ABORDAGEM DE DETECÇÃO DE MOTOS EM PONTO CEGO UTILIZANDO VISÃO COMPUTACIONAL

Leonardo Carvalho de Sant' Anna [1], Bruno Bastos Stoll [2]

[1] Acadêmico do curso de Engenharia de Computação

[2] Mestre – Professor Multivix - Vitória

RESUMO

Esse projeto tem como objetivo abordar o estudo sobre Inteligência artificial convolucional, na finalidade de tentar descontinuar os acidentes automobilístico usando redes neurais para identificar motos ou carros no seu ponto cego, alertando o motorista. Dessa forma, impedindo acidentes causados por conversão de pista. A solução proposta usa uma arquitetura de redes neurais chamada YOLO no qual vai identificar o objeto e mandar a mensagem para o condutor avisando que tem algum veículo se aproximando do seu carro e podendo entrar em seu ponto cego do carro.

Palavras Chaves: Visão Computacional, detectar motos em ponto cego

1. INTRODUÇÃO

A sociedade desde a revolução industrial vem se desenvolvendo, produzindo inovações a cada dia. A invenção dos automóveis quebrou paradigmas trazendo um novo modo de vida para a época trazendo mudanças socioeconômicas, transformando o modo que a indústria construía os seus produtos usando os modelos das empresas automobilísticas Fordismo e Toyotismo que acabaram sendo vanguarda para sua época puxando a fila de modelo de gestão da segunda revolução industrial trazendo mais otimização e melhor eficácia em seus produtos.

Uma tecnologia que vem chamando atenção do cenário de inovações é a implementação da Inteligência Artificial (IA), fazendo assim mudanças no cotidiano e tirando algumas funções da mão humana passíveis de ocorrer falhas e tendo uma taxa de acerto muito maior, diminuindo assim acidentes e desperdícios. A IA vem adicionando novos elementos no seguimento dos automóveis, encaminhando uma nova revolução no modo que a humanidade vai enxergar a maneira de direção, onde a IA praticara uma direção consciente e defensiva.

No mundo atual é bem difícil ver alguém que não use um meio de transporte, isso acaba causando um número excessivo de veículos em vias. Dessa forma, aumenta o número de Acidentes de Trânsito (AT). A Organização Mundial da Saúde (OMS) fez um estudo que mostra que as mortes decorrentes do AT ocupam a nona

posição de mortalidade pelo mundo, onde 1,25 milhão morre, no mundo, por ano em decorrência de acidentes de trânsito (WHO, 2015). E dessa total metade das vítimas são pedestres, ciclistas e motociclistas (USP, 2018). Isso mostra o quão perigoso é dirigir atualmente, pois o Brasil é considerado um dos países que mais tem acidente de trânsito, ocupando o 3ª lugar entre os países com maior número de mortes no trânsito, levando a óbitos 40 mil pessoas por ano (AMBEV e FALCONI, 2017; BRASIL, 2016; WHO, 2015).

Esta pesquisa tem como objetivo avançar nos estudos de redução de acidentes de trânsito, em situações que o motorista é surpreendido por algum veículo em seu ponto cego. Assim, usando a câmera traseira do carro para acudir o motorista, auxiliando de maneira que evite um eventual acidente. Dessa forma, foi implementada um sistema para detectar as motos em pontos cegos utilizando Visão Computacional. Foram usados dois tipos de modelos pré-treinados para ver qual atendia melhor o nosso desenvolvimento. A pesquisa é caracterizada como uma Prova de Conceito, do inglês *Proof of concept* (PoC), que corresponde a um teste de um software para mostrar que o programa funciona e através de feedback chegar a um hiper parâmetro que seja aceitável e condizente com a aferição de viabilidade técnica (MCADAM, 2009). Na metodologia, foi utilizada a abordagem qualitativa e quantitativa, de natureza aplicada, com o objetivo descritivo e com procedimentos experimentais.

Para validar a pesquisa, foram utilizados dois modelos pré-treinados e cinco (5) imagens para validação da solução, na finalidade de detectar motos em movimento. E os resultados preliminares indicam que a abordagem proposta pode melhorar significativamente a redução de acidentes.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. Inteligência artificial

Iniciando sua história no ano de 1943 com o primeiro artigo sobre redes neurais após 7 anos Alan Turing conhecido como pai da computação, criou o teste de Turing onde ele avalia se uma máquina consegue se passar por um humano em uma conversa por escrito, um ano depois em 1951 surgia a calculadora SNARC onde ela calculava as operações matemáticas simulando sinapse. Mas o “Marco Zero” ocorreu em 1956 na conferência de Dartmouth onde reuniu as maiores mentes da computação da

época, os participantes saíram de lá entusiados que anos depois surgi-o *Advanced Research Projects Agency* (ARPA). (Kleina 2018).

Nesse período teve um grande avanço em 57, Frank Rosenblatt apresenta o perceptron. Esse algoritmo com nome de personagem de Transformers é uma rede neural de uma camada que classifica resultados e começou como uma máquina chamada Mark 1. Já em 58, surge a linguagem de programação Lisp, que na época virou padrão em sistemas de Inteligência artificial e hoje inspira uma família inteira de linguagens. Em 59, vemos pela primeira vez o termo *machine learning*, descrevendo um sistema que dá aos computadores a habilidade de aprender alguma função sem serem programados diretamente pra isso. Basicamente, significa alimentar um algoritmo com dados, para que a máquina aprenda a executar uma tarefa automaticamente. Em 64, teve o primeiro chatbot do mundo, ELIZA, que conversava de forma automática imitando uma psicanalista, usando respostas baseadas em palavras-chave e estrutura sintática. E em 69 é demonstrado o Shakey, primeiro robô que unia mobilidade, fala e certa autonomia de ação. Ele era lento e cheio de falhas, mas funcionava. (KLEINA 2018).

2.2. Aprendizado de Máquina

O aprendizado de máquina é uma vertente dentro da inteligência artificial, na qual ela é dividida em subcamadas podendo ter várias ramificações em suas composições. O Aprendizado de Máquina Supervisionado, o programador sempre coloca dois tipos de informações os dados de entrada e os dados resultantes da determinada informação para que o programa possa ter um auxilio em seu aprendizado um exemplo, é o filtro de Spam, ele é treinado com muitos exemplos de e-mails junto a suas classes (spam ou não spam) e conforme novos dados são fornecido a este modelo, ele consegue fazer a Classificação pois já foi ensinado a reconhecer padrões dentro daquelas classes (spam ou não spam) (MAIA, 2020).

Algoritmos comuns nesse tipo de Aprendizado:

- K-NN (*K-Nearest Neighbours*).
- *Decision Tree* e *Random Forest* (Árvores de Decisão e Florestas Aleatórias).
- Regressão Linear.
- Regressão Logística.
- SVM - Máquinas de Vetores de Suporte.
- Redes Neurais

Em contrapartida o Aprendizado de Máquina Não Supervisionado o programador fornece os dados, porém não são rotulados, fazendo com que o programa tente aprender sozinho. Um exemplo, vamos supor que você tenha um e-commerce e que tenha um volume muito grande de dados sobre os visitantes do seu site. Como já foi dito, dados são informações, você pode fazer inferências em cima desses dados. Nesse caso, você pode querer aplicar um algoritmo de *clustering*, algoritmo de generalização, para detectar grupos de visitantes semelhantes para tentar identificar comportamentos parecidos em relação as compras do seu site. Como esses dados não são rotulados, você não tem a informação de quais grupos existem e nem em qual grupo cada visitante pertence. O algoritmo tentará encontrar essas conexões sem a sua ajuda, sem a sua supervisão (MAIA, 2020).

Alguns algoritmos desse tipo de Aprendizado:

- Clustering:
- Clustering Hierárquico (HCA)
- k-Means Aprendizado da regra de associação:
- Apriori
- Eclat
- Redes Neurais

O Aprendizado de Máquina Semi-Supervisionado um bom exemplo desse aprendizado é o Google fotos, você já deve ter notado isso nas suas fotos em família ou amigos. O serviço carrega todas as suas fotos e automaticamente reconhece as pessoas que estão ali. Por exemplo, ele reconhece que a pessoa A aparece nas fotos 1, 3 e 5, enquanto a pessoa B aparece nas fotos 2, 4 e 6. O algoritmo faz um agrupamento, sendo assim a parte não supervisionada. Já a parte supervisionada é quando você precisa rotular aquelas pessoas, ou seja, dizer ao aplicativo quem são aquelas pessoas (MAIA, 2020).

O Aprendizado de Máquina por Reforço, também chamado de agente. Que irá interagir com o ambiente externo para realizar uma ação e atingir um determinado objetivo. O ambiente irá recompensar ou punir o agente de acordo com suas ações. E assim, com base no feedback que ele recebeu por ter executado tal ação, ele aprenderá por si só qual é a melhor estratégia (MAIA, 2020).

Figura 1 – Tipos de Aprendizado de Máquina

Fonte: (MAIA, 2020)

2.3. Transfer Learning

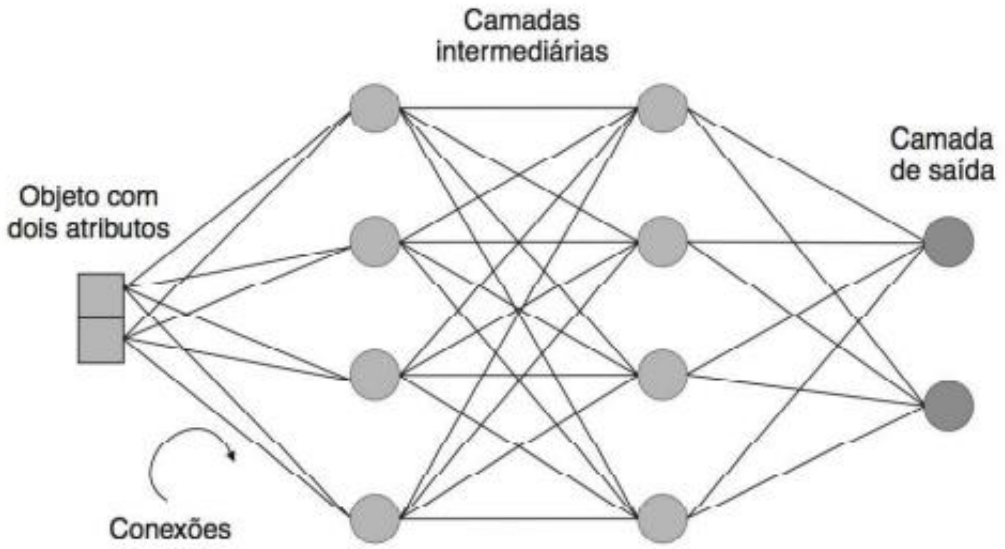
Transfer learning é o método onde se aproveita os treinamentos realizados em um dado momento, esse recurso é bastante aplicado pois reduz-se tempo de treinamentos e formação de *dataset*, dessa forma é possível que os pesos das redes já treinadas possam ser reutilizados em uma rede neural e assim as mesmas podem passar por um novo ciclo de treinamento com uma base de dados específicas a fim de conseguir resultados rápidos com pouco treinamento a ser realizado, essa técnica também possibilita a utilização de uma rede neural sem nenhuma necessidade de realização de treinamento, para isso basta que conheça a estrutura da rede utilizada assim como os seus parâmetros (BROWNLEE, 2019).

2.4. Redes Neurais

Quando os primeiros modelos de computadores eletrônicos foram produzidos deram ao início ao estudo de redes neurais artificiais em 1943, onde os pesquisadores McCulloch e Pitts apresentaram um modelo matemático de neurônio artificial no qual podiam realizar funções logicas simples, com isso sacramentando que a combinação de vários neurônios artificias em sistemas neurais tem um elevado poder computacional (FACELI *et al.*, 2021). Anos depois Rosenblatt em 1958 cria as redes Perceptron que possuem duas camadas usando simples adição e subtração,

entretanto as pesquisas perderam notabilidade pois em 1970 após a publicação do livro de Minskey e Papert (1969).

Figura 2 - Estrutura da Rede Neural



Fonte: (FACELI, 2011)

Conforme apresentada na Figura 2, uma Rede Neural Artificial (RNA) é um modelo computacional muito simplificado do Sistema Nervoso Central. É um gráfico orientado rotulado onde os vértices são organizados em camadas. Os vértices são unidades computacionais bastante simples que correspondem a neurônios biológicos simplificados. Eles integram, de forma não linear, os sinais que recebem de outras unidades. De acordo com o sinal que recebem, eles calculam seu próprio nível de ativação. As bordas do gráfico correspondem às sinapses entre as unidades. Eles codificam a influência de uma determinada unidade sobre qualquer outra unidade que recebe uma conexão daquela. Cada conexão pode aumentar ou diminuir a ativação da unidade receptora e esta influência é representada pelo peso associado à borda. Assim como o Sistema Nervoso Central, uma RNA é um sistema de aprendizado; ele aprende a realizar uma tarefa a partir de dados que exemplificam a tarefa que está sendo executada. Um algoritmo de aprendizagem de RNA é um procedimento para adaptar os pesos da rede de forma que a rede comece a realizar uma tarefa computacional, conforme expresso pela associação da ativação de unidades na camada de entrada à ativação correspondente de unidades na camada de saída (COELHO, 2017).

As redes neurais artificiais são redes interconectadas por elementos unitários conhecidos como neurônios, esse nome existe visto que essas redes foram baseadas no funcionamento do cérebro humano. Cada neurônio é constituído pelas entradas, que são os sinais provenientes de dados que podem ser de dados externos ou dados de demais neurônios, eles têm também sua função de ativação, que recebe a soma de todas as entradas com seus respectivos pesos. O resultado da entrada passa por uma função de ativação que determina quando a saída do neurônio será ativada ou não, ou seja, pode-se dizer que a saída do neurônio dependera se o conjunto de entradas atingirão um valor suficiente para ativar a saída do neurônio ou determinará um valor de saída dependendo da função de ativação do neurônio (BRAGA, LUDEMIR e CARVALHO, 2000).

Existem diversos tipos de funções de ativações, pode-se destacar a seguir:

- **Tanh**: abreviação para tangente hiperbólica e tem sua saída variando conforme a função matemática entre -1 a 1; (PONTI MA, 2017)
- **Logistic**: representa a função logística com sua variação de 0 a 1; (PONTI MA, 2017)
- **Relu**: inicia uma rampa ou um função de segundo grau somente para valores positivos de entradas, caso contrário a função manterá em valor nulo; (PONTI MA, 2017)
- **Prelu**: é uma função Relu onde é possível ajustar alguns parâmetros tais como a forma de trabalho dá em caso de entradas negativas. (PONTI MA, 2017)

2.5. Aprendizado profundo

As Redes Neurais foram limitadas durante várias décadas devido às restrições de hardware. O avanço nos estudos de Redes com topologias mais complexas, como o treinamento dezenas de camadas e vários milhões de conexões em conjuntos de dados de milhões, foram resolvidos em meados da década passada. Esse novo paradigma é chamado de Aprendizado Profundo, do inglês *Deep Learning*, ou simplesmente chamadas de Redes Profundas. Essas Redes Profundas são treinadas na GPUs em vez de CPUs e se beneficiam do ambiente de software personalizado. Pois é possível resolver problemas antigos impostos pela Inteligência Artificial, como reconhecimento de fala, tradução de texto, geração de sequência textual, classificação imagem e geração de texto a partir de imagem (LECUN, et. al 2015, GOODFELLOW, 2016, BENGIO et al. 2013).

Uma Rede Neural Convolucional (RNC) é um tipo de rede neural de múltiplas camadas com arquitetura profunda, inspirada no funcionamento biológico de

processamento de dados visuais, que utilizam a operação de convolução ao invés da multiplicação de matrizes em pelo menos uma das camadas. As RNCs vêm sendo amplamente utilizadas pela comunidade da computação, alcançando resultados práticos muito positivos (veículos e robôs autônomos), e apresentando o maior sucesso no tratamento de problemas envolvendo detecção, segmentação e reconhecimento de objetos em imagens (LECUN et al. 2015). Convolução é um tipo especializado de operação linear, que combina operações para extrair (filtrar) determinadas informações de um conjunto de dados. Essa operação possui características importantes que permitem melhorar o desempenho do Aprendizado de Máquina: conectividade esparsa e compartilhamento de parâmetros (GOODFELLOW et al. 2016).

2.6. Visão Computacional

Visão computacional é a ciência responsável pela visão de uma máquina, pela forma como um computador enxerga o meio à sua volta, extraindo informações significativas a partir de imagens capturadas por câmeras de vídeo, sensores, scanners, entre outros dispositivos. Estas informações permitem reconhecer, manipular e pensar sobre os objetos que compõem uma imagem. É a construção de uma descrição explícita e significativa de objetos físicos a partir de imagens. A compreensão da imagem é muito diferente do processamento de imagens, que estuda a transformação imagem-imagem, não a construção explícita de descrição. Visão computacional é a construção de descrições explícitas e significativas de objetos físicos a partir de imagens. A compreensão da imagem é muito diferente do processamento de imagens, que estuda transformações de imagem a imagem, não construção explícita de descrição. Descrições são um pré-requisito para reconhecer, manipular e pensar em objetos. (BALLARD, 1982).

2.7. Carros autônomos

O conceito relacionado a Carros Autônomos consiste no uso de computação aplicada, utilizando inteligência artificial para que os veículos façam trajetos sozinhos sem a intervenção do condutor. Para isso, são utilizados algoritmos de aprendizado profundo para detecção de placas, outros veículos, pedestres, curvas etc. Porém essa realidade parece estar um pouco distante, mesmo que existam hoje carros desse tipo, nenhum é 100% autônomo. Para ser mais exato hoje existem 6 níveis de autonomia do veículo, sendo nível 0 o que depende 100% do condutor, mas possui alguns

auxílios como sistemas que ajudam o motorista a dar ré com sensores e entre outros. Já no nível 5 o motorista seria totalmente desnecessário para comandar o veículo, sendo seu o único trabalho é de configurar a rota. Atualmente estamos no nível 3 no qual o sistema do carro é capas de conduzir o veículo, mas ele pede para que o motorista assuma o controle do veículo em algumas situações (FONSECA, 2020).

O trabalho (DE ANDRADE, 2020) estrutura uma ferramenta com a finalidade de contar de veículos em variados tipos de rodovia e diferentes cenários gravados por câmeras de segurança, sendo que para alcançar seu objetivo foi utilizado YOLO em conjunto com DeepSORT, para o rastreamento dos veículos que serão contabilizados.

3. DESENVOLVIMENTO

O trabalho apresenta um estudo na área de Visão Computacional com ênfase na detecção de motocicletas em pontos cegos de automóveis, na finalidade de auxiliar os condutores na direção defensiva. Foram validadas ferramentas que poderiam auxiliar o desenvolvimento do projeto, sendo inicialmente planejada a criação um modelo preditivo capaz de detectar motos. No entanto, optou-se pelo uso de transferência de aprendizado com uso das redes neurais já treinadas: Darknet[1] e OpenImages [2].

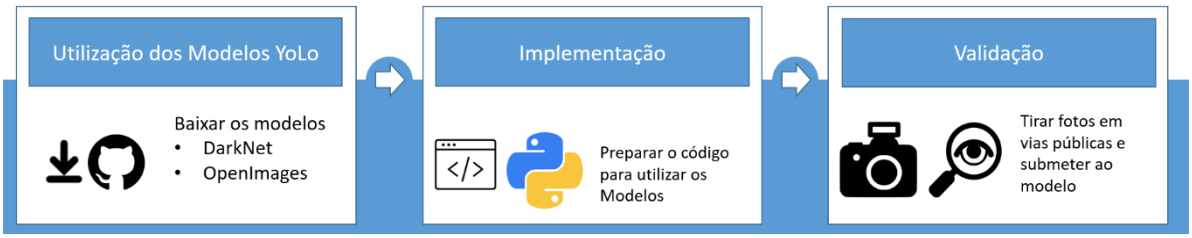
- Darknet: modelo multiclasses capaz de detectar 80 objetos
- OpenImages: mdoelo multiclasses capaz de detectar 601 objetos

Conforme apresentado na Figura 3, a pesquisa foi organizada em três etapas. Sendo que a primeira é a utilização do YOLO, que consiste na utilização de caixas que prevê em 3 escalas diferentes. Nosso sistema extrai recursos dessas escalas usando um conceito semelhante para redes em pirâmide de recursos. De nosso extrator de recurso base, adicionamos várias camadas convolucionais (REDMON, 2018). Os sistemas de detecção anteriores adaptam classificadores ou localizadores para realizar a detecção, que aplicam o modelo a uma imagem em vários locais e escalas. As regiões de alta pontuação da imagem são consideradas detecções. Usamos uma abordagem totalmente diferente. Aplicamos uma única rede neural à imagem completa. Essa rede divide a imagem em regiões e prevê caixas

[1] https://github.com/AlexeyAB/darknet/
[2] https://pjreddie.com/darknet/

delimitadoras e probabilidades para cada região. Essas caixas delimitadoras são ponderadas pelas probabilidades previstas (REDMON, 2018). A segunda etapa é a utilização dos modelos pré-treinados Darknet e OpenImages. Já a terceira e última etapa é a validação da pesquisa.

Figura 3 - Método da pesquisa



Fonte: próprio autor

Etapa 1) Utilização dos Modelos YoLo

Entrando no servidor do Gooble Colab[3], conforme apresentado na Figura 4, foram feitas o passo a passo: (a) foi feito o download do framework Darknet para poder trabalhar com *Deep Learning*; (b) Após sua instalação foi executado o comando para checando se a pasta do Darknet está integrada corretamente; (c) E uso dos modelos pré-treinados que faz uso do YOLO.

Figura 4 - Utilização do YoLo e Darknet

[3] https://colab.research.google.com/?utm_source=scs-index

Fonte: próprio autor

Após instalar o programa Darknet os modelos pré-treinados foram instalados, conforme apresentado na Figura 5. O Darknet é um código aberto de redes neurais onde se desenvolvem trabalhos na área de Visão Computacional para implementar pesquisas ou construir modelos do zero com objetivos específicos. No caso desse trabalho foram utilizados dois modelos fornecidos pela comunidade, citados anteriormente.

Figura 5 - Instalação dos modelos pré treinados

Etapa 2) Implementação

Na implementação foi realizada a: importação das bibliotecas; configuração do arquivo de compilação do Cuda [4]; validação do uso da NVIDIA; implementação das funções de detecção de motos e apresentação de imagens.

A) Importação das bibliotecas: As bibliotecas cv2 (OpenCV), matplotlib (Apresentar gráficos) e TensorFlow (configuração e compilação de redes neurais) é utilizada na aplicação.

B) Configuração do Makefile: Conforme apresentado na Figura 6, é feita a configuração do arquivo MakeFile, na finalidade de deixá-lo no com informações de uso de GPU, em vez do CPU, uso do OpenCV e usar o Cuda. E ao final do processo de configuração é feita a compilação do arquivo.

Figura 6 - Configuração do arquivo Makefile

```
[ ]  !sed -i 's/OPENCV=0/OPENCV=1/' Makefile
     !sed -i 's/GPU=0/GPU=1/' Makefile
     !sed -i 's/CUDNN=0/CUDNN=1/' Makefile
```

Fonte: próprio autor

C) Validar NVIDIA: A validação se as configurações estão ativas através do comando nvidia-smi, conforme figura 7.

[4] https://developer.nvidia.com/cuda-zone

Figura 7 - Validação das configurações da NVIDIA

```
!nvidia-smi
```

```
Mon Oct 25 19:57:42 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.74       Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   35C    P0    32W / 250W |    345MiB / 16280MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Fonte: Próprio autor

D) Implementação das funções: Através das funções detectar e mostrar é possível validar o uso do Modelo, conforme apresentado na Figura 8, Bloco A e Bloco B.

Figura 8 – Bloco A - Função Detectar; Bloco B - Função Mostrar

Bloco A

```
import os
def detectar(imagem):
  os.system(f"./darknet/yolov4.weights {imagem}")
  mostrar('predictions.jpg')
```

Bloco B

```
def mostrar(caminho):
  imagem = cv2.imread(caminho)
  fig = plt.gcf()
  fig.set_size_inches(18,10)
  plt.axis('off')
  plt.imshow(cv2.cvtColor(imagem, cv2.COLOR_BGR2RGB))
  plt.show()
```

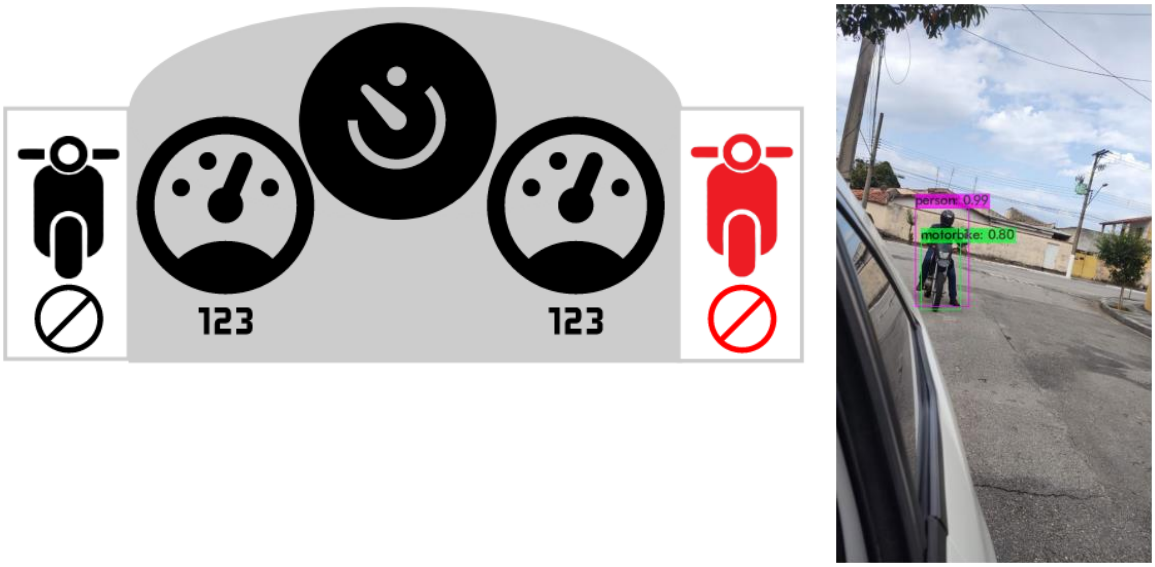
Fonte: Próprio autor

Etapa 4) Validação

Para a validação da solução proposta de detectar motocicletas em pontos cegos, foram tiradas fotos de um veículo em movimento e após isso foram submetidas à ferramenta. Conforme apresentada na Figura 9, é exemplifica a detecção de uma moto em ponto cego. Dessa forma, o sistema localiza uma moto, no Bloco A da Figura 9, assim será sinalizado ao motorista a existência de moto(s), Bloco B da Figura 9.

Figura 9 - (A) Detecção de Moto; (B) Sinalização de moto

Bloco A **Bloco B**

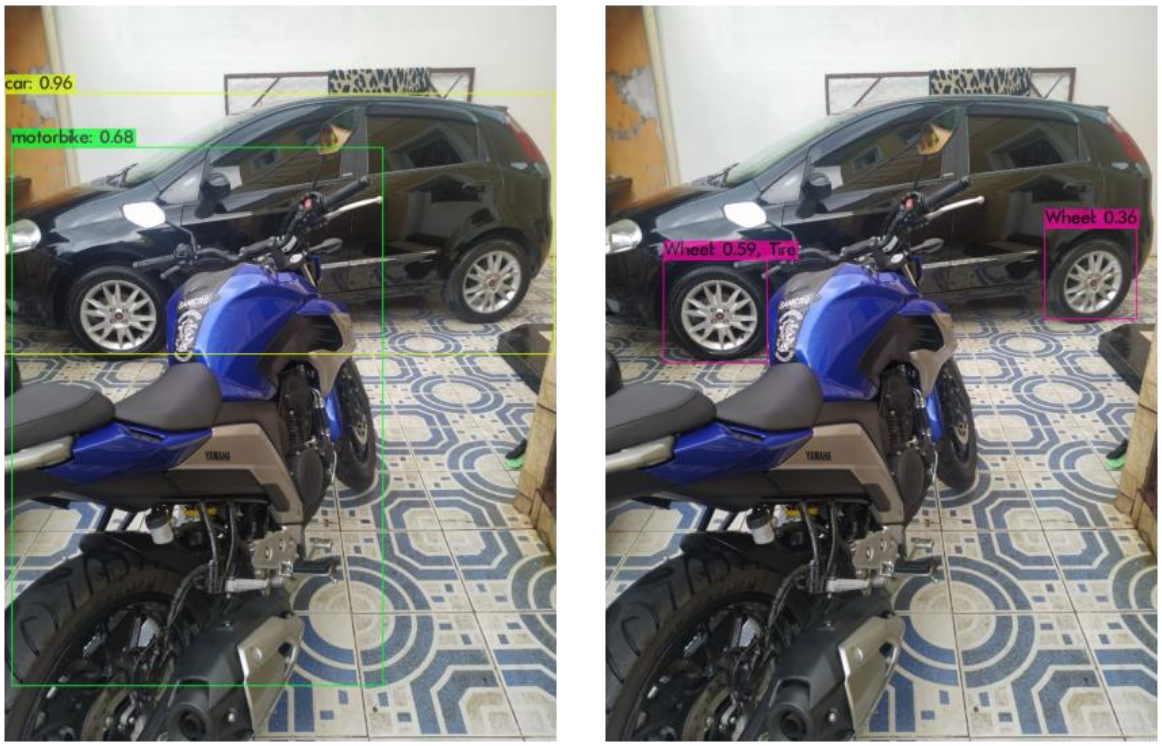


Fonte: próprio autor

O modelo OpenImages, conforme apresentado na Figura 10, Bloco B, não é capaz de detectar a existência de automóveis e acaba focando nas rodas do carro. Já o modelo Darknet apresenta ótimo desempenho conseguindo identificar os dois veículos precisamente.

Figura 10 – (A) Predição Modelo Darknet; (B) Predição Modelo OpenImages

Imagem (A) **Imagem (B)**



Fonte: próprio autor

Nas Figuras 11 e 12 são apresentadas as comparações de detecção de motos com os Modelo Darknet e Modelo OpenImages. Na Figura 12, Bloco A, o Modelo Darket obteve confiabilidade de 92%, já o modelo B obteve uma acurácia de 49% expondo a superioridade dos Modelo Darknet em comparação ao mesmo. Percebesse que ele acabada se focando em outros objetos e com isso acaba diminuindo a sua acurácia.

Figura 11 – (A) Predição Modelo Darknet; (B) Predição Modelo OpenImages

Bloco (A) Bloco (B)

Figura 12 – (A) Predição Modelo Darknet; (B) Predição Modelo OpenImages

Fonte: Próprio autor

Conforme descrito nesta seção e apresentado nas figuras 11 e 12, a ferramenta foi submetida a experimentos, na finalidade de avaliar se as funcionalidades estão de acordo com o objetivo proposto. Dessa forma, a arquitetura tecnológica proposta atende aos requisitos descritos na solução. Também é possível verificar que o Modelo Darknet é mais adequado de utilizar, quando é utilizada a transferência de aprendizado podemos empregar que os modelos utilizados possuem métricas diferentes de identificação que a carreta na hora da execução do programa.

4. CONCLUSÃO

Este trabalho apresentou uma pesquisa sobre Visão Computacional com abordagem de detecção de motos em pontos cegos de carros para tentar desvair os acidentes ocorridos em vias. Com as alterações do padrão de vida e nos hábitos da sociedade, vivemos rotinas exaustivas e muitas vezes corrida. Que se reflete diretamente no trânsito, uma vez que temos mais pressa para realizar as atividades, o que acaba tendo como consequência mais motoristas imprudentes no trânsito. A tecnologia

aplicada pode auxiliar as pessoas em tarefas básicas, como na direção defensiva. Dessa forma, a presente pesquisa apresentou uma solução tecnológica para auxiliar motoristas de carros na condução avisando ao condutor a detecção de motos em pontos cego utilizando redes neurais com o framework YoLo. Os resultados preliminares usando modelos pré-treinados foram muito satisfatórios, principalmente utilizando o Modelo Darknet, quando comparado com o Modelo OpenImages. Sendo possível que a visão computacional com ênfase na diminuição de acidentes de trânsito tem um potencial de crescimento muito grande na prevenção de acidentes, apoiando motoristas de carros tornando o trânsito mais seguro.

REFERÊNCIAS

BALLARD, Dana H.; BROWN, Christopher M. Computer vision, 1982. **Prenice-Hall, Englewood Cliffs, NJ**, 1982.

BRAGA, A. P.; LUDEMIR, T. B.; CARVALHO, A. C. P. L. F. **Redes Neurais Artificiais:** Teoria e Aplicações. 1. ed. Rio de Janeiro: LTC, 2000.

BROWNLEE, J. A Gentle Introduction to Transfer Learning for Deep Learning. **machinelearningmastery.**, 2019.

CARLEY, Michael et al. Looking to the future. In: **Engineering to survive: Global solutions for sustainable development**. Thomas Telford Publishing, 1996. p. 3: 41-46.

COELHO, Orlando Bisacchi; SILVEIRA, Ismar. Deep learning applied to learning analytics and educational data mining: A systematic literature review. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. 2017. p. 143.

DE ANDRADE, Matheus Molin. **APLICAÇÃO DE VISÃO COMPUTACIONAL PARA RASTREAMENTO E CONTAGEM DE VEÍCULOS EM RODOVIAS**. 2020. Tese de Doutorado. Universidade Federal do Rio de Janeiro.

FACELI, Katti et al. Inteligência Artificial: Uma abordagem de aprendizado de máquina. 2011.

FONSECA, GUSTAVO. Carro Autônomo: Veja a História Desse Tipo de Veículo e Entenda Seu Funcionamento. *In*: FONSECA, GUSTAVO. **Carro Autônomo: Veja a História Desse Tipo de Veículo e Entenda Seu Funcionamento**. Https://doutormultas.com.br/carro-autonomo-7-curiosidades-tecnologia/, 2020. Disponível em: https://doutormultas.com.br/carro-autonomo-7-curiosidades-tecnologia/. Acesso em: 8 out. 2021.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep learning**. MIT press, 2016.

KLEINA, Nilton. **A história da inteligência artificial**. 2018. Disponível em: https://www.tecmundo.com.br/mercado/135413-historia-inteligencia-artificial-video.htm. Acesso em: 21 out. 2021.

LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. **nature**, v. 521, n. 7553, p. 436-444, 2015.

LIMA, Tamires Feitosa de *et al*. **ANÁLISE EPIDEMIOLÓGICA DOS ACIDENTES DE TRÂNSITO NO BRASIL**. 2018. 7 f. Tese (Doutorado) - Curso de Mestrado de Saúde Pública, Universidade Federal do Ceará, Ceara, 2018. Disponível em: http://publicacoesacademicas.unicatolicaquixada.edu.br/index.php/eedic/article/view/31 02. Acesso em: 21 set. 2021.

MAIA, Beatriz. **Tipos de Aprendizado de Máquina**: saiba mais sobre os tipos de aprendizado de machine learnig. Saiba mais sobre os tipos de aprendizado de Machine Learnig. 2020. Disponível em: https://dev.to/beatrizmaiads/tipos-de-aprendizado-de-maquina-3-5d66. Acesso em: 21 out. 2021.

MCADAM, Rodney; MCADAM, Maura; BROWN, Valerie. Proof of concept processes in UK university technology transfer: an absorptive capacity perspective. **R&d Management**, v. 39, n. 2, p. 192-210, 2009.

MEDEIROS, João Bosco. TOMASI, Carolina. Redação de artigos científicos: métodos de realização, seleção de periódicos, publicação – São Paulo: Atlas, 2016.

PONTI MA, C. G. D. Como funciona o deep learning. In: SBC **Tópicos em gerenciamento de dados e informações**. Uberlandia: [s.n.], 2017. Disponivel em: <http://sbbd.org.br/2017/wpcontent/uploads/sites/3/2017/10/topicos-em-gerenciamento-de-dados-e-informacoes-2017.pdf>.

REDMON, Joseph; FARHADI, Ali. Yolov3: An incremental improvement. **arXiv preprint arXiv:1804.02767**, 2018.

SCHMIDT, Diogo et al. Um modelo de predição de mortalidade em unidades de terapia intensiva baseado em deep learning. In: **Anais do XVIII Simpósio Brasileiro de Computação Aplicada à Saúde**. SBC, 2018.

USP , Acidentes de trânsito no Brasil, um problema de saúde pública, Jornal da USP. 2018. Disponível em: https://jornal.usp.br/atualidades/acidentes-de-transito-no-brasil-um-problema-de-saude-publica/. Acesso em: 27 set. 2021.

WORLD HEALTH ORGANIZATION. Global status report on road safety 2015. World Health Organization, 2015.