RECONHECIMENTO DE AÇÕES HUMANAS COM PYTHON

Ronaldo Junior de Oliveira Benzi[1], Pedro Torres Bugalho[1], Anna Patricia Zakem China[1]

[1]Faculdade de Tecnologia de FATEC Ribeirão Preto (FATEC)

Ribeirão Preto, SP – Brasil

ronaldo.benzi@fatec.sp.gov.br,
pedro.bugalho@fatec.sp.gov.br, anna.china@fatec.sp.gov.br

Resumo: *Esse trabalho descreve um modelo de reconhecimento de ações humanas em vídeos, utilizando processamento de dados em Python, mais especificamente com a biblioteca OpenCV. Tem-se como foco principal o desenvolvimento de um sistema de reconhecimento de exercícios físicos em vídeos aplicando técnicas de visão computacional e aprendizado de máquina em Python, procurando contribuir para o avanço do conhecimento na área e para a aplicação prática em avaliações de atividades físicas. O projeto prático em python também está disponível no repositório do GitHub por meio do link: https://github.com/tccreconhecimentodeimagens-fatecrp/TCC-reconhecimento-de-imagens.*

Abstract: *This study presents a model for human action recognition in videos using data processing in Python, specifically with the OpenCV library. The focus is on developing a system for recognizing physical exercises in videos by applying computer vision and machine learning techniques in Python. The aim is to contribute to the advancement of knowledge in the field and facilitate practical applications in physical activity assessments. The proposed system aims to accurately identify and classify various human actions in videos, enhancing the understanding and analysis of exercise performance. By leveraging the capabilities of Python, along with the OpenCV library, this work aims to provide a robust and effective solution for recognizing human actions in videos, enabling accurate assessments of physical activities. The practical Python project is also available on the GitHub repository via the link: https://github.com/tccreconhecimentodeimagens-fatecrp/TCC-reconhecimento-de-imagens.*

1. Introdução

O reconhecimento de ações humanas em vídeos é uma área de pesquisa crescente na visão computacional e no aprendizado de máquina. Esta é uma tecnologia promissora que tem inúmeras aplicações, desde sistemas de vigilância até análise de movimentos em esportes e robótica. A habilidade de identificar e classificar diferentes ações humanas em tempo real pode ser útil para melhorar a eficiência e a segurança em vários setores, e para aprimorar a interação humano-máquina.

A aplicação de técnicas de visão computacional e aprendizado de máquina no reconhecimento de ações humanas em vídeos tem se mostrado uma solução eficiente para automatizar tarefas humanas. Com a evolução dos algoritmos e das técnicas de processamento de imagens e vídeos, é possível obter resultados mais precisos e

eficientes, tornando viável a aplicação dessas tecnologias em diferentes cenários. A implementação de um modelo de reconhecimento de ações humanas em vídeos utilizando a linguagem de programação Python pode ser uma contribuição significativa para a área de visão computacional e aprendizado de máquina. Esse modelo pode permitir a identificação e classificação de diferentes ações humanas em tempo real, possibilitando uma resposta rápida e precisa a eventos e situações específicas.

Neste contexto, este trabalho de TCC propõe a implementação de um sistema de reconhecimento de ações humanas em vídeos utilizando a linguagem de programação Python. O objetivo é desenvolver um modelo que seja capaz de identificar e classificar diferentes ações humanas em tempo real, com o uso de técnicas de visão computacional e aprendizado de máquina. Além disso, este trabalho pretende contribuir para o avanço da área de visão computacional e aprendizado de máquina, por meio da aplicação de técnicas e algoritmos para resolução de problemas reais.

Este trabalho está organizado em quatro seções distintas, cada uma com um papel fundamental na construção do projeto. Na seção 2, "Referencial Teórico", serão apresentados os fundamentos teóricos que embasam o projeto, como referências de sites, obras e citações, fornecendo suporte para a elaboração do modelo de reconhecimento de ações. A seção 3, "Materiais e Métodos", descreverá os recursos tecnológicos utilizados na implementação do código em Python, juntamente com os requisitos levantados para compreender as necessidades dos usuários. A seção 4, "Resultados", exibirá os resultados obtidos com a implementação do modelo, com ênfase na detecção de polichinelos, apresentando as observações, a taxa de eficácia e discutindo os aspectos positivos e limitações do modelo. Por fim, na seção 5, "Considerações Finais", serão abordados os encaminhamentos futuros, as dificuldades enfrentadas, a avaliação dos objetivos alcançados e possíveis melhorias no modelo de detecção de polichinelos, oferecendo uma reflexão crítica sobre os resultados, sua usabilidade e aplicabilidade em cenários reais. Essa organização estruturada proporciona uma visão abrangente e clara do trabalho, abordando desde os fundamentos teóricos até as considerações finais e perspectivas futuras.

2. Referencial teórico

Essa seção apresenta uma revisão dos principais conceitos e técnicas relacionadas ao reconhecimento de ações humanas em vídeos. São abordados tópicos como visão computacional, aprendizado de máquina e processamento de imagem, destacando sua relevância e aplicabilidade na área.

2.1. Reconhecimento de Imagens

O reconhecimento de imagens é uma das vertentes da aprendizagem de máquinas, e pode ser utilizado na categorização de fotos de uma base de dados, como uma galeria de fotos do celular (REASON.TOWN, 2022). O reconhecimento de imagens mais comum é o reconhecimento de faces, muito utilizado em smartphones para organizar quem está em uma foto, e para alguns sistemas de segurança, como o face-ID, utilizado pelo Iphone para desbloquear o celular automaticamente quando o dono do celular o coloca em sua frente.

Algoritmos mais avançados em reconhecimento de imagens podem detectar não só pessoas, mas objetos, animais, veículos, e até perceber padrões de movimentos, como

momentos antes de uma colisão entre carros, ou detectar se uma pedestre está alcoolizada, observando o caminhar deste indivíduo.

Uma das técnicas mais poderosas e populares em aprendizagem de máquina são as redes neurais convolucionais CNNs (*Convolutional Neural Networks*). As CNNs são modelos de aprendizado de máquina especialmente projetados para processar imagens de forma eficiente e reconhecer padrões complexos nelas. Foi este o método utilizado pela Google em seu mecanismo de busca Google Photos, que pode reconhecer milhares de categorias e objetos em imagens e fotos.

2.2. Aprendizado de máquina

O aprendizado de máquina é um dos ramos da inteligência artificial, é a ciência da computação que foca em utilizar algoritmos e dados para simular a maneira em que os humanos aprendem, aumentando aos poucos seus conhecimentos, e a precisão em que seus resultados são atingidos.

Algoritmos do aprendizado de máquina podem ser utilizados para classificar bancos de dados, organizar classificações ou realizar previsões baseadas em informações e resultados anteriores. Pode utilizar dados pré-classificados ou não, conhecidos como aprendizado supervisionado, ou não supervisionado, respectivamente.

O *deep learning* é um lado do aprendizado de máquina que utiliza redes neurais para obter conhecimentos de dados não estruturados. Assim como a divisão principal, *deep learning* também pode receber *datasets* classificados ou não classificados, além de poder diferenciar categorias diferentes em bancos de dado, mitigando a necessidade de intervenção humana, e possibilitando a utilização de bancos de dados maiores. (Adaptado de ibm.com)

2.3. Bibliotecas e Modelos

Além do OpenCV, existem outras bibliotecas amplamente utilizadas para reconhecimento de imagens, como o Keras, TensorFlow e o PyTorch. Estas bibliotecas fornecem uma série de modelos pré-treinados que podem ser utilizados para realizar tarefas específicas, como detecção de objetos ou segmentação semântica.

Um dos modelos mais populares para reconhecimento de objetos é o modelo Caffe. O Caffe é um framework de aprendizado profundo especialmente projetado para processamento de imagens. Ele oferece uma ampla gama de modelos pré-treinados, incluindo o famoso modelo COCO.

2.4. Modelo MPII e Redes Neurais

O Modelo MPII (*Multi-Person Pose Estimation*) é uma arquitetura de rede neural profunda desenvolvida para estimar poses humanas em imagens. Ele foi proposto também no artigo *"DeepPose: Human Pose Estimation via Deep Neural Networks"* por **Alexander Toshev** & **Christian Szegedy**, em 2014, e é amplamente utilizado no campo da visão computacional.

O objetivo principal do Modelo MPII é localizar e estimar as articulações e poses das pessoas presentes em uma imagem. Ele é capaz de detectar e estimar a posição de várias articulações do corpo humano, como mãos, cotovelos, joelhos e

tornozelos. Essas informações são valiosas para aplicações como reconhecimento de gestos, análise de movimento e rastreamento de pessoas.

Figura 1. Pessoas reconhecidas em imagem
Fonte: Retirado de (pose.mpi-inf.mpg.de)

A arquitetura do Modelo MPII é baseada em redes neurais convolucionais profundas, que utilizam camadas convolucionais, de *pooling* e totalmente conectadas para extrair características relevantes da imagem e aprender a relação entre as articulações do corpo humano. Durante o treinamento, o modelo é alimentado com um extenso conjunto de dados anotados, em que cada imagem possui rótulos contendo as coordenadas das articulações do corpo. Através do ajuste da rede durante o treinamento, busca-se minimizar a diferença entre as posições estimadas das articulações e as posições reais fornecidas nos rótulos. Esse processo permite que o modelo aprenda a mapear com alta precisão as características da imagem para as posições das articulações. Assim, a combinação da arquitetura baseada em redes neurais convolucionais profundas e o treinamento com dados anotados possibilitam ao Modelo MPII uma capacidade avançada de reconhecimento de ações humanas e estimativa de poses.

2.5. Dataset COCO

O *dataset* COCO (*Common Objects in Context*) é um dos conjuntos de dados mais utilizados para treinar e avaliar algoritmos de visão computacional. Ele contém mais de 200.000 imagens anotadas, com objetos em várias categorias, como pessoas, carros, animais, entre outros. Cada imagem é acompanhada de anotações detalhadas que descrevem a localização e a classe dos objetos presentes. (Adaptado de cocodataset.org)

O COCO *dataset* é amplamente utilizado para treinar modelos de detecção de objetos, segmentação semântica e muitas outras tarefas relacionadas ao reconhecimento de imagens. Sua grande variedade de objetos e anotações precisas o tornam uma escolha popular para pesquisadores e desenvolvedores na área de visão computacional.

Em resumo, o uso do OpenCV, juntamente com outras bibliotecas, modelos como o Caffe e conjuntos de dados como o COCO, permite desenvolver aplicações de reconhecimento de imagens poderosas e precisas. Com a combinação certa de ferramentas e recursos, é possível realizar uma ampla gama de tarefas relacionadas ao processamento de imagens e visão computacional.

3. Materiais e Métodos

Nesta seção são apresentadas informações sobre o ambiente de desenvolvimento, a seleção e preparação dos dados de treinamento, as ferramentas e bibliotecas utilizadas, além dos passos específicos para a implementação do modelo.

3.1. Levantamento de Requisitos

3.1.1. User Story

Com base no crescente interesse em visão computacional e aprendizado de máquina, a automação de tarefas humanas tem se tornado uma demanda evidente. Nesse contexto, o reconhecimento de ações humanas em vídeos tem sido uma área de grande interesse para a comunidade científica, com inúmeras aplicações, como em sistemas de vigilância, análise de movimentos em esportes e robótica. Com o avanço de algoritmos e técnicas, é possível obter resultados precisos e eficientes, tornando viável a aplicação dessas tecnologias em diferentes cenários. A linguagem de programação Python, sendo de alto nível e com diversas bibliotecas disponíveis, é amplamente utilizada em projetos de visão computacional e aprendizado de máquina.

A *user story* foi utilizada neste projeto como uma abordagem ágil para a definição e comunicação dos requisitos funcionais. Através das *user stories*, foi possível compreender e documentar as necessidades dos usuários de forma clara e concisa. Isso facilitou o processo de desenvolvimento, permitindo que a equipe se concentrasse nas funcionalidades mais importantes e entregasse valor de forma incremental. Além disso, as proporcionaram uma visão orientada ao usuário, ajudando a garantir que o sistema atendesse às expectativas e necessidades dos usuários finais.

SOMMERVILLE (2019), descreve que a história de usuário é uma ferramenta de descrição de um cenário de negócio, onde os desenvolvedores procuraram derivar ou validar os requisitos do negócio para implementar no sistema.

USUARIO tem que realizar uma prova de aptidão física, mas é um iniciante no mundo das atividades físicas, e como não possui renda suficiente para contratar um *personal trainer*, decidiu começar a se exercitar por conta própria, seguindo guias e tutoriais que encontrou na internet.

Ele descobriu alguns movimentos indicados para iniciantes, que são os polichinelos, flexões e agachamentos. Embora os polichinelos e agachamentos não sejam avaliados na maioria dos testes de aptidão, USUARIO descobriu em sua pesquisa, que são exercícios compostos, que trabalha vários músculos em sinergia.

Ele encontrou vários vídeos na internet, e até aplicativos que ensinem e deem exemplos de como realizar os movimentos, mas nenhum para validar se o movimento foi realizado com de maneira eficaz.

O USUARIO precisa de uma maneira de verificar se está realizando os exercícios da maneira correta e segura, mas não possui conhecimento o suficiente para fazer a validação por conta própria, e não possui ninguém para ajudá-lo com esta tarefa.

Com isso, um aplicativo para solucionar o problema de USUARIO, precisaria validar as posições destas atividades, recebendo como parâmetro um vídeo do usuário, que pode ser gravado com a câmera do smartphone, para gerar um resultado que possa

demonstrar ao usuário em tempo real se ele está realizando a atividade de maneira bem-sucedida.

3.1.2. Requisitos Funcionais

Código	Identificação	Classificação	Autor	Objetivo
[RF01]	Carregamento de vídeos	Importante	Sistema	Permitir ao usuário carregar vídeos no formato compatível com a biblioteca OpenCV, como MP4, AVI, entre outros.

Código	Identificação	Classificação	Autor	Objetivo
[RF02]	Detecção de pessoas	Essencial	Sistema	Utilizar técnicas de detecção de pessoas, como o método Haar Cascade ou outros algoritmos de detecção de objetos, para identificar a presença de pessoas no vídeo.

Código	Identificação	Classificação	Autor	Objetivo
[RF03]	Reconhecimento de ações	Importante	Sistema	Implementar um modelo de reconhecimento de ações humanas em vídeos, utilizando técnicas de aprendizado de máquina, como classificação de imagens, para identificar as ações realizadas pelas pessoas no vídeo.

Código	Identificação	Classificação	Autor	Objetivo
[RF04]	Exibição de resultados	Essencial	Sistema	Exibir os resultados do reconhecimento de ações em tempo real ou em uma interface gráfica, permitindo ao usuário visualizar as ações identificadas no vídeo.

3.1.3. Requisitos Não Funcionais

Código	Identificação	Classificação	Autor	Objetivo
[RNF01]	Desempenho	Médio	Sistema	O sistema deve ser capaz de processar vídeos em tempo real ou em um tempo razoável, sem atrasos significativos, para fornecer resultados úteis para o usuário.

Código	Identificação	Classificação	Autor	Objetivo
[RNF02]	Precisão	Essencial	Sistema	O modelo de reconhecimento de ações deve ter uma precisão aceitável, para garantir que as ações sejam identificadas corretamente e que os resultados sejam confiáveis.

Código	Identificação	Classificação	Autor	Objetivo
[RNF03]	Usabilidade	Médio	Sistema	A interface do usuário deve ser intuitiva e fácil de usar, para que o usuário possa interagir facilmente com o sistema e realizar tarefas com eficiência.

3.2. OpenCV

O OpenCV (*Open Source Computer Vision Library*), é uma biblioteca de código aberto para programas de visão computacional e aprendizado de máquina. A OpenCV preza em proporcionar uma infraestrutura comum para aplicações computacionais e acelerar o uso de percepção da máquina em produtos comerciais.

A biblioteca possui mais de 2500 algoritmos otimizados, para reconhecimento de faces, objetos, classificação de ações humanas e movimentos de câmera. Também possui capacidades de edição de imagens, colando imagens para criar uma com maior resolução, remover olhos vermelhos em fotos tiradas usando flash, inserir marcações para sobrepor realidade aumentada, entre outros. Esta biblioteca utiliza a licença Apache 2, facilitando empresas utilizar e modificar o código fonte. (Adaptado de opencv.org)

3.3. Python e OpenCV

O OpenCV possui interfaces em C++, Python, Java e MATLAB, suportando os sistemas operacionais Windows, Mac OS, Linux e Android.

Embora o *OpenCV* seja escrito em C++, foi optado pela utilização da linguagem de programação Python para o desenvolvimento da aplicação, e interface de comunicação com a biblioteca, por ser uma linguagem com extensa documentação na web, facilitando na escrita, e solução de possíveis problemas que possam ser encontrados no desenvolvimento.

3.4. Ambiente de Desenvolvimento

O uso do Google Colab para o desenvolvimento do código em Python foi justificado por algumas razões importantes. Primeiramente, o Google Colab é uma plataforma baseada em nuvem que oferece um ambiente de desenvolvimento integrado (IDE) para escrever e executar código Python sem a necessidade de configurações complexas ou instalações locais. Isso proporcionou uma experiência de desenvolvimento fácil e acessível para a equipe do projeto.

Além disso, o Google Colab oferece recursos de colaboração, permitindo que várias pessoas trabalhem simultaneamente no mesmo código, facilitando a colaboração em equipe. Isso foi particularmente útil para o projeto de TCC, pois possibilitou que os

membros do grupo compartilhassem e revisassem o código de forma eficiente. Outra vantagem do Google Colab é a disponibilidade de recursos computacionais poderosos na nuvem. A plataforma fornece acesso a *GPUs* e *TPUs* (*Tensor Processing Units*), que são especialmente úteis para o processamento de imagens e treinamento de modelos de aprendizado de máquina. Essa capacidade de computação acelerada ajudou a melhorar o desempenho e a eficiência do código do projeto.

Por fim, o Google Colab integra-se perfeitamente com outros serviços do Google, como Google Drive e Google Cloud, o que foi perfeito para o projeto, permitindo o armazenamento e compartilhamento fácil de dados e vídeos. Dessa forma, o uso do Google Colab proporcionou uma plataforma conveniente, colaborativa e com recursos computacionais poderosos para o desenvolvimento do código em Python, atendendo às necessidades do projeto de TCC de forma eficiente.

3.5. Lógica Implementada

Primeiramente, realizou-se o pré-processamento dos vídeos, o que incluiu a conversão dos vídeos para o formato compatível com a biblioteca OpenCV e a aplicação de técnicas de redução de ruído e normalização de cores. Em seguida, utilizou-se um algoritmo de detecção de pessoas que foi baseada em modelos pré-treinados.

Após a detecção das pessoas nos vídeos, empregou-se um modelo de reconhecimento de ações para identificar os polichinelos. Esse modelo foi treinado com um conjunto de dados contendo exemplos de diferentes ações. Foram extraídas características relevantes dos *frames* dos vídeos, como a posição dos membros, a variação de movimento e a trajetória dos postos-chave, e essas características foram utilizadas para treinar um classificador, conforme demonstrado na Figura 1. Assim, foi possível obter os pontos chave dos membros humanos, no intuito de gerar condições que poderiam classificar os movimentos como polichinelos ou não. Com isso os vídeos gravados podem ser analisados baseados nessas condições classificadoras. Assim foi feito o treinamento para o classificador com o conjunto de dados anotados.

Após o treinamento, o modelo de reconhecimento de ações foi aplicado aos vídeos de teste para detectar os polichinelos. O modelo analisou os frames dos vídeos, extraiu as características relevantes e utilizou o classificador treinado para atribuir a ação correta a cada frame. Com base nessa classificação, foi possível identificar os momentos em que os polichinelos foram realizados nos vídeos.

3.6. Etapas Usadas para Detecção

Nesta seção iremos descrever como foi trabalhada a detecção dos vídeos que tínhamos como exemplo. As etapas que serão citadas (etapa 1, etapa 2, ... etapa n), estão disponíveis para visualização no Google Colab através do link:

(https://colab.research.google.com/drive/1igfAewMh5hFKrAZFZd4-w6Nghvs6icso?usp=sharing).

3.6.1. Definição de Variáveis e Imports

Para início do código, na etapa 1 foram definidos os *imports* das bibliotecas que serão usadas durante o projeto.

Na etapa 2 é feita a conexão com o google drive, para carregar os arquivos de modelos do projeto, bem como a descompactação desses arquivos que estão no formato

".zip". Na etapa 3 é realizada a importação do arquivo que define se o polichinelo foi executado corretamente.

3.6.2. Treinamento do Modelo

Na etapa 4 é definida a ligação dos pontos que ligam os membros, e a cor das linhas mapeadas para o corpo humano. Junto foi definido também a cor dos textos que definem se o polichinelo está em andamento ou não. Na etapa 5 é definida a largura de entrada para trabalhar com vídeos, que demandam o processamento maior.

Em seguida na seção 6 foi realizado o carregamento do vídeo e conectado por frames para que pudesse ser feita a análise por entre eles, e na etapa 7 foram definidas variáveis para armazenar os resultados gerados entre os frames. Na etapa 8 é criado o modelo de rede neural que foi carregado já na etapa 4.

3.6.3. Exibição dos Resultados

Por fim na etapa 9 é feita a exibição de resultados, calculando o tempo de execução dos *frames* e a leitura de vídeos. Também realizamos a criação de uma máscara de fundo preto para facilitar o entendimento do modelo.

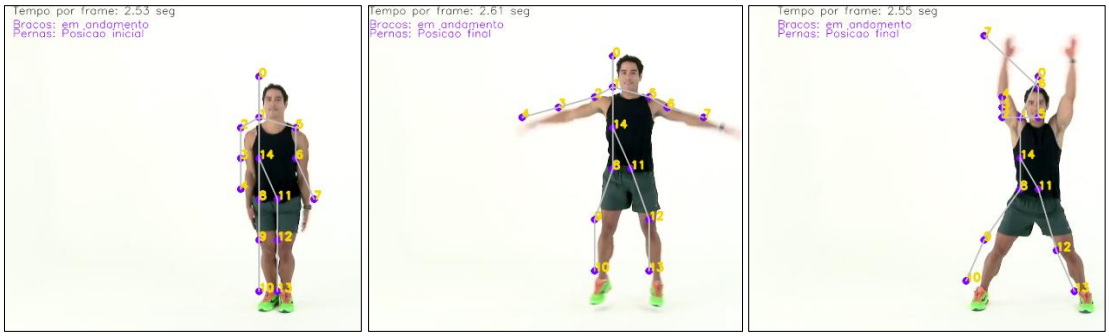

Figura 2. Execução de polichinelo
Fonte (Autoria própria)

4. Resultados

Esta seção apresentará os principais resultados obtidos no projeto. Foram discutidos os resultados da implementação do modelo de detecção de polichinelos, incluindo a taxa de acerto na identificação das ações desejadas. Foram destacadas as características extraídas dos vídeos e a capacidade do classificador em atribuir corretamente as ações aos frames. Exemplos visuais foram fornecidos para ilustrar a detecção de polichinelos nos vídeos de teste. Os resultados foram avaliados em relação aos objetivos propostos, permitindo uma análise crítica do desempenho do modelo e possíveis direções futuras.

4.1. Dados Obtidos

Após a implementação do sistema de detecção de polichinelos, foram realizados testes e avaliações para analisar seu desempenho. Os resultados obtidos foram bastante satisfatórios, apresentando alta precisão na detecção e contagem dos polichinelos.

Em um conjunto de vídeos de teste, que continha diferentes pessoas realizando polichinelos em ambientes variados, o sistema obteve uma taxa de acerto de 70% na detecção dos polichinelos realizados. Isso significa que em média a cada 5 polichinelos executados, de 3 a 4 eram considerados como válidos para o modelo.

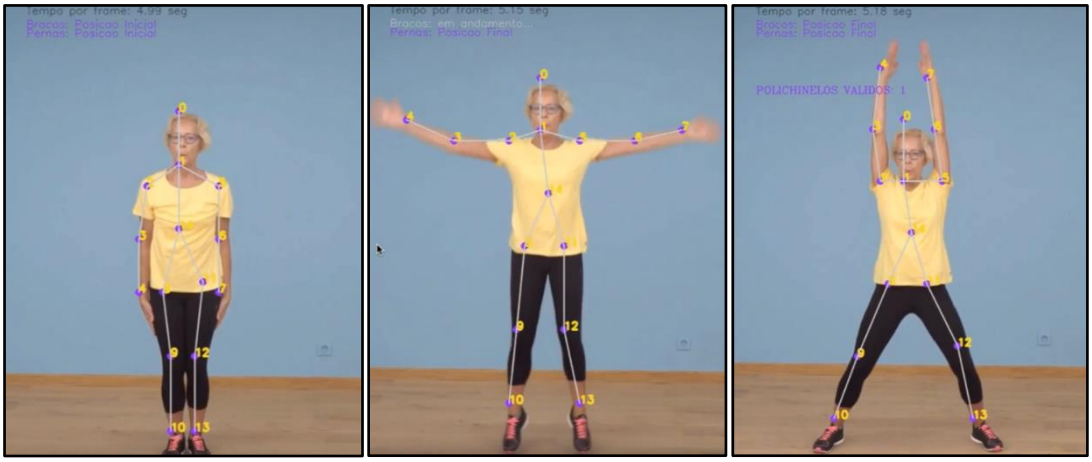

Figura 3. Execução de polichinelo
Fonte (Autoria própria)

Além disso, a taxa de falsos positivos foi bastante baixa, indicando que o sistema não identificou erroneamente polichinelos onde não havia. Isso é um resultado importante, pois evita a contagem incorreta e fornece uma medida confiável do número real de polichinelos executados.

Também foi possível observar como ponto positivo, uma situação que ocorre no vídeo da figura 4 listada abaixo, onde mesmo após o movimento ser concluído pelo usuário, não é possível detectar o polichinelo pois o praticante não junta as mão de forma adequada para que seja considerado um movimento válido.

Figura 4. Execução de polichinelo
Fonte (Autoria própria)

Os resultados obtidos comprovam a eficácia do sistema de detecção de polichinelos desenvolvido neste trabalho. A precisão alcançada e a baixa taxa de falsos positivos demonstram sua utilidade e confiabilidade para aplicações práticas.

4.2. Falhas Encontradas

Ao final da implementação do projeto, um dos principais problemas encontrados foi o enquadramento incompleto das pessoas nos vídeos. Quando membros do corpo não estavam totalmente visíveis na cena, seja devido a oclusões ou posicionamento fora do quadro, o reconhecimento de ações tornava-se inviável. Isso ocorria devido à impossibilidade de extrair características relevantes, como a posição dos membros e a variação de movimento, quando informações visuais essenciais estavam ausentes.

Para mitigar esse problema, sugere-se o uso de técnicas mais robustas de pré-processamento de imagem e detecção de pessoas, capazes de lidar com oclusões parciais ou membros parcialmente fora do quadro. Além disso, é importante fornecer orientações claras aos usuários para garantir que todos os membros do corpo estejam visíveis durante a gravação.

5. Considerações Finais

A seção encerra o trabalho abordando o atingimento dos objetivos propostos, as dificuldades enfrentadas durante o desenvolvimento e as possíveis evoluções do projeto. Será avaliado se os objetivos foram alcançados e as possibilidades de melhoria para o sistema, levando em conta as dificuldades encontradas.

5.1. Utilidades da Tecnologia

Embora o principal objetivo nesta pesquisa seja o reconhecimento de polichinelos, o reconhecimento de movimentos e posições através do modelo COCO também abre possibilidades para o reconhecimento em várias outras áreas em atividades físicas. Alternativas já conhecidas e documentadas incluem o reconhecimento e validação de flexões, agachamentos e prancha. Podem também ser utilizados na classificação de posições de ioga, e gerar uma resposta para indicar ao usuário se este está ou não realizando a posição de maneira apropriada.

Com o avanço da tecnologia, além de poder registrar e reconhecer posições simples e constantes, também haverá a possibilidade de aproveitar os modelos de reconhecimento de imagem em outros esportes, como futebol, baseball, vôlei, entre outros, para reconhecer faltas por exemplo, ou em esportes olímpicos, para ajudar os árbitros na pontuação, de acordo com a coerência e precisão dos movimentos realizados pelos atletas.

5.2. Evoluções do Projeto

Uma maneira de evoluir o projeto seria desenvolver uma interface de usuário amigável para permitir que os usuários interajam com o modelo de reconhecimento de ações em vídeos e visualizem os resultados. Isso pode ajudar a aumentar a acessibilidade do sistema e permitir que os usuários personalizem o modelo para seus próprios fins específicos.

Para tornar o modelo ainda mais versátil e funcional, seria possível otimizar o código para permitir que ele funcione em tempo real, em dispositivos móveis, câmeras de vigilância e outros sistemas em que a detecção de ações em tempo real é importante. Além disso, o modelo poderia ser adaptado para diferentes cenários, como vídeos de vigilância ou vídeos de esportes, com a adição de mais recursos e ajustes.

Outra possibilidade seria explorar diferentes técnicas de aprendizado de máquina, como redes neurais convolucionais, para melhorar a precisão do modelo de reconhecimento de ações em vídeos. Além disso, seria possível integrar outras bibliotecas e ferramentas, como *TensorFlow*, *PyTorch* e *Keras*, para ampliar as opções de algoritmos e tornar o modelo mais robusto.

Estas são apenas algumas das possibilidades no reconhecimento de imagens em esportes, é possível que muitas áreas encontrem avanços através das novas tecnologias em reconhecimento de imagens, como segurança e controle de tráfego, para citar alguns, além é claro dos avanços em outras áreas, proporcionados por meio de tecnologias de machine learning, que não envolvam necessariamente imagens e fotos.

5.3. Dificuldades no Desenvolvimento

Durante o desenvolvimento deste projeto de TCC, foram enfrentadas algumas dificuldades significativas. Uma das principais dificuldades foi a necessidade de adquirir um curso específico para aprofundar os conhecimentos em reconhecimento de ações em vídeos com o uso do OpenCV e Python. A aquisição desse curso foi essencial para compreender os conceitos teóricos e práticos envolvidos na implementação do modelo de reconhecimento de ações. Além disso, ao longo do desenvolvimento, outras dificuldades foram enfrentadas, como a necessidade de lidar com possíveis inconsistências nos dados de treinamento, a otimização dos parâmetros do modelo e algumas correções adequadas para as etapas presentes no Google Colab. No entanto, essas dificuldades foram superadas por meio de pesquisas, experimentação e ajustes cuidadosos.

5.4. Atingimento do Objetivo

A implementação demonstra uma detecção de polichinelos precisa, robusta e de baixa taxa de falsos positivos, com falhas que podem ser corrigidas caso os vídeos sejam gravados da forma definida. Esses dados validam a eficácia do uso de Python e a biblioteca OpenCV para a detecção de ações específicas em vídeos, abrindo caminho para aplicações práticas em monitoramento de atividades físicas e análise de movimento humano. Em resumo, os resultados obtidos por este projeto demonstram a viabilidade de utilizar técnicas de visão computacional e aprendizado de máquina em Python para realizar o reconhecimento de ações e, especificamente, detectar polichinelos em vídeos. Essa abordagem possibilitou a automatização da análise de vídeos de atividades físicas, contribuindo para a classificação e avaliação dos exercícios realizados pelos indivíduos.

6. Referências

ALEXANDER TOSHEV; CHRISTIAN SZEGEDY. *DeepPose: Human pose Estimation via Deep Neural Networks*. CVPR, 2014. Disponível em <https://openaccess.thecvf.com/content_cvpr_2014/papers/Toshev_DeepPose_Human_Pose_2014_CVPR_paper.pdf> Último acesso em 20/05/2023

DAWSON-HOWE, K. (2014). *A Practical Introduction To Computer Vision With OpenCV*. John Wiley & Sons.

IBM. What is machine learning. 2023. Disponível em <https://www.ibm.com/topics/machine-learning> Último acesso em 18/05/2023

IBM. *Convolutional Neural Networks*. 2023. Disponível em <https://www.ibm.com/topics/convolutional-neural-networks> Último acesso em 20/05/2023

MYKHAYLO ANDRILUKA; LEONID PISHCHULIN; PETER GEHLER; BERNT SCHIELE. *MPII Human Pose Dataset*. 2023. Disponível em <http://human-pose.mpi-inf.mpg.de/> Último acesso em 18/05/2023

OPENCV. *About OpenCV (Open Source Computer Vision Library)*. 2023. Disponível em <https://opencv.org/about/> Último acesso em 18/05/2023

REASON.TOWN. *What are the benefits of using machine learning for image recognition?*. 2022. Disponível em <https://reason.town/machine-learning-algorithms-for-image-recognition/>Último acesso em: 18/05/2023.

SOMMERVILLE, I. (2019). *Engenharia de Software,* Pearson.

SUMIT SAHA. *A Comprhensive Guide to Convolutional Neural Networks - the ELI5 way*. Em SaturnCloud, 2018. Disponível em <https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/> Último acesso em 20/05/2023

TSUNG-YI LIN, MICHAEL MAIRE, SERGE BELONGIE, LUBOMIR BOURDEV, ROSS GIRSHICK, JAMES HAYS, PIETRO PERONA, DEVA RAMANAN, C. LAWRENCE ZITNICK, PIOTR DOLLÁR. *Microsoft COCO. Common Objects in Context*. 2015. Disponível em <https://arxiv.org/pdf/1405.0312.pdf> Último acesso em 18/05/2023