Style transfer for human motion with transfer with adversarial learning

Sebastian Stelter

July 6, 2021

Abstract

In this paper, the concept of using generative adversarial networks for creating human motions in different styles is explored. By extending an adversarial sequence auto-encoder with a conditional input, motions can be transferred into different emotions. This way, new unique motions can be generated easily.

1 Introduction

In the context of humanoid robotics, motion is one of the most important components. Most of the motions a human can conduct, e.g. walking or throwing, are extremely complex and, therefore, most conventional ways of generating motions cannot hold up on a larger scale.

Modern humanoid robots have at least 20 degrees of freedom (DoF). Creating a motion by, for example, using keyframe animations would therefore require manually fine tuning 20 motor positions for each frame, with each animation often having 20 or more keyframes. Due to the amount of time necessary to create such a motion, this concept is not feasible for anything other than a quick prototype.

As an alternative, most researchers currently resort to motion tracking solutions, which reduces the need for manual labor, as well as intuition about the desired motion. However, motion tracking requires hiring actors, as well as getting access to expensive equipment, which is often unfeasible for smaller projects.

One way of resolving these problems might be generative adversarial networks (GANs) [1]. GANs consist of two different networks. The generator is trained to create motions that are indistinguishable, yet different from the training set. Simultaneously, a discriminator network is trained to distinguish between real data and generated data. As both networks counteract each other, they improve by learning from their counterparts. This way, new realistic motions can be generated quickly from a given set of known motions.

An approach like this also allows for more sophisticated generator patterns, like transferring known motions to different robot platforms, conducting motions

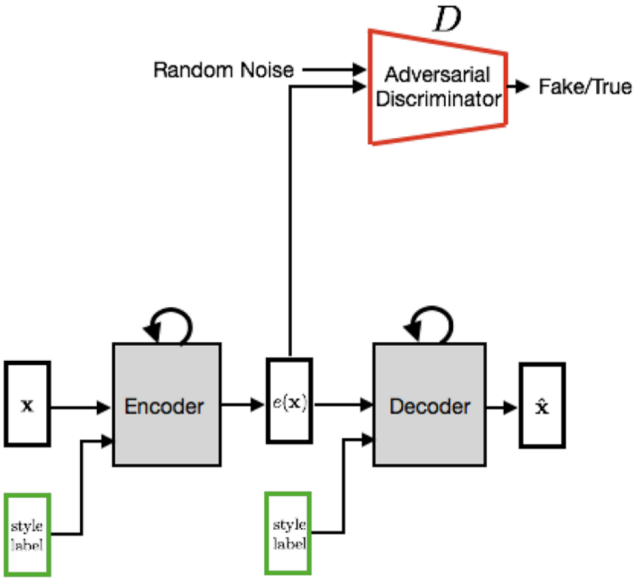

Figure 1: Schematic representation of a conditional adversarial sequence auto-encoder. Modified from [2].

with different styles or emotions, and continuing a motion. When aiming at making humanoid robots act as realistically as possible, emotion is a huge factor. Therefore this paper mainly focuses on creating a conditional adversarial sequence auto-encoder (CASAE) that can generate new motions from a given input emotion and motion name.

In section 2 an overview of GANs and SAEs will be given. Section 3 displays related work in the field. Section 4 explains the model used in this paper. In section 5 the results of the experiment will be discussed, while section 6 concludes the paper.

2 Background Information

The model used in this paper is a conditional adversarial sequence auto-encoder, one variant of the traditional GAN. A GAN consists of two opposed neural networks, one generator and one discriminator. The task of the generator is creating a target output from some latent input, while the discriminator distinguishes these generated outputs from real outputs. While the discriminator is rewarded by correctly distinguishing real data from generated, the generator is rewarded by fooling the discriminator. This way, both networks iteratively improve each other.

The adversarial auto-encoder [3] used here basically works the opposite way. An encoder network converts real data into smaller latent vectors. These vectors are then used as an input for a decoder network that attempts to recreate the real data from the latent vector. A discriminator is used to control the improvement

of the networks. As the latent vectors are smaller than the input data, some information has to get lost during the encoding step, which makes it impossible to perfectly recreate the input data. Therefore an auto-encoder will always generate data that is similar to the input, but not the same.

As we are dealing with motions in this paper, we extend our network to be sequential. This means, the input data has an additional dimension along time. In order to use this information, the network has to be able to relate to the past. This can be done by using long-short-term memory (LSTM). LSTM cells, compared to traditional neurons, can keep an internal state and have an additional forget gate, allowing data to pass through unchanged. This makes them suitable to process sequential data while also avoiding the vanishing gradient problem[4].

In order to make the network adversarial, we add a discriminator between the encoder and decoder. This discriminator is bound to the encoder reward function and directs the encoder in a way to create the latent vectors according to a predefined distribution. By doing this, the decoder can later be used independently with any latent vector from that distribution to generate new outputs.

Finally, a conditional aspect is added: Both encoder and decoder accept two inputs. Besides the sequential motion data for the encoder and the sequential latent vector for the decoder, both networks also accept a conditional vector that holds additional information. This vector can later be used to determine which kind of motion our decoder should generate. A full overview of the CASAE network type can be seen in figure 2.

3 Related Work

Generative adversarial networks and sequence auto-encoders have been used for various machine learning tasks in the past, most of them focusing on image generation or enhancement. In the realm of GANs, StackGAN [5] has been used to successfully and efficiently synthesize images from text input. StarGAN [6] excels in image-to-image translation, generating images of one person or animal while keeping features of a reference image. DeNoiseGAN [7] is capable of reducing noise on an image, making it easier to then recognize objects.

Auto-encoders have also been for different tasks, such as text compression or video anomaly detection. For instance, the KATE [8] auto-encoder can learn a meaningful representation of text data. The network proposed by Zhao et. al. [9] managed to detect anomalies in traffic surveillance footage, outperforming other approaches to this problem.

In the area of humanoid motions GANs and SAEs have not been represented quite as prominently, however several other types of networks have been used to generate motions. The DeepMimic Network uses reinforcement learning with additional goal inputs to generalize motion tracking data to dynamic motions. It reliably manages a variety of tasks, such as kicking a target while performing spin-kicks and traversing rough terrain. The Mix-StAGE [10] network generates hand motions that naturally correlate to spoken text. The CASAE shown in this

work is based on a paper by Wang et. al. [2], which also highlights various other networks solving different motion generating tasks. Finally, the Adult2Child network [11] encodes motions recorded by adult actors into a representation of a child-like motion. This system might eventually replace the need for child-actors in animations or video games.

4 Approach

While the original paper displayed various network types, this paper solely focuses on recreating the CASAE network. To do so, the Keras [12] library is being used.

In a first step, the training data had to be converted into a format that can be interpreted by the networks. We use the body movement library for the training process, which contains data in the `csm` format. As this format is mostly human readable, some simple text parsing transforms our data into numpy arrays containing either the motion data or the label data. The motion data consists of numerical position values for each joint in each frame, while the label data uses a one hot encoding to represent motion type and emotion. We furthermore store some metadata with each motion, in order to correctly reconstruct the `csm` file later. As it is unfeasible to hold the entire dataset in memory at once, a helper function remembers, which files have not yet been used for training and sequentially reads data samples when needed. As learning a motion in its entirety is difficult, with motions of the training set often containing thousand or more frames, and to avoid different length samples, training samples are generated by moving a window across each file, splitting it into multiple smaller size samples. While this process indicates, that all samples of one file have to be processed, before moving on to the next, which biases the training set, this restriction seems reasonable, as these samples naturally should follow each other and should not impact the training process too much.

The CASAE itself consists of three different networks, whose structure can be seen in figure 4. The Encoder and Decoder networks are structured similarly, but inverted. Since the motion vector and the label vector are different shapes, the label vector is repeated to fit the length of the motion vector for the encoder. The vectors are then concatenated and fed through multiple LSTM layers with tanh activation functions, until finally passing through a linear dense layer to create the latent vector.

This data is passed on to both the discriminator and the decoder. The discriminator is first trained to distinguish between latent codes from a gaussian distribution and latent codes generated by the encoder. This forces the encoder to create gaussian latent vectors. The discriminator has dense ReLU layers, alternating with dropout layers to avoid overfitting.

The generated latent vectors are also passed into the decoder, along with the same label vector, which is again repeated to fit the data. After concatenating both inputs, they are again fed through several LSTM layers, with the opposite order of shapes compared to the encoder. The last layer of the decoder is a sigmoidal

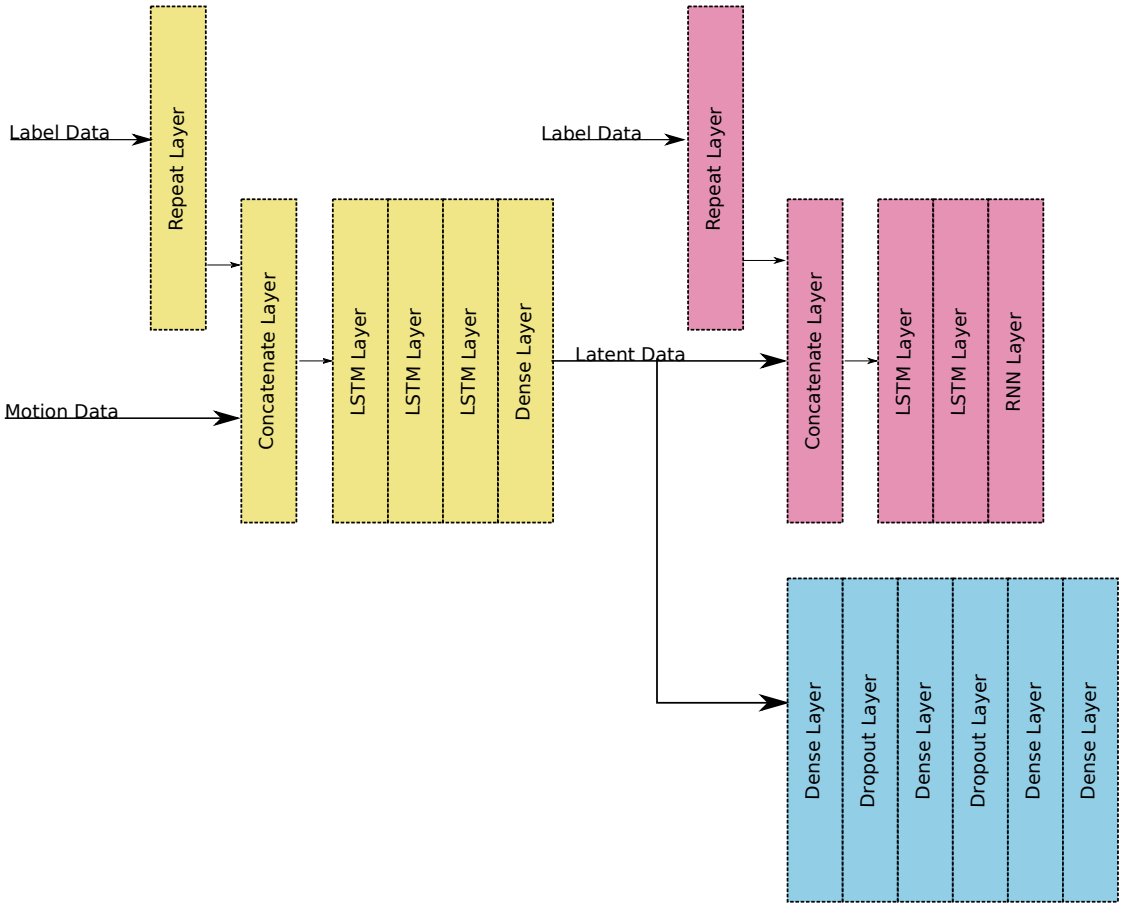

Figure 2: Structural model of the CASAE. The encoder is displayed in **yellow**, the decoder in **red** and the discriminator in **blue**.

RNN which fits the data back onto the original shape. After training, new motions can be generated by invoking the decoder with a label vector and a random latent vector. The network is compiled using a SGD optimizer.

5 Evaluation

After training the network, the results were evaluated by manually inspecting generated samples and comparing them to their recorded counterparts. However, as can be seen in figure 3 and 4, even after thorough training, the network was not able to generate reasonable motions. Instead the robot only seems to curl up into a ball and then teleport into the upper left corner, where it stops moving.

The reasons for this failure could be various and are difficult to discern. One of the reasons could be the chosen network size, as well as the size of the latent vectors. If the latent vectors are too small, not enough information can be stored in these vectors that would allow a reasonable reconstruction of the motion.

Another problem could be the shape of the chosen dataset. As the motion tracking samples have been recorded with different setups and therefore partially

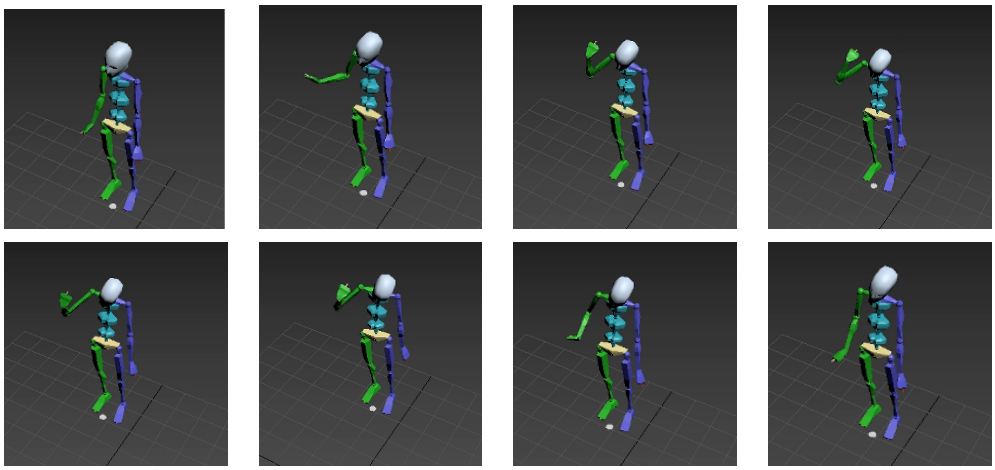

Figure 3: Snapshots from one training sample in 3DS Max using the default model. The displayed motion is labeled as "knocking", the emotion is labeled as "angry"

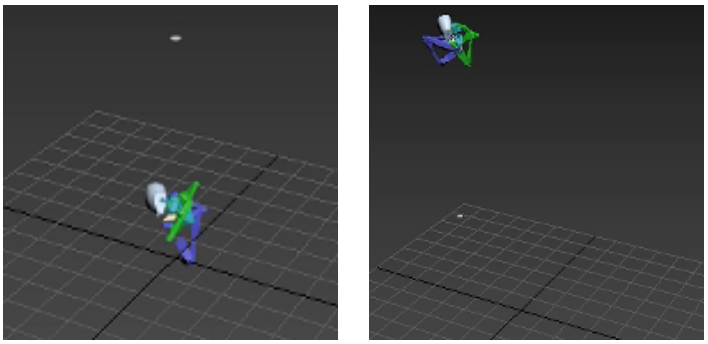

Figure 4: Snapshots of a generated sample from the CASAE. Target motion was "angry knocking".

include recordings with different amounts of joints, the overall size of the dataset is reduced to only 1323 different recordings. Spreading this over 25 possible combinations of emotion and motion, this simply does not leave many examples to learn from. Furthermore, the data is presented in the `.csm` format, which is a strong abstraction from classical keyframe data and heavily relies on two calibration files. These files were not presented during training, nor generated with the samples. This missing information could have lead the network to learning wrong patterns and generating motions, that do not work with the calibration file used for testing. Using a simpler file format would have avoided these issues.

6 Conclusion

Overall, the generation of specific motion - emotion combinations using CASAE seems possible and might be a promising approach, looking into the future. How-

ever, the experiments in this paper also demonstrate, that this is not the case for all possible setups. A lot of consideration has to be put into the right network shape and the right data.

As a proof of concept, this paper shows, that motions can be generated using conditional adversarial sequence auto-encoders. And with some more fine tuning and more carefully selected data, more realistic goals could be achieved. This has also been shown in the paper this work is based on [2], which performed significantly better.

The source code for this work can be found at `https://github.com/16stelter/NN-Seminar`.

Acknowledgements

This work is in large parts based on the "Adversarial learning for modeling human motion" paper by Wang et. al. [2]. This paper further uses the Body Movement Library provided by Ma et. al. [13].

References

[1] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[2] Qi Wang, Thierry Artières, Mickael Chen, and Ludovic Denoyer. Adversarial learning for modeling human motion. *The Visual Computer*, 36(1):141–160, 2020.

[3] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[5] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915, 2017.

[6] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[7] Daniel Speck, Pablo Barros, and Stefan Wermter. De-noise-gan: De-noising images to improve robocup soccer ball detection. In *International Conference on Artificial Neural Networks*, page 738–747. Springer, 2018.

[8] Yu Chen and Mohammed J Zaki. Kate: K-competitive autoencoder for text. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 85–94, 2017.

[9] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1933–1941, 2017.

[10] Chaitanya Ahuja, Dong Won Lee, Yukiko I Nakano, and Louis-Philippe Morency. Style transfer for co-speech gesture animation: A multi-speaker conditional-mixture approach. In *European Conference on Computer Vision*, pages 248–265. Springer, 2020.

[11] Yuzhu Dong, Andreas Aristidou, Ariel Shamir, Moshe Mahler, and Eakta Jain. Adult2child: Motion style transfer using cyclegans. In *Motion, Interaction and Games*, pages 1–11. 2020.

[12] François Chollet et al. Keras. `https://keras.io`, 2015.

[13] Yingliang Ma, Helena M Paterson, and Frank E Pollick. A motion capture library for the study of identity, gender, and emotion perception from biological motion. *Behavior research methods*, 38(1):134–141, 2006.