Universidade Fernando Pessoa

Mestrado em Computação Móvel

Inteligência Artificial

Fábio Carvalho - 35553

João Reis - 37190

Human Activity Recognition usando deep learning (ML) e video datasets

---

Porto

abril/2021

Universidade Fernando Pessoa

Praça 9 de Abril, 349

P-4249-004 Porto

Tel. +351-22550.82.70

Fax. +351-22550.82.69

geral@ufp.pt

Índice de imagens

Índice

I. Introdução

O comportamento humano é a capacidade expressa de indivíduos responderem a estímulos internos e externos ao longo das suas vidas. O reconhecimento de atividade humana é uma aprendizagem computacional para detetar e identificar este comportamento humano, face às ações humanas no ambiente onde se encontram. Logo, é de extrema importância monitorar estes comportamentos, nomeadamente, através de vídeos ou frames, onde os mesmos serão classificados, levando em consideração vários fatores, como o ambiente, a sequência temporal de acontecimentos e o posicionamento.

II. Descrição do Problema

O reconhecimento da atividade humana tem vindo a ser uma necessidade crescente nos últimos anos por diversas razões, entre as quais se pode destacar a deteção automática de atividades de caráter ilegal através de imagens de vídeo. Com isto, o desafio lançado neste presente projeto passa pela análise e comparação de vários métodos já implementados de classificação de vídeo que, ao analisar o mesmo, fazem uma previsão da atividade a ser desenvolvida com base em modelos previamente treinados. Para este presente relatório, será realizada uma análise e comparação entre os métodos Single-Frame CNN e CNN com LSTM.

III. Estado da Arte

Nos cenários deste relatório, o reconhecimento de atividade humana utiliza métodos de *deep learning*. Destes foram analisados apenas dois, nomeadamente o Single-Frame CNN e o CNN com LSTM.

O Single Frame CNN é um dos métodos, anteriormente comentados, no campo do reconhecimento de atividade humana. Consiste, em algumas frames espalhadas pelo vídeo em análise, calcular a probabilidade de uma determinada atividade estar a acontecer, fazendo, após isso, a média de todas as probabilidades individuais, obtendo um vetor de probabilidades final. Ainda assim, é uma abordagem com um desempenho muito bom. Este método foi implementado, por motivo de estudo para este relatório, contendo um dataset com vários vídeos contendo atividades humanas, como por exemplo, passear com um cão, andar de baloiço, corrida de cavalos e prática de tai chi. Todavia, este método estende-se além destes exemplos, sendo muito útil quando o objetivo se trata de identificar o nome de uma atividade e a probabilidade de estar a acontecer de facto.

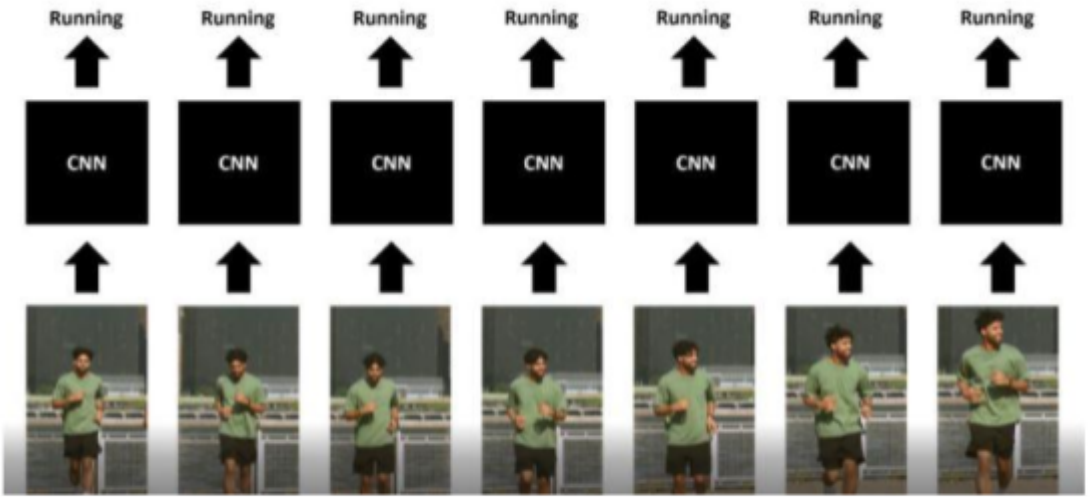


Figura 1 - Arquitetura Single Frame CNN

A arquitetura do CNN com LSTM envolve o uso de camadas CNN para extração de recursos em dados de entrada combinados com LSTM para oferecer suporte à previsão de sequência. Foram desenvolvidos para problemas de previsão visual de séries temporais e a prática de gerar descrições textuais provenientes de sequências de imagens. Ao contrário de um LSTM, um CNN não é

recorrente, o que significa que ele não retém a memória de padrões de séries temporais anteriores. Em vez disso, ele só pode treinar com base nos dados inseridos pelo modelo numa determinada instância de tempo

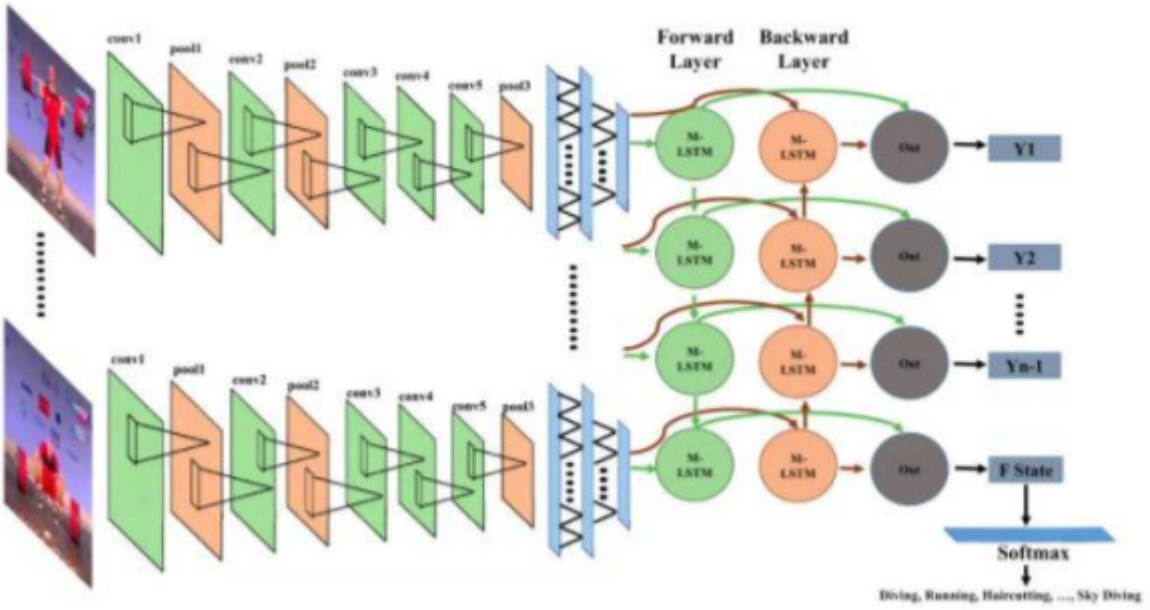


Figura 2 - Arquitetura CNN com LSTM

IV. Descrição do trabalho realizado pelo autor

Para a concretização deste projeto, inicialmente foi replicado o algoritmo de classificação de vídeo Single-Frame CNN, implementado em python com a biblioteca Keras por Taha Anwar [1]. Dadas as limitações computacionais dos autores deste relatório, e sendo este projeto GPU bound, decidiu-se que a toda a implementação e execução de código necessárias seriam feitas via Google Colab, onde é possível o acesso a GPU's da Google para executarem o código de forma mais rápida e eficiente. Um pré-requisito ao desenvolvimento deste projeto, foi algum tipo de video dataset para que se pudesse testar os vários métodos de Human Activity Recognition. Para o efeito, foi utilizado o Human Activity Recognition Using Smartphones Data Set, disponibilizado pelo UCI Machine Learning Repository [2]. Este dataset foi então carregado para o Google Drive para que pudesse ser acessado pelo Google Colab e apresenta uma estrutura de pastas definida pela pasta root "UCF50", e várias subpastas, cada uma identificando uma atividade, com diversos vídeos representativos da mesma. Esta estrutura está representada pela figura 3.

Figura 3 - Estrutura de pastas do Dataset

Com o código implementado, definiu-se um modelo Single Frame CNN que será o primeiro algoritmo utilizado neste relatório. Este modelo está representado pela figura 4.

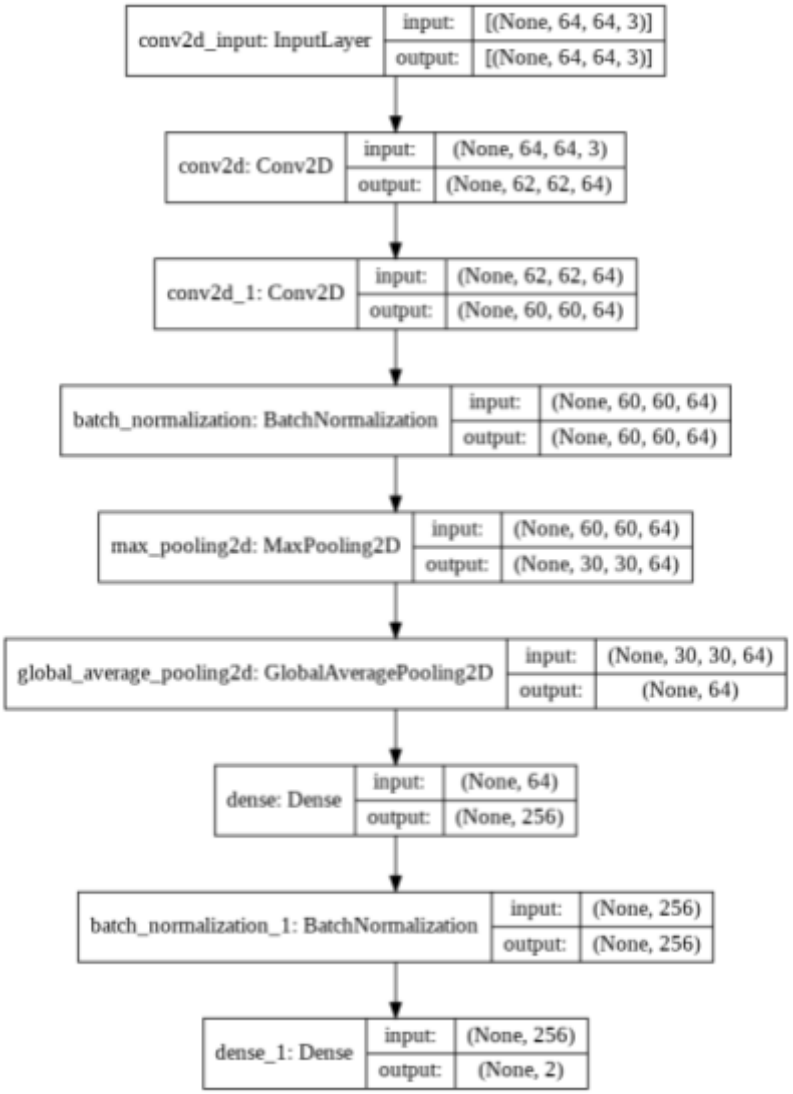


Figura 4 - Modelo Single Frame CNN

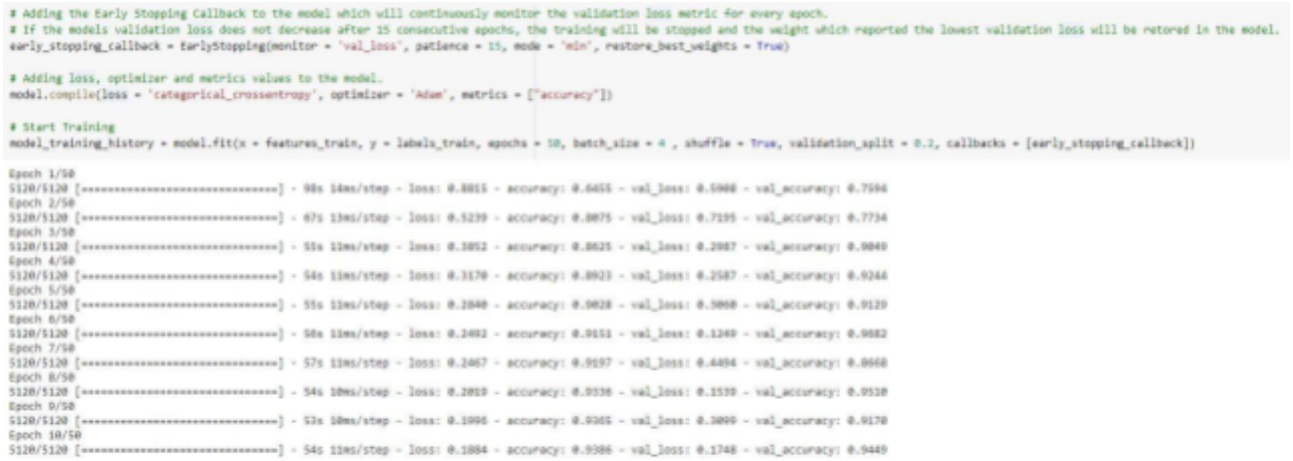
De seguida, teve que ser selecionado um grupo de N classes, 4 inicialmente ("WalkingWithDog", "TaiChi", "Swing" e "HorseRace"), para os quais um modelo de Single Frame CNN será treinado utilizando um conjunto de vídeos relativos a cada uma das atividades de cada classe. Numa segunda fase, foram consideradas 2 outras classes, Punch e Push Up.

```
image_height, image_width = 64, 64
max_images_per_class = 8000

dataset_directory = "drive/MyDrive/UCF50"
classes_list = ["WalkingWithDog", "TaiChi", "Swing", "HorseRace"]
```

Figura 5 - Lista de classes iniciais

Após a extração dos dados das classes previamente selecionadas, foi necessário o treino do modelo tendo - as em conta. Para o treino do modelo, foi necessário definir um número de épocas de treino. Quanto maior o número de épocas, maior a accuracy do sistema a detetar atividades. O número de épocas definido para este projeto foi 50.



```
# Adding the Early Stopping Callback to the model which will continuously monitor the validation loss metric for every epoch.
# If the models validation loss does not decrease after 15 consecutive epochs, the training will be stopped and the weight which reported the lowest validation loss will be retored in the model.
early_stopping_callback = EarlyStopping(monitor = 'val_loss', patience = 15, mode = 'min', restore_best_weights = True)

# Adding loss, optimizer and metrics values to the model.
model.compile(loss = 'categorical_crossentropy', optimizer = 'Adam', metrics = ["accuracy"])

# Start Training
model_training_history = model.fit(x = features_train, y = labels_train, epochs = 50, batch_size = 4 , shuffle = True, validation_split = 0.2, callbacks = [early_stopping_callback])
```

```
Epoch 1/50
5120/5120 [==============================] - 98s 14ms/step - loss: 0.8815 - accuracy: 0.6455 - val_loss: 0.5908 - val_accuracy: 0.7594
Epoch 2/50
5120/5120 [==============================] - 67s 13ms/step - loss: 0.5239 - accuracy: 0.8075 - val_loss: 0.7195 - val_accuracy: 0.7734
Epoch 3/50
5120/5120 [==============================] - 55s 11ms/step - loss: 0.3852 - accuracy: 0.8625 - val_loss: 0.2987 - val_accuracy: 0.9049
Epoch 4/50
5120/5120 [==============================] - 54s 11ms/step - loss: 0.3170 - accuracy: 0.8923 - val_loss: 0.2587 - val_accuracy: 0.9244
Epoch 5/50
5120/5120 [==============================] - 55s 11ms/step - loss: 0.2840 - accuracy: 0.9028 - val_loss: 0.5068 - val_accuracy: 0.9129
Epoch 6/50
5120/5120 [==============================] - 56s 11ms/step - loss: 0.2492 - accuracy: 0.9151 - val_loss: 0.1249 - val_accuracy: 0.9682
Epoch 7/50
5120/5120 [==============================] - 57s 11ms/step - loss: 0.2467 - accuracy: 0.9197 - val_loss: 0.4494 - val_accuracy: 0.8668
Epoch 8/50
5120/5120 [==============================] - 54s 10ms/step - loss: 0.2019 - accuracy: 0.9336 - val_loss: 0.1539 - val_accuracy: 0.9518
Epoch 9/50
5120/5120 [==============================] - 53s 10ms/step - loss: 0.1996 - accuracy: 0.9365 - val_loss: 0.3099 - val_accuracy: 0.9170
Epoch 10/50
5120/5120 [==============================] - 54s 11ms/step - loss: 0.1884 - accuracy: 0.9386 - val_loss: 0.1748 - val_accuracy: 0.9449
```

Figura 6 - Treino do modelo por épocas

Com isto, o algoritmo recebe como input ficheiros de vídeo que irá analisar e, consequentemente, classificar quanto ao tipo de atividade, devolvendo a probabilidade do mesmo input ser de uma das classes definidas e treinadas no modelo. Uma análise dos resultados será feita no capítulo V deste relatório onde serão experimentadas novas classes.

Em relação ao CNN com LSTM, todos os passos anteriores foram replicados, tendo como a única diferença a construção do modelo. O modelo CNN com LSTM utilizado está representado na figura 7.

Figura 7 - Modelo CNN com LSTM

V. *Análise de Resultados*

Conforme explicado no capítulo anterior, um conjunto de passos foram executados para a produção deste relatório. Estes passos produziram resultados que resultam agora numa análise demonstrada neste mesmo capítulo.

Começando pelo Single Frame CNN. Utilizando inicialmente as classes "WalkingWithDog", "TaiChi", "Swing" e "HorseRace", foram analisados vários vídeos com diferentes taxas de sucesso.

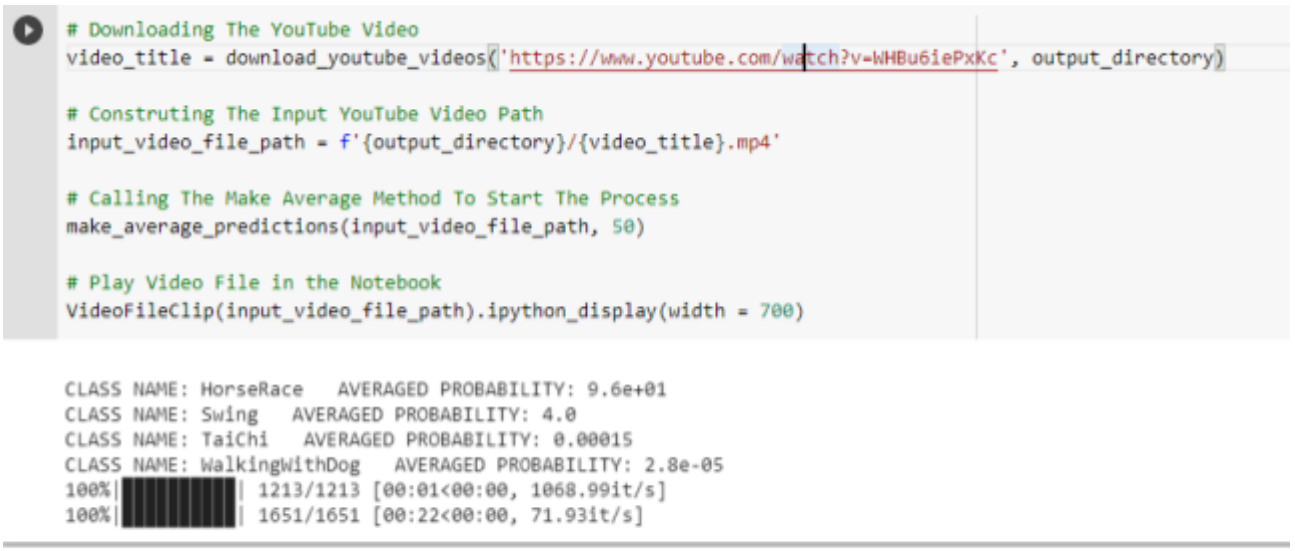


Figura 8 - Estimativa da atividade "Horse Race" com Single Frame CNN

Como pode ser observado na figura 8, ao receber um input de um vídeo de Youtube de uma corrida de cavalos, o algoritmo de Single Frame CNN classifica, com cerca de 96% de certeza, que se trata de uma corrida de cavalos, O algoritmo também afirma uma chance de 4% do vídeo representar a atividade "swinging", porém, sendo este valor muito inferior aos 96% anteriores, não o consideramos como verdadeiro.

O mesmo acontece com outros inputs de outras atividades, cujas quais foram consideradas no modelo. Tem-se então uma taxa de 95.25% de certeza quanto às atividades previamente treinadas conforme a figura 9, chegando mesmo a praticamente 100% de certeza no caso do Taichi.

	Input			
Probabilidade de atividade	**Horse Race**	**Taichi**	**Walking with Dog**	**Swing**
Horse Race	9.6e+01	9.5e-05	1.2e-07	0.59
Taichi	0.00015	1E+02	1.8e-06	2.7e-07
Walking with Dog	2.8e-05	0.0011	9.3e+01	7.4
Swing	4	0.00039	7.3	9.2e+01

Figura 9 - Resultados da classificação do primeiro conjunto de atividades utilizando Single Frame CNN

Após estes resultados, o modelo foi re-treinado para aprender duas atividades diferentes, nomeadamente as atividades Punch e Push Ups, e tal como primeiro teste, os resultados foram bastante positivos conforme demonstrado pela figura 10.

	Input		
Probabilidade de atividade	**Mayweather vs Logan**	**Push Ups**	**Top 25 best Punches**
Punch	6.2e+01	5.1e-08	8.6e+01
Push up	3.8e+01	1E+02	1.4e+01

Figura 10 - Resultados da classificação do segundo conjunto de atividades utilizando Single Frame CNN

Neste caso apesar de, no primeiro input, se obteve apenas 62% de certeza quanto à atividade, pode ser declarado como um valor bastante positivo, uma vez que o input é constituído por um vasto conjunto de momentos de um combate de boxe, dos quais apenas uma pequena parte se destina ao combate em si. Quanto aos restantes inputs, o modelo conseguiu classificar com alto grau de confiança.

No entanto, foi possível observar que, quando era feito um input com uma atividade que não as treinadas pelo modelo, obteve - se uma alta taxa de erro, pois o algoritmo declarava, com grandes taxas de certeza, que reconhecia a atividade do input como sendo uma das atividades treinadas no modelo. Isto foi possível observar quando se re-treinou o modelo para as atividades de “Push Ups”

e “Punch”, e se tentou reclassificar as atividades de “Taichi” e “Swing” como podemos observar na figura 11.

	Input	
Probabilidade de atividade	**Swing**	**Taichi**
Punch	85%	98%
Push Ups	15%	2%

Figura 11 - Classificação de atividades não treinadas no modelo com Single Frame CNN

Dada à natureza Single Frame deste algoritmo, onde frames isoladas são analisadas sem contexto temporal para a classificação de atividade, verifica-se uma alta taxa de insucesso quando os inputs não coincidem com as atividades treinadas no modelo. No caso específico do Taichi, foi atribuída uma taxa de 98% de certeza em como se tratava de um Punch. Isto pode acontecer pelo facto de, sem uma noção temporal, uma frame de Taichi possa parecer com um Punch, no entanto, a classificação continua incorreta, especialmente no caso do Swing que, mesmo sendo uma atividade totalmente distinta de Punch, foi declarada como tal com uma taxa de 85% de certeza.

Devido a estas incoerências classificativas por parte do Single Frame CNN, foi decidido a repetição do processo, mas com o algoritmo de CNN com LSTM, com o objetivo de melhorar os resultados obtidos.

Com o processo repetido, usando CNN com LSTM com os mesmos inputs, conseguiu - se uma melhoria em alguns dos resultados, mas um decréscimo da certeza de uma atividade em particular (Horse Race) e também uma falha numa atividade anteriormente bem qualificada (Walking with Dog). Estes resultados estão apresentados nas figuras 12 e 13. Esta falha no reconhecimento da atividade Walking with Dog, apesar de inesperada, acredita-se que poderá ser corrigida com algum ajustamento do algoritmo e com um aumento de épocas de treino do modelo.

	Input			
Probabilidade de atividade	**Horse Race**	**Taichi**	**Walking with Dog**	**Swing**
Horse Race	6E+01	4.1e-09	0.0087	0.0098
Taichi	0.6	1E+02	0.0075	0.002
Walking with Dog	2.3	1.5e-05	1.4e+01	9.5
Swing	3.7e+01	7.8e-08	8.6e+01	9.1e+01

Figura 12 - Resultados da classificação do primeiro conjunto de atividades utilizando CNN com LSTM

	Input		
Probabilidade de atividade	**Mayweather vs Logan**	**Push Ups**	**Top 25 best Punches**
Punch	7.5e+01	1.2e+01	9.4e+01
Push up	2.5e+01	8.8e+01	6

Figura 13 - Resultados da classificação do segundo conjunto de atividades utilizando CNN com LSTM

Comparando os resultados de ambos os algoritmos, em média, registou -se um aumento de certeza de cerca de 7,3% (não contando com o caso de classificação errada) por parte do algoritmo de CNN com LSTM, em relação ao Single Frame CNN. Este valor está contido naquilo que seria o intervalo esperado de melhoria em relação ao trabalho de outros autores, tendo os mesmos atingidos melhorias entre os 3% e os 7%.

Input	Single Frame CNN	CNN com LSTM	Melhoria (%)
Horse Race	9.6e+01	6E+01	36
Taichi	1E+02	1E+02	0
Walking with Dog	9.3e+01	Falhou	N/A
Swing	9.2e+01	9.1e+01	-1
Mayweather vs Logan	6.2e+01	7.5e+01	13
Push Ups	1E+02	8.8e+01	-12
Top 25 best Punches	8.6e+01	9.4e+01	8
		Média de Melhoria	7,3

Figura 14 - Comparação de resultados entre Single Frame CNN e CNN com LSTM

Para além da figura 14, pode ser observado as melhorias, tanto em termos de taxa de acerto, como também em termos de perdas, do CNN com LSTM em relação ao Single Frame CNN nas figuras 15 e 16, onde estão representados através de gráficos de linhas.

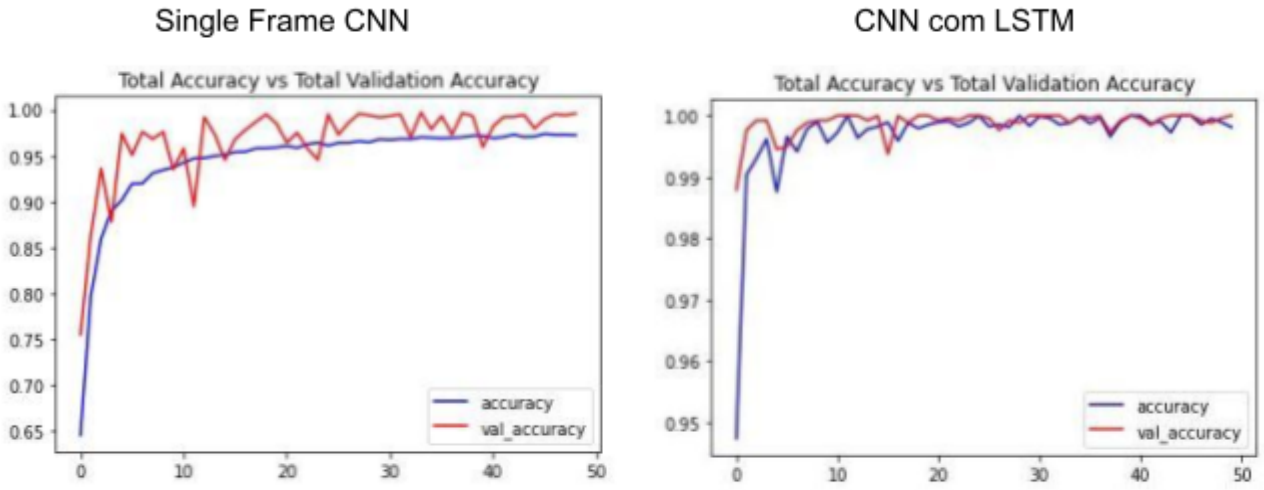


Figura 15 - Comparação de taxa de acerto entre os algoritmos

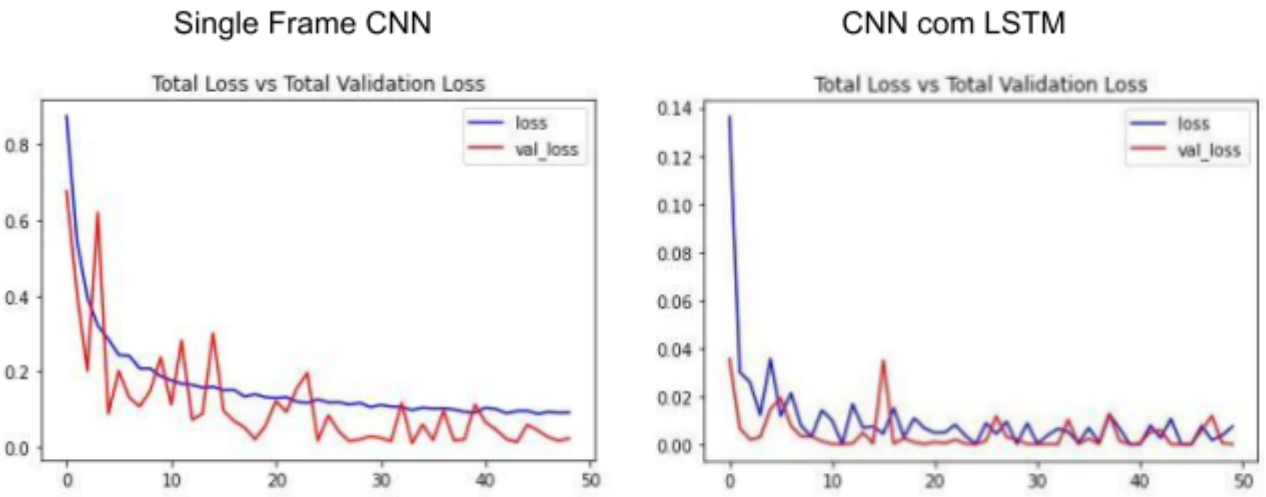


Figura 16 - Comparação de perdas entre os algoritmos

VI. Conclusões e Perspetivas de Desenvolvimento

Com o trabalho executado e depois de uma análise detalhada, pôde-se identificar que, com o CNN com LSTM, atinge-se, por norma, algumas melhorias em relação ao Single Frame CNN, atingindo uma maior taxa de acerto geral. Algumas reduções da taxa de acerto e erros na classificação com o CNN com LSTM em relação ao Single Frame CNN poderiam ser melhorados ou resolvidos com um ajuste do algoritmo e com maior treino do mesmo. Não obstante, com o contributo do LSTM conseguiu-se, em média, um aprimoramento dos resultados na ordem dos 7,3% em relação ao Single Frame CNN. Para efeitos de trabalho futuro, seria importante ajustar o algoritmo CNN com LSTM para obtenção de melhores resultados, sendo também relevante experimentar a mesma classificação com outros algoritmos de reconhecimento de atividades.

VII. Referências

1. Learn OpenCV, (2021, May 3). Introduction to Video Classification and Human Activity Recognition. Learn OpenCV | OpenCV, PyTorch, Keras, Tensorflow Examples and Tutorials.
https://learnopencv.com/introduction-to-video-classification-and-human-activity-recognition
2. UCI Machine Learning Repository. Human Activity Recognition Using Smartphones Data Set
https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones

VIII. Anexos

GitHub - https://github.com/cebada/IA_Video_Activity_Recognition

Dataset - https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones