

Equipe 5 - Detecção de Face

Alunos:

Felipe

Giovanna

Lucas

Tatiane

Sumário

- Introdução
- Metodologia
- Resultados
- Conclusões

Introdução

RESUMO DO PROBLEMA

- A detecção de faces em imagens;
- Desafios para garantir precisão e confiabilidade;
- Amplas aplicações em segurança, autenticação e interações digitais;
- Implementação de métodos e modelos distintos para alcançar uma maior eficiência.

Introdução

PORQUE ESSE PROBLEMA?

- A crescente demanda de sistemas de reconhecimento facial:
 - Eficiência de sistemas de vigilância;
 - Autenticação e reconhecimento facial.
- Desafios para a eficácia desses sistemas:
 - Variação do ambiente de detecção;
 - Diversidade facial;
 - Baixa qualidade da imagem.

Introdução

OBJETIVO GERAL

- Capacidade de localizar rostos em imagens e produzir resultados precisos;
- Detecção em condições variáveis;
- Integração de sistemas de reconhecimento facial no cotidiano;
- Desenvolver modelos de detecção de objetos utilizando a arquitetura YOLO (You Only Look Once) para identificar objetos em imagens;
- Efetuar séries de testes para comparar diretamente os benefícios e desvantagens proporcionados pelas distintas variações do modelo e seus parâmetros.

Metodologia: Componentes Principais

- **BASE DE DADOS**

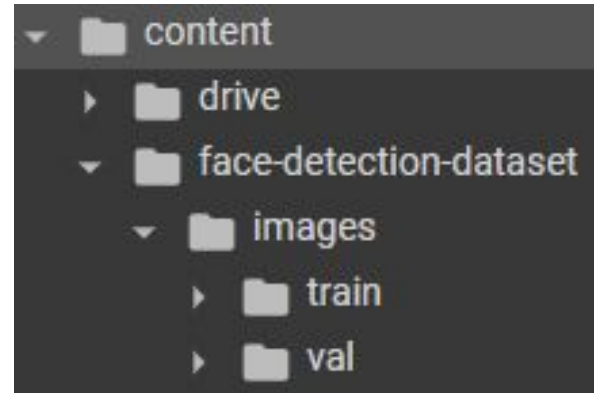
- Fonte: "fareselmenshawii/face-detection-dataset";
- Classe única especificada como ['face']:

Base: Original

- Treinamento: 13.400 imagens;
- Validação: 3.347 imagens.

Base: Reduzida

- Treinamento: 2.280 imagens;
- Validação: 564 imagens.



Metodologia: Configuração Inicial

CONFIGURAÇÃO PADRÃO DO MODELO ORIGINAL:

```
MODEL_ARCH = 'yolo_nas_l'  
DEVICE = 'cuda' if torch.cuda.is_available() else "cpu"  
BATCH_SIZE = 8  
MAX_EPOCHS = 20  
CHECKPOINT_DIR = f'/kaggle/working/'  
EXPERIMENT_NAME = f'yolo_nas_face'
```

CONFIGURAÇÃO PADRÃO DO MODELOS ALTERADOS:

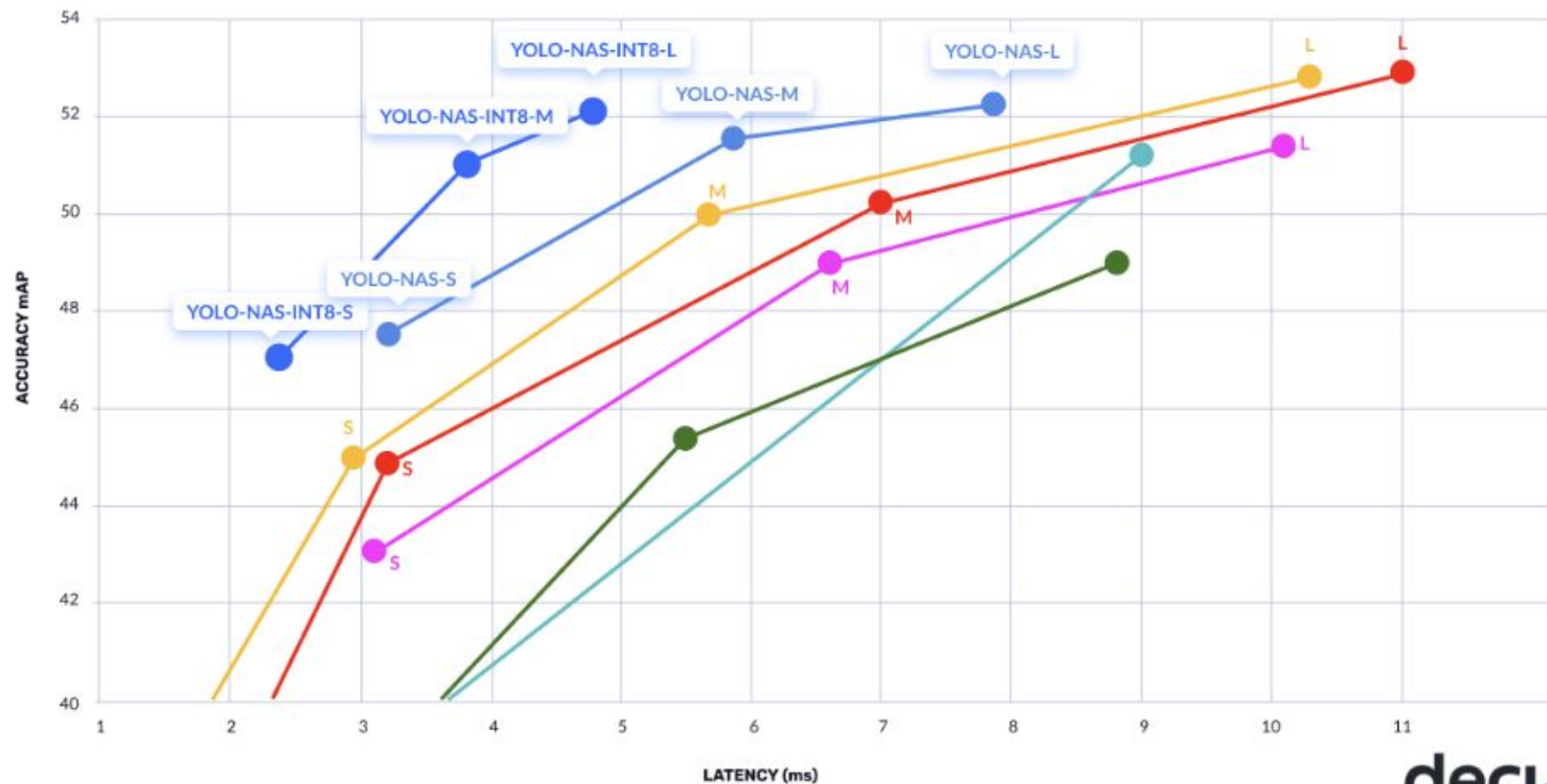
```
MODEL_ARCH = 'yolo_nas_l' #l/s/m  
DEVICE = 'cuda' if torch.cuda.is_available() else "cpu"  
BATCH_SIZE = 16  
MAX_EPOCHS = 35  
CHECKPOINT_DIR = f'/content/avantiData/kaggle/working/'  
EXPERIMENT_NAME = f'yolo_nas_face'
```

Metodologia: Arquitetura do Modelo

O modelo **YOLO-NAS** é pré-treinado em conjuntos de dados como COCO e Objects365, o que o torna adequado para aplicações do mundo real. Atualmente está disponível no **SuperGradients** da Deci , que é uma biblioteca baseada em PyTorch que contém cerca de 40 modelos pré-treinados para realizar diferentes tarefas de visão computacional, como classificação, detecção, segmentação, etc.

Efficient Frontier of Object Detection on COCO, Measured on NVIDIA T4

YOLO-NAS-FP16 YOLO-NAS-INT8 YOLO-V6-3.0 YOLO-V8 PPYOLOE YOLO-V7 YOLO-V5



Metodologia

TIPOS DE MODELOS:

O modelo **YOLO-NAS** contém três distintos tamanhos de modelos únicos implementados no projeto, visando oferecer diferentes medidas entre a precisão e a latência do programa, sendo eles:

| MODELO | YOLO-NAS-S | YOLO-NAS-M | YOLO-NAS-L |
|----------|----------------|----------------|-----------------|
| LATÊNCIA | Maior rapidez | Rapidez média | Menor rapidez |
| PRECISÃO | Menor precisão | Precisão média | Melhor precisão |

Metodologia

TIPOS DE OTIMIZADORES:

Adicionalmente às distintas configurações de modelos presentes no **SuperGradients**, a biblioteca distintos tipos de otimizadores testados comparativamente no projeto, oferecendo diferentes benefícios e desvantagens de acordo com suas especificações:

| OTIMIZADOR | ADAM | ADAM W | SGD |
|--------------|---|---|--|
| VANTAGENS | Automaticamente ajusta o learning rate | Melhor que o Adam no quesito de generalização | Simples e altamente utilizado (Melhor generalização) |
| DESVANTAGENS | Requer alterações cuidadosas em seus parâmetros | Requer muita memória em certos pontos | Sensível ao learning rate |

Metodologia

PARÂMETROS DE TREINAMENTO ALTERADOS:

Learning rate (lr): Controla o tamanho de passos durante o treinamento do modelo. Para isso, o CosineLRScheduler garante uma convergência mais suave, refinando a eficácia do treinamento;

Média Exponencial Móvel (EMA): É uma técnica de suavização usada em séries de dados, dando mais importância aos mais recentes e promovendo previsões e resultados mais estáveis;

Otimizadores: Algoritmos que ajustam os parâmetros de um modelo conforme a progressão do treinamento, desempenhando um papel crucial na otimização e estabilização do modelo.

TREINAMENTO PADRÃO DO MODELO ORIGINAL:

```
train_params = {
    'silent_mode': False,
    'average_best_models': True,
    'warmup_mode': "linear_epoch_step",
    'warmup_initial_lr': 1e-6,
    'lr_warmup_epochs': 3,
    'initial_lr': 5e-4,
    'lr_mode': "cosine",
    'cosine_final_lr_ratio': 0.1,
    'optimizer': "Adam",
    'optimizer_params': {"weight_decay": 0.0001},
    'zero_weight_decay_on_bias_and_bn': True,
    'ema': True,
    'ema_params': {"decay": 0.9, "decay_type": "threshold"},
    'max_epochs': MAX_EPOCHS,
    'mixed_precision': True,
    'loss': PPYOloELoss(
        use_static_assigner=False,
        num_classes=len(dataset_params['classes']),
        reg_max=16
    ),
}
```

TREINAMENTO PADRÃO DO MODELO ORIGINAL:

```
"valid_metrics_list": [  
    DetectionMetrics_050(  
        score_thres=0.1,  
        top_k_predictions=300,  
        num_cls=len(dataset_params['classes']),  
        normalize_targets=True,  
        post_prediction_callback=PPYoloEPostPredictionCallback(  
            score_threshold=0.01,  
            nms_top_k=1000,  
            max_predictions=300,  
            nms_threshold=0.7  
        )  
    )  
],  
"metric_to_watch": 'mAP@0.50'  
}
```

TREINAMENTO PADRÃO DE UM DOS MODELO ALTERADO:

```
train_params = {
    'silent_mode': False,
    "average_best_models": True,
    "warmup_mode": "LinearBatchLRWarmup",
    "warmup_initial_lr": 1e-6,
    "lr_warmup_epochs": 3,
    "initial_lr": 5e-4,
    "lr_mode": "CosineLRScheduler",
    "cosine_final_lr_ratio": 0.1,
    "optimizer": "AdamW",
    "optimizer_params": {"weight_decay": 0.0001},
    "zero_weight_decay_on_bias_and_bn": True,
    "ema": True,
    "ema_params": {"decay": 0.9999, "decay_type": "exp", "beta": 15},
    "max_epochs": MAX_EPOCHS,
    "mixed_precision": True,
    "loss": PPYoloELoss(
        use_static_assigner=False,
        num_classes=len(dataset_params['classes']),
        reg_max=16
    ),
}
```

TREINAMENTO PADRÃO DE UM DOS MODELO ALTERADO:

```
"valid_metrics_list": [  
    DetectionMetrics_050(  
        score_thres=0.1,  
        top_k_predictions=300,  
        num_cls=len(dataset_params['classes']),  
        normalize_targets=True,  
        post_prediction_callback=PPYoloEPostPredictionCallback(  
            score_threshold=0.01,  
            nms_top_k=1000,  
            max_predictions=300,  
            nms_threshold=0.5,  
        )  
    )  
],  
"metric_to_watch": 'mAP@0.50'  
}
```

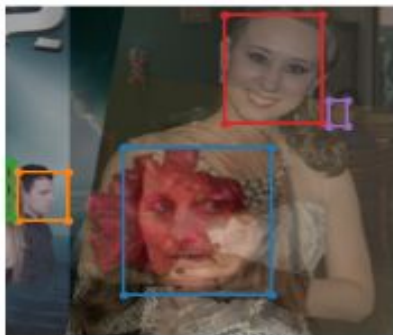
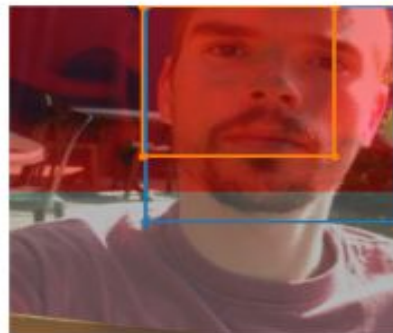
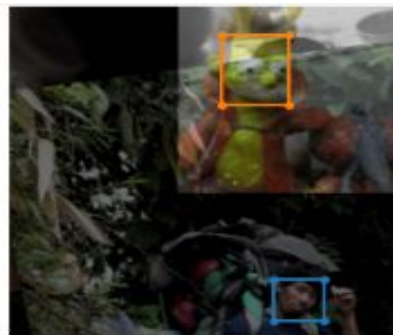

Metodologia

Esse script, em resumo, configura o treinamento de um modelo de detecção de objetos usando a biblioteca *SuperGradients*, definindo todos os parâmetros necessários para o processo de treinamento, desde a configuração do otimizador até a escolha da métrica de avaliação. O treinamento foi realizado utilizando essas configurações iniciais e, como demonstrado, foi alterado para avaliar a eficácia e comportamentos distintos propagadas pela inclusão dos novos parâmetros em um novo treinamento.

Metodologia: Pre-Processamento

MÉTRICAS - REFINAMENTO:

- Mistura (Mixup):
 - Combina duas imagens, ajustando os pesos das imagens e suas etiquetas correspondentes;
 - Ajuda a diversificar o conjunto de dados e melhorar o desempenho do modelo.
- Afinação (Augmentation):
 - Várias operações de aumento, como rotação, escala e recorte, para criar variações nas imagens de treinamento.
- HSV (Hue, Saturation, Value):
 - Informações de cor (matiz, saturação e valor) ajustadas para criar mais variações nas imagens.
- Normalização:
 - Garante que os valores dos pixels estejam na mesma escala.



Metodologia: Treinamento

MÉTRICAS DO MODELO:

loss_cls (Loss de Classificação): Refere-se à perda associada à classificação das caixas delimitadoras;

loss_iou (Loss de IoU): Indica a perda associada à sobreposição (Intersection over Union) entre as caixas delimitadoras previstas e as caixas delimitadoras reais;

loss_dfl (Loss de DFL - Dynamic Feature Learning): Representa a perda associada ao aprendizado de recursos dinâmicos;

loss (Loss Total): É a soma ponderada das perdas anteriores.

Metodologia: Treinamento

MÉTRICAS DO MODELO:

Precision@0.50: Representa a precisão do modelo, ou seja, a proporção de caixas previstas corretamente;

Recall@0.50: Indica a proporção de caixas reais que foram corretamente previstas em relação ao número total de caixas reais;

Map@0.50 (Mean Average Precision): É a média da precisão para diferentes valores de limiar de sobreposição;

F1@0.50: É a média harmônica da precisão e do recall.

=====

SUMMARY OF EPOCH 0

└ Train

- └ Ppyoloeloss/loss_cls = 1.7471
- └ Ppyoloeloss/loss_iou = 0.8271
- └ Ppyoloeloss/loss_dfl = 0.8337
- └ Ppyoloeloss/loss = 3.408

└ Validation

- └ Ppyoloeloss/loss_cls = 1.79
- └ Ppyoloeloss/loss_iou = 0.7266
- └ Ppyoloeloss/loss_dfl = 0.7995
- └ Ppyoloeloss/loss = 3.3161
- └ Precision@0.50 = 0.0346
- └ Recall@0.50 = 0.0
- └ Map@0.50 = 0.0124
- └ F1@0.50 = 0.0

=====

Validating epoch 19: 100%|██████████| 63/63 [00:29<

[2023-11-11 03:18:20] INFO - base_sg_logger.py - Ch

[2023-11-11 03:18:20] INFO - sg_trainer.py - Best cl

=====

SUMMARY OF EPOCH 19

└─ Train

| | |
|--|--|
| └─ Ppyoloeloss/loss_cls = 0.7163 | |
| └─ Epoch N-1 = 0.7341 (↘ -0.0178) | |
| └─ Best until now = 0.733 (↘ -0.0167) | |
| └─ Ppyoloeloss/loss_iou = 0.4577 | |
| └─ Epoch N-1 = 0.4688 (↘ -0.0111) | |
| └─ Best until now = 0.4642 (↘ -0.0065) | |
| └─ Ppyoloeloss/loss_dfl = 0.4594 | |
| └─ Epoch N-1 = 0.4641 (↘ -0.0048) | |
| └─ Best until now = 0.4608 (↘ -0.0014) | |
| └─ Ppyoloeloss/loss = 1.6333 | |
| └─ Epoch N-1 = 1.667 (↘ -0.0337) | |
| └─ Best until now = 1.658 (↘ -0.0247) | |

Validation

Ppyoloeloss/loss_cls = 0.6754

Epoch N-1 = 0.6844 (↘ -0.009)

Best until now = 0.6812 (↘ -0.0059)

Ppyoloeloss/loss_iou = 0.4573

Epoch N-1 = 0.453 (↗ 0.0044)

Best until now = 0.453 (↗ 0.0044)

Ppyoloeloss/loss_dfl = 0.4546

Epoch N-1 = 0.4568 (↘ -0.0021)

Best until now = 0.4568 (↘ -0.0021)

Ppyoloeloss/loss = 1.5874

Epoch N-1 = 1.5941 (↘ -0.0068)

Best until now = 1.5941 (↘ -0.0068)

Precision@0.50 = 0.0843

Epoch N-1 = 0.0963 (↘ -0.012)

Best until now = 0.1384 (↘ -0.0541)

Recall@0.50 = 0.9449

Epoch N-1 = 0.937 (↗ 0.0079)

Best until now = 0.9456 (↘ -0.0007)

Map@0.50 = 0.877

Epoch N-1 = 0.8675 (↗ 0.0095)

Best until now = 0.8739 (↗ 0.0031)

F1@0.50 = 0.1548

Epoch N-1 = 0.1746 (↘ -0.0198)

Best until now = 0.2404 (↘ -0.0855)

Resultados

Visualização












de

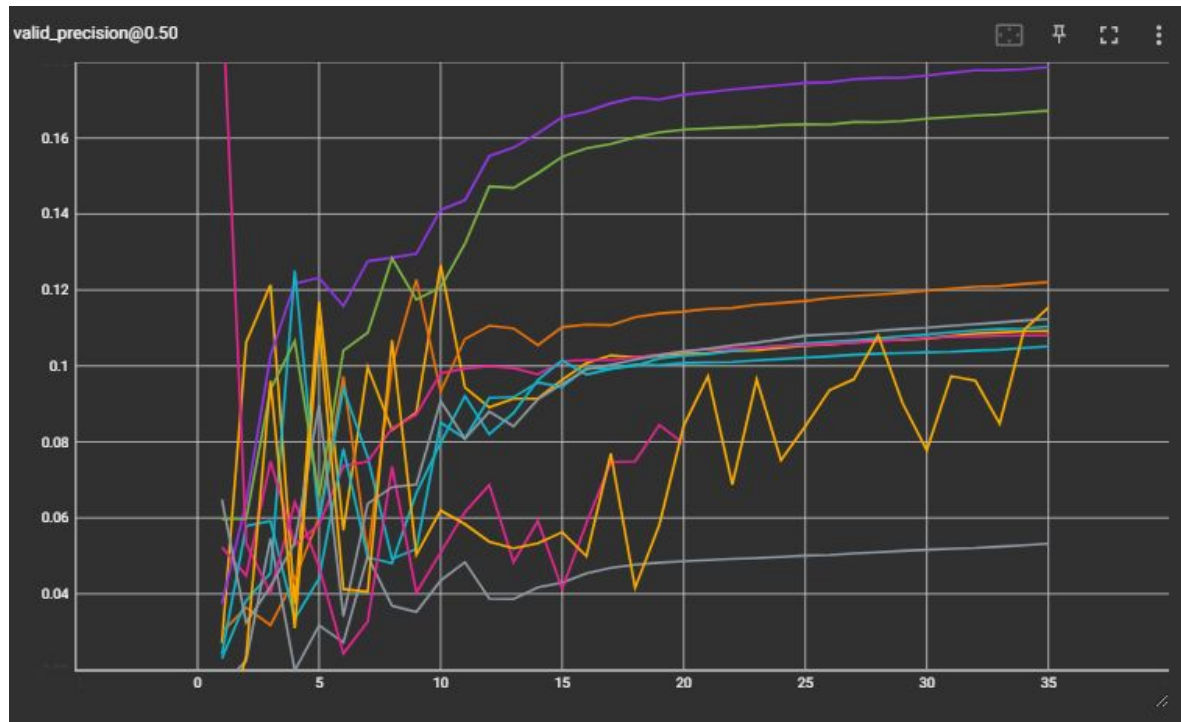
Resultados:

O Tensorboard é uma ferramenta de visualização e monitoramento desenvolvida pelo TensorFlow, uma biblioteca popular de aprendizado de máquina. Ele é usado para visualizar graficamente o treinamento de modelos de machine learning, demonstrando as métricas, gráficos e informações relevantes, por meio de uma interface interativa, para os olhos do usuário.

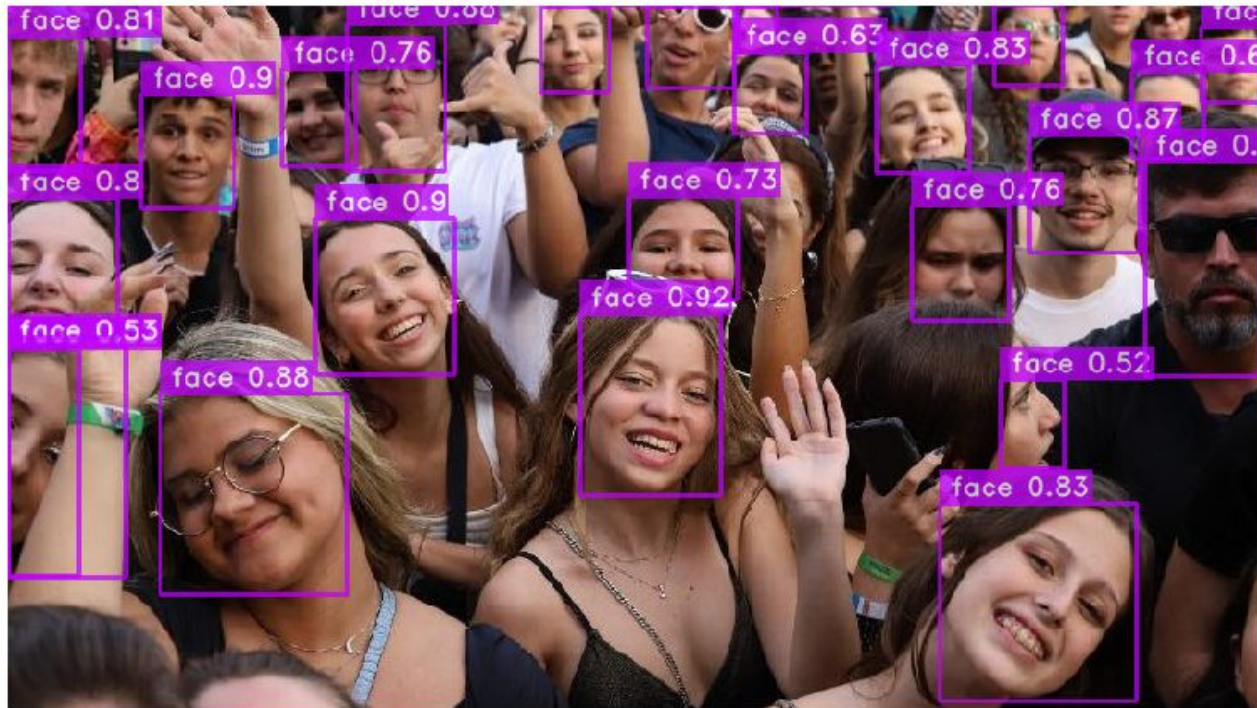


TensorBoard

| | | |
|-------------------------------------|--------------------------------|--|
| <input checked="" type="checkbox"/> | RUN_S_ADAM/RUN_S_ADAM |  |
| <input checked="" type="checkbox"/> | RUN_S_ADAMW/RUN_S_ADAMW |  |
| <input checked="" type="checkbox"/> | RUN_S_SGD/RUN_S_SGD |  |
| <input checked="" type="checkbox"/> | RUN_M_ADAM/RUN_M_ADAM |  |
| <input checked="" type="checkbox"/> | RUN_L_SGD/Yolo_L_SGD/RUN_L_SGD |  |
| <input checked="" type="checkbox"/> | RUN_M_SGD/RUN_M_SGD |  |
| <input checked="" type="checkbox"/> | RUN_L_ADAM/RUN_L_ADAM |  |
| <input checked="" type="checkbox"/> | RUN_L_ADAMW/RUN_L_ADAMW |  |
| <input checked="" type="checkbox"/> | RUN_M_ADAMW/RUN_M_ADAMW |  |
| <input checked="" type="checkbox"/> | RUN_ORIGINAL/RUN_ORIGINAL |  |
| <input checked="" type="checkbox"/> | RUN_BATCH/RUN_BATCH |  |

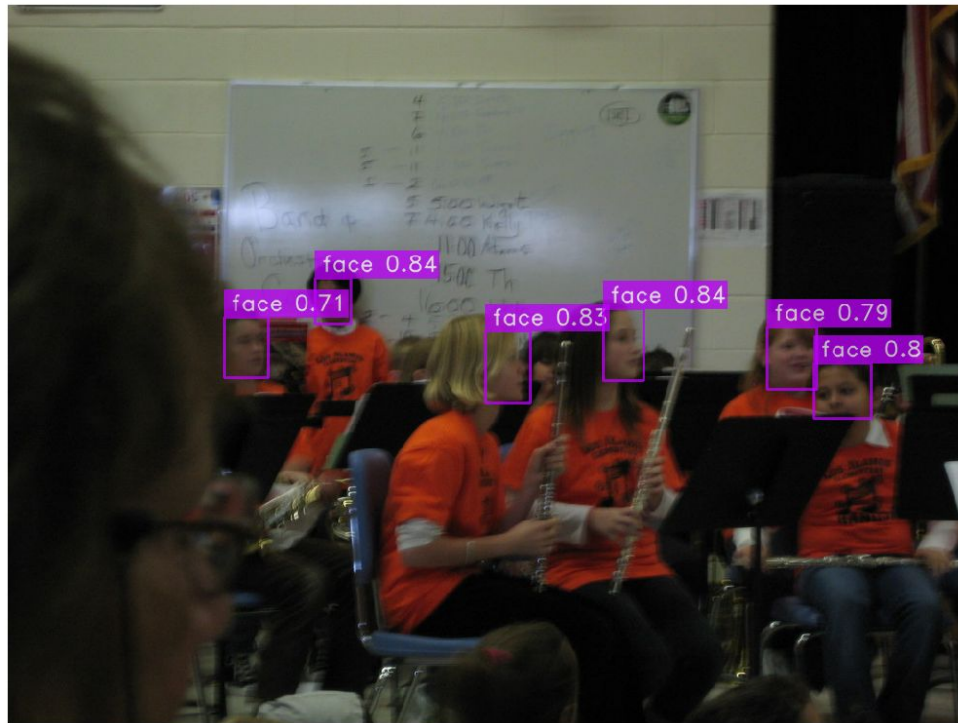


Resultados



Resultados

PARÂMETROS ORIGINAIS:



Resultados

PARÂMETROS ALTERADOS:

<https://imgur.com/a/AYwiAef>

Resultados

PARÂMETROS ORIGINAIS:



Resultados

PARÂMETROS ALTERADOS:

<https://imgur.com/a/yGrffhF>

Conclusões

- O modelo demonstrou uma detecção satisfatória de rostos, As estratégias avançadas de treinamento e a escolha da arquitetura YOLO NAS-L juntamente com o as mudanças de parâmetro, como o otimizador, contribuíram para o sucesso do projeto.
- Para melhores resultados:
 - Variação do BATCH_SIZE e MAX_EPOCHS;
 - Ajustes nos parâmetros de treinamento;
 - Escolha do otimizador ideal: SGD;
 - Escolha do modelo ideal: Yolo_nas_L.

Referências Bibliográficas

FACE Detection - YOLO-NAS. 15 ago. 2023. Disponível em: <https://www.kaggle.com/code/mohamedchahed/face-detection-yolo-nas/>. Acesso em: 6 dez. 2023.

INTRO - Deci AI Documentation Hub. Disponível em: <https://docs.deci.ai/super-gradients/latest/documentation/source/welcome.html>. Acesso em: 6 dez. 2023.

WHICH OPTIMIZER should I use for my ML Project? Disponível em: <https://www.lightly.ai/post/which-optimizer-should-i-use-for-my-machine-learning-project>. Acesso em: 6 dez. 2023.