Human Motion Prediction - Project Report

Ming-Da Liu Zhang
lming@student.ethz.ch

Harleen Hanspal
hhanspal@student.ethz.ch

ABSTRACT

Human motion prediction is forecasting future human body dynamics from visual observations. The number of applications where human motion prediction plays an integral role such as self-driving cars, surveillance, human-computer interaction, virtual reality, planning and predicting sports strategies in real time, etc. is growing and so is the interest among the graphics and computer vision community towards to this problem. Several works based on deep recurrent neural networks (RNNs) to model human motion and learn time-dependent representations for both short-term motion prediction and long-term human motion synthesis have been done to a reasonably good accuracy. In this project, we implement the sequence-to-sequence architecture based on [5] as the base and try to improve its accuracy by trying variants like Bidirectional RNN in place of simple RNN units or adding attention on top of the model, compare results and refine our understanding of RNNs. We find the Bidirectional RNN variation to work as good as the basic RNN model and the convergence was slightly faster. Adding attention does not improve the results.

1 INTRODUCTION

Human motion, stating simplistically, is the result of physical limitations (e.g. torque exerted by muscles, gravity, moment preservation), intentions of subjects (how to perform an intentional motion) and spontaneous reactions to situations, motivating the graphics and computer vision community to learn modeling this complex task from observations like static images, videos or motion capture (mocap) data. This interest is obvious given the multitude of fields where human motion prediction plays an integral role such as self-driving cars, 3D people tracking in surveillance, human-computer interaction, virtual reality, planning and predicting sports strategies in real time, etc. Several works based on deep recurrent neural networks (RNNs) to model human motion and learn time-dependent representations for both short-term motion prediction and long-term human motion synthesis have been done to a reasonably good accuracy. In this project, we implement the sequence-to-sequence architecture based on [5] and try to improve its accuracy with variants like Bidirectional RNN in place of simple RNN units or adding attention on top of the model.

1.1 Statement of Contributions

For better understanding of deep architectures such as RNN, we implement and compare results between Martinez's model and our variants. We found that Bidirectional RNN variation was as good as the basic RNN model and the convergence during training for BiRNN was faster (as shown in Figure 2). We also experimented with attention since for most practical scenarios, this mechanism seems promising [11, 12] and therefore, we term its implementation our second contribution. This mechanism does not improve the results because it is inadequate for the setting.

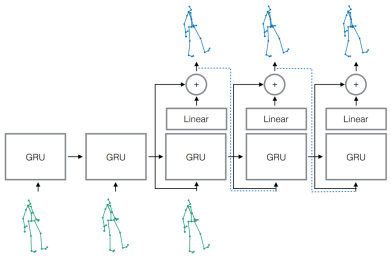

Figure 1: Model presented in Martinez et al. [5]. In this image, green figures represent ground truth, and blue figures represent predictions. This is the sequence-to-sequence architecture. During training, the ground truth is fed into the encoder network. The decoder feeds its own predictions and has a residual connection to model angle velocities.

2 LITERATURE REVIEW

In the last decade, a lot of work focused on modeling human motion using latent variable models like HMMs, Expectation Maximisation over Gaussian Mixtures, parametric temporal filters like Kalman Filter and its variants, etc. to model nonlinear motion. The use of these statistical models has limitations such as number of latent variables in the models grow quadratically with the number of states required to model more complex and non-linear dynamics. Optical flow and activity-specific spatio-temporal graph based methods try to leverage the inherent regularity and context dependency of natural motion, but they require a lot of human input in terms of parameters and case based predictions and hence is difficult to scale up to the various situations that can be encountered. Due to the above problems, motion modelling that are based on deep RNNs seems more promising. Architectures such that LSTM-3LR (3 layers of Long Short-Term Memory cells [6], ERD (Encoder-Recurrent-Decoder with non-linear space encoders for data pre-processing) [6], Dropout Autoencoder LSTM [10], or the more recent Convolutional Sequence to Sequence Model [1], etc. have shown decent success. A recent paper using a simple RNN based model trained on a Human coupled with the three modifications (explained below) [5] showed to have the best results for most activities. The modifications are shared weights for the encoder and decoder to accelerate convergence and produce results closer to reality; sampling based loss where the decoder takes its own samples as input to produce a sequence, which avoids hyper-parameter tuning and trains the model to recover from its own mistakes rather than feeding it with ground truth at each training time-step; and a residual connection between the input and the output of each RNN cell to model velocities. The second modification helps the model to produce plausible motion in the long-term while the third modification helps in minimizing the short-term error, especially the first frame discontinuity, both of which are generally traded off for one another. As an added advantage, this model gives better

results on large and diverse training datasets like Human3.6M than on small action-specific subsets.

3 METHOD

We first present the architecture for human motion prediction used in [5] as seen in Figure 1. Inspired on this model, we also present two variations. In the first one, we use a bidirectional encoder [8], and in the second one, we use the attention mechanism [2].

3.1 Main model

Sequence to sequence architecture. We address the human motion prediction as a sequence to sequence (seq2seq) problem [4]. A typical seq2seq model has two parts: an encoder and a decoder. The encoder is a network whose task is to understand the input sequence, and create an internal representation of it. This representation is then fed into a second network, the decoder, that generates the output.

Sampling based loss. RNNs are generally trained to maximize the likelihood of generating a target sequence given an input. In practice, this is done by maximizing the likelihood of each target, given the current state of the model (which summarizes the past) and the previous true target (ground truth). However, there is a discrepancy for inference: since true previous targets are unavailable, they have to be replaced by the predictions generated by the model itself. This causes a problem: mistakes made early are fed as input to the network and can be amplified because the model might be in a situation that it has never seen at training time. We address this problem by making the decoder produce a sequence by always taking as input its own predictions. This avoids hyper-parameter tuning, unlike other methods that rely heavily on it (e.g adversarial network or scheduling sampling [13]).

Residual architecture. A residual connection is added between the input and the output of each RNN cell. It helps in modeling prior knowledge about the statistics of human motion. Motion continuity is easier to express in terms of velocities than in poses. It solves the discontinuity problem between the conditioning sequence and the prediction [5].

3.2 Variations

Attention mechanism. A potential issue with the encoder decoder approach is that the neural network needs to be able to compress all the necessary information of a source sequence into a fixed-length vector. This may make it difficult for the neural network to cope with long sequences. The attention mechanism is an extension of the encoder-decoder model. Each time the proposed model generates a new pose, it searches for a set of positions in a source sequence where the most relevant information is concentrated. The model then predicts a target pose based on the context vectors associated with these source positions and all the previous generated target poses. The most important distinguishing feature of this approach from the basic encoder-decoder is that it does not attempt to encode a whole input sequence into a single fixed-length vector. Instead, it encodes the input sequence into a sequence of vectors and chooses a subset of these vectors adaptively while decoding. This frees a seq2seq model from having to squash all the information of a source sequence, regardless of its length, into a fixed-length vector [2].

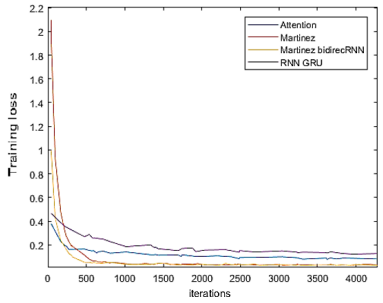

Figure 2: Training loss L2 per time step

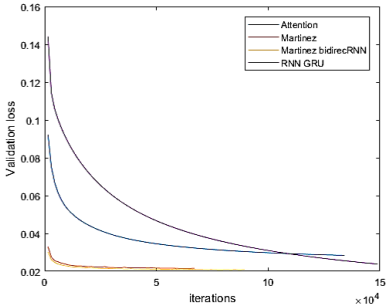

Figure 3: Validation loss L2 per time step

Bi-directional encoder. Bi-directional RNNs takes the RNN further and implements it in two ways: forward and backward directions over the input sequence [9]. The output vectors of the two models are normally concatenated as one to be used for the RNN model in the decoding part. The use of providing the bi-directional sequence was justified in the domain of speech recognition because there is evidence that the context of utterance is used to interpret what is being said rather than a linear unidirectional interpretation [3]. We believe that the bidirectional structure will provide a better context by effectively introducing additional short term dependencies. Learning a sequence of poses from the future to the past can be useful in predicting new poses.

4 EXPERIMENTS

We evaluate our models on the provided dataset. It contains 162 samples with variable sequence lengths.

4.1 Implementation Details

Data processing. The given skeleton code provides the data in an ineffective way: it separates each sample in disjoint chunks of a specified length. For example, if the total length of a sample is 800 frames, and the chunk length is 75, it will provide 11 chunks of length 75, having the last chunk padded with 50 zeros. This way of feeding the data may be problematic, since the model will not learn the intermediate sequences between different chunks. We handle this problem by sliding the chunks one by one through the entire sequence. In the last example, that means that the first chunk is composed by $x_0, x_1, \ldots, x_{74}$, the second chunk by $x_1, x_2, \ldots, x_{75}$,

Model	MSE
RNN_GRU	0.04453
RNN_GRU-one-hot	0.05921
Martinez	**0.01713**
Martinez_bi	0.01751
Attention	0.04746

Table 1: Kaggle score

and the last chunk by $x_{725}, \ldots, x_{799}$. With this approach, we solve the issue of the intermediate sequences, and we avoid to deal with the padding. Since the task is to predict the next 25 frames given 50, we use chunks of length 75: we feed into the encoder the frames $x_0, \ldots, x_{48}$, the decoder receives $x_{49}, \ldots, x_{73}$, and we compare its outputs with the ground truth composed of $x_{50}, \ldots, x_{74}$.

Training details. In our experiments, we use GRU with 1024 units, GRU cell is used instead of LSTM cells as it is computationally less expensive. We borrowed the extended GRU cell from Martinez et al. [5]. This cell implements the residual connection, and maps the 1024-dimension output of the GRU into the 75-dimension vector used for pose representation. We use a learning rate of 0.005 during training, clip the gradients to a maximum L_2-norm of 5, a decay rate of 0.95 every 10000 iterations, and a batch size of 128.

Models. At the beginning of the project, we were not very familiar with Tensorflow, so we implemented our models in an incremental complexity way.

(1) **RNN_GRU.** We started first with a basic RNN using a single GRU with 1024 units. We also tried to implement one-hot encoding, taking into account the provided actions labels from the data. The approach taken for the implementation of the one-hot worsen results, so we did not take it into account for the remaining models (more in Discussion).
(2) **Martinez.** Then we implemented the model presented in Martinez et al. [5]. Here we enforced weight sharing between the encoder and decoder to accelerate convergence.
(3) **Martinez_bi.** On top of Martinez, we implemented a bidirectional RNN on the encoder. We also enforced weight sharing between the bi-directional encoder, and the decoder. Moreover, instead of concatenating the forward and backward states of the encoder, we sum them to keep the 1024 dimensions. With this we avoid adding another dense layer to map it to the cell size required for the decoder.
(4) **Attention.** At last, we added the attention mechanism to the Martinez model. As in the related literature, we let the decoder to choose from the encoder outputs, and determine which of them are more useful for the prediction.

4.2 Results

In Table 1, we show the scores obtained in each model. The best result is highlighted in bold letters. Martinez obtains the best accuracy. We can see that our new approach Martinez_bi obtains a similar state of the art score as Martinez, while the remaining models are not even close to the easy baseline (0.02670). We can also see that adding a one-hot encoding decreases the performance of the RNN_GRU model, so we did not include it for the remaining models. Finally, the Attention model obtained results as bad as RNN_GRU.

https://github.com/una-dinosauria/human-motion-prediction

5 DISCUSSION

RNN_GRU. The first two models (RNN_GRU and RNN_GRU-one-hot) did not pass the easy baseline, since they lacked of enough complexity to predict human motion and were not able to capture the human motions. It faced problems like the first frame discontinuity described in [5], which resulted in a big and cumulative error in the prediction. Moreover, the one-hot encoding decreased the performance. At first, we concatenated a 15 dimensional one-hot vector to each 75 dimension vector pose, and tried to predict the one-hot vector as part of the output (another 90 dimensional vector). This resulted in disastrous results, since the task became an action classification with the additional task of computing the next pose. The result RNN_GRU-one-hot shown in the table is obtained by using one-hot encoding in the input, but removing it from the target. This fixes the previous problem, but we experimentally got worse results than RNN_GRU without one-hot encoding.

Martinez and Martinez_bi. It is not surprising that Martinez passed the hard baseline, since this model provides state of the art results. We were expecting Martinez_bi to have a stronger performance, since it was supposed to add more context information to predict the poses, but we experimentally got similar results. On the other hand, we can appreciate in Figure 2 and 3 that it takes more time for Martinez_bi to converge, since it has more parameters because of the backward RNN, which means that this model may be a bit inferior than Martinez.

Attention. Adding the attention mechanism did not end up with good results. Attention-based recurrent networks have been successfully applied to a wide variety of tasks, such as handwriting synthesis [3], machine translation [2], image caption generation [7] and visual object classification [14]. Such models iteratively process their input by selecting relevant content at every step. This allow them to have a more general context of the sequence, and not forget about the initial states by just using a final state of the encoder. In those settings, attention is be useful since the correlation may not be just at the end of the sequence, but also in other parts of it. After our experiments, we believe that in the human motion setting, the most important information come from the most recent outputs. Prior information from the beginning of the sequence is not needed that much to predict the next pose, so it is affordable to forget about it. This would explain why attention works so bad in the model. Using the attention mechanism just translated in having more parameters to fit and making the model more complex.

6 CONCLUSION

We have confirmed that a sequence to sequence architecture with residual connections, when trained on a sample-based loss, achieves state of the art results. We also have shown that trying to make a more complex model based on this one ends up making the architecture harder to train, slower, and with worse results. On the other hand, the predictions we obtained, generally have a slower motion than the given sequence. Modeling the velocities in a different manner to address this issue is left for future work.

ACKNOWLEDGMENTS

We would like to thank the Advanced Interactive Technologies Group for the provided skeleton code, valuable help and helpful suggestions.

REFERENCES

[1] Wee Sun Lee Gim Hee Lee Chen Li, Zhen Zhang. 2018. Convolutional Sequence to Sequence Model for Human Dynamics. *Computer Vision and Pattern Recognition* (May 2018).

[2] Yoshua Bengio Dzmitry Bahdanau, Kyunghyun Cho. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).

[3] A. Graves. 2013. Generating sequences with recurrent neural networks. *arXiv 1308.085* (2013).

[4] Quoc V. Le Ilya Sutskever, Oriol Vinyals. 2014. Sequence to Sequence Learning with Neural Networks. *NIPS* (2014).

[5] Javier Romero Julieta Martinez, Michael J. Black2. 2017. On human motion prediction using recurrent neural networks. *Computer Vision and Pattern Recognition* (2017).

[6] Panna Felsen Jitendra Malik Katerina Fragkiadaki, Sergey Levine. 2015. Recurrent Network Models for Human Dynamics. *Computer Vision and Pattern Recognition* (2015).

[7] Ryan Kiros Kyunghyun Cho Aaron Courville Ruslan Salakhutdinov Richard S. Zemel Yoshua Bengio Kelvin Xu, Jimmy Lei Ba. 2015. Show, attend and tell: Neural image caption generation with visual attention. *International Conference on Machine Learning*. (2015).

[8] Richard Zemel Mengye Ren, Ryan Kiros. 2015. Exploring models and data for image question answering. *NIPS* (2015).

[9] Kuldip K. Paliwal Mike Schuster. 1997. Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing 45, no. 11 (1997): 2673-2681* (1997).

[10] Emre Aksan Otmar Hilliges Partha Ghosh, Jie Song. 2017. Learning Human Motion Models for Long-term Predictions. *Computer Vision and Pattern Recognition* (2017).

[11] Roland Memisevic Samira Ebrahimi Kahou, Vincent Michalski. 2016. RATM: Recurrent Attentive Tracking Model. *Learning* (2016).

[12] Roland Memisevic Samira Ebrahimi Kahou, Vincent Michalski. 2017. Temporal Attention augmented Bilinear Network for Financial Time-Series Data Analysis. *Computational Engineering, Finance, and Science* (Dec 2017).

[13] Navdeep Jaitly Noam Shazeer Samy Bengio, Oriol Vinyals. 2015. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. *Advances in Neural Information Processing System* (2015).

[14] Alex Graves Koray Kavukcuoglu Volodymyr Mnih, Nicolas Heess. 2014. Recurrent models of visual attention. *Advances in neural information processing systems.* (2014).