LSTM RNN for ballet sequences

Martina Rudari

May 2024

The following paper presents a ballet classifier method using a Long Short-Term Memory Network and the pre-trained MediaPipe model for pose detection. Ballet is a popular art form that originated during the Italian Renaissance and developed as a concert dance form in France and Russia. Since then it has become a highly technical art with a broad vocabulary. Ballet provides the technical base for many other dance forms and has inspired new genres in cultures across the globe.
This paper presents an LSTM RNN able to classify ballet poses and list the sequence in which the detected poses were executed.
This project aims to support dancers in their training, recognizing dance figures and assisting the dancer with the correct execution of the movements according to ballet technique.
There are a variety of possibilities for development, and this LSTM network could be the base for generating synthetic ballet exercises as support for teachers, professional dancers, and choreographers.
The Dataset was collected in real-time, as well as the classification, which consisted of the real-time recognition of six ballet movements from ballet technique. When a movement was detected it was added to the exercise sequence.
The MediaPipe pre-trained model was used to collect and prepare the Dataset. The model permitted the detection of the 33 key points of the body to make the pose detection possible.
The choice of using an LSTM Network was made to obtain a high level of accuracy with a limitation on the number of necessary data for the training. The LSTM network resulted in being faster in the training and detection phases, using a lower number of parameters, and being best suited for real-time performances.
In addition, conclusions and future developments are presented to encourage the spread of interest and consciousness of ballet. It could help beginners make their first steps in the ballet world, whilst also providing useful tools for advanced training, building choreographies, and helping performers memorising steps.

1 Introduction

Most current approaches to the development of Artificial Intelligence (AI) are mainly on Machine Learning (ML).
ML is a branch of AI that is concerned with the use of data and algorithms inspired by the way human beings learn, gradually increasing and refining its accuracy. The goal of ML is to bring the machine to autonomy in its existence, i.e. to be able to carry out inductive reasoning and, to complete new tasks, which it has never tackled before, in an accurate manner, simply after gaining experience on a set of training examples. Machine Learning, Deep Learning (DL), and Artificial Neural Networks (ANNs) are subfields of Artificial Intelligence.
Firstly, it is possible to define Artificial Neural Networks as a type of Machine Learning, and Deep Learning as an Artificial Neural Network consisting of more than three layers. AI processes data to make decisions and predictions, and ML algorithms allow AI not only to process data but to use it to increase its intelligence, without the need for programming.
DL includes multiple communicating layers where the system can produce higher-level output. Unlike traditional ML algorithms, DL algorithms update parameters sequentially. The effectiveness of DL algorithms depends on the availability of large datasets, therefore its increasing usefulness has only been possible in the last few years with the exponential growth of data.

2 Neural Networks

Artificial Neural Networks are inspired by the biological neural organisation found in animal brains. The artificial neurons are connected by edges, which are inspired by the synapses in the brain. Each signal is a real number processed by the artificial neuron and sent to other connected neurons through an activation function. Weights associated with neurons and edges represent the strength of the signal, and it needs to be adjusted through the learning process. Neurons are typically aggregated in layers, each layer is associated with a different function to apply to its inputs. Signals propagate from the first layer, the input layer, to the last layer, the output layer. The intermediate layers are called hidden layers, if there are more than two intermediate layers the ANN is called Deep Neural Network.

2.1 Artificial Neural Networks (ANNs)

Artificial Neural Networks are inspired by the structure and the function of the human brain to solve various problems in the field of AI. These algorithms are influenced by the way that biological neurons communicate with each other through signal transmission. An ANN is composed of interconnected layers of nodes, including an input layer, hidden layers, and an output layer. Each node is associated with a linear regression model, which is a mathematical model that aims to predict future events from the input data set. The connections between the nodes are characterised by addressed weights, which define the influence that each input has on the output. Each node is defined by its weight, threshold, input data, and output. The data flows in only one direction, for this reason, ANNs are called Feed-forward networks. The level of accuracy of an ANN is evaluated by its cost function. The goal is to reduce the value of that function by adjusting the weights of the model and biases to better fit the training data set. ANNs operate with a distributed memory allowing the possibility to work on incomplete knowledge, and to store information through the entire network. However, ANNs can have a huge hardware dependency requiring significant computational resources. There are various types of neural networks specialised in solving specific tasks, for example, Convolutional Neural Networks (CNN) for image processing, and Recurrent Neural Networks (RNN) for sequential data analysis.

2.2 Convolutional Neural Networks (CNNs)

CNNs are mostly used to solve pattern recognition problems. The pattern recognition is performed by filters applied on the hidden layers. These filters specify which pattern the network has to look for and map the image returning a similarity score based on the matched features. As data progresses through layers the applied filters will be able to recognise increasing levels of abstraction and complexity. Unlike traditional ANN, CNNs are designed to solve image-related tasks thanks to their ability to learn feature similarities.

2.3 Recurrent Neural Networks (RNNs)

RNNs are primarily designed to solve problems characterized by sequential data or require a time-series analysis. The architecture is characterised by having connections that form a loop, where the information is kept in an internal memory to allow the understanding of dependencies between successive inputs. The ability to recognise dependencies between data is fundamental to solving tasks that require an understanding of the order and the context of the input data. For this reason, RNNs are mostly used for natural language processing tasks, speech recognition, and handwriting recognition. The main challenge faced by these networks has been solving the vanishing gradient, a typical issue faced with data that presents long-term dependencies. Long Short-Term Memory Networks (LSTM) have been developed to solve these limitations, helping to control the flow of information across the network.

2.3.1 Long Short-term Memory RNN (LSTM)

A Long Short-Term Memory Network is a type of Recurrent Neural Network (RNN) designed to deal with the vanishing gradient problem, commonly encountered in standard RNNs. LSTM networks consist of three main components: the Cell state, the Hidden state, and the Gates, which include the Forget-Gate, the Input-Gate, and the Output Gate. The Cell state behaves as a "memory" of the network reducing the short-term memory limitations of the traditional RNNs. The information flows through the Cell state and its content can be updated through the gates. The gates determine which information is relevant for the final output and which ones are not. The Forget-Gate controls the retention or removal of information from the Cell state. It returns a value of 0 to indicate that the data is irrelevant, and of 1 if it has to be kept for the final output. The Input-Gate is made of two components: The Sigmoid layer and the Tanh layer. The Sigmoid layer returns a value between 0 and 1 which indicates the importance of each part of the information. The Tanh Layer evaluates the importance of the information combining its evaluation with the result given by the Sigmoid function. The Output-Gate determines the next hidden state of the LSTM. The information is processed by a Sigmoid function and a Tanh Function, their outputs are multiplied together to define which information should be carried by the next hidden state. The updated hidden states are used for the subsequent time step.

3 LSTM and ballet

Traditional ML models or ANNs may struggle with the recognition of long-term dependencies in sequential data. The capacity of an LSTM network to learn temporal dependencies and patterns makes it well-suited for catching the sequential nature of ballet poses in ballet technique. First of all, ballet exercises can vary in length and design based on the specific needs of the dancer. This project aims to create an LSTM network able to capture and retain the relationship between various ballet poses in more advanced ballet exercises. Furthermore, it has been possible to analyse the sequential poses in real-time obtaining immediate feedback from the network. The LSTM network in combination with the MediaPipe holistic model allowed us to have a higher level of accuracy using a smaller dataset, and to obtain a dense neural network allowing for a faster training phase, using a lower number of parameters. Finally, due to its simplicity fast detection of poses was possible in real-time. Applying the MediaPipe holistic model it is possible to estimate the position of the key points of the body in the space, by analyzing video recordings, real-time videos, and still images. The obtained outputs are the coordinates in the space and the reliability of the estimation per frame of the 33 key points represented in Figure[1] below.

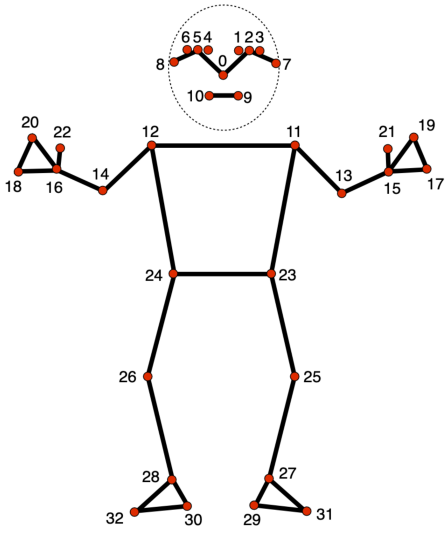

Figure 1: 33 key points estimated by the MediaPipe pre-trained pose detection model.

The Mediapipe holistic landmark allows the detection of gestures, movements, and actions performed by the human body.
There are three possible outputs:

- **the pose landmarks, which are the 33 key points defined above**
- **the face landmarks, which are 468**
- **the hand landmarks, 21 per hand**

The first approach to the problem was to apply the MediaPipe Holistic Landmarker on 30 videos of complex ballet exercises at the barre (Figure [2], [3], [4], [5]).

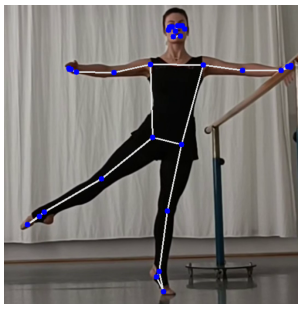

Figure 2: Balance second

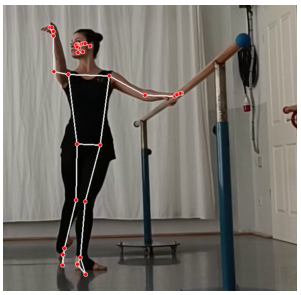

Figure 3: Tendu derriere

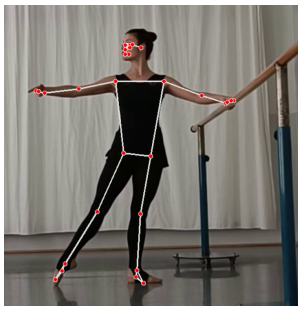

Figure 4: Tendu second

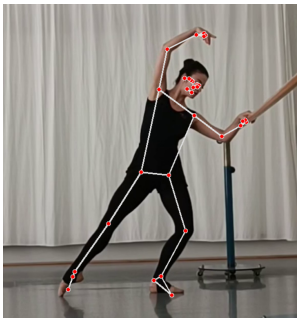

Figure 5: Tendu second

The videos were each more than one minute in length and performed using different musicality, coordination, and sequences. The data could not be used for the aim of the project, because it would have been necessary to label all the steps executed in each video, becoming too expensive in terms of time and computation.
However, it has been possible to apply the MediaPipe model on those recordings to prepare the algorithm applied to the main project. The choice of using real-time video for ballet pose detection was taken due to the necessity of a

normalised dataset and the need for at least 20 videos per movement.
The model was trained to detect six basic movements of ballet technique:

- **demi-plié, performed in first position (bent knees exercise keeping the heels together with the toes facing opposite sides)**
- **battement tendu performed to the side (stretching knee, ankle, and toes of the working leg to the side)**
- **Relevé, performed in the first position, means to rise, lifting the heels off the ground**
- **Attitude croisé, front leg stretched as the standing leg, second leg bent in the air.**
- **First arabesque, both legs stretched, the working leg is lifted back.**
- **Retiré devant, working leg slides on the standing leg reaching the height of the knee.**

The six movements were performed twenty times each and collected in real time. The pose landmarks were saved in large-dimension arrays using the dedicated Python library Numpy. The dataset contains six movements collected across 120 videos, 20 videos of 40 frames for each movement. Each frame is represented by the three coordinates of the 33 body key points and the accuracy of the detection.
In total, the dataset consisted of 4.800 images.

Data Collection The algorithm to collect the dataset and save the data in .npy files required the following tasks: Definition of the functions used throughout the program:

- **Image Processing Function, to process the image to make it usable by the pre-trained model**
- **Pose Drawing Function, to draw the landmarks and the connections between the key points, returning also a visual result.**
- **Extract Point Function, to extract the landmarks and save them as arrays, returning the array of the pose points.**
- **Creation of the DataSet Directory and the directories for the different movements**
- **Video Capture and Collection of the data: the functions defined above were used to detect the six movements which were performed twenty times, to capture 40 frames for each performance of the pose. The extracted points were then saved in .npy files using the Extract Point Function**

Data Preparation A 'label map' was created to label each movement, it consisted of a dictionary that associates each movement to a numerical identifier. To organise the data for the training two empty lists for the frames and their corresponding labels were created. Looping through the movements and the sequences of frames from the dataset, the values from the .npy files were appended to the list of sequences and the corresponding label to the list of labels. Obtaining arrays suitable for the LSTM network training. Before the compilation of the model, the data was split into training and testing data using a 95:5 ratio. A directory for storing the TensorBoard logs to monitor the training of the model, it is possible to notice the two graphs below [6][7], representing the accuracy function and the loss function of the model.

Model's Compilation The neural network model was initialised as a 'Sequential' adding the three LSTM layers with a RELU function each. Two fully connected layers were governed by a RELU activation function. The final fully connected layer with a softmax activation function gave the probabilities for each movement as the output. The optimisation function used was the Adam optimiser, a stochastic gradient descent method that has little memory requirement and is well suited for problems with a large number of parameters.

Model's training The model was trained using the 95% of data defined earlier as training data for 2000 epochs. It was possible to check the model's training process by calling the Tensorboard monitor.

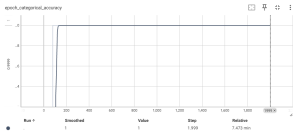

Figure 6: epoch accuracy graph

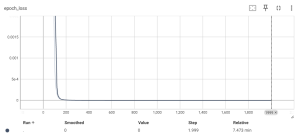

Figure 7: epoch loss graph

Figure 8: demi-pliè

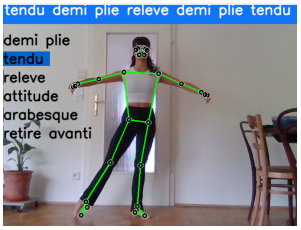

Figure 9: tendu second

Figure 10: relevè

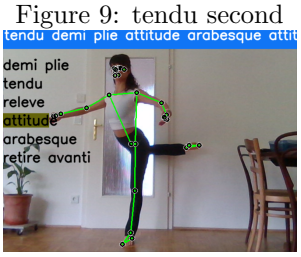

Figure 11: attitude croisè

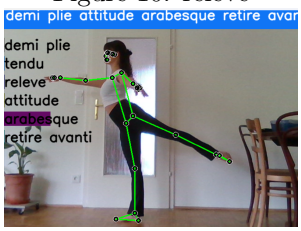

Figure 12: first arabesque

Figure 13: retirè devant

Conclusions Finally, using real-time detection and visualisation the program was able to predict the probability of the detected movement. If the probability was more than 80% the movement was added to a sequence of movements, to create an exercise of five movements.
It is possible to appreciate the obtained results above in Figures [8], [9],[10],[11], [12], [13]. The network can detect correctly the ballet movements with precision. At first, there were issues in detecting similar poses and mistakes in the detection of movements during the change of position. Adjusting the parameters and using a higher threshold made it possible to get more accurate detection. The five-movement exercise can be used to create synthetic exercises for the dancer's training. The generation of new exercises can be more efficient if the generative network is supported by the presented LSTM RNN.

4 Litterature

The idea of applying an LSTM RNN to classify dance poses for an easier approach to dance training is not unique to just this paper. Papers that present a similar approach have been published in the last few years.

Study on Dance Style Classification by LSTM RNN Yanika Misfud and Frankie Inguanez published a study on Dance Style Classification by LSTM RNN. Using OpenPose library for the pose estimation and an LSTM Network to classify Ballet and Breakdance styled choreography. The study was published in 2021 in IEEE 11th International Conference on Consumer Electronics

Advanced Dance Choreography System Using Bidirectional LSTMs Published by Hanha Yoo and Yunsick Sung. The paper presents a study to generate K-pop choreographies. The dataset consisted of dance videos collected in advance.

DanceNet Dance generator using Variational Autoencoder, LSTM, and Mixture Density Network. (Keras). A study published by Jainson Saji Chacko and Charlie Harrington for dance generation using a Variational autoencoder and an LSTM + Mixture Density Layer.

A Basic Study on Ballroom Dance Figure Classification with LSTM Using Multi-modal Sensor The paper presents a ballroom dance figure classification method with LSTM using video and wearable sensors. The study was conducted by Hitoshi Matsuyama, Kei Hiroi, Katsuhiko Kaji, Takuro Yonezawa, and Nobuo Kawaguchi.

5 Conclusions and future developments

The project proposes a modular structure for movement detection and classification, using the LSTM architecture for time sequence movement recognition. The network proved to be efficient despite the limited availability of data.
Collecting data in real-time gave an overview of the collecting process and model testing. The classification of the pose was returned only if the accuracy value of the detection was more than 80% probability. This high threshold guaranteed a high reliability of the program in building the movement sequence.
Despite the challenges presented by overtraining and the mistakes made by the model in the detection between movements, the model showed accurate results when the parameters were set correctly.
The trained model could be used in future developments, such as part of a Generative Adversarial Network for the generation of synthetic dance exercises, as a base for a reliable and robust discriminator. In the dance field, this project could be developed to guarantee training assistance for the artists, choreographers, and teachers. It could also be developed for injury prevention and an educational tool for beginners.
Dedicated improvements could help artists preserve their physical health and strength, creating training sequences adaptable to various training and performance environments and situations.
In conclusion, the project highlights the significant impact that these technologies could have on our society. Such networks could be a powerful training tool for artists and athletes to help them train more creatively and safely.

6 Bibliography

- https://developer.ibm.com/tutorials/iot-deep-learning-anomaly-detection-1/
- https://developers.google.com/mediapipe
- https://www.ibm.com/topics/neural-networks
- https://mediapipe-studio.webapps.google.com/demo/pose-landmarker
- https://en.wikipedia.org/wiki/Ballet
- https://en.wikipedia.org/wiki/Glossary-of-ballet
- https://en.wikipedia.org/wiki/Long-short-term-memory
- Mediapipe and CNNs for Real-Time ASL Gesture Recognition, 2023
- Stanford CS229 Machine Learning Full Course
- https://github.com/nicknochnack/ActionDetectionforSignLanguage