Human Motion Prediction by tessellating Attention weights Machine Perception FS 2020

Rafael Bischof
rabischof@student.ethz.ch

Tristan Cinquin
tcinquin@student.ethz.ch

Stanislas Gal
stagal@student.ethz.ch

ABSTRACT

Human motion prediction is a key computer vision task, involving complex concepts such as human body structure, spatial and temporal dependencies and repetitive patterns. In this paper, we introduce TesselNet, an attention-based model that leverages the repetitive aspect of human motion. TesselNet's foundations are bidirectional LSTM embeddings as well as vanilla synthesized multihead spatial and temporal attention layers. TesselNet uses the novel approach of predicting future movement by combining past poses using attention weights in the *Recovery* module, thus enabling better predictions for repetitive movements. We complete this module by an engineered *Discovery* module, responsible for the prediction of irregular movements. These methods allow to build an efficient model with less than 1 million parameters (866'862) which achieves good performance and produces natural-looking motions.

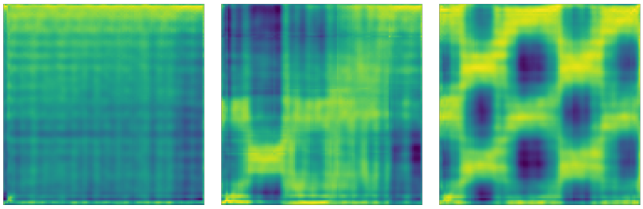
1 INTRODUCTION

Recently, the emergence of autonomous systems such as self-driving cars, gave rise to a vast number of new challenges, among which human pose estimation and motion prediction are of primary importance. When driving through crowded and busy areas, self-driving vehicles need to precisely model pedestrian movement in order to anticipate and avoid collisions.

In the literature, human motion prediction has predominantly been tackled using Recurrent Neural Networks (RNNs) combined with various regularization mechanisms [4][5] or methods incorporating prior knowledge about human morphology and its dynamics [2][7].

Recent progress in sequence-to-sequence translation, especially in Natural Language Processing (NLP), has seen the popular LSTM [6] being replaced by attention-based architectures such as Transformers [10]. This trend has also found its applications in human motion prediction. Unlike recurrent units, which suffer from catastrophic forgetting, attention models have shown to be more robust, especially for repetitive movements [1].

As illustrated by Figure 1, propagating or attenuating information based on the dot-product similarity allows the Transformer to combine previously seen poses to indefinitely extend a repetitive movement. Furthermore, stacking many self-attention layers enables the model to produce new poses in case the movement is non-repetitive. However, this architecture suffers from its high complexity. The need for depth, obtained by superposing several attention layers both in the encoder and decoder, quickly induces multiple millions of parameters. Given our small dataset, this level of complexity cannot be leveraged in this project.



(a) Static **(b) Irregular** **(c) Repetitive**

Figure 1: Attention weights over time of three different movements of the left elbow

2 METHOD

Our approach seeks to use past poses to predict future frames, while keeping the complexity of the model as low as possible. Consider the matrix of Attention weights $A \in \mathcal{R}^{T \times T}$ in Figure 1c. Thanks to the distinctive pattern in the image, it's possible to estimate the movement's period [3] and therefore easily predict t additional rows $A' \in \mathcal{R}^{t \times T}$ by extending the repeating shapes. The entry $a_{ij} \in A'$ then represents the amount of attention that frame f_{T+i}, $0 \leq i < t$ should pay to frame f_j, $0 \leq j < T$. In case of a perfectly repetitive movement, we could predict the future frames F' by calculating $F' = A'F$, $F \in \mathcal{R}^{T \times D}$, with D the dimensionality of the input. Since in our case $T = 120$ and $t = 24$, the task translates into taking a matrix of Attention weights $A \in \mathcal{R}^{120 \times 120}$ and predicting $A' \in \mathcal{R}^{24 \times 120}$. We propose to solve this by using a comparatively cheap encoder-decoder module with convolutional layers and incorporate it into an Attention-based architecture.

To the best of our knowledge, this network is the first of its kind. We name it TessellNet as its core task is to tessellate mosaical patterns of Attention weights.

3 DATA

To handle the given task, we use a dataset composed of pre-recorded human motion sequences. Within these sequences, human poses are encoded using the SMPL skeletal representation, where each major joint is parameterized as a rotation of its parent joint. This rotation is represented by a 3×3 matrix, and the skeletal representation is made of 15 major joints (hips, 3 vertebrae along the spine, knees, neck, clavicles, head, shoulders and elbows).

Many of the provided training sequences are longer than our required length of 144 (seed sequences 120 frames (2 s), target ground-truth sequences 24 frames (400 ms)). We therefore generate our training batches by randomly sampling *batch-size* sequences following a probability distribution proportional to the sequences' length and then pick a random window inside the chosen items. Concerning the sequences shorter than 144, we randomly left-pad them either with a static pose, or by mirroring the entire sequence until

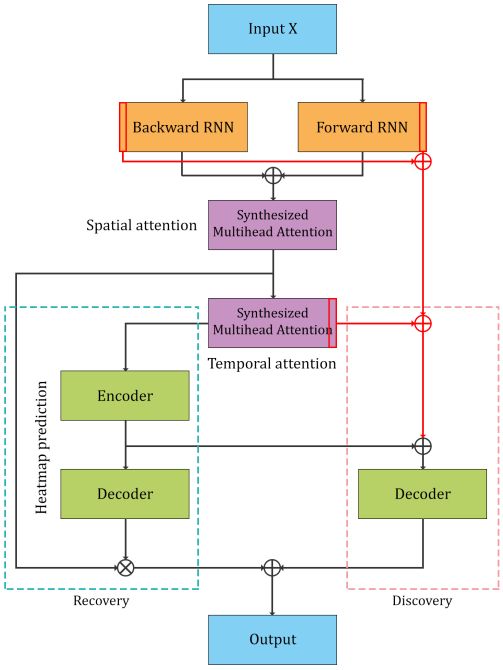


Figure 2: Illustration of TessellNet

getting the desired length. This procedure allows us to augment our training data from less than 5'000 to over 4 million samples.

4 BASELINES

In order to evaluate the performance of TessellNet, we compare it to simpler architectures presented below:

- `CNN-simple`: 6-layer deep convolutional network with batch normalization, a pooling layer and a final fully-connected layer.
- `CNN-multiple`: combination of 6 parallel convolutional networks with different kernel sizes and activation functions concatenated before a fully-connected layer.
- `CNN-gated`: a 6-layer gated convolution model with skip connections and a final fully-connected layer.
- `RNN-simple`: composed of 2 consecutive LSTM layers. The first one's output is repeated 24 times and passed as input to the second one. All 24 output sequences of the second LSTM are finally passed to a point-wise feed-forward network.
- `RNN-multiple`: a composition of 9 parallel RNN-simple networks with different LSTM embedding sizes and fully-connected layer activation functions, outputs are projected through a final fully-connected layer.

5 MAIN MODEL

Our architecture (Figure 2) takes as input every joint's 120 past rotation matrices and predicts the 24 following matrices. To reduce complexity, we feed all joints through the same model and will therefore, with the exception of a spatial attention layer, be employing weight sharing between the joints.

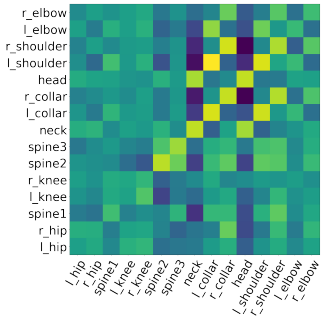


Figure 3: Variables learned in the spatial Factorized Random Synthesizer. They capture some static dependencies (f.ex. between head and neck, shoulders and collars, etc.), which in turn can free up the self-attention mechanism.

5.1 Embeddings

A bidirectional LSTM projects the inputs' rotation matrix $R \in \mathcal{R}^{15\times120\times9}$ to $\mathcal{R}^{15\times120\times128}$ while capturing sequence wide knowledge used in the discovery module to predict new frames. Empirically, we observed that recurrent units performed better than fully-connected or convolutional embeddings.

5.2 Attention Layers

Following Y. Tay et al. in [9], we use a combination of vanilla Multi-Head Attention [10] and Factorized Random Synthesizer Attention modules [9]. The authors demonstrated on NLP tasks that adding learnable weights to A improves performance, reasoning that inherent structure in the data could be directly learned in the Random Synthesizer instead of having to be captured in the dot-product Multi-Head Attention, thus enabling the latter to focus more on dynamic dependencies. Compared to text, one could argue that human motion data has even more static structure. Indeed, joint movements are highly correlated with their parent, child or symmetric counterparts on the other side of the body (see Figure 3). Similarly, poses at a given time t are bound to resemble poses in a neighbouring time-window.

The calculation for our attention layers L can be described by the following:

$$L(Q,K,V) = \text{Softmax}(\frac{QK^T}{\sqrt{d_k}} + R_{L1}{R_{L2}}^T)V \ , L \in \{S,T\}$$

with $R_{L1}{R_{L2}}^T$ being the trainable variables forming the Factorized Random Synthesizer for layer L.

We employ a variable number n_s of spatial attention layers S, followed by a variable number n_t of temporal attention layers T, i.e. in contrary to [1] we align all layers sequentially instead of in parallel. For ease of notation, we will hereafter consider $n_s = 1$, $n_t = 1$.

5.3 Recovery

We will refer to this module of TessellNet as *Recovery* as it is responsible for predicting future movement by recombining past poses.

The attention weights A_T calculated in T are fed into an encoder-decoder network consisting of multiple layers of 2D convolution followed by BatchNormalization and MaxPool2D in the *down* layers, or Upsampling2D in the *up* layers. To tie in with the notation introduced in Section 1, we calculate

$$A' = DB \, , \, B = EA_T$$

with E being the encoder, D the decoder, $B \in \mathcal{R}^{15\times360}$ the bottleneck and $A' \in \mathcal{R}^{15\times24\times120}$ the predicted heatmap.

A' is then used as attention weights and multiplied with the output of S:

$$Recovery(S, A') = \text{softmax}(A' + R_{Rec1}{R_{Rec2}}^T)S$$

with $R_{Rec1}{R_{Rec2}}^T$ again a Factorized Random Synthesizer and $Recovery \in \mathcal{R}^{15\times24\times128}$. Upon feeding this through a point-wise feed-forward network, the output is of the form $\mathcal{R}^{15\times24\times16}$.

As its name suggests, *Recovery* performs well when the movement is repetitive (Figures 4c and 4d). It will however struggle at producing new, previously unseen poses. In case of irregular movements, it will simply return the last pose (Figure 4b) and rely on the *Discovery* module to extend the movement.

5.4 Discovery

This module is the result of pure engineering, i.e. we tried different ways of connecting the modules in the network to create a module capable of coming up with new poses and found that this particular implementation worked best. We add the last state of the bidirectional LSTM to the last state of T, essentially creating a skip connection, and concatenate it with the bottleneck B before passing it to a point-wise feed-forward network. The output is then reshaped and up-sampled (using similar up-sampling layers as in D) until getting the shape $\mathcal{R}^{15\times24\times16}$.

5.5 Output

Finally, the outputs of both the *Recovery* and *Discovery* modules are added together, the dimensions permuted and projected down to $\mathcal{R}^{24\times135}$ to get the valid output shape.

6 SPATIO-TEMPORAL ATTENTION NETS

To measure the contribution of the *Recovery* module in TessellNet, we also implemented three variations of Attention Nets composed of multiple spatio-temporal Attention layers as described in [1] followed by a point-wise feed-forward network. The variations are standard Multi-Head Attention, Random Synthesizer Multi-head Attention and Factorized Random Synthesizer Multi-head Attention.

7 RESULTS

We present the results for our different models in Table 1. The metric used to evaluate the performance of our models is defined as the mean joint angle difference between all joints, summed over all 24 target time-steps. Here the joint angle difference is computed as the norm of the Rodrigues function [8] of the product of the angle matrices.

From the table, we can see that TessellNet outperforms our baseline models by a significant margin with respect to the metric as well as producing more natural-looking predictions. Note that we're able to beat the hard baseline using only one spatial and temporal Attention layer, totalling a surprisingly low number of parameters (650'414). Even the best-performing model is still far from employing 1 million parameters (866'862).

Interestingly, the Attention Nets without *Recovery* module perform worse than TessellNet. This indicates that the encoder-decoder architecture is more efficient at processing the spatial-temporal attention weights than the traditional attention-value product used in the original Transformer in our setting.

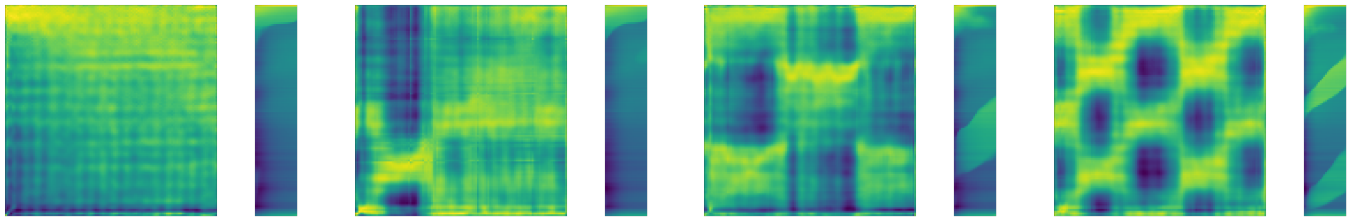
	Model				J. Angle
Baselines	CNN-simple				6.27
	CNN-multiple				3.93
	CNN-Gated				4.07
	RNN-simple				3.10
	RNN-multiple				**2.58**
Attention Nets	Factorized Synthetiser				2.52
	MultiHead Attention				2.50
	Synthetiser				**2.47**
TessellNet	n_s	n_t	heads	Synthesizer	
	2	2	16	yes	2.30
	2	2	8	no	2.28
	1	1	8	yes	2.25
	3	3	8	yes	2.21
	2	2	8	yes	**2.13**

Table 1: Performance of different models and implementations on public test set

8 DISCUSSION

In almost every work employing an Attention-mechanism, the authors show Attention-weights to illustrate the inner workings of their architectures. But surprisingly enough, the information contained in these matrices, which are very intuitive even to humans, is never explicitly utilized. TessellNet shows that exploiting the weights yields good results with comparatively low complexity. We don't expect TessellNet to outperform the Transformers whenever there's enough data and computing-power at hand. But there are many settings, especially in the medical domain, where data is scarce and where a cheaper version of a Transformer might be welcome.

Future work would therefore consist in extending TessellNet to a Transformer-like architecture. For instance, we could imagine integrating the Encoder/Decoder concept used in Transformers: while the Encoder processes A, the Decoder continuously builds A', thus turning TessellNet autoregressive. It would be interesting to investigate, whether such an extended TessellNet could be used on similar tasks as Transformers, or if its use case is more restrained.



(a) Static pose **(b) Irregular movement** **(c) Repetitive movement (slow)** **(d) Repetitive movement (fast)**

Figure 4: Attention weights and predicted weights for future 24 frames of four different movements. 4a and 4b show that the recovery part just returns the last pose whenever the movement is static or irregular, while it returns a linear combination of past poses when it detects a regular pattern (4c, 4d).

REFERENCES

[1] Emre Aksan, Peng Cao, Manuel Kaufmann, and Otmar Hilliges. 2020. A Spatio-temporal Transformer for 3D Human Motion Prediction. *arXiv e-prints*, Article arXiv:2004.08692 (April 2020), arXiv:2004.08692 pages. arXiv:cs.CV/2004.08692

[2] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. 2019. Structured Prediction Helps 3D Human Motion Modelling. In *The IEEE International Conference on Computer Vision (ICCV)*. First two authors contributed equally.

[3] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. 2020. Counting Out Time: Class Agnostic Video Repetition Counting in the Wild. *arXiv e-prints*, Article arXiv:2006.15418 (June 2020), arXiv:2006.15418 pages. arXiv:cs.CV/2006.15418

[4] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. 2017. Learning Human Motion Models for Long-term Predictions. *arXiv e-prints*, Article arXiv:1704.02827 (April 2017), arXiv:1704.02827 pages. arXiv:cs.CV/1704.02827

[5] Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and Jose M. F. Moura. 2018. Adversarial Geometry-Aware Human Motion Prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural computation* 9 (12 1997), 1735–80. https://doi.org/10.1162/neco.1997.9.8.1735

[7] Dario Pavllo, Christoph Feichtenhofer, Michael Auli, and David Grangier. 2019. Modeling Human Motion with Quaternion-based Neural Networks. *arXiv e-prints*, Article arXiv:1901.07677 (Jan. 2019), arXiv:1901.07677 pages. arXiv:cs.CV/1901.07677

[8] Olinde Rodrigues. 1840. Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendants des causes qui peuvent les produire. *Journal de Mathématiques Pures et Appliquées* 5 (1840), 380–440.

[9] Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2020. Synthesizer: Rethinking Self-Attention in Transformer Models. *arXiv e-prints*, Article arXiv:2005.00743 (May 2020), arXiv:2005.00743 pages. arXiv:cs.CL/2005.00743

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf