

---

# Dimension Reduction Approach for Interpretability of Sequence to Sequence Recurrent Neural Networks

---

**Kun Su**

Department of Electrical & Computer Engineering  
University of Washington Seattle, WA 98195  
suk4@uw.edu

**Eli Shlizerman**

Departments of Applied Mathematics and Electrical & Computer Engineering  
University of Washington Seattle, WA 98195  
shlizee@uw.edu

## Abstract

Encoder-decoder recurrent neural network models (Seq2Seq) have achieved great success in ubiquitous areas of computation and applications. It was shown to be successful in modeling data with both temporal and spatial dependencies for translation or prediction tasks. In this study, we propose a dimension reduction approach to visualize and interpret the representation of the data by these models. We propose to view the hidden states of the encoder and the decoder as spatio-temporal snapshots of network dynamics and to apply proper orthogonal decomposition to their concatenation to compute a low-dimensional embedding for hidden state dynamics. Projection of the decoder states onto such interpretable embedding space shows that Seq2Seq training to predict sequences using gradient-descent back propagation effectively performs dimension reduction consisting of only a small percentage of dimensions of the network's hidden units. Furthermore, sequences are being clustered into well separable clusters in the low dimensional space each of which corresponds to a different type of dynamics. The projection methodology also clarifies the roles of the encoder and the decoder components of the network. We show that the projection of encoder hidden states onto the low dimensional space provides an initializing trajectory directing the sequence to the cluster which corresponds to that particular type of distinct dynamics and the projection of the decoder hidden states constitutes the embedded cluster attractor. Inspection of the low dimensional space and the projections onto it during training shows that the estimation of clusters separability in the embedding can be utilized to estimate the optimality of model training. We test and demonstrate our proposed interpretability methodology on synthetic examples (dynamics on a circle and an ellipse) and on 3D human body movement data.

## 1 Introduction

Sequence to sequence (Seq2Seq) network models belong to the class of Recurrent Neural Networks (RNNs) which use internal states to process sequences of inputs [1, 2, 3, 4]. The speciality of Seq2Seq is that these models consist of encoder and decoder components. While both components share the same internal neural units, the encoder typically processes input sequences and constructs a latent representation of the sequences. In addition to sharing the internal units with the decoder, the encoder passes the last internal state to the decoder as an "initialization" of the decoder. With this information

the decoder is able to transform or generate novel sequences with a similar distribution. Such an architecture and its variants, e.g. attention-based Seq2Seq [4], showed compelling performance in applications of machine translation [3], speech recognition [5], and human motion prediction [6].

While Seq2Seq and its variants achieve strong performance on various applications, a consistent interpretation of how the encoder-decoder structure capable to embed the data for general time-series data (i.e. multi dimensional ordered sequences) and how such interpretation can be used to estimate the performance of the model is an active research topic. In this paper, we propose a dimension reduction approach to visualize and interpret representation within Seq2Seq model of general time-series data. The main contribution of this paper is to provide constructive insights on the properties which allow the encoder and the decoder components to operate optimally and to propose a low dimensional visualization method of the representation that the encoder and the decoder construct.

Interpretability has been an important aspect of any artificial neural network (ANN) and is expected to provide generic methodologies to assess the capabilities of different models and evaluate performance of models for given tasks. Associating interpretability with various types of ANN is often a challenging undertaking as the typical state-of-the-art network models are high dimensional and include many components and processing stages (layers). The problem becomes more challenging when dealing with timeseries models (i.e. RNN sequence models) and in unsupervised tasks such as synthesis and translation, in which encoder-decoder networks were found to be prevalent. In these tasks, interpretability is aimed to capture the model synthesis procedures and to demonstrate how these are being improved over training and since learning is unsupervised interpretation methodologies have the potential to provide a framework to assist with enhancing the optimization and learning.

Typically, existing interpretation methods applicable to ANNs are application oriented. For Convolutional Neural Networks (CNN), models were explained within image classification problems [7, 8, 9]. For RNN, interpretation and visualization tools focus on natural language processing applications [10, 11, 12, 13]. Our proposed work is based and inspired by these works and extends the methodology to generic sequences, in which textual sequences are a sub-class with special time dependence (semantics). In general, multidimensional time series are spatiotemporal sequences including nontrivial correlations in space and time. Beyond text data, there are works inspired from neuronal networks investigating the dynamics of RNN [14, 15]. In this work, we aim to provide interpretation of encoder-decoder (Seq2Seq) network models for general time series data. We test our methods on prediction tasks of synthetic data and of typical movements of human body joints. There are several Seq2Seq based models that achieve success on human motion prediction [16] and outperform the previous non-Seq2Seq based RNN models such as ERD [17] and S-RNN [18]. Recently, Generative Adversarial Networks [6] have achieved better performance on this task, however the predictor network is still Seq2Seq.

We show that Seq2Seq optimization with gradient descent based methods forms a low dimensional embedding of internal states. The embedding can be mapped and visualized through proper orthogonal decomposition of concatenated encoder and decoder internal states - the interpretable embedding. Within this embedding the decoder evolution for each distinct sequence (decoder trajectory) is well separable from other distinct sequences, i.e., distinct projections to the interpretable embedding space are well clustered (in K-means++ sense). Furthermore, each distinct decoder trajectory preserves the temporal properties of the sequence. The encoder trajectory initiated from various starting points connects them in the interpretable embedding space with the appropriate decoder trajectory. Monitoring the interpretable embedding space and projected trajectories in it during training epochs shows the effect of training on the representation of data to identify an optimal regime in between under- and over- fitting. We construct synthetic data examples to demonstrate the construction of the interpretable embedding space. Next, we apply the construction of the interpretable embedding and analyze Seq2Seq performance on human joints movements dataset containing 15 different types of real body movement sequences, H3.6M, such as walking, eating, etc [19]. Without loss of generality, our interpretable methods can also be applied to any spatiotemporal data.

**Setup and Spatiotemporal States of the Seq2Seq Model:** Seq2Seq model is an encoder-decoder style RNN architecture that maps a sequence to another sequence and thereby can be used for sequence prediction (where it synthesises a predicted sequence) or translation (where it maps the sequence to a new representation) (Fig. 1). For general time series prediction, given a sequence of spatiotemporal data as input, Seq2Seq predicts the future sequence. We stack the sequence of input to the encoder as a matrix and call it the **encoder input matrix**  $\mathbf{X} \in \mathbb{R}^{T_e \times M}$ , where each row

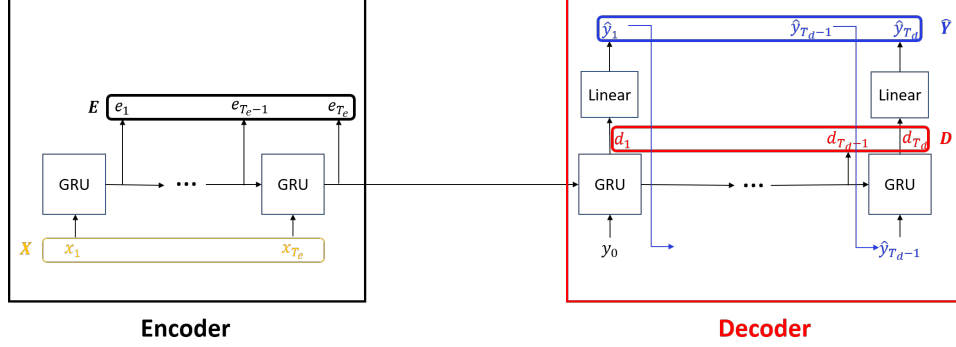


Figure 1: Seq2Seq architecture: We consider the inputs, encoder states, decoder states and outputs as spatiotemporal matrices  $X, E, D, \hat{Y}$  respectively. Here the recurrent network block is Gated Recurrent Unit (GRU).

$x_t \in \mathbb{R}^{1 \times M}$  is a time step of the input sequence at time  $t$ ,  $T_e$  is the number of time steps and  $M$  is the number of dimensions in the input data. Similarly, we construct the target sequences as **target output matrix**  $\mathbf{Y} \in \mathbb{R}^{T_d \times M}$ , where  $T_d$  is the number of output sequence steps to be predicted.

In addition, forward propagation of the input in Seq2Seq computes the internal states of the encoder at each time step. We concatenate them and denote as the **encoder states matrix**  $\mathbf{E} \in \mathbb{R}^{T_e \times N}$  where each row  $e_t \in \mathbb{R}^{1 \times N}$  represents the states of all internal units at  $t$  and  $N$  is the number of hidden units. Similarly, we also define the **decoder states matrix**  $\mathbf{D} \in \mathbb{R}^{T_d \times N}$ . Typically, there is an additional fully connected linear map transforming the decoder states to the dimensions of the output space. We denote the **decoder output matrix**  $\hat{\mathbf{Y}} \in \mathbb{R}^{T_d \times M}$  where  $\hat{y}_t$  is the predicted  $\hat{y}_t$  output at time  $t$ . Fig. 1 demonstrates the structure of the components of Seq2Seq matrices. In the figure, the encoder and the decoder are single layer GRU units sharing parameters with each other. Our approach is applicable to general types of units such as LSTMs/GRUs and number of layers. Additionally, as demonstrated, the decoder uses the output of the previous time step as an input to the current time step in both training and testing, except the initial time step in which it receives its input from the last step of the encoder. In terms of time series prediction, the cost function is typically defined as the MSE between **target outputs** and **decoder outputs**,  $J = \frac{1}{T_d} \sum_{t=1}^{T_d} (y_t - \hat{y}_t)^2$ , however other norms or cost functions can be considered.

**Proper Orthogonal Decomposition of Spatiotemporal Matrices:** Since forward propagation within the Seq2Seq network can be represented through spatiotemporal matrices, we propose to apply the Proper Orthogonal Decomposition (POD) as a dimensionality reduction algorithm to construct a low dimensional interpretable embedding [20]. Specifically, we use the Singular Value Decomposition (or PCA) which decomposes the matrices into orthogonal spatial modes (PCs) and time dependent coefficients and singular values (scaling) associated with each mode. Particularly, given a matrix  $A \in \mathbb{R}^{T \times N}$  we center the matrix around the origin, obtaining  $A_c$ , and then apply SVD such that  $A_c = U \Sigma V^T$ , where  $U \in \mathbb{R}^{T \times T}$  is an orthogonal matrix of time dependent coefficients,  $\Sigma \in \mathbb{R}^{T \times N}$  is the matrix of singular values and  $V \in \mathbb{R}^{N \times N}$  is the matrix of spatially dependent components. To determine the number of PCs, we compute the singular value energy (SVE),  $\text{SVE} = \sum_{i=1}^n \sigma_i^2$  where  $\sigma_i$  is singular value corresponding to PC mode  $i$ . We compute the number of modes such that 90% or 99% of the energy is retained ( $\text{SVE}_k = \sum_{i=1}^k \sigma_i^2, \frac{\text{SVE}_k}{\text{SVE}} \leq p$  where  $k$  is the number of modes and  $p$  is the percentage). With the number of dominant spatial features, we can truncate  $A_c$  by projecting it onto the PC modes to get a low dimensional matrix  $A_{(\text{PC}=n)} \in \mathbb{R}^{T \times n}$  where (PC= $n$ ) denotes  $n$  principal components are retained. If we choose  $n = 2$  or  $3$ , we can visualize the representation in 2D or 3D. The axes are the orthogonal PC modes.

**Clustering:** While visualization of projected dynamics could be informative [21], 2D or 3D visualized dynamics do not reveal the intricacies in the representation of different datasets. In particular, here we would like to evaluate the separability of projections of distinct trajectories in the interpretable embedding space. We thereby propose to augment the embedding with clustering approaches such as **K-means++** clustering, an extended version of the standard K-means algorithm [22]. Since the number of trajectories and time steps are known, we can use the **Adjusted Rand Index** and

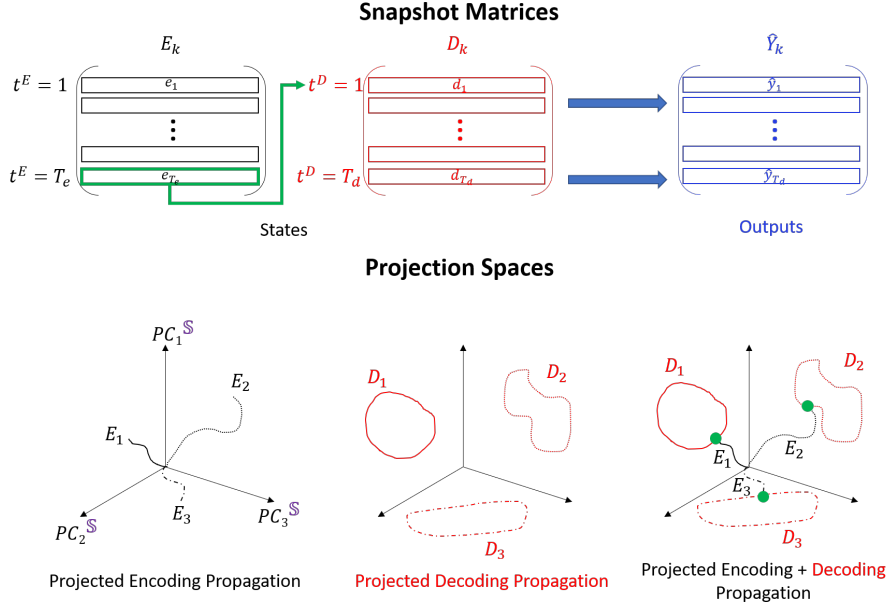


Figure 2: POD is performed on  $E_k$  and  $D_k$ , obtained for each type of data  $k$  and stacked to  $\mathbb{S}$ . Three distinct trajectories are shown in the low dimensional embedding space. The encoder trajectories (black) start from the origin and diverge in different directions. The decoder trajectories (attractors; red) are hence placed in separate locations in the space. The last point of each encoder trajectory (green) connects the encoder and the decoder trajectories.

**V-measure** to evaluate the clustering performance. In addition, we also directly compute the distance between different trajectories to quantify their proximity to each other [23, 24].

## 2 Interpretable Embedding for Seq2Seq Networks

The generic goal of Seq2Seq model is to continue (predict) the evolution of each given input sequence chosen from a (test) dataset that includes  $K$  different types of multidimensional time series. Such a goal is challenging as it requires the network to generate a sequence which superimposes the typical dynamics of that particular type and the individual dynamics of the given tested input sequence. We show that the methods described above can be applied to construct an interpretable embedding for the Seq2Seq model depicted in Fig.2.

To construct the embedding space basis we concatenate the matrices  $\mathbf{E}$  and  $\mathbf{D}$  for each single forward propagation into a **state matrix**

$$\mathbf{S} = \begin{bmatrix} \mathbf{E} \\ \mathbf{D} \end{bmatrix} \in \mathbb{R}^{(T_e+T_d) \times N}, \quad \mathbb{S} = \begin{bmatrix} S_1 \\ \vdots \\ S_K \end{bmatrix}.$$

Then concatenation of forward propagation evolution for all considered time series in the dataset will result with a **global state matrix** denoted as  $\mathbb{S}$ . POD application on  $\mathbb{S}$  provides the PC modes which are the axes of the interpretable embedding space. Dimension reduction of the embedding space is performed by considering only  $n$  PCn modes which singular values are included in representation of particular total SVE (e.g. 90% or 99%) and truncating the rest of the modes.

To inspect the propagation of single time series through the network, we can project the matrices  $\mathbf{E}$ ,  $\mathbf{D}$  or  $\mathbf{S}$  onto the low dimensional space spanned by PCn modes. As we show below, we find that the dimension of the embedding space turns out to be very low even for high dimensional data. We depict the structure of such projections onto PC3 embedding space in Fig. 2 bottom. Encoder trajectories (black, left) start from initial points projected to the embedding space (we choose initial states as zeros and therefore the trajectories are always initiated at the origin) and evolve in different directions in the space. Decoder trajectories (red, middle) appear as attractors in the space and clustering approaches

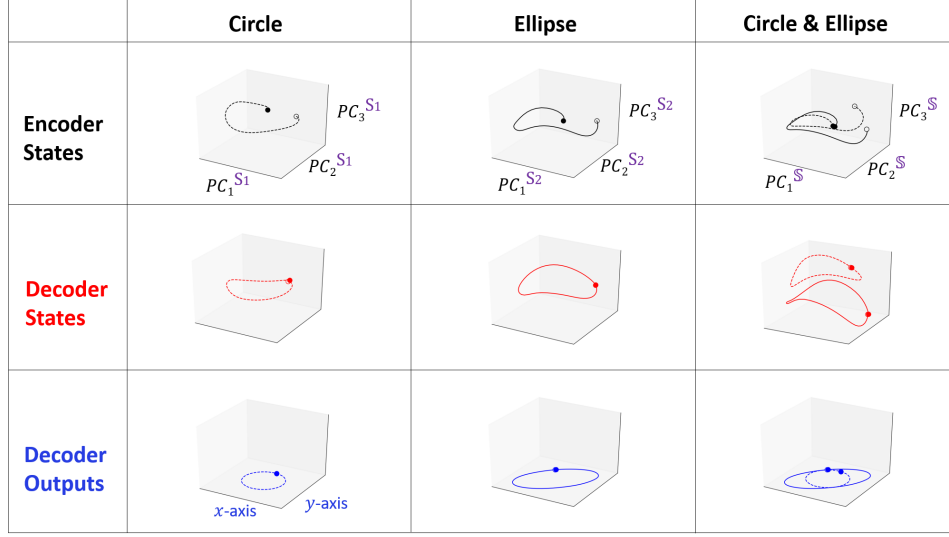


Figure 3: Projected trajectories in the embedding space spanned by PC3 for the encoder states, decoder states and decoder outputs matrices. Dashed and solid lines represent  $X_{circle}$  and  $X_{ellipse}$  respectively. Opaque and transparent points denote the starting and the ending points of the trajectory.

are used to determine (e.g. K-means++, ARI and V-measure) how separable they are in the space. The encoder and the decoder connect via a single time step which corresponds to a point (green, right). Composition of the three types of projections corresponds to an interpretation of the propagation in Seq2Seq. In particular, we conclude that the encoder trajectory takes the sequence from an initial point and evolves it to the corresponding starting point on the decoder trajectory (we call it attractor or cluster). The decoder continues the evolution from there. As we show below, it appears that the gradient descent training succeeds to place the decoder attractors in the low dimensional embedding spaces such that they are easily clustered. Such an arrangement explains the uniqueness of Seq2Seq training in which the cost function minimizes the error between the decoder output and the actual output and there is no minimization on the encoder. Therefore, the decoder is trained to predict different features for different types of inputs (training to optimize clustering of types of data and capture unique features) with the encoder trajectory (and not only the last time step of it) being a sequential constraint that connects the cluster to the initial state. In practice, the longer the encoding sequence the better is the prediction (clustering property) and such an interpretation is evident from the low dimensional embedding space as well.

It is possible to monitor the construction and the changes in the embedding space and projected trajectories within it during the training of Seq2Seq. Specifically, at each training iteration  $i$ , for each type of data, we can construct the matrices  $E^i$ ,  $D^i$ ,  $S^i$  and perform the aforementioned procedures.

### 3 Interpretable Embedding Space Applied to Synthetic Data

We create two types of synthetic 2D trajectories which follow a circumference of (i) a unit circle  $X_{circle}$  (ii) an ellipse  $X_{ellipse}$ . The encoder input is binned into  $T_e$  time steps such that  $X_{circle}, X_{ellipse} \in \mathbb{R}^{T_e \times 2}$  with  $T_e = 50$  such that it completes a full period. Given full circle or ellipse dynamics as the encoder input, the objective is to predict another full circle or ellipse by the decoder, i.e., the target sequence  $Y$  is expected to be the same as the input sequence ( $Y \equiv X$ ) and the **decoder output matrix**  $\hat{Y}$  has the same dimension as  $Y$  and  $X$ . The cost function  $J = \frac{1}{T_d} \sum_{t=1}^{T_d} (y_t - \hat{y}_t)^2$  is the mean square error between the target and the prediction with  $T_d = T_e$ . For both the encoder and the decoder we use a single layer GRU with 16 neurons ( $E, D \in \mathbb{R}^{T_e \times 16}$ ). We use the ADAM optimizer and train the model for 10,000 iterations where the cost function almost converges to zero such that the model can predict the trajectory with high accuracy.

We show in Fig. 3 the projections of forward propagation in Seq2Seq trained on (i) a circle only (ii) an ellipse only (iii) both circle and ellipse. In each model, we obtain the PC3 embedding space from

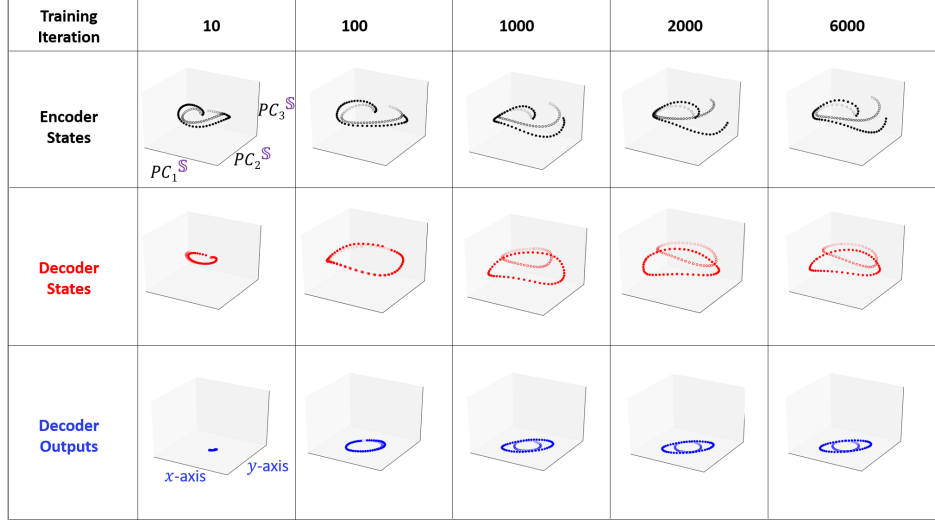


Figure 4: Snapshots of the evolution of trajectories projected to the interpretable embedding space when they undergo training. We use similar line identifiers and colors as in Fig. 3. At the beginning of the training, two trajectories are very similar and positioned close to each other. As training proceeds the two type of trajectories emerge and separate from each other.

the matrix  $\mathbb{S}$  and examine the projections of the encoder states matrix  $\mathbf{E}$ , decoder states matrix  $\mathbf{D}$  onto the space and also show the decoder output matrix  $\hat{\mathbf{Y}}$  projected onto  $x - y$  space. We observe that the projections of the encoder and the decoder states are deformed and not necessarily preserve the same form as the input and the output, however, linear transformation of the decoder deforms the trajectory to be correctly represented in the  $x - y$  space. Notably, all encoder projections start near the origin since our initial states are zero by default.

The last state of the encoder in NLP applications is typically considered to contain the information of all previous states, an assumption that gives rise to the attention-based Seq2Seq model. However, our results indicate that the full encoder sequence is important and the last encoder state is simply providing a starting point for the decoder to continue. Such an effect of the encoder is easily observed in considering continuous time series (as we show in prediction of human movement data) and deviates from the interpretation of the encoder role in textual semantic sequences.

We focus on the model trained on both the circle and the ellipse to better understand how Seq2Seq is able to predict both time series in an unsupervised way without any guidance. Projections onto the PC embedding space (right column of Fig. 3). Encoder projections indicate that the two trajectories corresponding to distinct types of output sequences, start from points near the origin however diverge as the sequence evolves and end up at more distant points. The projected trajectories of the decoder states start from these distinct points and continue to perform prediction in completely separable shapes. Effectively we observe that the decoder states projections are clustered in the embedding space.

To visualize how Seq2Seq learns to differentiate the two shapes with training, we keep track of the evolution of states representation and depict in Fig. 4 the three projections as in Fig. 3 at iteration 10, 100, 1000, 2000, 6000. In the beginning of training (iteration = 10), all projections of the two shapes are very similar. Projections of the decoder states appear to be distinct for a few several initial points but when the sequence evolves forward, the trajectories appear as converging to the same point. When the model undergoes additional training, after 100 iterations, Seq2Seq appears to have learned a single pattern (ellipse like) however is unable to generate two distinct predicted shapes. At iteration 1000 the two trajectories separate and obtain accurate predictions at the same time. The evolution over training reveals that the model learns one general pattern first and then gradually evolves into two different separable patterns.

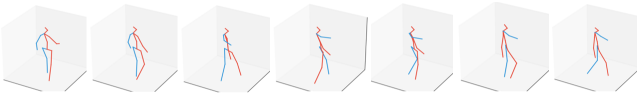
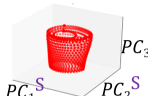
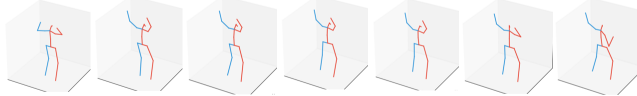
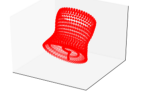
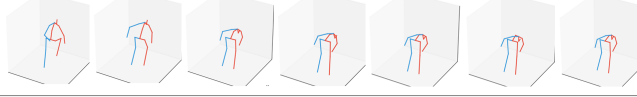
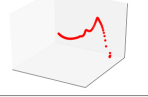
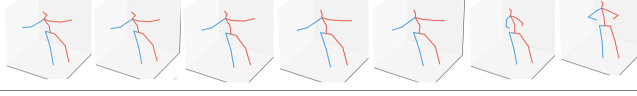
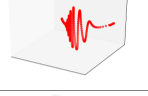
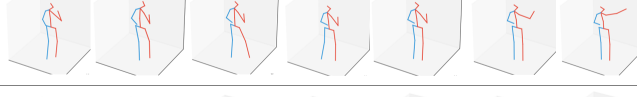



Movement Type	Human Movements 3D Coordinates	Decoder Projections
Walking		
Taking Photos		
Sitting		
Posing		
Directions		
Purchases		

Figure 5: Different human joint movement types and (long-term) projected decoder states in the interpretable embedding space.

#### 4 Human Body Joints Movements Data

To test the proposed interpretable embedding methodology on realistic data, we use the Human 3.6 (H3.6M), which is currently one of the largest publicly available data sets of motion capture data [19]. It includes 7 actors performing 15 various activities such as walking, engaging in a discussion, taking pictures, shopping and talking on the phone. Each movement is repeated in 2 different trials. We use 6 of the actors motion as the training set and the other actors motion as a testing set. Human pose is represented as an exponential map representation of each joint, with a special pre-processing of global translation and rotation. The number of features of body joints dynamics is  $M = 54$ . For consistency with the previous synthetic example, we provide  $T_e = 50$  frames of input data and predict next  $T_d = 50$  frames. As a result, the **actual input matrix**, **actual output matrix** and **decoder output matrix** will be  $X, Y, \hat{Y} \in \mathbb{R}^{T_d \times 54}$ . Seq2Seq model was shown to be successful in such a prediction task [16]. We use a similar setup where we use a single layer GRU with  $N = 1024$  units for both the encoder and the decoder. Thereby the **encoder states matrix** and **decoder states matrix** will be  $E, D \in \mathbb{R}^{T_e \times N}$ . The cost function is the mean square error between the ground truth and the predicted output  $J = \frac{1}{T_d} \sum_{t=1}^{T_d} (y_t - \hat{y}_t)^2$ . We use a batch size of  $B = 15$  such that every training iteration can contain all actions. We train the model with various gradient descent based optimizers and show our results for ADAM (the fastest converging optimizer for this model).

We visualize the projections onto PC3 embedding space for every type of action. In order to see the pattern clearly, we use a trained model and predict 1000 more time steps, i.e. ( $D \in \mathbb{R}^{1050 \times 1024}$ ). In Fig. 5, we show examples of the joints evolution (connected with lines) alongside with the decoder states projection onto PC3 embedding space. We clearly observe that each individual action has its own trajectory evolution (attractor) pattern even in 3D. For example, as expected, walking data corresponds to periodic circular pattern with rather fixed period and actions such as sitting correspond to non-periodic trajectories in the embedding space.

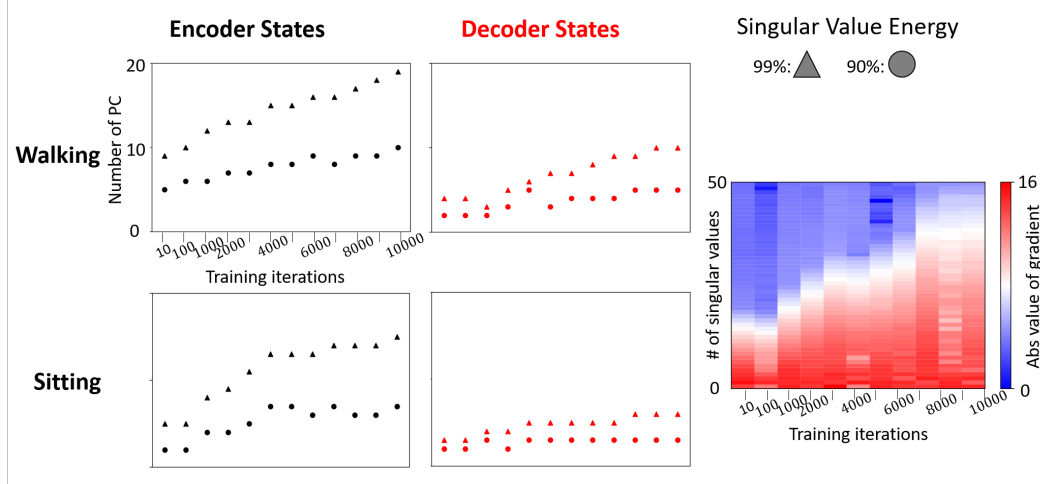


Figure 6: Left: Examples of evolution of the number of dominant modes to reach 90% and 99% in encoder and decoder for walking and sitting, respectively. Right: evolution of absolute changes of singular values along the training for "walking" decoder states matrix.

To get a better understanding on what should be the appropriate dimensions of the embedding space we monitor the number of dominant PC modes to reach 90% and 99% SVE for encoder and decoder matrices POD for every type of action during training (Fig. 6). We observe a low number of dominant modes ( $< 10$  and  $< 20$  on average for 90% and 99% respectively) needed to span most of the energy. The number of modes increases with training and the requirement of the energy threshold (accuracy of the embedding). The rate of the increase depends on the type of movement. For example, as shown in Fig. 6, the number of dominant modes for "sitting" is much smaller than for walking. Furthermore, the number of dominant modes required to reach 90% SVE increases more slowly than numbers of modes to reach 99%. These results indicate that the model learns additional details with training. However, most of the main features captured in (90%) are learned in the first iterations. Such an observation is supported by analyzing the gradients of the singular values during training (Fig. 6, right). As training proceeds, additional singular values are gradually being optimized. Notably, we observe that encoder states would have more modes than decoder states. The reason stems from the encoder trajectory starting from the origin and connecting to the decoder attractors in various parts of the embedding space. Such trajectories are hence including mixed characteristics of the attractor and of the path to it resulting in more irregular trajectories requiring additional modes to be represented.

Table 1: Clustering results

	Training iterations	K-means++	ARI (%)	V-measure (%)
Encoder states	10	dim=3	52	72
		dim=1024	61	78
	2000	dim=3	56	74
		dim=1024	79	89
	10000	dim=3	59	77
		dim=1024	86	93
Decoder states	10	dim=3	29	56
		dim=1024	32	60
	2000	dim=3	97	98
		dim=1024	<b>100</b>	<b>100</b>
	10000	dim=3	94	96
		dim=1024	<b>100</b>	<b>100</b>
Joints data		dim=3	59	76
		dim=54	80	90



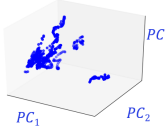
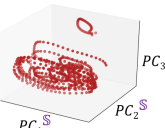
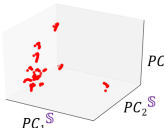
Human Movements Clustering		
Joints Coordinates	Decoder States Trained on Single Movement (Walking)	Decoder States Trained on ALL Movements
		
ARI: 59% (3d) 80% (1024d)	ARI: 25% (3d) 58% (1024d)	ARI: 97% (3d) 100% (1024d)

Figure 7: Visualization of clustering in 3D. Left: Joints coordinates projected data, Middle: Overfitting for walking data. Right: Decoder states at training iterations=2000.

To understand how Seq2Seq forms distinct attractors and differentiates various actions, we construct matrices  $\mathbf{E}$ ,  $\mathbf{D}$ ,  $\hat{\mathbf{Y}}$  for all 15 actions for which the model was trained. We apply the K-means++ clustering to  $\mathbb{D}$  to evaluate the separability of Seq2Seq in the interpretable embedding space and the dimension of the space which provides efficient clustering property (Table 1).

We compare clustering of the encoder and the decoder trajectories with respect to ARI and V-measures at different training iterations and for different dimensions (dim= 3 and dim= 1024) of the embedding space. In addition, we cluster the body joints data (in dim= 3 and dim= 54). Only clustering of the decoder attractors is able to reach 100% clustering in both measures for dim= 1024 of the embedding space. The next best clustering performance is for the decoder attractors in dim= 3 (97% ARI which is significantly higher than joints data for dim=3 and dim=54) after considerable number of iterations = 2000. After this number of iterations the attractors start to approach each other instead of diverging. Encoder trajectories are not clustered well in any dimension and their clustering does not significantly change with training. In Fig. 7 we visualize the clusters (for joints coordinates, overfitted trained model, trained model on all movements) in 3D. We show that even visually, the decoder attractors are scattered in the space indicating almost perfect clustering property of PC3 space. If we keep adding the number of PCs to 10, it would be enough to achieve the perfect clustering result.

One of our key observations is that the clustering performance in low dimensions reaches a peak after considerable training ( 2000 iterations) and then if training continues, clustering becomes inferior. Eventually, all actions will have similar properties such that the model overfits to certain actions. We show such a case by training Seq2Seq with more "walking" action data and then test the model with all actions. Indeed, no matter what type of action is given as an input, the decoder states will always be similar to a circular trajectory that represents the "walking" action. Hence, we propose that the clustering property of the low dimensional embedding space (90% PC) could be used as an indication for sufficient and optimal training of the model.

To understand how the training shapes the decoder attractors and their clustering, we mark each distinct movement attractor by a different color and monitor their representation in PC3 embedding space for various training iterations (Fig. 8). When the training is initiated the clusters appear close to each other and as training proceeds distinct attractors emanate, also indicated by measuring the distances between the clusters shown in (Fig. 8 right).

## 5 Conclusion

We propose to utilize the proper orthogonal decomposition to construct a low-dimensional interpretable embedding for the internal states within the Seq2Seq model. The embedding clarifies the role of the encoder and the decoder components of Seq2Seq such that encoder embedded trajectories direct the evolution from the origin to the decoder trajectories represented as attractors. We demonstrate

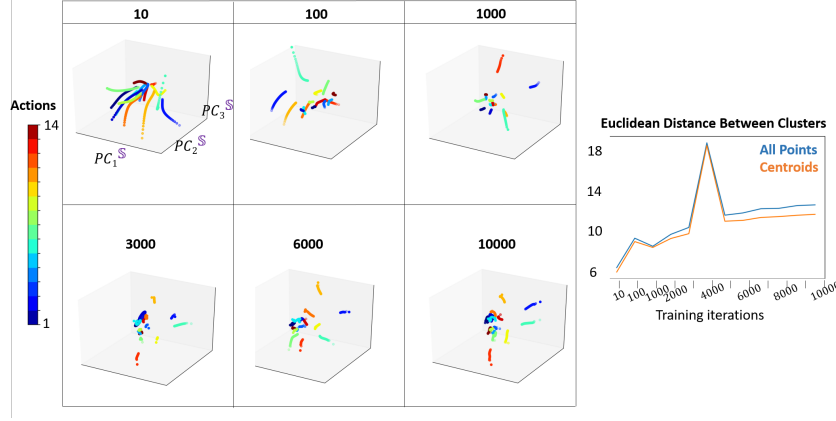


Figure 8: Left: projection of  $\mathbb{D}$  on the basis of  $\mathbb{S}$  at different training iterations, each color represents one type of actions. Right: distance between clusters.

the construction and the utilization of the embedding space on both synthetic and human body joints datasets. Our findings include important realizations on the function of Seq2Seq model. For properly trained Seq2Seq, the embedding appears to be low dimensional and attractors within it are well separated from each other such that they can be easily clustered. Using the described properties of the interpretable embedding space assists with choosing model properties and encoding/decoding sequences. We show that inspecting training by projecting forward propagation onto the embedding space interprets the performance of the network and can be used to determine the status of the network being under- or over-fitted.

## References

- [1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [2] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.
- [3] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
- [4] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [5] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.
- [6] Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and José MF Moura. Adversarial geometry-aware human motion prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 786–803, 2018.
- [7] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.
- [8] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *arXiv preprint arXiv:1711.11279*, 2017.
- [9] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- [10] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.
- [11] Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. Capacity and trainability in recurrent neural networks. *arXiv preprint arXiv:1611.09913*, 2016.

- [12] Jakob N Foerster, Justin Gilmer, Jascha Sohl-Dickstein, Jan Chorowski, and David Sussillo. Input switched affine networks: An rnn architecture designed for interpretability. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1136–1145. JMLR. org, 2017.
- [13] Hendrik Strobelt, Sebastian Gehrmann, Michael Behrisch, Adam Perer, Hanspeter Pfister, and Alexander M Rush. Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *IEEE transactions on visualization and computer graphics*, 25(1):353–363, 2019.
- [14] Stefano Recanatesi, Matthew Farrell, Guillaume Lajoie, Sophie Deneve, Mattia Rigotti, and Eric Shea-Brown. Signatures and mechanisms of low-dimensional neural predictive manifolds. *bioRxiv*, page 471987, 2018.
- [15] Matthew S Farrell, Stefano Recanatesi, Guillaume Lajoie, and Eric Shea-Brown. Dynamic compression and expansion in a classifying recurrent network. *bioRxiv*, page 564476, 2019.
- [16] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2891–2900, 2017.
- [17] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4346–4354, 2015.
- [18] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5308–5317, 2016.
- [19] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2014.
- [20] Eli Shlizerman, Konrad Schroder, and J Nathan Kutz. Neural activity measures and their dynamics. *SIAM Journal on Applied Mathematics*, 72(4):1260–1291, 2012.
- [21] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- [22] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
- [23] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.
- [24] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 2007.