pode ser aplicada apenas a entradas de tamanho fixo. Substituir as camadas totalmente conectadas por camada de convolução reduz o número de parâmetros e pode ser aplicada a vários tamanhos de entrada [11]. Para a MobileVGG, as camadas totalmente conectadas da rede VGG16 foram substituídas por camadas convolucionais do tipo 1 x 1 [6].

IV. METODOLOGIA

Este capítulo descreve a metodologia experimental utilizada para obtenção dos resultados.

A. *Datasets*

Para obtenção dos resultados, utilizamos três diferentes datasets que possuem as mesmas dez classes. O primeiro dataset é o *American University in Cairo* (AUC) [7]. Este dataset é composto por dez posturas diferentes do motorista, sendo uma delas do motorista dirigindo com atenção e as outras nove posturas são do motorista desatento, isto é, escrevendo ou lendo mensagem no celular com a mão direita, falando ao celular com a mão direita, escrevendo ou lendo mensagem no celular com a mão esquerda, falando ao celular com a mão esquerda, ajustando o rádio, bebendo, retocando maquiagem no retrovisor, pegando algo no banco traseiro, e olhando e conversando com o passageiro. Este dataset é composto por imagens de 31 motoristas diferentes gravados em 4 carros diferentes e em condições de tempo diferentes, como na luz do sol, sombra, e outras. O dataset contém um total de 17308 imagens, sendo 12977 imagens usadas para treino, e 4331 imagens usadas para teste, tendo a divisão das imagens de treino e teste sido realizada de forma aleatória.

O segundo dataset utilizado é o dataset State Farm [12]. Este dataset contém 22424 imagens para treino e 79726 imagens para teste. Entretanto, as imagens de teste não estão categorizadas, sendo portanto utilizadas 75% das imagens de treino (16818 imagens) para treino e 25% das imagens de treino (5606 imagens) para validação.

Além dos dois datasets utilizados no paper original, a rede foi treinada com um dataset criado especificamente para este trabalho.

O dataset **FFD** (Friends and Family Dataset) foi criado com a colaboração de familiares e amigos dos autores deste trabalho, que aceitaram fornecer suas imagens para gravação de vídeos de simulação das posturas de motorista desatento. A Figura 1 apresenta exemplos das dez posturas com os motoristas colaboradores do dataset FFD. No total, seis motoristas foram gravados em três dias diferentes, em

(a) C0: Dirigindo com atenção | (b) C1: Mensagem no celular com mão direita | (c) C2: Falando ao celular com mão direita | (d) C3: Mensagem no celular com mão esquerda | (e) C4: Falando ao celular com mão esquerda

(f) C5: Ajustando rádio | (g) C6: Bebendo | (h) C7: Maquiagem no retrovisor | (i) C8: Pegando algo no banco traseiro | (j) C9: Conversando com passageiro

Fig 1: As 10 posturas do dataset FFD

condições de tempo diferentes. Para maior variação das imagens, foram utilizados três carros nas gravações, que também foram realizadas locais diferentes. Como a quantidade de motoristas que tivemos para gerar as imagens de cada classe é menor que nas outras bases de dados, dois dos motoristas colaboradores foram gravados duas vezes, mas em dias, locais, roupas e carros diferentes para que as imagens tivessem maior variação.

Os motoristas colaboradores foram gravados realizando cada uma das dez posturas por aproximadamente 35 segundos. Os vídeos foram gravados a uma taxa de 30 quadros por segundo, e na resolução de 1920 x 1080 pixels. Os frames dos vídeos foram extraídos em resolução 640 x 480 pixels, e após uma seleção para remoção de frames a base de dados resultante teve um total de 40505 imagens úteis, das quais 29499 foram usadas para treinamento e teste da rede. As 11006 imagens restantes foram descartadas por limitação de memória da máquina após a execução de testes preliminares.

B. Pré-processamento de Imagens

A resolução original das imagens na base de dados AUC é 1920 × 1080 × 3. As bases de dados State Farm e FFD têm imagens de resolução 640 × 480 × 3. No procedimento original, todas as imagens são redimensionadas para 224 × 224 × 3 por interpolação bilinear antes de fornecer entrada para CNN, entretanto, devido a limitação de memória da máquina utilizada, redimensionamos as imagens para 128 × 128 × 3. Como uma etapa de pré-processamento, a média por canal de planos RGB é subtraída de cada pixel da imagem para centralizar os dados em torno da origem de ambos os conjuntos de dados separadamente.

C. Aumento de Dados

Para treinar o modelo com mais eficiência e desenvolver um classificador mais robusto, também é realizado um aumento de dados. Com o aumento de dados, um novo conjunto de imagens de entrada é gerado em cada época por várias transformações especificadas. Usamos as quatro transformações: deslocamento vertical, deslocamento horizontal, zoom e corte com valor máximo de 0.2. O número de imagens de entrada é duplicado aplicando as transformações geradas.

D. Experimentos

Um total de quatro experimentos foram realizados sobre as três bases de dados. Os dois primeiros experimentos visam reproduzir os resultados obtidos por [6], utilizando os datasets AUC e State Farm da mesma forma como foram divididos, isto é, a rede MobileVCC foi treinada com aproximadamente 75% das imagens selecionadas aleatoriamente para treino e 25% para teste. Os outros dois experimentos executam o treinamento da rede MobileVCC com o dataset FFD.

Para o primeiro treino realizado com o dataset FFD (treino FFD-1), ele foi dividido de forma aleatória com 23163 imagens usadas para treino e 6336 imagens usadas para validação da performance da rede. Desta forma, a base de validação contém imagens dos motoristas que também foram utilizados para treinar a rede, mas com frames diferentes.

Para um segundo treino com o dataset FFD (treino FFD-2), foram selecionados aleatoriamente dois motoristas, e todas imagens destes dois motoristas (7206 imagens) foram utilizadas apenas para validação, enquanto as imagens dos outros motoristas (22293 imagens) foram utilizadas para treino. Desta forma podemos avaliar a performance da rede com imagens de motoristas que não fizeram parte do treinamento.

E. Setup

Os pesos da rede foram inicializados com o inicializador normal Glorot. O otimizador Adam é usado com uma taxa de aprendizado de 0.0001, decaimento de 0.000001, beta1 com valor 0.9 e beta2 com valor 0.999. O número de épocas e o tamanho do lote são definidos como 256 e 32, respectivamente.

Todos os experimentos foram realizados em Python utilizando Keras e Tensorflow, sendo executados em uma máquina com processador Intel i5-10400f de 2.9GHz, 16GB de RAM, e com placa de vídeo GeForce RTX 3060 com 12GB de RAM.

V. Resultados e Discussões

As Figuras 2 e 3 apresentam os valores de acurácia e loss, respectivamente, como resultado do treinamento da rede utilizando a base de dados AUC. A matriz de confusão resultante deste experimento é exibida na Tabela II.

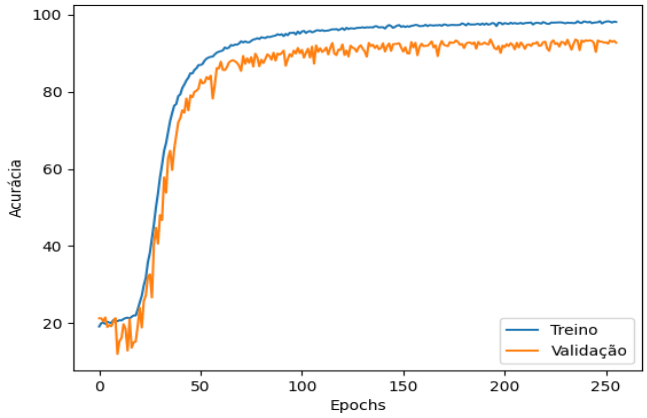

Fig 2: Acurácia do experimento com a base de dados AUC.

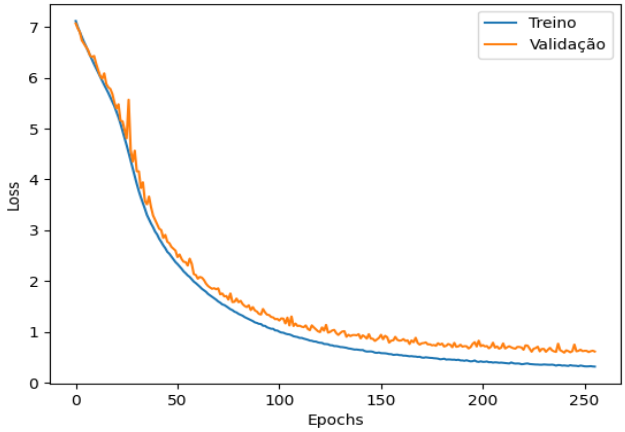

Fig 3: Loss do experimento com a base de dados AUC.

TABELA II

Matriz de Confusão do experimento com dataset AUC

Classe Verdadeira \ Classe Predita	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
C0	812	1	6	12	1	16	20	9	3	42
C1	0	311	8	1	3	2	1	0	0	0
C2	3	17	313	2	0	5	0	0	0	1
C3	10	5	2	465	4	0	4	0	1	3
C4	0	3	0	19	281	0	3	0	0	0
C5	10	0	1	1	0	287	4	0	1	1
C6	6	0	0	2	4	1	388	1	0	1
C7	19	0	0	0	3	0	4	268	1	6
C8	7	0	0	0	0	2	5	0	272	4
C9	13	0	0	0	0	0	9	0	2	619

As Figuras 4 e 5 apresentam os valores de acurácia e loss, respectivamente, como resultado do treinamento da rede utilizando a base de dados State Farm. A matriz de confusão resultante deste experimento é exibida na Tabela II.

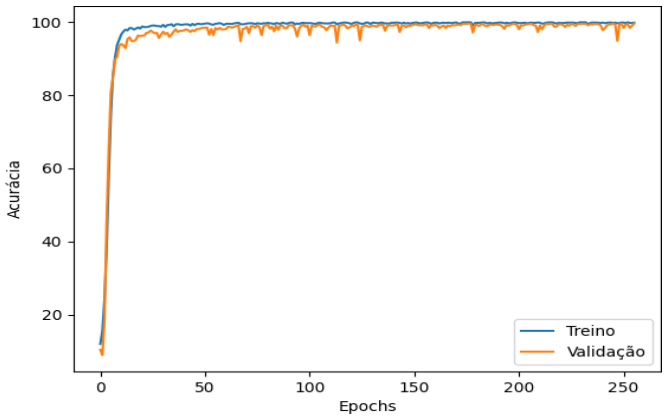

Fig 4: Acurácia do experimento com a base de dados State Farm.

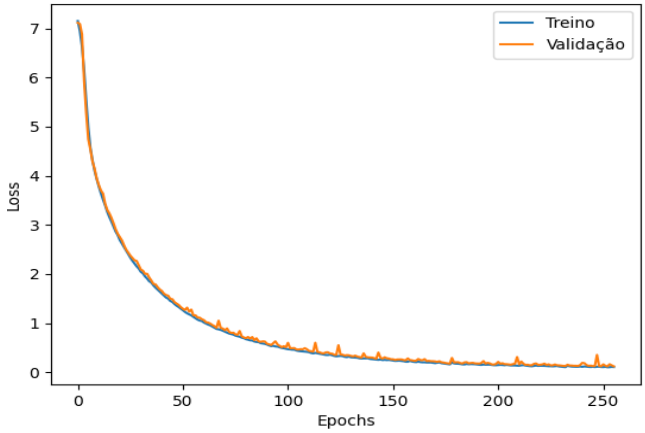

Fig 5: Loss do experimento com a base de dados State Farm.

TABELA III

Matriz de Confusão do experimento com dataset State Farm

Classe Verdadeira \ Classe Predita	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
C0	495	1	0	0	1	0	0	0	0	0
C1	0	453	0	0	0	0	0	0	0	0
C2	0	0	462	0	0	0	1	0	0	0
C3	1	0	0	468	0	0	0	0	0	0
C4	0	0	0	0	465	0	0	0	0	0
C5	2	0	0	0	1	458	0	1	0	0
C6	0	0	1	0	0	0	463	0	0	0
C7	0	0	0	0	0	0	0	399	0	1
C8	0	0	0	0	0	0	1	1	378	2
C9	0	0	0	0	0	0	0	1	2	422

As Figuras 5 e 6 apresentam os valores de acurácia e loss, respectivamente, como resultado do treinamento da rede utilizando a base de dados FFD. A matriz de confusão resultante deste experimento é exibida na Tabela III.

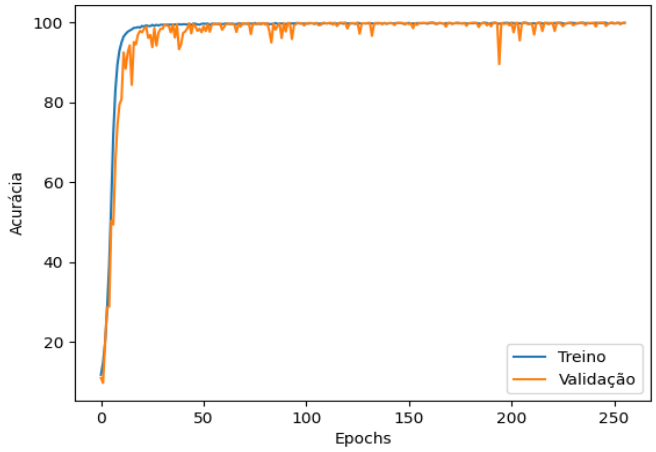

Fig 6: Acurácia do experimento FFD-1.

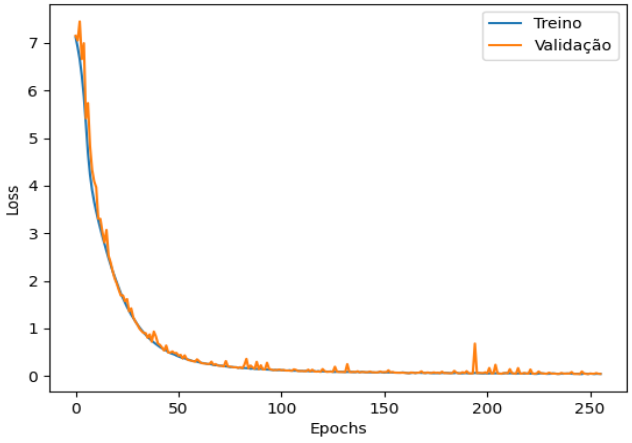

Fig 7: Loss do experimento FFD-1.

TABELA IV
MATRIZ DE CONFUSÃO DO EXPERIMENTO FFD-1

Classe Verdadeira \ Classe Predita	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
C0	600	0	0	0	0	0	0	0	0	0
C1	0	320	0	0	0	0	0	0	0	0
C2	0	0	230	0	0	0	0	0	0	0
C3	0	0	0	230	0	0	0	0	0	0
C4	0	0	0	0	240	0	0	0	0	0
C5	0	0	0	0	0	200	0	0	0	0
C6	0	0	0	0	0	0	200	0	0	0
C7	0	0	0	0	0	0	0	180	0	0
C8	0	0	0	0	0	0	0	0	180	0
C9	0	0	0	0	0	0	0	0	0	455

As Figuras 7 e 8 apresentam os valores de acurácia e loss, respectivamente, como resultado do treinamento da rede utilizando a base de dados FFD. A matriz de confusão resultante deste experimento é exibida na Tabela IV.

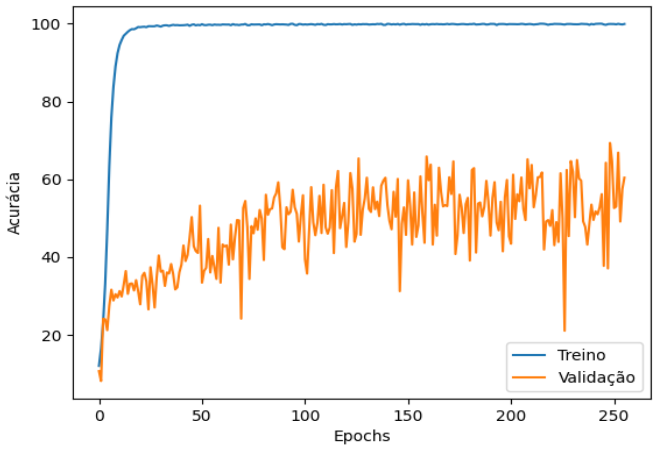

Fig 8: Acurácia do experimento FFD-2.

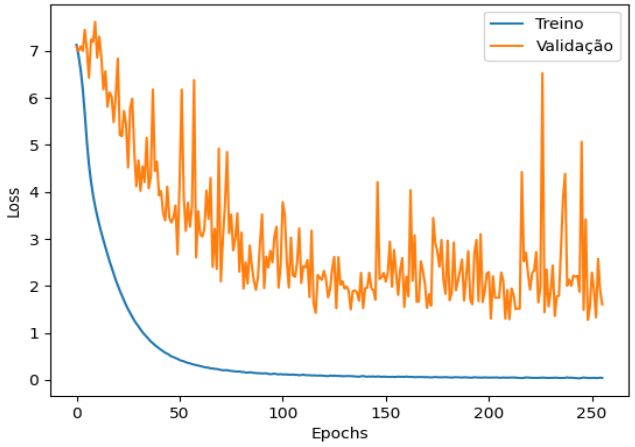

Fig 9: Loss do experimento FFD-2.

TABELA V
MATRIZ DE CONFUSÃO DO EXPERIMENTO FFD-2

Classe Verdadeira \ Classe Predita	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
C0	57	1	0	0	342	0	1	0	27	354
C1	0	283	7	22	52	0	142	0	24	101
C2	0	0	507	0	105	0	34	0	46	0
C3	23	0	0	640	9	0	0	0	0	11
C4	0	0	0	35	648	0	0	0	0	0
C5	11	0	8	0	14	593	0	0	95	13
C6	17	1	61	3	87	0	443	0	152	6
C7	0	186	13	0	7	0	181	173	124	0
C8	0	1	24	0	71	0	231	2	442	2
C9	14	0	18	1	172	0	0	0	1	568

TABELA VI
ACURÁCIA POR CLASSE USANDO A ARQUITETURA MOBILEVGG NO EXPERIMENTO FFD-1

Class	Total de Amostras	Predições Corretas	Predições Incorretas	Sensibilidade (%)
C0: Dirigindo com atenção	600	600	0	100.00
C1: Mensagem no celular com mão direita	320	320	0	100.00
C2: Falando ao celular com mão direita	230	230	0	100.00
C3: Mensagem no celular com mão esquerda	230	230	0	100.00
C4: Falando ao celular com mão esquerda	240	240	0	100.00
C5: Ajustando rádio	200	200	0	100.00
C6: Bebendo	200	200	0	100.00
C7: Pegando algo no banco traseiro	180	180	0	100.00
C8: Maquiagem no retrovisor	180	180	0	100.00
C9: Conversando com passageiro	455	455	0	100.00

TABELA VII
ACURÁCIA POR CLASSE USANDO A ARQUITETURA MOBILEVGG NO EXPERIMENTO FFD-2

Class	Total de Amostras	Predições Corretas	Predições Incorretas	Sensibilidade (%)
C0: Dirigindo com atenção	782	57	725	7.29
C1: Mensagem no celular com mão direita	631	283	348	44.85
C2: Falando ao celular com mão direita	692	507	185	73.27
C3: Mensagem no celular com mão esquerda	683	640	43	93.70
C4: Falando ao celular com mão esquerda	683	648	35	94.88
C5: Ajustando rádio	734	593	141	80.79
C6: Bebendo	770	443	327	57.53
C7: Pegando algo no banco traseiro	684	173	511	25.29
C8: Maquiagem no retrovisor	773	442	331	57.18
C9: Conversando com passageiro	774	568	206	73.39

A sensibilidade por classe, também chamada de True Positive Rate (TPR), é calculada para cada classe de acordo com a equação 1.

$$Sensitivity = \frac{TP}{TP + FN} \quad (1)$$

Os valores obtidos de sensibilidade por classe são apresentados nas tabelas VI e VII para os resultados dos experimentos FFD-1 e FFD-2 respectivamente.

A partir dos gráficos das Figuras 2 e 4, observamos que a rede atingiu resultados semelhantes aos apresentados em [6], com valores de acurácia próximo a 100% para as duas bases de dados nos dados de treino, e de 93% para a base de dados AUC e de 99% para a base State Farm nos dados de validação.

A matriz de confusão da base de dados AUC (Tabela II), mostra que as classes C0 e C9 são que mais se confundem neste dataset, enquanto no dataset State Farm isso não ocorreu. Esse resultado reafirma o que foi afirmado em [6], que tal discrepância pode ser resultado de imagens rotuladas incorretamente na base de dados AUC, enquanto este problema não está presente na base de dados State Farm.

O gráfico da Figura 6 apresenta os resultados de 100% de acurácia tanto para treino quanto para validação obtidos no experimento FFD-1, que utiliza a base de dados FFD que criamos para treinar a rede. Como este experimento utiliza frames de todos os motoristas no treino e teste selecionados aleatoriamente, acreditamos que tal resultado tenha se dado pela rede ter selecionado para o teste frames intermediários entre frames que foram usados no treinamento da rede, havendo assim pouca variação nas imagens de treino e teste. As tabelas IV e VI reafirmam que neste experimento, a rede não errou as predições em nenhum caso de validação.

O experimento FFD-2 atingiu acurácia no treino próxima a 100%, e um valor de acurácia de 69% nos dados de validação. Neste experimento, diferentemente dos outros três experimentos, a rede foi treinada com imagens de motoristas, e validada com imagens de motoristas que não fizeram parte do treino, e observamos assim um valor menor de acurácia da rede, apesar de ainda apresentar um resultado satisfatório.

Assim como nos experimentos da base AUC, podemos ver pela matrix de confusão apresentada na Tabela V, que a rede nesse experimento errou muitas predições da classe C0, confundindo com a classe C9. Consideramos que um motivo para isso ocorrer possa ser o fato de ter pouca variação das posições das mãos dos motoristas nas imagens, apenas mudando a posição do rosto do motorista, havendo maior possibilidade de confusão entre essas duas classes. Podemos também perceber que a rede nesse experimento confunde também a classe C0 muitas vezes com a classe C4, e a classe C6 com a classe C8, e vice-versa em todos os casos. Um motivo para isso pode ser a semelhança entre as imagens, que em ambas as classes possuem a mão direita do motorista interagindo com a frente do rosto.

VI. CONCLUSÕES E TRABALHOS FUTUROS

A distração do motorista é um problema sério que leva a um grande número de acidentes de trânsito em todo o mundo. Consequentemente, a detecção de motorista distraído torna-se um componente essencial do sistema em carros autônomos.

Pelos experimentos apresentados neste trabalho, podemos concluir que a rede MobileVCC desempenha de forma muito satisfatória a tarefa de detectar motoristas distraídos quando a rede é treinada previamente com os dados dos motoristas que ela irá classificar. Caso contrário, ela apresenta um bom desempenho, mas não ótimo.

Fomos capazes de reproduzir resultados muito semelhantes aos obtidos em [5] utilizando os datasets públicos AUC e State Farm, fazendo apenas pequenas modificações devido a limitações de memória.

Também criamos um dataset próprio chamado FFD com 40505 imagens com as mesmas classes dos datasets AUC e State Farm. A criação do dataset com a extração e seleção das imagens foi trabalhosa mas necessária para obtermos um treinamento da rede com um maior número de imagens e analisarmos o comportamento da rede com um dataset diferente dos datasets públicos. Com este dataset, treinamos a rede MobileVCC de duas formas diferentes, obtendo valor de acurácia de 100% quando usamos frames intermediários aos usados no treino para validação da rede, e 69% quando usamos motoristas que não fizeram parte do treino para teste.

Para trabalhos futuros sugeriremos aumentar a base de dados FFD com mais imagens de motoristas diferentes, possibilitando fazer mais experimentos, a fim de deixar a rede desenvolvida por [5] mais robusta e mais próxima do estado da arte para a tarefa de classificação de motoristas distraídos.

REFERENCES

[1] Organização Mundial da Saúde, Segurança Viária no Brasil. Accessado em 10/09/2019. https://www.paho.org/bra/index.php?option=com_content&view=article&id=5147:acidentes-de-transitofolha-informativa&Itemid=779

[2] National Highway Traffic Safety Administration, "The impact of driver inattention on near-crash/crash risk: an analysis using the 100-car naturalistic driving study data", Department of Transportation, NHTSA, Washington, 2006.

[3] S.P. Mcevoy et al., "Role of mobile phones in motor vehicle crashes resulting in hospital attendance: a case-crossover study". Br Med J331:428–430, 2005.

[4] Backes, A. "Introdução à visão computacional usando Matlab", 1a Edição. Brasil: Elsevier, 2016. 290 pgs.

[5] B. Baheti, S. Talbar and S. Gajre, "Towards Computationally Efficient and Realtime Distracted Driver Detection With MobileVGG Network," in IEEE Transactions on Intelligent Vehicles, vol. 5, no. 4, pp. 565-574, Dec. 2020, doi: 10.1109/TIV.2020.2995555.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. Int. Conf. Learn. Representations, 2015, pp. 1–14.

[7] Y. Abouelnaga, H. Eraqi, and M. Moustafa. "Real-time Distracted Driver Posture Classification". Neural Information Processing Systems (NIPS 2018), Workshop on Machine Learning for Intelligent Transportation Systems, Dec. 2018.

[8] A. Behera, A. H. Keidel, and C. Science, "Latent body-pose guided densenet for recognizing driver's fine-grained secondary activities," in Proc. 15th IEEE Int. Conf. Adv. Video Signal-Based Surveillance, Nov. 2018, pp. 1–6.

[9] A. Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In Proceedings of the Twenty-first International Conference on Machine Learning, ICML '04, pages 78–, New York, NY, USA, 2004. ACM.

[10] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, pages 448–456. JMLR.org, 2015.

[11] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(4):640–651, April 2017.

[12] State Farm Distracted Drivers Dataset. Acessado em 10/09/2019. https://www.kaggle.com/c/state-farm-distracted-driver-detection/data