Recurrent Network Models for Human Dynamics

Katerina Fragkiadaki Sergey Levine Panna Felsen Jitendra Malik
University of California, Berkeley
Berkeley, CA
{katef,svlevine@eecs,panna@eecs,malik@eecs}.berkeley.edu

Abstract

We propose the Encoder-Recurrent-Decoder (ERD) model for recognition and prediction of human body pose in videos and motion capture. The ERD model is a recurrent neural network that incorporates nonlinear encoder and decoder networks before and after recurrent layers. We test instantiations of ERD architectures in the tasks of motion capture (mocap) generation, body pose labeling and body pose forecasting in videos. Our model handles mocap training data across multiple subjects and activity domains, and synthesizes novel motions while avoiding drifting for long periods of time. For human pose labeling, ERD outperforms a per frame body part detector by resolving left-right body part confusions. For video pose forecasting, ERD predicts body joint displacements across a temporal horizon of 400ms and outperforms a first order motion model based on optical flow. ERDs extend previous Long Short Term Memory (LSTM) models in the literature to jointly learn representations and their dynamics. Our experiments show such representation learning is crucial for both labeling and prediction in space-time. We find this is a distinguishing feature between the spatio-temporal visual domain in comparison to 1D text, speech or handwriting, where straightforward hard coded representations have shown excellent results when directly combined with recurrent units [31] .

1. Introduction

Humans have a remarkable ability to make accurate short-term predictions about the world around them conditioned on prior events [41]. Predicting the movements of other humans is an important facet of these predictions. Although the number of possible movements is enormous, conditioning on visual history can reduce the range of probable outcomes to a manageable degree of variation. For example, a walking pedestrian will most likely continue walking, and will probably not begin dancing spontaneously. Short term predictions of human kinematics allows people to adjust their behavior, plan their actions, and properly direct their attention when interacting with others. Similarly, for Computer Vision algorithms, predicting human motion is important for timely human-computer interaction [17], obstacle avoidance [22], and people tracking [8]. While simpler physical phenomena, such as the motion of inanimate objects, can be predicted using known physical laws, there is no simple equation that governs the conscious movements of a person. Predicting the motion of humans instead calls for a statistical approach that can model the range of variation of future behavior, and presents a tremendous challenge for machine learning algorithms.

We address this challenge by introducing Encoder-Recurrent-Decoder (ERD) networks, a type of Recurrent Neural Network (RNN) model [49, 24] that combines representation learning with learning temporal dynamics. We apply this model to generation, labeling, and forecasting of human kinematics. We consider two data domains: motion capture ("mocap") and video sequences. For mocap, conditioning on a mocap sequence so far, we learn a distribution over mocap feature vectors in the subsequent frame. At test time, by supplying mocap samples as input back to the model, long sequences are synthesized. For video, conditioning on a person bounding box sequence, we predict the body joint locations in the current frame or, for the task of body pose forecasting, at a specific point in the future. In the mocap case, the input and output domains coincide (3D body joint angles). In the video case, the input and output domains differ (raw video pixels versus body joint locations).

RNNs are network models that process sequential data using recurrent connections between their neural activations at consecutive time steps. They have been successfully applied in the language domain for text and handwriting generation [16, 30, 9], image captioning [43], action recognition [6]. Ranzato *et al.* [23] applies RNNs for visual prediction by quantizing the visual signal into a vocabulary of visual words, and predicts a distribution over those words in the next frame, given the visual word sequence observed at a particular pixel location.

We advocate a visual predictive model that is "Lagrangian" in nature [48]: it predicts future outcomes conditioning on an object tracklet rather than on a tube fixated at a particular pixel location, as [23] (the "Eulerian" approach). Such object-centric conditioning exploits more relevant visual history of the object for prediction. In contrast, a visual tube fixated at a particular pixel location encounters dramatically different content under camera or object motion.

In the ERD, the encoder transforms the input data to a representation where learning of dynamics is easy. The decoder transcribes the output of the recurrent layers to the desired visual form. For mocap generation, the encoder and decoder are multilayer fully connected networks. For video pose labeling and prediction, the encoder is a Convolutional Neural Network (CNN) [4] initialized by a CNN per frame body part detector and decoder is a fully connected network. ERDs simultaneously learn both the representation most suitable for recognition or prediction (input to the recurrent layer), as well as its dynamics, represented in the recurrent weights, by jointly training encoding, decoding and recurrent subnetworks. We found such joint finetuning crucial for empirical performance.

We test ERDs in kinematic tracking and forecasting in the H3.6M video pose dataset of Ionescu *et al.* [13]. It is currently the largest video pose dataset publicly available. It contains a diverse range of activities performed by professional actors and recorded with a Vicon motion capture system. We show that ERDs effectively learn human dynamics in video and motion capture. In motion generation, ERDs synthesize mocap data across multiple activities and subjects. We demonstrate the importance of the nonlinear encoder and decoder in ERDs by comparing to previous multilayer LSTM models [9]. We show that such models do not produce realistic motion beyond very short horizons. For video pose labeling, ERDs outperforms a per frame body part CNN detector, particularly in the case of left-right confusions. For future pose forecasting, ERDs forecast joint positions 400ms in the future, outperforming first order motion modeling with optical flow constancy assumptions.

Our experiments show that the proposed ERD models can simultaneously model multiple activity domains, implicitly detect the right activity scenario, and adapt their output labels or predictions accordingly. The action transitioning is transparent, in contrast to previous switching dynamical systems or switching HMMs [21, 7] for activity modeling.

2. Related work

Motion generation Generation of naturalistic human motion using probabilistic models trained on motion capture data has previous been addressed in the context of computer graphics and machine learning. Prior work has tackled synthesis of stylized human motion using bilinear spatiotemporal basis models [1], Hidden Markov Models [3], linear dynamical systems [21], and Gaussian process latent variable models [46, 40], as well as multilinear variants thereof [12, 45]. Unlike methods based on Gaussian processes, we use a parametric representation and a simple, scalable supervised training method that makes it practical to train on large datasets.

Dynamical models based on Restricted Boltzmann Machines (RBMs) have been proposed for synthesis and infilling of motion data [34, 29, 33, 35]. While such approaches have the advantage of learning probabilistic models, this also results in a substantially more complex training algorithm and, when multilayer models are used, requires sampling for approximate inference. In contrast, our RNN-based models can be trained with a simple stochastic gradient descent method, and can be evaluated very efficiently at test time with simple feedforward operations.

Video pose labeling and forecasting Temporal context has been exploited in kinematic tracking using dynamic programming over multiple per frame body pose hypotheses [20, 2], where unary potentials encore detectors' confidence and pairwise potentials encode temporal smoothness. Optical flow has been used in [26, 27] to adjust the temporal smoothness penalty across consecutive frames. Optical flow can only estimate the motion of body joints that do not move too fast and do not get occluded or dis-occluded. Moreover, the temporal coupling is again pairwise, not long range. ERDs keep track of body parts as they become occluded and disoccluded by aggregating information in time across multiple frames, rather than the last frame.

Parametric temporal filters such as Kalman filtering [47], HMMs or Gaussian processes for activity specific dynamics [39, 19, 28] generally use simple, linear dynamics models for prediction. Such simple dynamics are only valid within very short temporal horizons, making it difficult to incorporate long range temporal information. Switching dynamic systems or HMMs [21, 7] detect activity transitioning explicitly. In contrast, in ERD, action transitioning is transparent to the engineer, and also more effective. Moreover, HMM capacity increases linearly with increasing numbers of hidden states, but its parameter count increases quadratically. This makes it difficult to scale such models to large and diverse datasets. ERDs scale better than previous parametric methods in capturing human dynamics. RNNs use distributed representations: each world "state" is represented with the ensemble of hidden activations in the recurrent layer, rather than a single one. Thus, adding a neural unit quadratically increases the number parameters yet doubles the representation power - assuming binary units.

Standard temporal smoothers or filters are disconnected from the pose detector and operate on its output, such as the space of body joint locations. This representation discards context information that may be present in the original

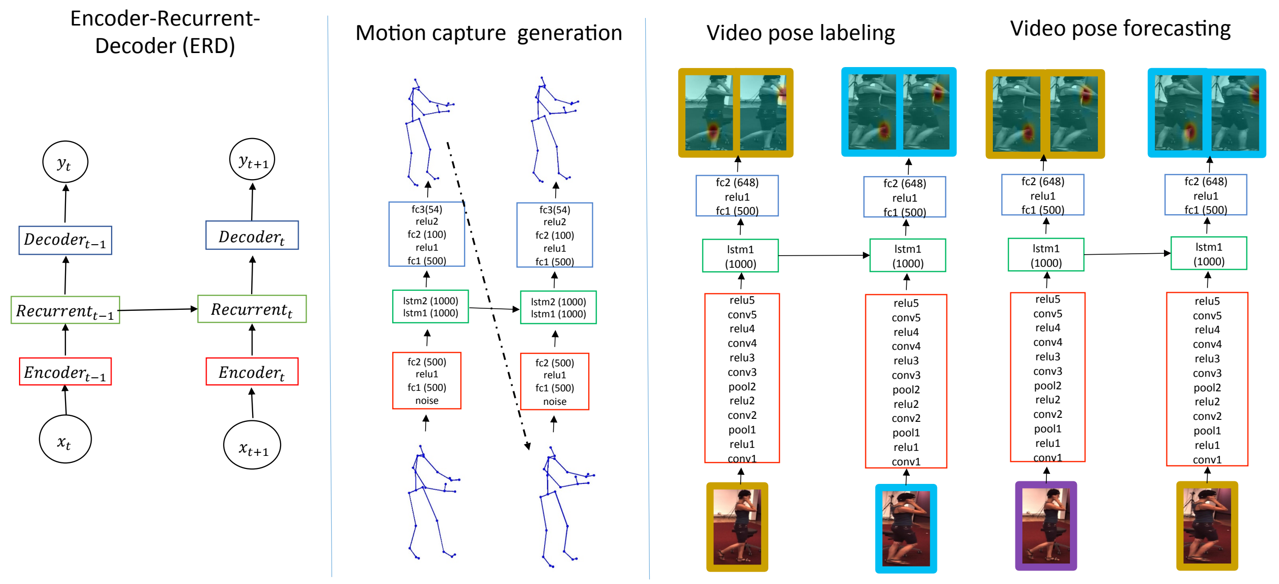
Encoder-Recurrent-Decoder (ERD)
Motion capture generation
Video pose labeling
Video pose forecasting
y_t
y_{t+1}
$Decoder_{t-1}$
$Decoder_t$
$Recurrent_{t-1}$
$Recurrent_t$
$Encoder_{t-1}$
$Encoder_t$
x_t
x_{t+1}
fc3(54) relu2 fc2 (100) relu1 fc1 (500)
lstm2 (1000) lstm1 (1000)
fc2 (500) relu1 fc1 (500) noise
fc2 (648) relu1 fc1 (500)
lstm1 (1000)
relu5 conv5 relu4 conv4 relu3 conv3 pool2 relu2 conv2 pool1 relu1 conv1

Figure 1. **ERDs for human dynamics in video and motion capture.** Given a mocap sequence till time t, the ERD for mocap generation predicts the mocap vector at time instance $t+1$. Given a person tracklet till time t, ERD for video forecasting predicts body joint heat maps of the *next* frame $t+1$. ERD for video labeling predicts heat maps of the *current* frame instead.

video. In contrast, ERDs learn the representation suitable for temporal reasoning and can take advantage of visual appearance and context.

3. ERDs for recurrent kinematic tracking and forecasting

Figure 1 illustrates ERD models for recurrent kinematic tracking and forecasting. At each time step t, vector x_t of a sequence $\mathbf{x} = (x_1, \cdots, x_T)$ passes through the encoder, the recurrent layers, and the decoder network, producing the output y_t. In general, we are interested in estimating some function $f(x)$ of the input x at the current time step, or at some time in the future. For example, in the case of motion capture, we are interested in estimating the mocap vector at the next frame. Since both the input and output consists of mocap vectors, f is the identity transformation, and the desired output at step t is $f(x_{t+1})$. In case of video pose labeling and forecasting, $f(x)$ denotes body joint locations corresponding to the image in the current bounding box x. At step t, we are interested in estimating either $f(x_t)$ in the case of labeling, or $f(x_{t+H})$ in the case of forecasting, where H is the forecast horizon.

The units in each recurrent layer implement the Long Short Term Memory functions [11], where writing, resetting, and reading a value from each recurrent hidden unit is explicitly controlled via gating units, as described by Graves [9]. Although LSTMs have four times more parameters than regular RNNs, they facilitate long term storage of task-relevant data. In Computer Vision, LSTMs have been used so far for image captioning [43] and action classification in videos [6].

ERD architecture extends prior work on LSTMs by augmenting the model with encoder and decoder networks. Omitting the encoder and decoder networks and instead using linear mappings between the input, recurrent state, and output caused underfitting on all three of our tasks. This can be explained by the complexity of the mocap and video input in comparison to the words or pen stroke 2D locations considered in prior work [9]. For example, word embeddings were not crucial for RNNs to do well in text generation or machine translation, and the standard one hot encoding vocabulary representation also showed excellent results [31].

3.1. Generating Motion Capture

Our goal is to predict the mocap vector in the next frame, given a mocap sequence so far. Since the output y_t has the same format as the input x_{t+1}, if we can predict x_{t+1}, we can "play" the motion forward in time to generate a novel mocap sequence by feeding the output at the preceding time step as the input to the current one.

Each mocap vector consists of a set of 3D body joint angles in a kinematic tree representation. We represent the orientation of each joint by an exponential map in the coordinate frame of its parent, corresponding to 3 degrees of freedom per joint. The global position of the body in the x-y plane and the global orientation about the vertical z axis are predicted relative to the previous frame, since each clip has an arbitrary global position. This is similar to the approach taken in previous work [34]. We standardize our input by mean subtraction and division by the standard deviation along each dimension.

We consider both deterministic and probabilistic predictions. In the deterministic case, the decoder's output y_t is a single mocap vector. In this case, we train our model by minimizing the Euclidean loss between target and predicted body joint angles. In the probabilistic case, y_t parametrizes a Gaussian Mixture Model (GMM) over mocap vectors in the next frame. We then minimize the GMM negative log-likelihood during training:

$$\mathcal{L}(\mathbf{x}) = -\sum_{t=1}^{T} \log\Pr(x_{t+1}|y_t) \tag{1}$$

We use five mixture components and diagonal covariances. The variances are outputs of exponential layers to ensure positivity, and the mixture component probabilities are outputs of a softmax layer, similar to [9]. During training, we pad the variances in each iteration by a fixed amount to ensure they do not collapse around the mixture means. Weights are initialized randomly. We experimented with initializing the encoder and decoder networks of the mocap ERD from the (first two layers of) encoder and (last two layers of) decoder of a) a ten layer autoencoder trained for dimensionality reduction of mocap vectors [10], b) a "skip" autoencoder trained to reconstruct the mocap vector in few frames in the future given the current one. In both cases, we did not observe improvement over random weight initialization. We train our ERD model with stochastic gradient descent and backpropagation through time [50] with momentum and gradient clipping at 25, using the publicly available Caffe package [15] and the LSTM layer implementation from [6].

We regularize our mocap ERD with denoising: we provide mocap vectors corrupted with zero mean Gaussian noise [42] and have the model predict the correct, uncorrupted mocap vector in the next frame. We found it valuable to progressively increase the noise standard deviation, learning from non-corrupted examples first. This corresponds to a type of curriculum learning. At test time, we run the model forward by feeding the predictions as input to the model in the following time step. Without denoising, this kind of forward unrolling suffers from accumulation of small prediction mistakes at each frame, and the model quickly falls into unnatural regions of the state space. Denoising ensures that corrupted mocap data are shown to the network during training so that it learns to correct small amounts of drift and stay close to the manifold of natural poses.

3.2. Labeling and forecasting video pose

In the previous section, we described how the ERD model can be used to synthesize naturalistic human motion by training on motion capture datasets. In this section, we extend this model to identify human poses directly from pixels in a video. We consider a pose labeling task and a pose forecasting task. In the labeling task, given a bounding box sequence depicting a person, we want to estimate body joint locations for the *current* frame, given the sequence so far. In the forecasting task, we want to estimate body joint locations for a specific future time instance instead.

We represent K body joint locations as a set of K $N \times N$ heat maps over the person's bounding box, that represent likelihood for each joint to appear in each of the N^2 grid locations, similar to [38]. Predicting heat maps naturally incorporates uncertainty over body joint locations, as opposed to predicting body joint pixel coordinates.

Figure 1*right* illustrates our ERD architecture for video pose labeling and forecasting. The encoder is a five layer convolutional network with architecture similar to Krizhevsky *et al.* [18]. Our decoder is a two layer network with fully connected layers interleaved with rectified linear unit layers. The output of the decoder is body joint heat maps over the person bounding box in the current frame for the labeling task, or body joint heat maps at a specified future time instance for the forecasting task.

We train both our pose labeler and forecaster ERDs under a Euclidean loss between estimated and target heat maps. We initialize the weights of the encoder from a six layer convolutional network trained for per frame body part detection, in which the final CONV6 layer corresponds to the body joint heat maps.

Empirically, we found it valuable to input to the recurrent layer not the per frame estimated heat maps (CONV6), but rather the preceding CONV5 feature maps. These feature maps capture rich appearance information, rather than merely body joint likelihood. Rich appearance information assists the network in discriminating between different actions and pose dynamics without explicit switching across activity domains, as previous switching dynamical linear systems or HMMs [21].

We use two networks on different image scales for our per frame pose detector and ERD: one where the output layer resolution is 6×6 and one that works on double image size and has output resolution of 12×12. The heat maps of the coarser scale are upsampled and added to the finer scale to provide the final combined 12×12 heat maps. Multiple scales have shown to be beneficial for static pose estimation in [38, 36, 37].

4. Experiments

We test our method on the H3.6M dataset of Ionescu *et al.* [13], which is currently the largest video pose dataset. It consists of 15 activity scenarios, performed by seven different professional actors and recorded from four static cameras. For each activity scenario, subject, and camera viewpoint, there are two video sequences, each between 3000 and 5000 frames. Each activity scenario features rich gestures, pose variations and interesting subactions performed by the actors. For example, the walking activity includes holding hands, carrying a heavy load, putting hands in the pockets, looking around etc. The activities are recorded using a Vicon motion capture system that tracks markers on actors' body joints and provides high quality 3D body joint locations. 2D body joints locations are obtained by projecting the 3D positions onto the image plane using the known camera calibration and viewpoint. For all our experiments, we treat subject 5 as the test subject and all others as our training subjects.

Motion capture generation We compare our ERD mocap generator with a) an LSTM recurrent neural network with linear encoder and decoders that has 3 LSTM layers of 1000 units each (architecture found through experimentation to work well), b) Conditional Restricted Boltzmann Machines (CRBMs) of Taylor *et al.* [34], c) Gaussian Process Dynamic Model (GPDM) of Wang *et al.*[44], and d) a nearest neighbor N-gram model (NGRAM). For CRBM and GPDM, we used the code made publicly available by the authors. For the nearest neighbor N-gram model, we used a frame window of length $N = 6$ and Euclidean distance on 3D angles between the conditioning prefix and our training set, and copy past the subsequent frames of the best matching training subsequence. We applied denoising during training to regularize both the ERD and the LSTM-3LR. For all models, the mocap frame sequences were subsampled by two. ERD, LSTM-3LR and CRBM are trained on multiple activity scenarios (Walking, Eating and Smoking). GPDM is trained on Walking activity only, because its cubic complexity prohibits its training on a large number of sequences. Our comparison focuses on motion forecasting (prediction) and synthesis, conditioning on motion prefixes of our test subject. Mocap in-filling and denoising are non-trivial with our current model but developing this functionality is an interesting avenue for future work.

We show qualitative motion synthesis results in Figure 2 and quantitative motion prediction errors in Table 1. In Figure 2, the conditioning motion prefix from our test subject is shown in green and the generated motion is shown in blue. In Table 1, we show Euclidean norm between the synthesized motion and ground-truth motion for our test subject for different temporal horizons past the conditioning motion prefix, the largest being 560msecs, averaged across 8 different prefixes. The stochasticity of human motion prevents a metric evaluation for longer temporal horizons, thus all comparisons in previous literature are qualitative. LSTM-3LR dominates the short-term motion generation, yet soon converges to the mean pose, as shown in Figure 2. CRBM also provides smooth short term motion completions, yet quickly drifts to unrealistic motions. ERD provides slightly less smooth completions, yet can generate realistic motion for long periods of time. For ERD, the smallest error was always produced by the most probable GMM sample, which was similar to the output of an ERD trained under a standard Euclidean loss. N-gram model exhibits a sudden change of style during transitioning from the conditioning prefix to

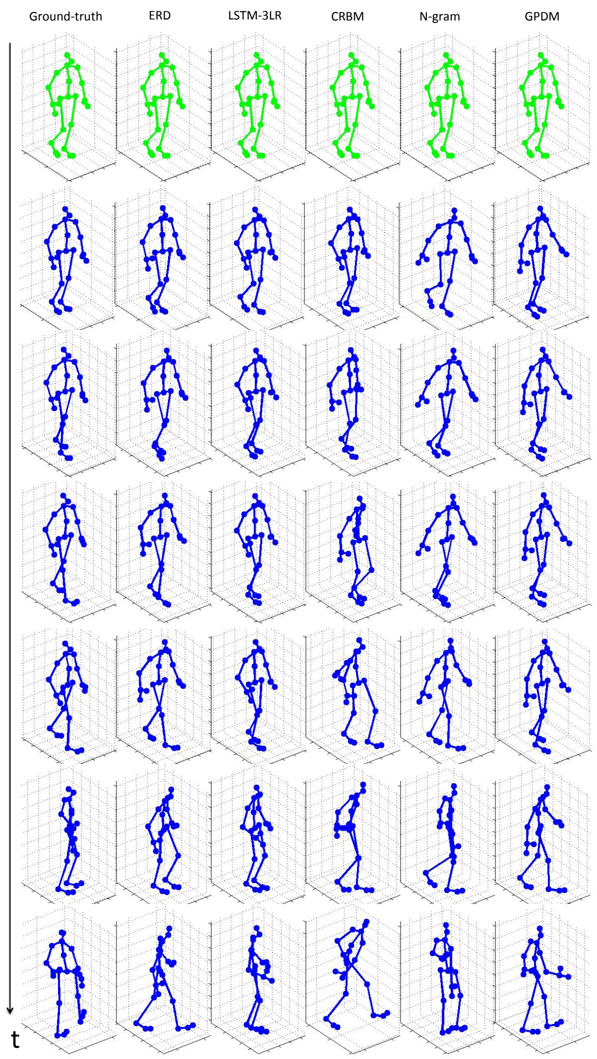

Figure 2. **Motion synthesis.** LSTM-3LR and CRBMs [34] provide smooth short-term motion completions (for up to 600msecs), mimicking well novel styles of motion, (e.g., here, walking with upright back). However, ERD generates realistic motion for longer periods of time while LSTM-3LR soon converges to the mean pose and CRBM diverges to implausible motion. NGRAM has a non-smooth transition from conditioning to generation. Per frame mocap vectors predicted by GPDM [44] look plausible, but their temporal evolution is far from realistic. You can watch the corresponding video results at `https://sites.google.com/site/motionlstm/` .

the first generated frame, and cannot generate anything outside of the training set. Due to low-dimensional embedding, GPDM cannot adequately handle the breadth of styles in the training data, and produces unrealistic temporal evolution.

The quantitative and qualitative motion generation results of ERD and LSTM-3LR suggest an interesting tradeoff between smoothness of motion completion (interesting motion extrapolations) and stable long-term motion generation. Generating short-term motion that mimics the style of the test subject is possible with LSTM-3LR, yet, since the network has not encountered similar examples during training, it is unable to correctly generate motion for longer periods of time. In contrast, ERD gears the generated motion towards similarly moving training examples. ERD though cannot really extrapolate, but rather interpolate among the training subjects. It does provides much smoother motion completions than the N-gram baseline. Both setups are interesting and useful in different applications, and in between architectures potentially lie somewhere in between the two ends of that spectrum. Finally, it is surprising that LSTM-3LR outperforms CRBMs given its simplicity during training and testing, not requiring inference over latent variables.

	80	160	240	320	400	480	560
ERD	0.89	1.39	1.93	2.38	2.76	3.09	3.41
LSTM-3LR	**0.41**	**0.67**	**1.15**	**1.50**	**1.78**	**2.02**	**2.26**
CRBM [34]	0.68	1.13	1.55	2.00	2.45	2.90	3.34
6GRAM	1.67	2.36	2.94	3.43	3.83	4.19	4.53
GPDM [44]	1.76	2.5	3.04	3.52	3.92	4.28	4.61

Table 1. **Motion prediction error** during 80, 160, 240, 320, 400, 480 and 560 msecs past the conditioning prefix for our test subject during Walking activity. Quantitative evaluation for longer temporal horizons is not possible due to stochasticity of human motion.

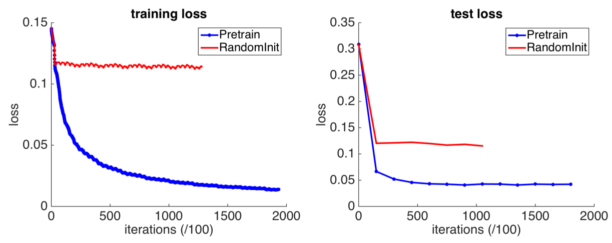

Figure 3. **Pretraining.** Initialization of the CNN encoder with the weights of a body pose detector leads to a much better solution than random weight initialization. For motion generation, we did not observe this performance gap between pertaining and random initialization, potentially due to much shallower encoder and low dimensionality of the mocap data.

Video pose labeling Given a person bounding box sequence, we want to label 2D pixel locations of the person's body joint locations. Both occluded and non-occluded body joints are required to be detected correctly: the occluder's appearance often times contains useful information regarding the location of an occluded body joint [5]. Further, for transcribing 2D to 3D pose, all body joints are required [32].

We compare our ERD video labeler against two baselines: a per frame CNN pose detector (PF) used as the encoder part of our ERD model, and a dynamic programming approach over multiple body pose hypotheses per frame (VITERBI) similar in spirit to [20, 2]. For our VITERBI baseline, we consider for each body joint in each frame all possible grid locations and encode temporal smoothness as the negative exponential of the Euclidean distance between the locations of the same body joint across consecutive frames. The intuition behind VITERBI is that temporal smoothness will help rule out isolated, bad pose estimates, by promoting ones that have lower per frame scores, yet are more temporally coherent.

We evaluate our model and baselines by recording the highest scoring pixel location for each frame and body joint. We compute the percentage of detected joints within a tolerance radius of a circle centered at the ground-truth body joint locations, for various tolerance thresholds. We normalize the tolerance radii with the distance between left hip and right shoulder. This is the standard evaluation metric for static image pose labeling [25]. We show pose labeling performance curves in Figure 4. For a video comparison between ERD and the per frame CNN detector, please see the video at `https://sites.google.com/site/motionlstm/`.

discriminatively learning to integrate temporal information for body joint tracking, instead of employing generic motion smoothness priors. ERD's performance boost stems from correcting left and right confusions of the per frame part detector, as Figure 5 qualitatively illustrates. Left and right confusion is a major challenge for per frame part detectors, to the extent that certain works measure their performance in image centric coordinates, rather than object centric [51, 25]. Last, VITERBI is marginally better than the per frame CNN detector. While motion coherence proved important when combined with shallow and inaccurate per frame body pose detectors [20, 2], it does not improve much upon stronger multilayer CNNs.

Figure 3 compares ERD training and test losses during finetuning the encoder from (the first five layers of) our per frame CNN pose detector, versus training the encoder from scratch (random weights). CNN encoder's initialization is crucial to reach a good solution.

We further compare our video labeler in a subset of 200 video sequences of around 50 frames each from the FlicMotion dataset of [14, 25] that we annotated densely in time with person bounding boxes. We used 170 video sequences for training and 30 for testing. We show performance curves for the upper body joints in Figure 7. VITERBI has simi-

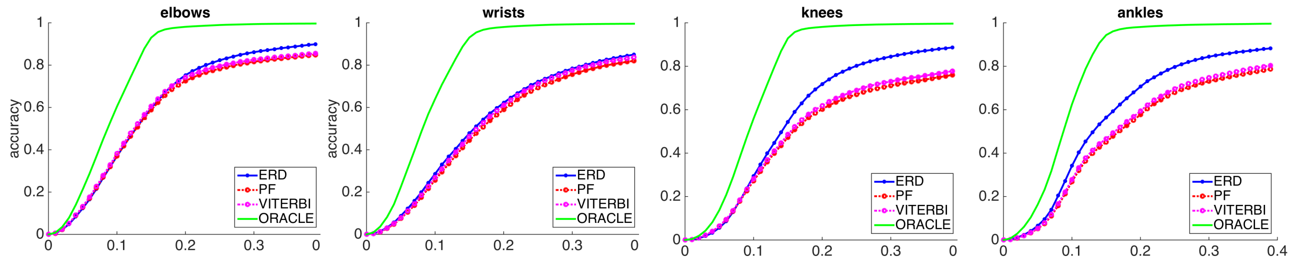

Figure 4. **Video pose labeling in H3.6M.** Quantitative comparison of a per frame CNN body part detector of [38] (PF), dynamic programming for temporal coherence of the body pose sequence in the spirit of [20, 2] (VITERBI), and ERD video pose labeler. ERD outperforms the per frame detector as well as the dynamic programming baseline. Oracle curve shows the performance upper-bound imposed by our

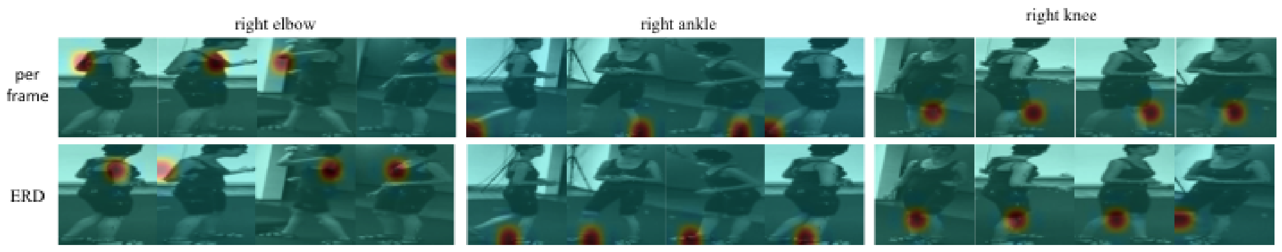

Figure 5. **Left-right disambiguation.** ERD corrects left-right confusions of the per frame CNN detector by aggregating appearance features (CONV5) across long temporal horizons.

lar performance as in H3.6M, marginally exceeding the per frame CNN detector. However ERD does much worse since the training set is too small to learn effectively. Finetuning from the model learnt from H3.6M did not help since H3.6M concerns full body motion while FlicMotion captures upper body only. We did not change the architecture in comparison to the ERD used in H3.6M. It is probable that a smaller recurrent layer and decoder would improve performance preventing overfitting. Large training sets such as in H3.6M allow high capacity discriminative temporal smoothers as our video labelled ERD to outperform generic motion smoothness priors for human dynamics.

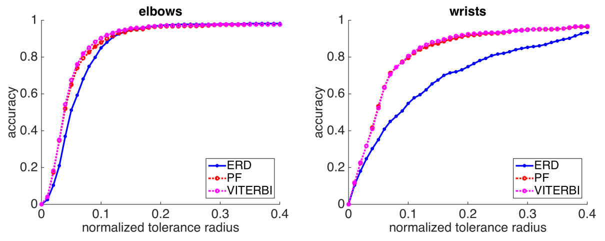

Figure 7. **Video pose labeling in FlicMotion.** ERD does not succeed in learning effectively from the small set of 170 videos of about 50 frames each. Large training sets, such as those provided in H3.6M, are necessary for ERD video labeler to outperform generic motion smoothness priors.

Video pose forecasting We predict 2D body joint locations 400ms ahead of the current frame. Figure 6 shows pose forecasting performance curves for our ERD model, a model that assumes zero object and camera motion (NoMotion-*NM*), and a model that assumes constant optical flow within the prediction horizon (*OF*). ERD carries out more accurate predictions than the zero order and first order motion baselines, as also shown qualitatively in Figure 8. Optical flow based motion models cannot make reasonable predictions for occluded body joints, since their frame to frame displacements are not observed. Further, standard motion models suffer from separation of the observation model (part detector) and temporal aggregation, which ERD combines into a single network.

Discussion Currently, the mocap ERD performs better on periodic activities (walking, smoking etc) in comparison to non periodic ones (sitting etc.). Interesting directions for future research is predicting 3D angle differences from frame to frame as opposed to angles directly. Such transformation prediction may generalize better to new subjects, focusing more on motion rather than appearance of the skeleton. We are also investigating using large frame volumes as input to our video prediction ERDs with spatio-temporal convolutions in CONV1 as opposed to a single frame LSTM, in order to exploit short temporal horizon more effectively.

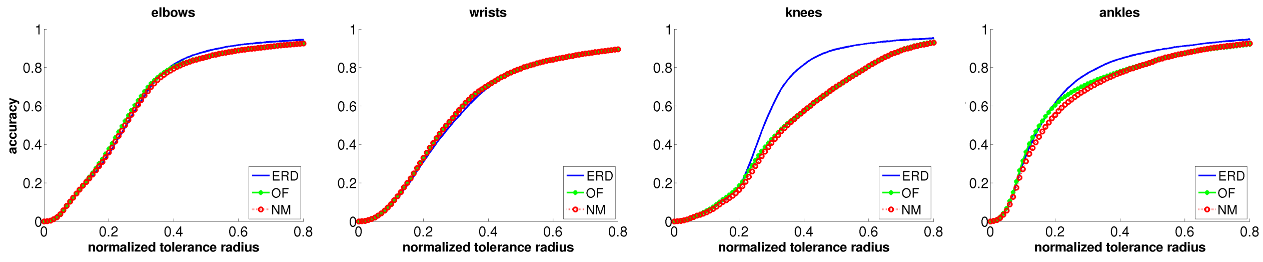

Figure 6. **Video pose forecasting.** Quantitative comparison between the ERD model, a zero motion (NM), and constant velocity (OF) models. ERD outperforms the baselines for the lower body limbs, which are frequently occluded and thus their per frame motion is not frequently observed using optical flow.

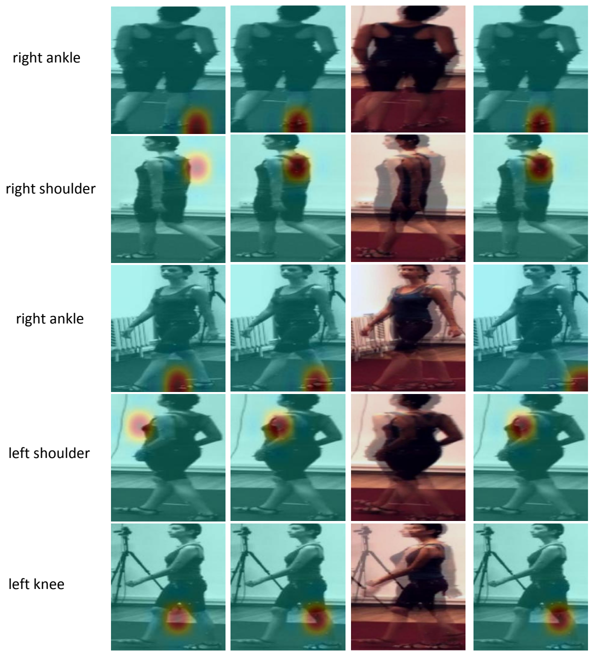

Figure 8. **Video pose forecasting** 400ms in the future. *Left:* the prediction of the body part detector 400ms before superimosed on the frame to predict pose for (zero motion model). *MiddleLeft:* Predictions of the ERD. The body joints have been moved towards their correct location. *MiddleRight:* The current and 400ms ahead frame superimposed. *Right:* Ground-truth body joint location (discretized in a $N \times N$ heat map grid). In all cases we show the highest scoring heat map grid location.

5. Conclusion

We have presented end-to-end discriminatively trained encoder-recurrent-decoder models for modeling human kinematics in videos and motion capture. ERDs learn the representation for recurrent prediction or labeling, as well as its dynamics, by jointly training encoder recurrent and decoder networks. Such expressive models of human dynamics come at a cost of increased need for training examples. In future work, we plan to explore semi-supervised models in this direction, as well learning human dynamics in multi-person interaction scenarios.

Acknowledgements We would like to thank Jeff Donahue and Philipp Krähenbühl for useful discussions. We gratefully acknowledge NVIDIA corporation for the donation of K40 GPUs for this research. This research was funded by ONR MURI N000014-10-1-0933.

References

[1] I. Akhter, T. Simon, S. Khan, I. Matthews, and Y. Sheikh. Bilinear spatiotemporal basis models. *ACM Transactions on Graphics*, 31, 2012. 2

[2] D. Batra, P. Yadollahpour, A. Guzman-Rivera, and G. Shakhnarovich. Diverse ”m”-best solutions in markov random fields. In *ECCV*, 2012. 2, 6, 7

[3] M. Brand and A. Hertzmann. Style machines. SIGGRAPH, 2000. 2

[4] L. Cun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *NIPS*, 1990. 2

[5] C. Desai and D. Ramanan. Detecting actions, poses, and objects with relational phraselets. In *ECCV*, 2012. 6

[6] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, 2014. 1, 3, 4

[7] E. Fox, E. Sudderth, M. Jordan, and A. Willsky. Bayesian nonparametric methods for learning Markov switching processes. *Signal Processing Magazine, IEEE*, 27(6):43–54, 2010. 2

[8] H. Gong, J. Simy, M. Likhachev, and J. Shi. Multi-hypothesis motion planning for visual object tracking. In *ICCV*, 2011. 1

[9] A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013. 1, 2, 3, 4

[10] G. E. Hinton and R. R. Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, July 2006. 4

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. 3

[12] E. Hsu, K. Pulli, and J. Popović. Style translation for human motion. *ACM Transactions on Graphics*, 24(3):1082–1089, July 2005. 2

[13] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI*, 36(7), 2014. 2, 4

[14] A. Jain, J. Tompson, Y. LeCun, and C. Bregler. Modeep: A deep learning framework using motion features for human pose estimation. In *ACCV*, 2014. 6

[15] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding, 2013. 4

[16] S. Kombrink, T. Mikolov, M. Karafiát, and L. Burget. Recurrent neural network based language modeling in meeting recognition. In *INTERSPEECH*, 2011. 1

[17] H. S. Koppula and A. Saxena. Anticipating human activities for reactive robotic response. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, page 2071, 2013. 1

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*. 2012. 4

[19] X. Lan and D. Huttenlocher. A unified spatio-temporal articulated model for tracking. In *CVPR*, volume 1, 2004. 2

[20] D. Park and D. Ramanan. N-best maximal decoders for part models. In *ICCV*, pages 2627–2634, 2011. 2, 6, 7

[21] V. Pavlovic, J. M. Rehg, and J. MacCormick. Learning switching linear models of human motion. In *NIPS*, 2000. 2, 4

[22] S. Pellegrini, A. Ess, K. Schindler, and L. J. V. Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, 2009. 1

[23] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. *CoRR*, abs/1412.6604, 2014. 1, 2

[24] P. Rodriguez, J. Wiles, and J. Elman. A recurrent neural network that learns to count. . *Connection Science*, 1, 1999. 1

[25] B. Sapp and B. Taskar. Modec:multimodal decomposable models for human pose estimation. In *CVPR*, 2013. 6

[26] B. Sapp, D. Weiss, and B. Taskar. Parsing human motion with stretchable models. In *CVPR*, 2011. 2

[27] L. Sigal, M. Isard, H. Haussecker, and M. J. Black. Loose-limbed people: Estimating 3D human pose and motion using non-parametric belief propagation. *IJCV*, 98, 2012. 2

[28] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Discriminative density propagation for 3D human motion estimation. In *In CVPR*, 2005. 2

[29] I. Sutskever, G. E. Hinton, and G. W. Taylor. The recurrent temporal restricted boltzmann machine. In *NIPS*, 2008. 2

[30] I. Sutskever, J. Martens, and G. Hinton. Generating text with recurrent neural networks. In *ICML*, 2011. 1

[31] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014. 1, 3

[32] C. J. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. *CVIU*, 80, 2000. 6

[33] G. W. Taylor and G. E. Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In *ICML*, 2009. 2

[34] G. W. Taylor, G. E. Hinton, and S. T. Roweis. Modeling human motion using binary latent variables. In *NIPS*, 2006. 2, 3, 5, 6

[35] G. W. Taylor, L. Sigal, D. J. Fleet, and G. E. Hinton. Dynamical binary latent variable models for 3d human pose tracking. In *CVPR*, 2010. 2

[36] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014. 4

[37] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659, 2013. 4

[38] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *CVPR*, 2015. 4, 7

[39] R. Urtasun, D. J. Fleet, and P. Fua. 3d people tracking with gaussian process dynamical models. In *CVPR*, 2006. 2

[40] R. Urtasun, D. J. Fleet, A. Geiger, J. Popovic, T. Darrell, and N. D. Lawrence. Topologically-constrained latent variable models. In *ICML*, 2008. 2

[41] D. Vernon, C. von Hofsten, and L. Fadiga. A roadmap for cognitive development in humanoid robots. 11, 2011. 1

[42] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010. 4

[43] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014. 1, 3

[44] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models. In *In NIPS*, pages 1441–1448. MIT Press, 2006. 5, 6

[45] J. M. Wang, D. J. Fleet, and A. Hertzmann. Multifactor gaussian process models for style-content separation. In *ICML*, 2007. 2

[46] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *TPAMI*, 30, 2008. 2

[47] S.-K. Weng, C.-M. Kuo, and S.-K. Tu. Video object tracking using adaptive kalman filter. *J. Vis. Comun. Image Represent.*, 17(6):1190–1208, 2006. 2

[48] Wikipedia. Lagrangian and eulerian specification of the flow field. 2

[49] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.*, 1(2), 1989. 1

[50] R. J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity, 1995. 4

[51] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011. 6