Relatório 28 - Trabalho Final do Bootcamp: monte você mesmo

Guilherme Loan Schneider

1. Introdução do Projeto

O projeto final escolhido para o meu Projeto Final do Bootcamp do LAMIA consiste em uma abordagem de identificação automatizada de chassi com visão computacional para o reconhecimento de veículos.

A Figura abaixo demonstra a arquitetura do modelo de reconhecimento proposto e que foi desenvolvido neste trabalho, bem como um fluxograma das atividades que relacionam as fases do projeto.

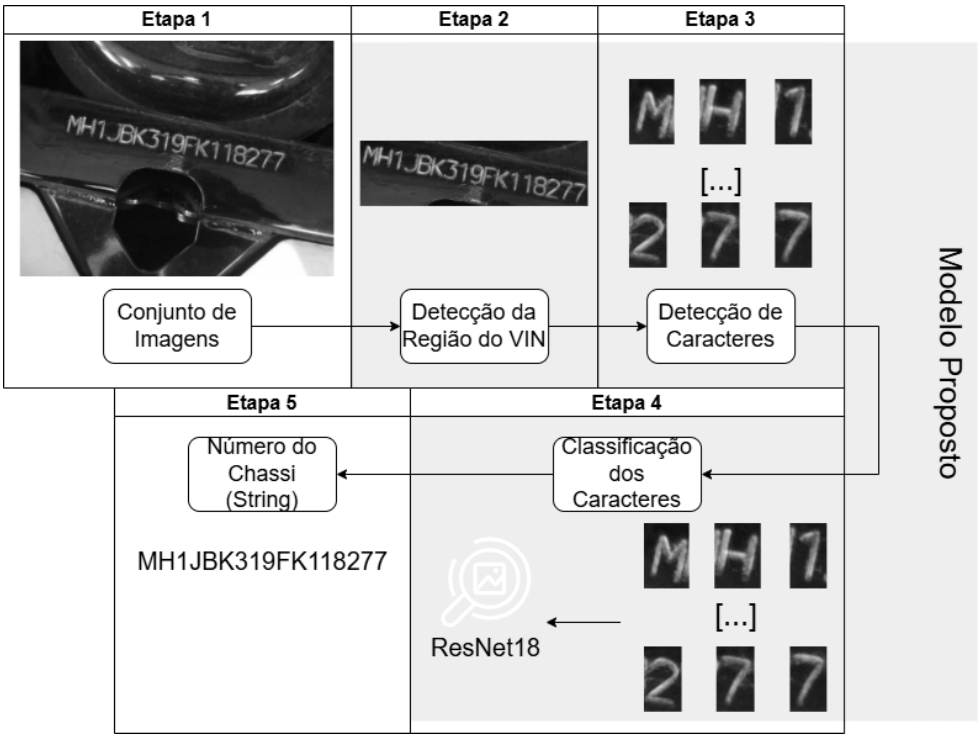


Figura 1. Descrição das etapas do projeto

O processo de localização, identificação e classificação de um Número de Identificação de Veículos (VIN) se dá pelas seguintes sequências de etapas:

1. A primeira etapa tem objetivo de receber a imagem, ou um conjunto de imagens, e aplicar o pré-processamento definido com técnicas como: conversão para escala de cinza, equalização de histograma com CLAHE, aplicação de Gaussian Blur e por fim Sharpening para realce de bordas;
2. Em seguida, a detecção da área de interesse (os 17 caracteres alfanuméricos) é realizada, onde será feito um corte contendo apenas o VIN visível. Nessa etapa, um modelo YOLOv11 é utilizado especificamente para esse processo;
3. Após o recorte, essa etapa foca na detecção e segmentação dos 17 caracteres alfanuméricos. Nessa etapa, um segundo modelo YOLOv11 é utilizado especificamente para esse processo;
4. Essa etapa consiste na classificação dos caracteres obtidos na etapa anterior. 17 imagens de caracteres são passadas para uma ResNet18 que fará a classificação de cada um;

5. Por fim, será retornado ao usuário, em formato de texto, o VIN encontrado.

Note que os caracteres I, O e Q não estão presentes no trabalho visto que a norma NBR 6066 e ISO 3779 delimitam que esses caracteres não são permitidos em números de chassi de veículos.

2. Datasets utilizados

O dataset montado para este trabalho é fruto de três datasets coletados no website Roboflow. Para as etapas 2, 3 e 4 descritas na Figura 1, serão empregados conjuntos de imagens diferentes para cada.

O primeiro dataset consiste em imagens do VIN de veículos com a anotação de onde ele se encontra na imagem, similarmente a Figura abaixo. Note que esse dataset foi montado a partir do CVAT, de forma manual pelo autor do projeto, que permite fazer anotações em imagens para modelos de visão computacional. Isso acaba impactando na reprodutibilidade do projeto.

O segundo dataset é similar ao anterior, onde haverá a anotação individual de cada caractere presente no VIN, a fim de treinar um segundo modelo YOLOv11 para efetuar a detecção e classificação preliminar. A Figura abaixo demonstra essa diferença.

O terceiro e último dataset é utilizado para treinar a ResNet18 presente no final do pipeline. Aqui foi utilizado as imagens do segundo dataset (17 caracteres anotados), onde é feito o recorte de cada caractere individual e posteriormente salvo em uma pasta com a sua respectiva classificação (Pasta “0”: caractere “0”, Pasta “1”: caractere “1”, ...).

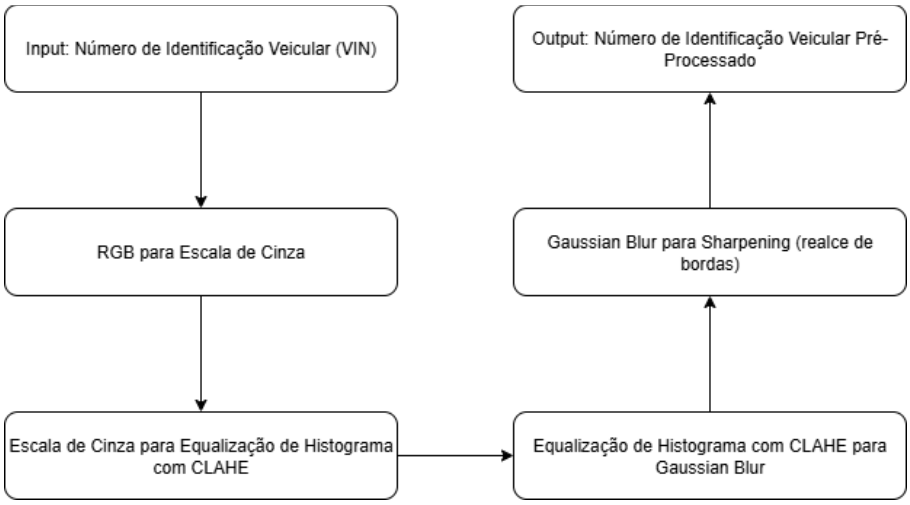
2.1 Cuidados com os datasets

Ao analisar os datasets e verificar as imagens presentes, podemos verificar alguns tipos incorretos de dados, que nos trarão problemas mais pra frente. É o caso das imagens abaixo, onde a Figura a esquerda representa um VIN com um tamanho fora do padrão, de 10 caracteres. A Figura a direita representa um número de motor, que não segue as normas delimitadas pela NBR 6066 e ISO 3779.

Para isso, é necessário que sejam feitos tratamentos na base de dados antes de treinar os modelos. Para isso, filtramos apenas imagens que possuem 17 caracteres presentes na imagem, eliminando assim essas imagens indesejadas.

3. Pré-processamentos utilizados

Para lidar com problemas frequentes em imagens, como reflexos, sombras, sujeiras, iluminação ruim, dentre outros, o projeto apresenta um pipeline para minimizar erros de leitura, detecção e classificação das imagens. A Figura abaixo demonstra os pré-processamentos utilizados.

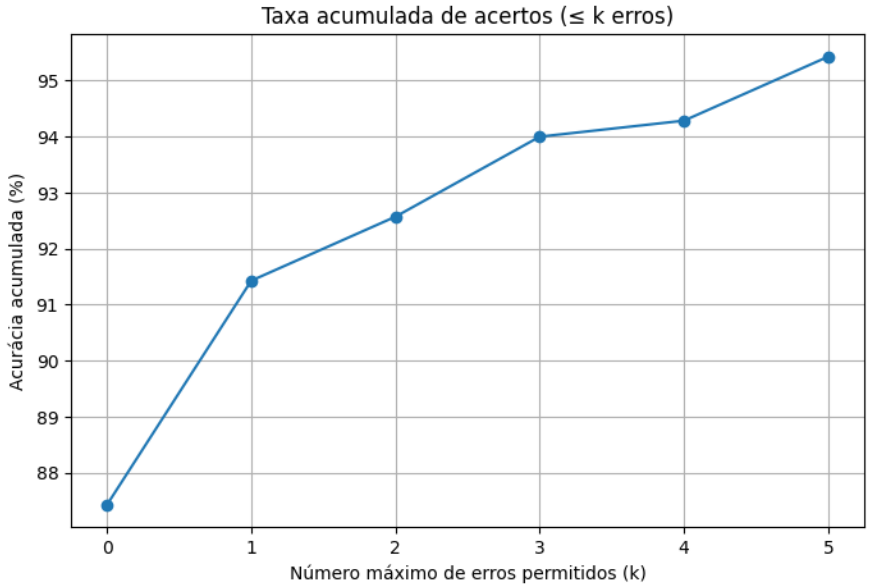


A Figura abaixo demonstra a aplicação desse pré-processamento (A primeira figura a esquerda é a imagem original, e quanto mais à direita, os processos são aplicados).

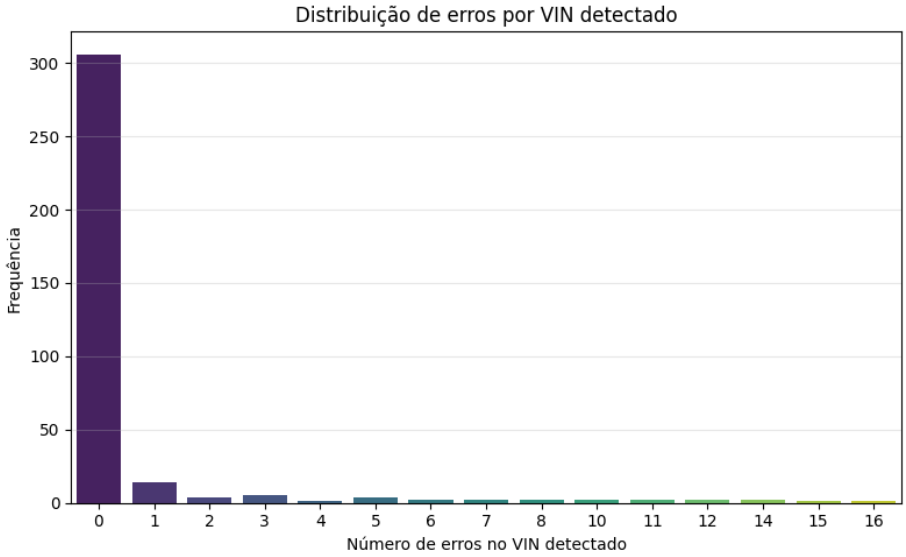
5. Resultados

O projeto proposto atingiu acurácia acumulada de aproximadamente 88% na leitura, sem erros, de um conjunto de VINs selecionados aleatoriamente no dataset com as

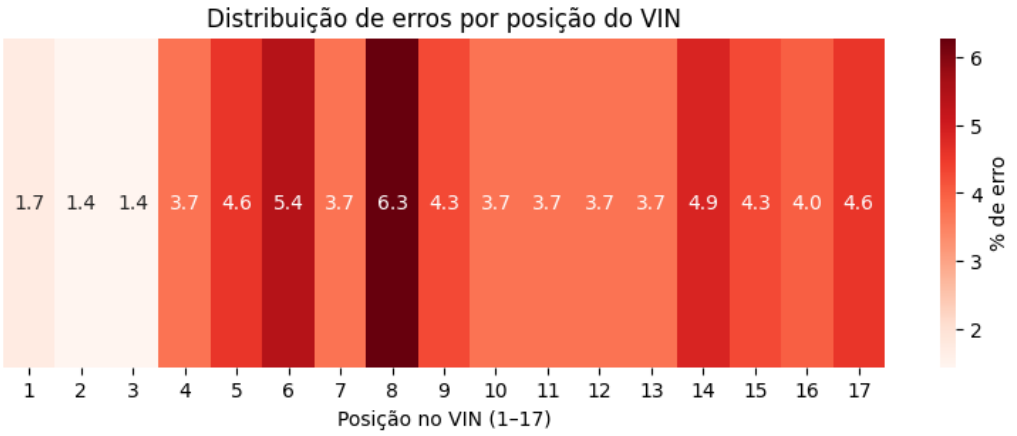
anotações de cada caracter. Com a tolerância de 1 erro de leitura, podemos verificar que existe um salto na acurácia para mais de 91%, isso indica que podemos utilizar uma verificação após a leitura para indicar a correção daquele determinado caractere. A Figura abaixo demonstra a acurácia atingida pelo modelo.



Na Figura abaixo podemos analisar a distribuição de erros do pipeline para essa execução. Podemos perceber que existe um pico de frequência com zero erros, indicando que o nosso modelo consegue apresentar bom desempenho na leitura quando comparado com tolerância de 1 até 5 erros (região que se concentra a maior frequência de erros).



Outro caso que podemos verificar é a posição do VIN que o modelo mais apresenta erros de leitura. Isso pode indicar alguma falta de dados para algum caractere presente do dataset, como um desbalanceamento. A Figura abaixo demonstra a distribuição de erros por posição do VIN, que se concentra principalmente na 8ª posição (note que estamos falando de porcentagem de erro).



5.1 Resultados dos modelos utilizados

Nessa seção daremos enfoque aos resultados obtidos em cada modelo individualmente. O primeiro deles é um modelo YOLOv11 utilizado para efetuar o reconhecimento da área de interesse (ROI) do VIN e posteriormente recorte da região. O segundo modelo é também da família YOLOv11, mas treinado especificamente para detectar e classificar os caracteres presentes na figura recebida pelo modelo anterior. Por fim, uma ResNet18 é treinada para efetuar a classificação em conjunto com o modelo de caracteres e posteriormente retornar ao usuário em formato de texto.

5.1.1 Modelo YOLOv11 – Detecção e recorte da ROI

Nesse modelo, utiliza-se um treinamento com 27 épocas, resolução de imagens de 640x640, tamanho do batch de 16, parâmetro “augment” como True e um modelo YOLOv11 de tamanho small, conseguimos atingir as seguintes métricas com o modelo:

- Precisão: 0.9709;
- Recall: 0.9759;
- mAP@50: 0.9925;
- mAP@50-95: 0.8608;
- Fitness: 0.8740.

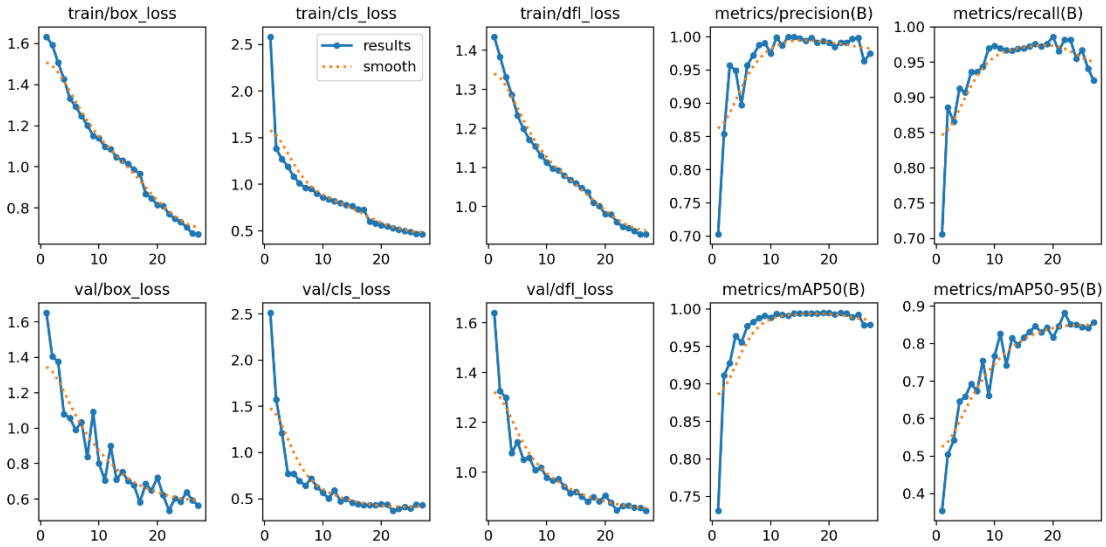
Os resultados obtidos com esse modelo foram muito bons, com algumas métricas chegando próximo a 1.0, indicando que o modelo é altamente eficaz na identificação e localização da região de interesse (ROI) correspondente ao VIN.

As métricas de precision e recall, próximo a 1.0, indicam que o modelo comete poucos falsos positivos e falsos negativos, respectivamente — ou seja, quase todas as detecções realizadas realmente correspondem a um VIN, e praticamente todos os VINs presentes nas imagens são detectados.

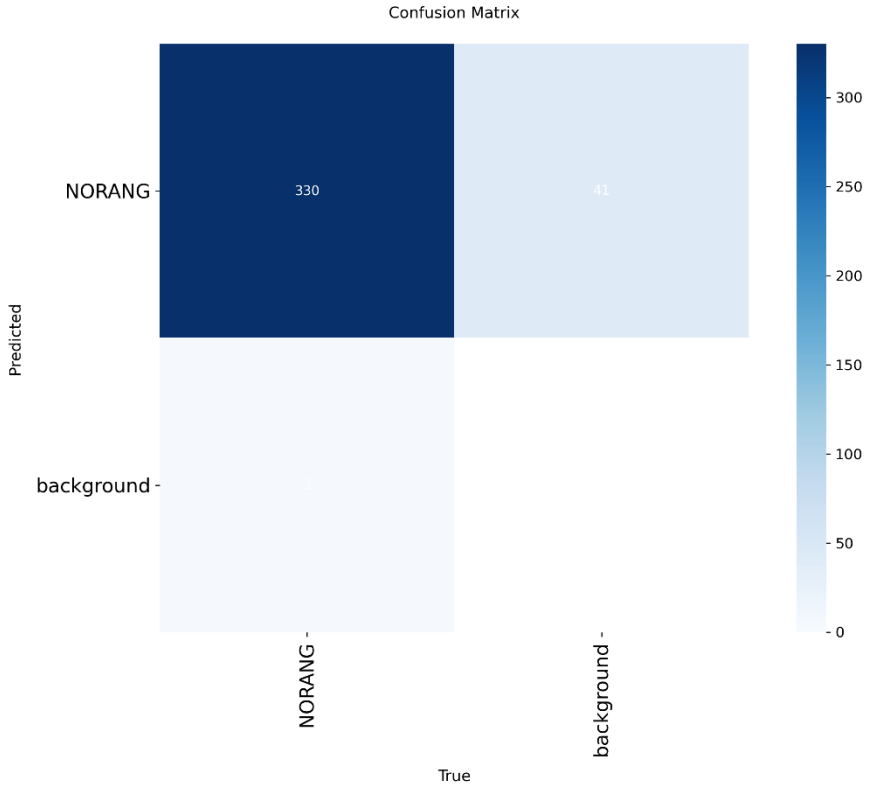
O valor de mAP@50 de 0.9925 mostra que o modelo consegue localizar com grande precisão as ROIs corretas. Já o mAP@50–95, de 0.8608, embora mais rigoroso, ainda apresenta desempenho elevado, evidenciando boa consistência da detecção em múltiplos limiares de confiança (IoU).

A Figura abaixo demonstra também alguns outros valores que são utilizados no treinamento do modelo, como “box_loss”, que indica o erro do modelo perante ao formato da caixa, se ela não está grande demais ou pequena demais, de acordo com o ground truth. O “cls_loss” indica se o erro de classificação de um VIN, como classificar um background como um VIN, ou vice-versa. Já o “dfl_loss” é utilizado para refinar a

predição das coordenadas das caixas delimitadoras, ele faz isso aprendendo uma distribuição de probabilidades para cada borda da caixa.



A matriz de confusão abaixo demonstra o desempenho obtido pelo modelo. Curiosamente, os casos negativos não estão sendo mostrados nessa matriz de confusão, mas eles existem e compõem cerca de 20-40% das imagens utilizadas.



5.1.2 Modelo YOLOv11 – Detecção e classificação de caracteres

Admitindo um treinamento com 80 épocas, resolução das imagens de 640x640, tamanho do batch de 16 e um modelo YOLOv11 de tamanho small, conseguimos atingir as seguintes métricas com o modelo:

- Precisão: 0.9512;
- Recall: 0.8936;
- mAP@50: 0.9552;
- mAP@50-95: 0.7077;
- Fitness: 0.7325.

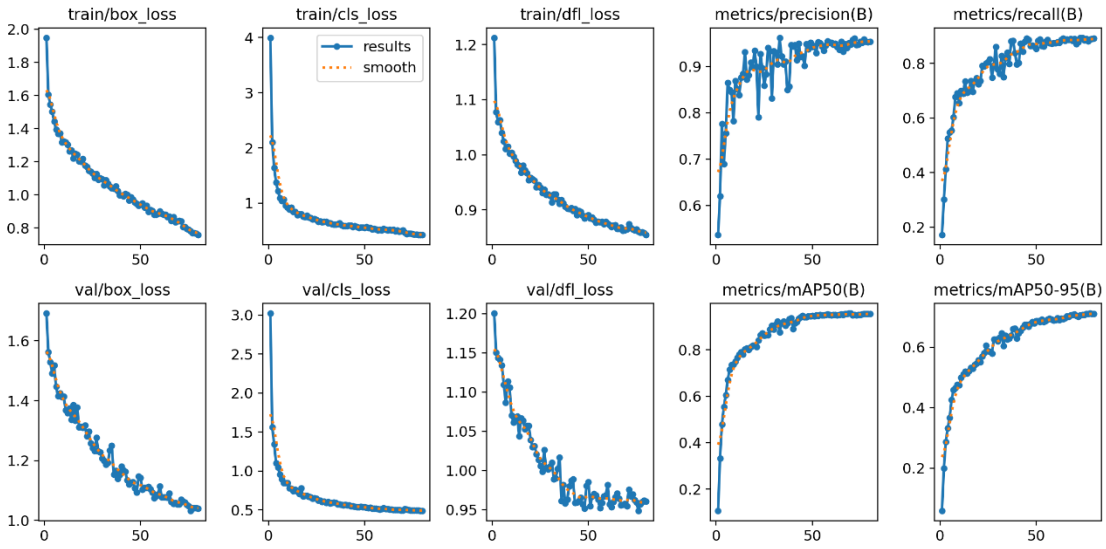
A precisão de 0.9512 indica que, dentre todas as predições positivas realizadas, mais de 95% correspondem a detecções corretas. Isso demonstra que o modelo tem alta capacidade de evitar falsos positivos, ou seja, raramente detecta caracteres onde não há nenhum.

O valor de Recall 0,8936 mostra que aproximadamente 89% dos caracteres reais foram corretamente identificados pelo modelo. Embora levemente inferior à precisão, esse valor ainda é considerado alto, o que reflete uma boa capacidade de detectar a maioria dos caracteres presentes nas imagens. Pequenas perdas de recall podem estar relacionadas à presença de imagens com baixo contraste, ruído ou caracteres parcialmente ilegíveis.

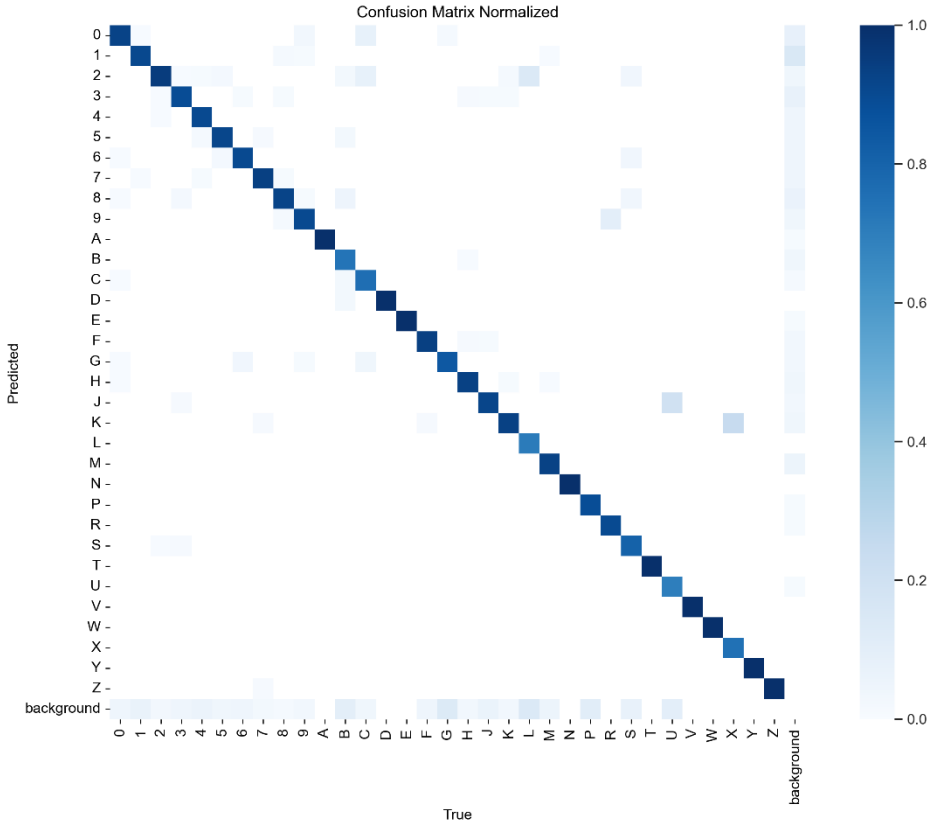
O mAP@50 de 0,9552 demonstra que o modelo apresenta excelente desempenho geral de detecção quando o limiar de interseção entre as caixas preditas e as reais (IoU) é de 50%. Já o mAP@50-95 de 0,7077 — que é uma métrica mais rigorosa por considerar limiares de IoU entre 50% e 95% — mostra que o modelo mantém um bom desempenho mesmo sob valores de confiança maiores.

Por fim, o fitness de 0,7325 resume as principais métricas de desempenho (precisão, recall e mAP), indicando que o modelo se encontra com bom desempenho e estabilidade, capaz de generalizar bem sobre dados não vistos, mas ainda existem melhorias a serem feitas.

De modo geral, os resultados demonstram que o modelo treinado é eficaz na detecção e classificação de caracteres VIN, com boa precisão e um recall suficientemente alto para garantir a confiabilidade do pipeline. Pequenas melhorias poderiam ser obtidas por meio de ajustes de hiperparâmetros, aumento da diversidade do dataset (particularmente em casos de baixa luminosidade ou ângulos distintos) e o uso de técnicas adicionais de regularização ou fine-tuning.



Por fim, temos então a matriz de confusão normalizada do modelo, que apresenta uma diagonal principal muito destacada, indicando que o modelo consegue realizar a classificação preliminar dos caracteres de forma razoável, com poucos erros. Além disso, é possível verificar também que o modelo apresenta algumas confusões ainda com o fundo da imagem e alguns caracteres em específico, como o K e X, U e J, devido a sua similaridade e questões da própria fotografia (sujeira, reflexo, dentre outros).

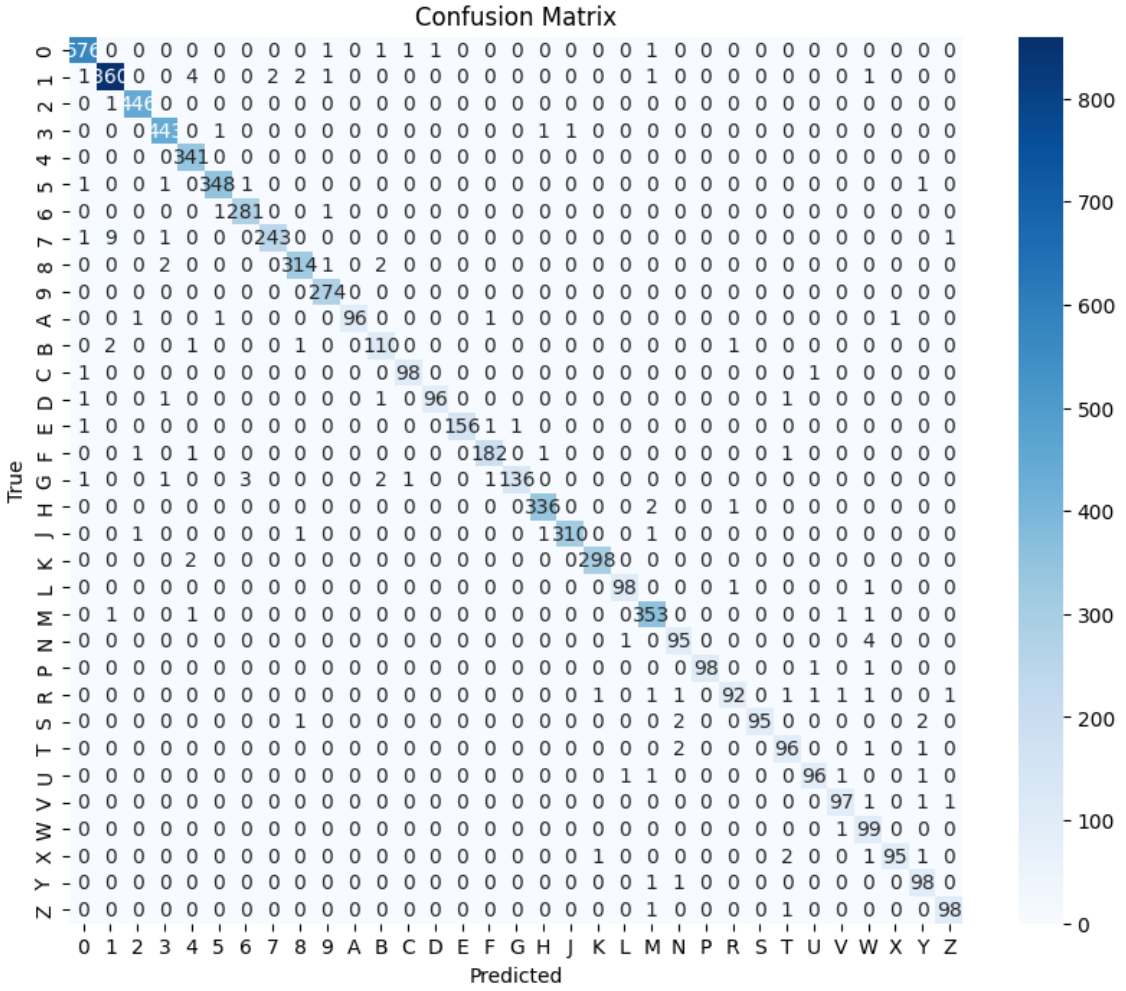


5.1.3 ResNet18 – Classificação e refinamento

Esse modelo é voltado para o refinamento em conjunto com o modelo anterior a fim de melhorar o desempenho do pipeline e torna-lo menos suscetível a erros. Atualmente, o modelo atinge 97.77% de precisão macro (média das precisões por classe) e loss de aproximadamente 0.05.

Note que esse modelo também foi utilizado a técnica de data-augmentation, para ser possível minimizar casos de overfitting e até mesmo permitir maior variabilidade de iluminação, rotação, brilho, contraste, das imagens que serão enviadas para o modelo.

A matriz de confusão abaixo demonstra as previsões do modelo. Podemos verificar que a maior confusão presente é entre o caractere 1 sendo classificado erroneamente com o caractere 7, com um total de 9 confusões.



6. Conclusão

Com esse trabalho, foi possível concluir que os modelos apresentados neste relatório obtiveram resultados muito bons perante a dificuldade que é de conseguir efetuar, com precisão, a leitura de um VIN a partir de uma imagem. É de saber de todos que quando falamos de fotografias, é inerente a presença de ruídos, sombras, desfoques, dentre outros, que dificultam a leitura de um caractere presente em uma imagem.

No entanto, apesar dos resultados satisfatórios, ainda existem melhorias a serem feitas no projeto, como, principalmente a melhora das métricas do segundo modelo. Como se trata de uma imagem com anotações individuais de caracteres, não é possível realizar o data augmentation aplicando rotações, visto que essa anotação não representará mais exatamente o caractere, sendo apenas possível aplicar alterações visuais.

Além disso, esse modelo (modelo de caracteres) apresenta algumas situações difíceis de serem lidadas, como a ausência de leitura de algum caractere em qualquer posição do VIN. Alguns testes foram feitos e o resultado é inconsistente quando alteramos o modelo de detecção da ROI.

Fiquei muito contente com os resultados, como falei ainda existem pontos a serem melhorados, mas a atual situação já me agrada bastante ainda mais quando comparados com os primeiros resultados que obtive.