

# Detecção de Objetos Através de Rede Neural do tipo FOMO

---

## Detecção de Objetos Através de Rede Neural do tipo FOMO

Neste projeto, apresentamos uma evolução do modelo de rede neural para detecção de objetos do tipo **FOMO** clássico.

O modelo **FOMO** ( Faster Objects, More Objects ) foi originalmente proposto pela empresa **Edge Impulse**.

Um modelo tipo **FOMO** clássico utiliza um trecho de rede **Mobilenet V2** ( truncado em uma certa camada escolhida ), sendo terminado por duas camadas de convolução (Conv2D), que conclui a inferência na dimensão desejada ( uma matriz 2D com vetores tipo “one-hot encoding” ).

A nova arquitetura apresentada estende o modelo **FOMO** clássico, acrescentando camadas customizadas ao final da rede, introduzindo a capacidade do modelo em reconhecer os tamanhos dos objetos detectados.

Nos referenciaremos à nova arquitetura pelo nome de **VOLTAMO**.

Foram implementados 2 variantes do modelos **VOLTAMO**:

## Modelo VOLTAMO

Um detector de objetos de 2 classes (cubos plásticos demarcados com “X” ou “O”), capaz de detectar 4 tamanhos distintos (ou proximidade entre objeto e câmera) a partir de imagens monocromáticas em baixa resolução (96 x 96 pixels) em taxa 5 FPS ( Frames por Segundo ).

## Modelo VOLTAMO VEHICLE

Um detector de veículos (1 classe) trafegando sob viadutos, treinado para detectar carros ou ônibus a partir de imagens coloridas (RGB) em resolução um pouco superior (160 x 160 pixels), em taxa de pelo menos 3 FPS. Imagens coloridas neste resolução aumentam significativamente a carga computacional no microcontrolador. Esta variante do modelo foi refinada em sucessivas etapas de ajustes e treinamentos, visando-se a obtenção de taxa de inferência não inferior a 3 FPS ( um mínimo aceitável para processar imagens em tempo real, nesta aplicação específica ).

A detecção de objetos é fundamental em diversas aplicações. Porém, em microcontroladores com restrições de memória, surgem desafios adicionais devido às limitações de processamento e armazenamento.

# Detecção de Objetos Através de Redes Neurais do tipo FOMO

---

O projeto utiliza o **Tensorflow**, um framework popular para Machine Learning (ML). **Tensorflow Lite** e **Tensorflow Lite Micro** são utilizados para se criar modelos ML otimizados para microcontroladores com recursos limitados.

São exploradas técnicas de pré-processamento, extração de características e treinamento de modelos ML baseados em redes neurais convolucionais (CNNs).

Técnicas de otimização (compressão e quantização de parâmetros) são aplicadas, visando-se a redução da complexidade computacional dos modelos ( compressão dos modelos ).



# Detecção de Objetos Através de Redes Neurais do tipo FOMO

---

## FOMO e VOLTAMO

A ferramenta **Edge Impulse** foi utilizada no início do projeto para prova de conceito e modelagem de uma arquitetura de rede neural tipo **FOMO** com detecção de 2 classes de objetos.

Posteriormente, com a evolução do ciclo de vida do aprendizado de máquina do projeto, criou-se a arquitetura de rede neural tipo **VOLTAMO**, capaz de detectar 8 classes ( 2 classes x 4 sub-classes ) em ambiente interno.

Em virtude da extensão do prazo do projeto, foi possível a criação de uma segunda variante denominada **VOLTAMO VEHICLE**, um modelo de rede neural um pouco mais desafiador ao microcontrolador ( por processar imagens coloridas em resolução superior ).

Um modelo **FOMO** típico utiliza a rede **Mobilenet V2** truncada na camada '**block\_6\_expand\_relu**' como base para extração de features. O modelo **VOLTAMO** trunca a rede **Mobilenet V2** um pouco antes, na camada '**block\_4\_expand\_relu**', resultando em um modelo menor (menos parâmetros). Entretanto, o modelo **VOLTAMO** possui camadas finais de classificação um pouco mais complexas. Tais camadas finais se especializam na detecção do tamanho dos objetos ( ou 'bounding boxes' ).

O **MobileNet V2**, um modelo de classificação, foi originalmente treinado no conjunto de dados ImageNet, que contém 1000 classes diferentes. Como pretende-se detectar poucos objetos de geometria simples, a intuição de se utilizar menos parâmetros da rede **Mobilenet V2** se justifica.

No modelo **VOLTAMO VEHICLE**, por processar imagens um pouco maiores ( de dimensão 160 pixels x 160 pixels x 3 cores), o truncamento ocorre na camada '**block\_2\_add**', resultando em uma rede mais compacta. Entretanto, as camadas iniciais e finais são muito maiores ( manipulando um volume maior de informação ). Um maior volume de dados na camada de entrada significa mais operações de leitura da Flash, mais operações de leitura e escrita na PSRAM e mais ciclos de processamento, resultando em taxas FPS menores. Então, neste caso, a motivação para a compactação da rede neural foi a necessidade de obtenção de taxa inferência não inferior a 3 FPS. Com a compactação, o modelo resultante é um pouco menos preciso (KPI F1 menor), mas ainda capaz de produzir resultados aceitáveis.

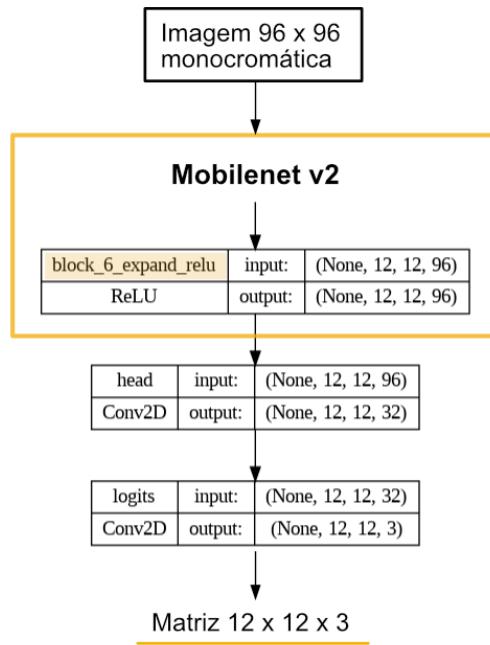
# Detecção de Objetos Através de Redes Neurais do tipo FOMO

---

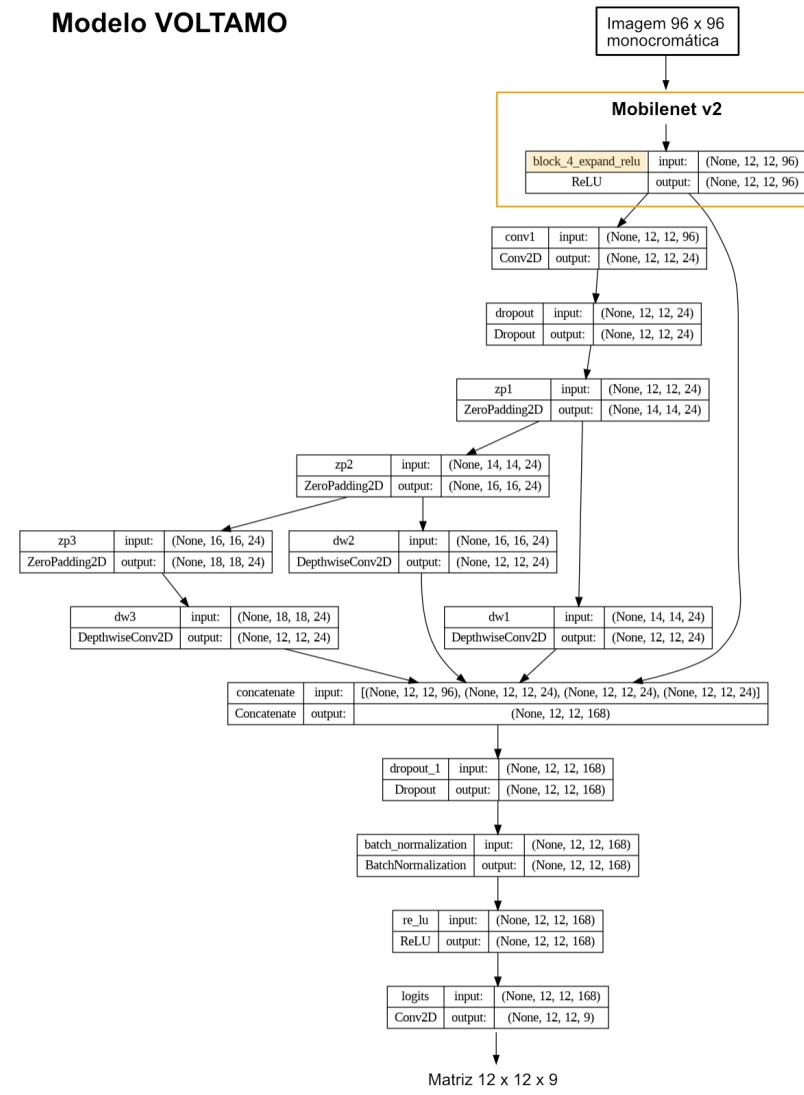
	FOMO	VOLTAMO	VOLTAMO VEHICLE
<b>Camada de truncamento Mobilenet V2</b>	<b>block_6_expand_relu</b>	<b>block_4_expand_relu</b>	<b>block_2_add</b>
<b>Parâmetros Treináveis</b>	<b>20499</b>	<b>14345</b>	<b>4110</b>
<b>Tipo de Detecção</b>	<b>Centróides (X,Y)</b>	<b>Centróides (X,Y) e Tamanho aproximado</b>	<b>Centróides (X,Y) e Tamanho aproximado</b>
<b>Rótulos “one-hot encoding”</b>	( <b>1</b> , 0, 0, 0, 0, 0, 0, 0, 0 ) : classe 0 – background  ( 0, <b>1</b> , 0, 0, 0, 0, 0, 0, 0 ) : classe 1 – pequeno (1x1)  ( 0, 0, <b>1</b> , 0, 0, 0, 0, 0, 0 ) : classe 1 – médio (3x3)  ( <b>1</b> , 0, 0 ) : classe 0 – background  ( 0, <b>1</b> , 0 ) : classe 1  ( 0, 0, <b>1</b> ) : classe 2	( <b>1</b> , 0, 0, 0, 0, 0, 0, 0, 0 ) : classe 0 – background  ( 0, <b>1</b> , 0, 0, 0, 0, 0, 0, 0 ) : classe 1 – veículo (menor)  ( 0, 0, <b>1</b> , 0, 0, 0, 0, 0, 0 ) : classe 1  ( 0, 0, 0, <b>1</b> , 0, 0, 0, 0, 0 ) : classe 1 – grande (5x5)  ( 0, 0, 0, 0, <b>1</b> , 0, 0, 0, 0 ) : classe 1 – extra-grande (7x7)  ( 0, 0, 0, 0, 0, <b>1</b> , 0, 0, 0 ) : classe 2 – pequeno (1x1)  ( 0, 0, 0, 0, 0, 0, <b>1</b> , 0, 0 ) : classe 2 – médio (3x3)  ( 0, 0, 0, 0, 0, 0, 0, <b>1</b> , 0 ) : classe 2 – grande (5x5)  ( 0, 0, 0, 0, 0, 0, 0, 0, <b>1</b> ) : classe 2 – extra-grande (7x7)	( <b>1</b> , 0, 0, 0, 0, 0 ) : classe 0 – background  ( 0, <b>1</b> , 0, 0, 0, 0 ) : classe 1 – veículo (maior)
<b>Dimensão da Entrada</b>	( 96, 96, 1 )	( 96, 96, 1 )	( 160, 160, 3 )
<b>Dimensão da Saída</b>	( 12, 12, 3 )	( 12, 12, 9 )	( 20, 20, 6 )

# Detecção de Objetos Através de Redes Neurais do tipo FOMO

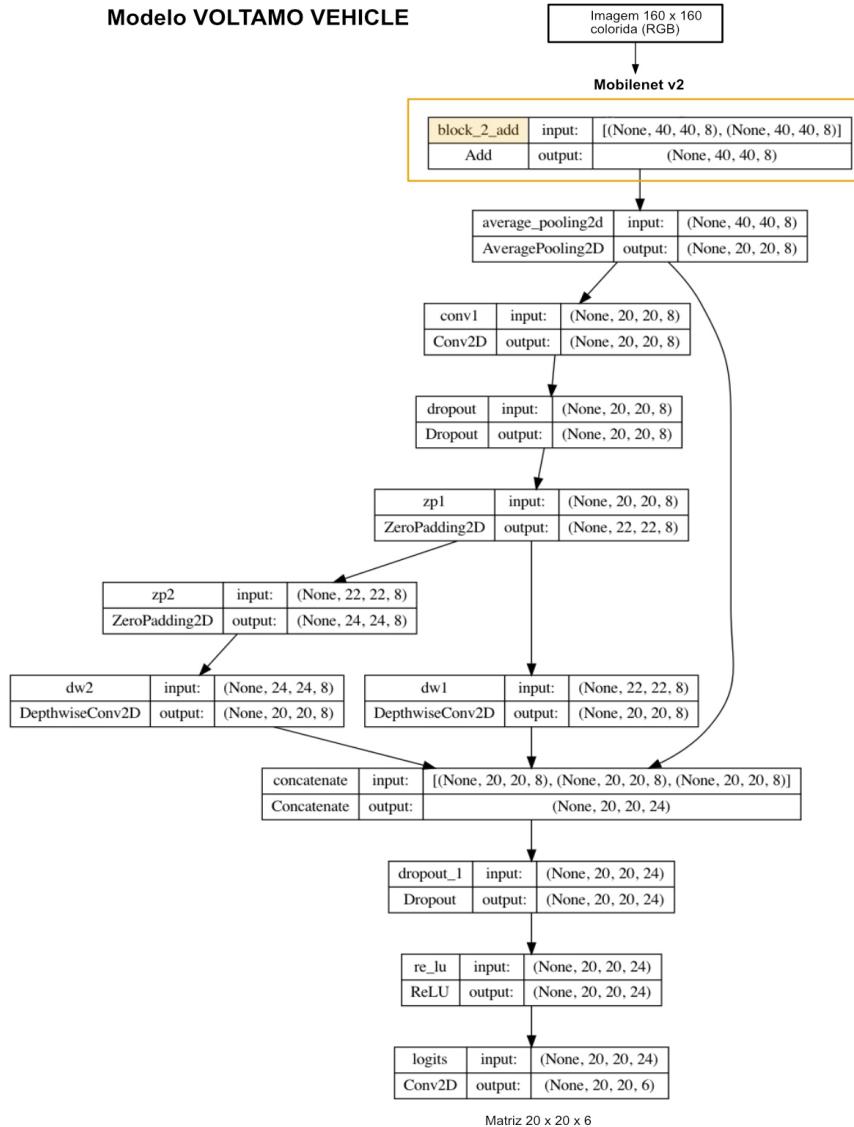
Modelo FOMO



Modelo VOLTAMO

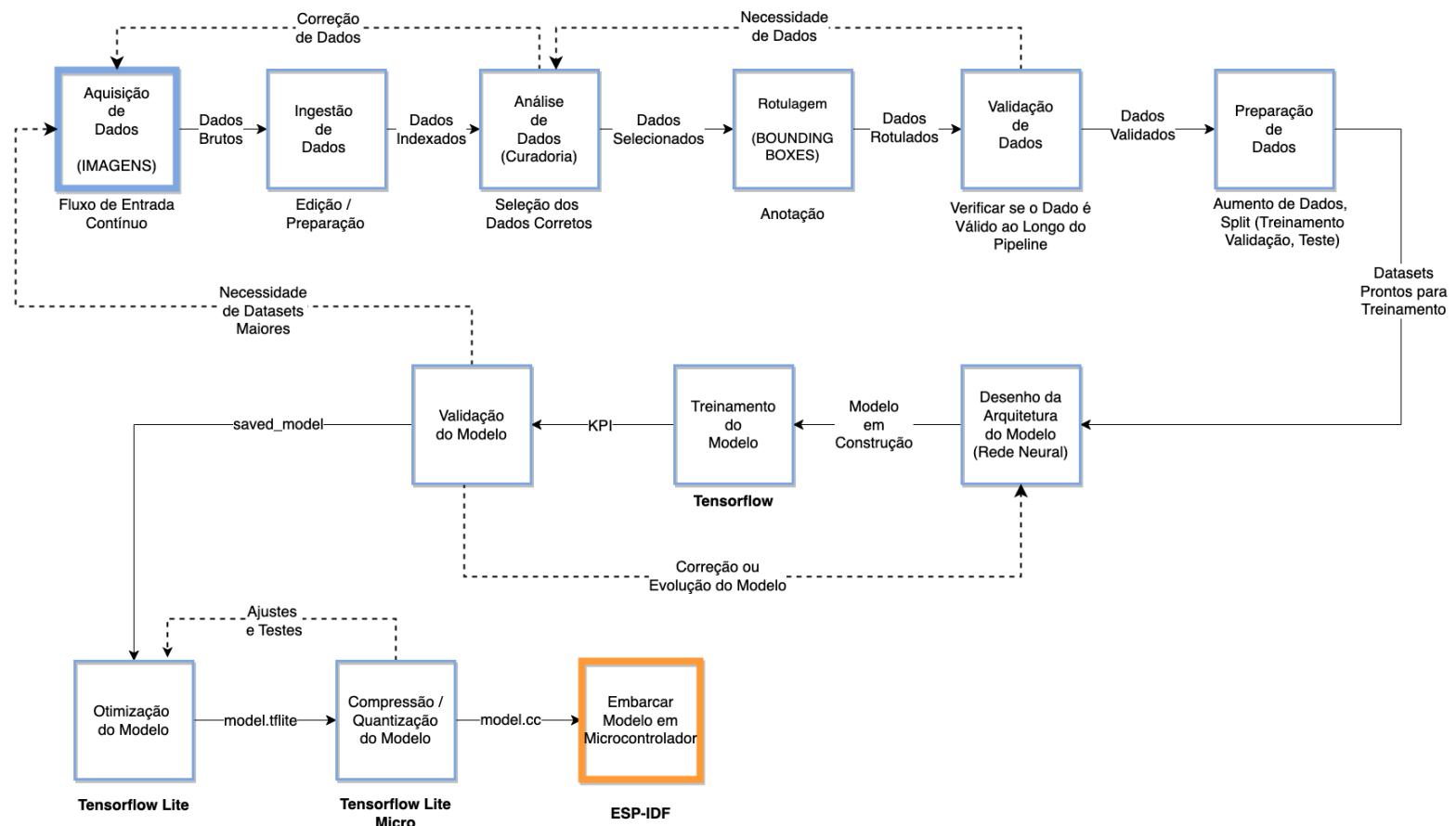


# Detecção de Objetos Através de Redes Neurais do tipo FOMO



# Detecção de Objetos Através de Redes Neurais do tipo FOMO

## Ciclo de Vida do Aprendizado de Máquina do Projeto



# Detecção de Objetos Através de Redes Neurais do tipo FOMO

---

## CICLO DE VIDA DO APRENDIZADO DE MÁQUINA DO PROJETO

### INFRAESTRUTURA DE IA

A INFRAESTRUTURA DE IA (AI INFRASTRUCTURE) geralmente envolve várias etapas em seu ciclo de vida, desde a preparação dos dados até a implantação do modelo. Embora as etapas específicas possam variar dependendo do projeto, as seguintes **quatro etapas** principais são comumente envolvidas na infraestrutura de IA:

### ENGENHARIA DE DADOS (DATA ENGINEERING)

Aquisição e Preparação de Dados: Essa etapa inicial envolve a coleta de dados relevantes para o treinamento e avaliação dos modelos de IA. Os dados podem vir de várias fontes, como bancos de dados, APIs, sensores, **câmeras** ou outros sistemas de informação. Pode ser necessário limpar, pré-processar e transformar os dados para garantir sua qualidade, consistência e compatibilidade com os modelos de IA.

### ENGENHARIA DO MODELO (MODEL ENGINEERING)

Desenvolvimento e Treinamento do Modelo: Nesta etapa, os modelos de IA são projetados, desenvolvidos e treinados usando os dados preparados. Isso envolve a seleção de algoritmos ou arquiteturas adequadas, dividindo os dados em conjuntos de treinamento e validação e otimizando os modelos usando técnicas de aprendizado supervisionado ou não supervisionado. O processo de treinamento inclui etapas iterativas de ajuste dos parâmetros do modelo, avaliação de desempenho e refinamento do modelo até que resultados satisfatórios sejam alcançados.

### IMPLANTAÇÃO DO MODELO (MODEL DEPLOYMENT)

Implantação e Integração do Modelo: Uma vez que o modelo de IA é treinado e validado, ele precisa ser implantado no ambiente de produção, onde pode ser utilizado para gerar previsões ou tomar decisões. Essa etapa envolve a integração do modelo na infraestrutura ou hardware/software existente ou sistema empresarial. Pode ser necessário considerar a escalabilidade, desempenho, segurança e compatibilidade com o ambiente de implantação.

# Detecção de Objetos Através de Redes Neurais do tipo FOMO

## ANÁLISE DE PRODUTO (PRODUCT ANALYTICS)

Monitoramento, Avaliação e Manutenção: Após a implantação do modelo, é crucial monitorar continuamente seu desempenho e avaliar sua eficácia em cenários do mundo real. Isso envolve o acompanhamento de várias métricas, como precisão, latência, utilização de recursos e feedback do usuário (ou processo), para garantir a confiabilidade e o desempenho contínuo do modelo. Além disso, manutenção regular e atualizações podem ser necessárias para lidar com problemas como mudanças de conceito, distribuições de dados em evolução, degradação de sensores ou obsolescência do modelo ao longo do tempo.

Essas **quatro etapas** formam um framework típico para construir e gerenciar a infraestrutura de IA.

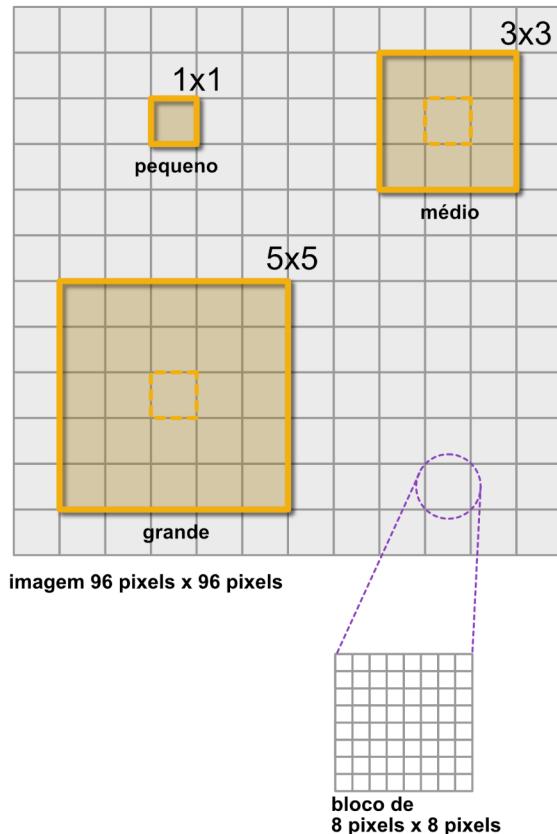
Neste projeto, as **3 primeiras etapas** da INFRAESTRUTURA de IA são parcialmente realizadas :

<b>ENGENHARIA DE DADOS</b>	Coleta de imagens utilizando-se diferentes tipos de câmeras, em diversos cenários, diferentes condições de iluminação e distâncias entre objeto e câmera. Além da coleta de dados, utilizou-se a técnica de pré-processamento para "data augmentation" baseado em rotação de imagem, visando aumentar artificialmente os datasets e maximizar os KPIs de avaliação do modelo ( F1, Precision e Recall ). Os objetos foram rotulados manualmente por "bounding boxes", sendo posteriormente codificados por "one hot" em sub-classes relacionadas ao tamanho dos objetos ( pequeno, médio, grande e extra-grande ).
<b>ENGENHARIA DO MODELO</b>	Criação de modelos customizados de rede neural ( <b>VOLTAMO</b> e <b>VOLTAMO VEHICLE</b> ) que estendem a capacidade do modelo <b>FOMO</b> . Os modelos utilizam uma rede <b>Mobilenet V2</b> (truncada) como 'feature extraction', complementada por camadas que propiciam o inferência dos tamanhos dos objetos ( em blocos de tamanho 1x1, 3x3, 5x5 e 7x7 ).
<b>IMPLEMENTAÇÃO DO MODELO</b>	Um dos requisitos do projeto é a criação de uma rede neural compacta, própria para ser embarcada em um microcontrolador <b>ESP32-S3</b> com quantidades limitadas de Flash e PSRAM. Além disso, buscou-se o equilíbrio entre <u>precisão</u> e <u>taxa FPS</u> aceitáveis em aplicações práticas de visão computacional.

# Detecção de Objetos Através de Redes Neurais do tipo FOMO

## Dicionário de "Bounding Boxes"

grids (1x1), (3x3), (5x5) e (7x7)



## ROTULAGEM DOS DADOS

Os dados de treinamento foram rotulados manualmente ("bounding boxes"), utilizando-se a ferramenta **Edge Impulse** (no caso do modelo **VOLTAMO**), e a ferramenta **LabelImg** (para o modelo **VOLTAMO VEHICLE**).

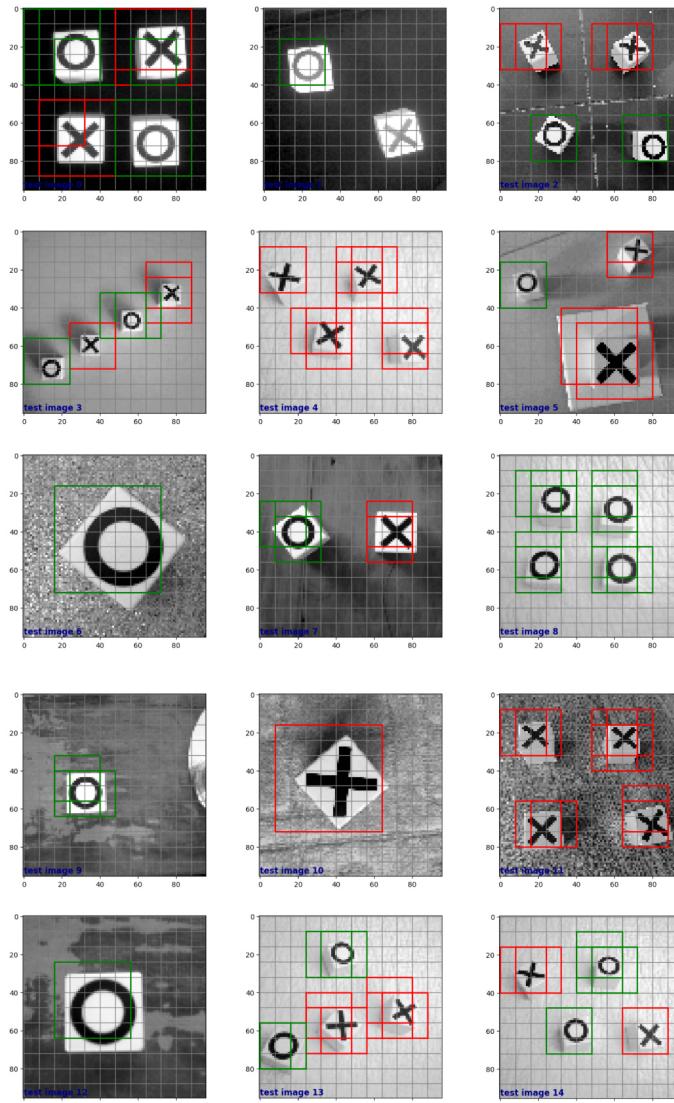
Uma etapa de pré-processamento converte os "bounding boxes" em rótulos do tipo "one-hot encoding" posicionados no centro de cada objeto (centróide).

Conforme o tamanho de cada "bounding box", escolhe-se a classe que melhor representa a dimensão do objeto rotulado (**1x1**, **3x3**, **5x5** ou **7x7**).

# Detecção de Objetos Através de Redes Neurais do tipo FOMO

## INFERÊNCIA SOBRE O DATASET DE TESTE

A imagem ao lado apresenta uma coleção de inferências realizadas pelo modelo **VOLTAMO** sobre um dataset de teste. Cada tipo de objeto detectado é identificado por quadrados verdes ou vermelhos com tamanhos variados ( de acordo com o dicionário de “bounding boxes” ).



# Detecção de Objetos Através de Redes Neurais do tipo FOMO

---

## VÍDEOS DE DEMONSTRAÇÃO

Foram criados dois vídeos de demonstração do processamento das redes neurais (**VOLTAMO** e **VOLTAMO VEHICLE**) no microcontrolador **ESP32-S3** da placa **ESP32-S3-EYE**.

O vídeo da rede neural **VOLTAMO** demonstra a detecção de cubos plásticos demarcados com “X” ou “O” em ambiente interno. A placa **ESP32-S3-EYE** captura as imagens e realiza as inferências de detecção de objetos localmente. O firmware da placa também implementa um **servidor HTTP** e um **encoder MJPEG**, podendo realizar o streaming das inferências (imagens e “bounding boxes” sobre os objetos detectados) a um navegador de internet (Chrome) via WiFi. As inferências são realizadas sobre imagens monocromáticas em resolução 96x96 pixels, a uma taxa de **5 FPS**. No display da placa **ESP32-S3-EYE** e no streaming de imagens, apresenta-se as imagens originais coloridas (RGB) captadas pela câmera, em resolução 192x192, com os “bounding boxes” pintados em verde ou vermelho.

O vídeo da rede neural **VOLTAMO VEHICLE** demonstra o resultado de 4 testes realizados em ambiente externo, capturando imagens do tráfego de veículos sob viadutos. A rede neural foi otimizada (com ajustes e treinamentos sucessivos) para produzir uma taxa de **3 FPS**. Esta rede

neural processa imagens coloridas em resolução 160x160, tornando o processamento mais intenso.

Outras variantes do mesmo modelo (**VOLTAMO VEHICLE**), contendo mais camadas, foram treinadas e testadas. Porém, tais variantes mais complexas produziram resultados mais precisos (KPI F1 maiores) com taxas FPS inferiores. A detecção de objetos em tempo real em taxas FPS maiores exigiria o uso de microcontroladores mais rápidos que o **ESP32-S3**.

O **ESP32-S3** possui 2 núcleos de CPU com clock máximo de 240 MHz. Um teste futuro interessante será distribuir as inferências entre os 2 núcleos, em 2 processos distintos. Entretanto, ambos compartilhariam as mesmas memórias **Flash** e **PSRAM**, o que pode ser um gargalo (“bottleneck”) de desempenho. Testes poderão confirmar se é vantajoso distribuir as inferências desta maneira.

Alguns novos modelos de módulo **ESP32-S3** (série **WROOM 2**) possuem octal **SPI FLASH**, o que pode tornar a execução do firmware mais rápida. Testar a execução dos modelos neste tipo de módulo também será interessante.