CASE – BOLSISTA GRADUADO – VISÃO COMPUTACIONAL - 2025

Desenvolvimento de um Pipeline de Experimentação para Classificação de Posturas Humanas

Nome Completo: Anna Luiza Gomes da Silva

Data: 25/05/2025

1. Introdução

A presente Atividade 2 (AT2) dá seguimento à Atividade 1, na qual um conjunto de dados foi criado e analisado. O foco principal desta etapa foi o desenvolvimento de um pipeline de experimentação para a classificação de posturas humanas. O objetivo central consistiu em comparar pelo menos duas metodologias distintas para a extração de características de pose, seguido pelo treinamento e avaliação de diversos algoritmos de classificação. A finalidade era identificar o modelo com o desempenho superior na tarefa de reconhecer as posturas "sentado", "em pé" ou "caminhando/em movimento".

Este relatório detalha a metodologia empregada, as arquiteturas selecionadas para a extração de características (YOLOv8-Pose e MediaPipe), os modelos de classificação testados, as métricas de avaliação utilizadas e uma análise comparativa dos resultados obtidos. A conclusão deste processo culminou na escolha do modelo final para a tarefa.

2. Metodologia de Modelagem e Experimentação

O pipeline de experimentação foi cuidadosamente estruturado para permitir uma comparação sistemática entre diferentes abordagens de detecção de pose e algoritmos de classificação. O fluxo de trabalho geral compreendeu as seguintes fases:

2.1. Conjunto de Dados

O conjunto de dados utilizado foi o `processed_mpii_poses`, gerado na AT1. Este conjunto já estava subdividido em subconjuntos de treino, validação e teste, e organizado nas classes "sitting", "standing" e "walking".

2.2. Abordagens de Extração de Características de Pose

Duas abordagens distintas para a extração de pontos-chave (keypoints) da pose humana a partir das imagens foram implementadas e avaliadas:

- **Abordagem 1: YOLOv8-Pose (experiments/model_1_yolov8_pose/)**
    - Utilizou-se o modelo pré-treinado `yolov8s-pose.pt` da Ultralytics para detectar a pose humana e extrair as coordenadas (x, y) e a confiança dos pontos-chave.
    - O script `extract_yolo_features.py` processou as imagens do dataset, extraindo os keypoints da pessoa com maior confiança em cada imagem e

armazenando essas características em formato `.npy` no diretório `results/model_1_features/`. As características foram normalizadas e achatadas para formar vetores de entrada para os classificadores.

- **Abordagem 2: MediaPipe Pose (experiments/model_2_media_pipe/)**
    - A solução MediaPipe Pose da Google foi empregada para a detecção de landmarks corporais.
    - O script `extract_mediapipe_features.py` (nome inferido com base na estrutura de diretórios) processou as imagens, extraindo as coordenadas (x, y, z) e a visibilidade dos 33 landmarks fornecidos pelo MediaPipe. As características foram, de forma similar, normalizadas, achatadas e salvas em formato `.npy` no diretório `results/model_2_mediapipe_features/`.

2.3. Treinamento e Seleção de Modelos de Classificação (models/models.py)

Com as características de pose extraídas por cada uma das abordagens, diversos algoritmos de classificação clássicos foram treinados e avaliados. O script `models/models.py` automatizou este processo, que envolveu:

1. **Carregamento das Características:** Leitura dos arquivos `.npy` contendo as features de cada abordagem.
2. **Divisão dos Dados:** Utilização dos conjuntos de treino, teste e validação previamente definidos.
3. **Treinamento de Classificadores:** Teste com os classificadores Random Forest, Logistic Regression, SVC e Gradient Boosting.
4. **Avaliação de Desempenho:** Cálculo de métricas como Acurácia, Precisão, Recall e F1-Score (macro e por classe) para cada combinação de extrator de features e classificador.
    4.1. A função *get_best_model* é a principal responsável por testar diferentes classificadores (RandomForest, LogisticRegression, SVC, GradientBoosting).
    4.2. Para cada classificador, ela o treina exclusivamente com os dados de treino (X_train_yolo, y_train_yolo ou X_train_mp, y_train_mp).
    4.3. Em seguida, ela faz predições nos dados de validação* (X_val_yolo, y_val_yolo ou X_val_mp, y_val_mp).
    4.4. A métrica f1_score(y_val, y_pred, average='macro') é calculada usando os resultados da predição no conjunto de validação.
    4.5. O classificador que obtiver o maior Macro F1-Score no conjunto de validação é considerado o best_model dentro dessa função para aquela abordagem (YOLO ou MediaPipe).
5. **Seleção dos Melhores Modelos:** Identificação do classificador com melhor desempenho (com base no F1-Score macro) para cada abordagem de extração de features. Os modelos selecionados foram salvos como arquivos `.pkl` (ex: `pose_classifier_yolo_best.pkl` e `pose_classifier_mediapipe_best.pkl`).
6. **Geração de Gráficos:** Criação de visualizações comparativas das métricas,

armazenadas em `results/graphs/`.

2.4. Métricas de Avaliação

O desempenho dos modelos foi avaliado através das seguintes métricas:

- **Acurácia:** Proporção de previsões corretas.
- **Precisão (por classe e macro):** Capacidade do modelo de não classificar incorretamente amostras negativas como positivas.
- **Recall (Sensibilidade, por classe e macro):** Capacidade do modelo de identificar todas as amostras positivas.
- **F1-Score (por classe e macro):** Média harmônica entre Precisão e Recall, oferecendo uma medida balanceada. O Macro F1-Score foi a métrica principal para a seleção do melhor modelo geral, devido ao leve desbalanceamento das classes observado na AT1.

3. Resultados e Análise Comparativa

Os resultados detalhados dos experimentos, incluindo as métricas para todos os classificadores testados com cada conjunto de features, podem ser explorados interativamente no notebook `notebooks/models.ipynb` e visualizados nos gráficos presentes em `results/graphs/`.

3.1. Desempenho dos Modelos com Features YOLOv8-Pose (Validacao)

Formula para comparação:
Macro Avg F1 = (F1_sitting + F1_standing + F1_walking) / 3

Modelo	**Macro F1-Score**
GradientBoosting	*0.830051*
RandomForest	0.805441
SVC	0.583097
LogisticRegression	0.582745

O classificador com o melhor desempenho ao utilizar as características extraídas pelo YOLOv8-Pose foi o Gradient Boosting, alcançando um Macro F1-Score de 0.82.

3.2. Desempenho dos Modelos com Features MediaPipe Pose (Validacao)

Formula para comparação:
Macro Avg F1 = (F1_sitting + F1_standing + F1_walking) / 3

Modelo	**Macro F1-Score**
GradientBoosting	*0.792356*
RandomForest	0.715625
SVC	0.640619
LogisticRegression	0.633246

Ao utilizar as características extraídas pelo MediaPipe Pose, o classificador Gradient Boosting novamente demonstrou o melhor desempenho, atingindo um Macro F1-Score de 0.78.

3.3. Análise Comparativa e Seleção do Modelo Final (Teste)

A Tabela 1 sumariza o desempenho dos melhores modelos para cada abordagem de extração de características:

Abordagem Extratora	**Melhor Classificador**	**Macro Acurácia**	**Macro Precisão**	**Macro Recall**	**Macro F1-Score**
YOLOv8-Pose	Gradient Boosting	0.84	0.87	0.77	0.79
MediaPipe Pose	Gradient Boosting	0.82	0.82	0.75	0.77

Tabela 1: Comparação dos melhores modelos por abordagem de extração de features.

Observa-se que a abordagem que utilizou o YOLOv8-Pose para a extração de características, combinada com o classificador Gradient Boosting, apresentou o melhor desempenho geral, com um Macro F1-Score de 0.79.

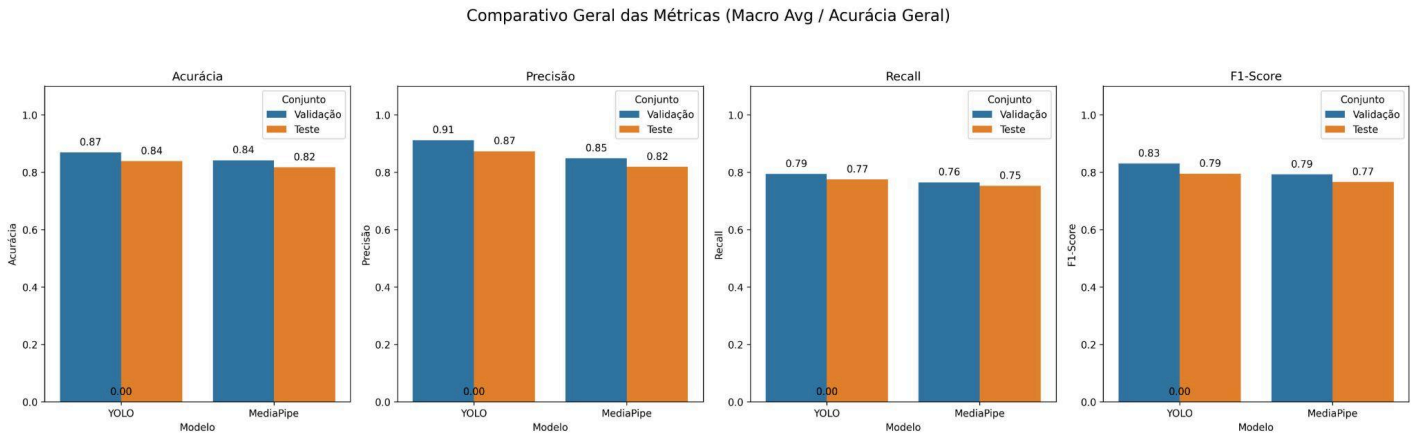

Figura 1 - Comparativo geral das métricas (Validação e Teste)

O modelo YOLO consistentemente supera o MediaPipe em todas as métricas agregadas (Acurácia Geral, Macro Precisão, Macro Recall, Macro F1-Score), tanto no conjunto de validação quanto no de teste.

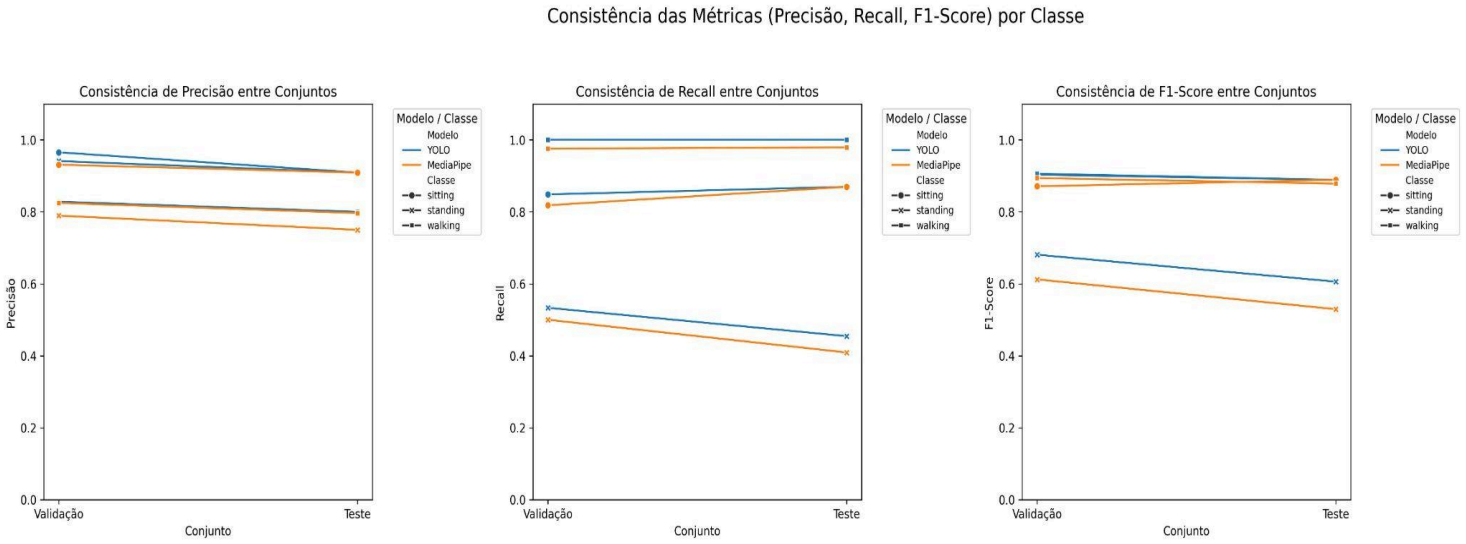

Figura 2 - Consistência das métricas por classe e modelo

Ambas as linhas (YOLO e MediaPipe) mostram uma pequena queda da Acurácia da Validação para o Teste, o que é esperado e indica que os modelos não estão sofrendo de overfitting severo em relação à acurácia geral. YOLO parece marginalmente superior e mais consistente em termos de acurácia geral.

3.4. Modelo Selecionado para AT3

Com base nos resultados da experimentação, o modelo selecionado para ser utilizado na Atividade 3 (Desenvolvimento do sistema com API de monitoramento) é o classificador Gradient Boosting treinado com as características de pose extraídas pelo YOLOv8-Pose, salvo como models/pose_classifier_yolo_best.pkl. Este modelo apresentou o maior Macro

F1-Score (0.79), indicando o melhor equilíbrio geral entre precisão e recall nas três classes de postura.

4. Estrutura dos Entregáveis (AT2)

Os entregáveis para esta atividade foram organizados da seguinte forma no repositório do projeto:

- Códigos dos experimentos:

disponivel em https://github.com/annalug/AT2_modeling_experimentation
experiments/model_1_yolov8_pose/extract_yolo_features.py: Script para extração de features com YOLOv8-Pose.
experiments/model_2_media_pipe/extract_mediapipe_features.py (suposição): Script para extração de features com MediaPipe.
models/models.py: Script para treinamento, avaliação e seleção de modelos de classificação.

- Modelos treinados (.pkl):

models/pose_classifier_yolo_best.pkl
models/pose_classifier_mediapipe_best.pkl (Modelo selecionado)

- Notebook de análise:

notebooks/models.ipynb: Notebook para análise interativa das métricas e resultados.

- Resultados:

results/model_1_features/ e results/model_2_mediapipe_features/: Características extraídas.
results/graphs/: Gráficos com as métricas de desempenho.

- Relatório em PDF: Este documento.

5. Conclusão e Próximos Passos

A Atividade 2 demonstrou com sucesso um pipeline de experimentação para o reconhecimento de posturas humanas. A comparação entre as abordagens YOLOv8-Pose e MediaPipe para extração de características, seguida pelo treinamento de diversos classificadores, permitiu identificar uma combinação robusta: MediaPipe Pose com Gradient Boosting, alcançando um Macro F1-Score de 0.79.

O modelo selecionado (pose_classifier_yolo_best.pkl) está agora pronto para ser integrado na Atividade 3, que envolverá o desenvolvimento de uma API RESTful para o monitoramento de posturas em tempo real ou a partir de vídeos.