Video Extrapolation with an Invertible Linear Embedding

Robert Pottorff[*] , **Jared T Nielsen** , **David Wingate**

Brigham Young University

rpottorff@byu.net, jaredtnielsen@gmail.com, wingated@cs.byu.edu

Abstract

We predict future video frames from complex dynamic scenes, using an invertible neural network as the encoder of a nonlinear dynamic system with latent linear state evolution. Our invertible linear embedding (ILE) demonstrates successful learning, prediction and latent state inference. In contrast to other approaches, ILE does not use any explicit reconstruction loss or simplistic pixel-space assumptions. Instead, it leverages invertibility to optimize the likelihood of image sequences exactly, albeit indirectly. Comparison with a state-of-the-art method demonstrates the viability of our approach.

1 Introduction

Video frame extrapolation is the generation of future frames conditioned on past ones. Due to the ubiquity of image and video sequences, video prediction plays a central role in diverse fields such as self-driving vehicles and reinforcement learning. In these domains, high quality video prediction correlates directly with improved practical performance.

Frame prediction also offers a well-posed unsupervised objective for representation learning. Any successful algorithm must have extracted salient features useful for describing both the content and dynamics of a scene. To some degree, video prediction and representation learning are essentially the same task. With the right representation, prediction is easy, stable, and efficient; with the wrong one, it may be difficult or impossible [Bengio *et al.*, 2013].

The grand vision of representation learning is to understand how useful encodings can be learned. Although there is no consensus as how this should be done, the video prediction objective offers a well-posed unsupervised task that may provide insight into how these productive representations may be learned and may produce them itself [Mathieu *et al.*, 2015].

Video prediction can be modeled as a probability distribution over future frames. This stochasticity presents a challenge to prediction tasks. Consider a video of a falling leaf - it may float in one direction or another with little indication of future direction. To approximate the true distribution of future frames, some approaches minimize pixel-wise mean
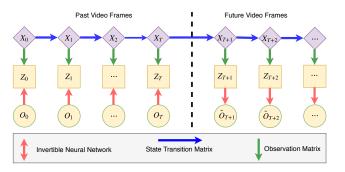

Figure 1: We model video frame generation as a nonlinear dynamic system. The task is to find a suitable invertible linear embedding which encodes pixel-space observations o_t as z_t. We treat z_t as observations of a Markov process and solve for the initial latent state x_0. The invertible neural network, state transition matrix, and observation matrix are all learned parameters. This enables exact maximum likelihood learning without reconstruction, adversarial, or lower-bound losses in the image domain.

squared error (MSE) reconstruction loss. Others optimize a lower bound, typical of variational auto-encoders (VAEs). Still others use generative adversarial (GAN) discriminators to approximate the likelihood in the data domain. However, none of these approaches model the true distribution.

Our main contribution, an invertible linear embedding (ILE), combines invertible neural networks and a latent linear dynamical system to explicitly model the true distribution. By leveraging an invertible function approximator and the change of variables formula, frame prediction likelihood can be precisely equated with the likelihood of an observation from a linear dynamical system.

1.1 Related Work

Modern approaches to frame prediction use advances in neural network research to make state-of-the-art gains in predictive performance, our workincluded. [Oh *et al.*, 2015] and [Chiappa *et al.*, 2017] use an action-conditioned deep auto-encoder to estimate the frames of Atari games. [Reda *et al.*, 2018] uses a network to predict optical flow that is then used to warp frames into the future. [Lee *et al.*, 2018] attempts to model the true distribution by fusing VAEs with GANs, using the VAE approach to discourage mode collapse in GANs, and the GAN discriminator to overcome the lower-bound approx-

---

[*]Contact Author

imation in VAEs. Similar to our work, [Watter *et al.*, 2015] and its extension [Banijamali *et al.*, 2017] use locally-linear dynamical models that use a latent structure, but ultimately rely on a variational lower bound to approximate the posterior. [Mathieu *et al.*, 2015] and its extension [Lotter *et al.*, 2016] use image gradient losses and adversarial training to achieve stable and crisp results. Our experiments compare against this particular algorithm. These works differ in both their network topology and the loss functions they employ to approximate the true distribution, but all use some form of reconstruction error as a regularization. We explore the impact this common assumption has on maximum likelihood estimation later in this work.

Conceptually, modeling a nonlinear dynamic system as an tuple of an encoding function and a linear (possibly time-invariant) dynamic system is known in control literature as a Hammerstein-Wiener block model[1] [Janczak, 2004]. This literature has historically focused on low-dimensional systems using methods which do not scale well to high dimensional systems like video. We extend it here with neural network techniques for high dimensional systems such as image sequences.

2 Method

Formally, we consider a video sequence as an ordered tuple of T frames, each denoted as o_t. The abstract problem of video extrapolation is to learn the conditional distribution over future frames, given past frames:

$$p(o_t \mid o_{t-1}, \ldots, o_0)$$

This distribution is extremely complicated with no closed form to tractably sample, score, or approximate directly. In lieu of direct approximation, we consider transformed frames $g_\theta(o_t) = z_t$, where g is a neural network parameterized by θ, which we refer to as *embeddings* or *encodings*:

$$p(g_\theta(o_t) \mid o_{t-1}, \ldots, o_0) = p_\theta(z_t \mid z_{t-1}, \ldots, z_0)$$

Provided that g_θ is sufficiently expressive and invertible, we can define an equivalence between a typically tractable distribution over observations p_θ parameterized by θ and the true distribution over frames p using a change of variables:

$$p(o_t \mid o_{t-1}, \ldots, o_0) = p_\theta(z_t \mid o_{t-1}, \ldots, o_0) \mid \det \frac{\partial z_t}{\partial o_t} \mid$$

This equality allows us to learn the true distribution using a maximum likelihood objective:

$$\max_\theta \ p_\theta(z_t \mid o_{t-1}, \ldots, o_0) \mid \det \frac{\partial z_t}{\partial o_t} \mid \qquad (1)$$

In this work, we use an invertible neural network as the model class for $g_\theta(o_t) = z_t$ and a linear time-invariant dynamic system (LTI) to define the tractable likelihood p_θ. To our knowledge, this is the first work to demonstrate successful learning in reversible flow networks using an LTI prior and one of the few works in video frame extrapolation to avoid making any assumptions about the data distribution.

---

[1]Technically the model presented here is a Wiener model, but we consider the connection to the generalized model in the literature to be important.

Algorithm 1 Invertible Linear Embedding

1: **Returns the following:**
2:    g_θ: a learned invertible neural network
3:    A: a learned state transition matrix
4:    C: a learned observation matrix
5: **while** $\mathcal{L}$ is not minimized **do**
6:    Sample $o_0, \ldots, o_{T-1}$ frames
7:    **for** $t = 0, \cdots, T-1$ **do**
8:      $z_t = g_\theta^{-1}(o_t) \in \mathbb{R}^D$
9:      $s_t = |\det \frac{\partial z_t}{\partial o_t}|$
10:    **end for**
11:    $\mathcal{Z} = \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_{T-1} \end{bmatrix} \quad \mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{T-1} \end{bmatrix}$
12:    $x_0^* = \mathcal{O}^+ \mathcal{Z}$
13:    $\hat{\mathcal{Z}} = \mathcal{O} x_0^*$
14:    $\gamma = \frac{1}{D} \|\mathcal{Z}\|_1$
15:    $\mathcal{L} = \frac{1}{2} \|\gamma^{-1}(\mathcal{Z} - \hat{\mathcal{Z}})\|_2^2 + \sum_t^T [\log s_t] - \log \gamma$
16:    Take gradient step in A, C, θ to minimize $\mathcal{L}$
17: **end while**
18: $o_T = CA^T x_0^*$

2.1 Reconstruction Error and Implicit Assumptions

To distinguish between a large class of prior work and our contribution, we highlight a distinction between a common candidate objective function and the true distribution described in Equation 1. A common framework for video prediction involves learning an encoding function e_θ, a separate decoding function d_θ, and a transition function f_θ. Learning the decoder has the practical purpose that the system avoids perfectly predictable but not particularly useful minima such as $e_\theta = 0$. A typical loss is usually defined:

$$\min_\theta \ \alpha \|f_\theta(e_\theta(o_t)) - e_\theta(o_{t+1})\| + \beta \|d_\theta(e_\theta(o_t)) - o_t\|$$

When using L^2 as the norm, minimizing this candidate objective function is equivalent to maximum log likelihood learning under three assumptions. First, that the conditional distribution of the error *of the embedding* is an isotropic Gaussian.

$$p_\theta(e_\theta(o_{t+1})|e_\theta(o_t)) = \mathcal{N}(e_\theta(o_{t+1}), \alpha^{-1})$$

Second, that the input images are isotropic Gaussian with a mean defined by the decoder

$$p(o_t|e_\theta(o_t)) = \mathcal{N}(o_t, \beta^{-1})$$

Third, that the determinant of the encoder's Jacobian is 1. If our observations are image pixel intensities, these may not hold. While the first assumption is valid (given a sufficiently expressive encoder), the second and third are not. The term $\|d_\theta(e_\theta(o_t)) - o_t\|$, which we call *reconstruction loss*, compares an observation o_t with its reconstruction $d_\theta(e_\theta(o_t))$ using pixel-wise mean-squared error, known to perform poorly in the case of translations and brightness variations. More

generally, it completely ignores valuable higher-order information in images: pixel intensities are neither independent nor do they share the same variance. The third assumption is likewise almost certainly not true for traditional autoencoders. Put simply, this loss implies false assumptions and results in a different objective than the one we would truly like to minimize.

In GAN approaches, the error between the true and approximate distribution is theoretically bounded by the complexity of the discriminator which acts as a data-driven direct approximation to $p(o_t|\cdot)$. In practice, adversarial losses are difficult to train and are generally used as an additional regularizing term in loss functions which make similar simplifying assumptions about the data distribution [Mathieu *et al.*, 2015].

2.2 Background: Invertible Networks

Recent work on invertible networks, also known as reversible flows, are a relatively new approach to deep generative modeling [Dinh *et al.*, 2014; Dinh *et al.*, 2016; Kingma and Dhariwal, 2018] which introduces techniques for learning exactly invertible neural networks. This work represents the backbone of our method. We consider a generative model using a known parameterized distribution $p_\theta(z)$ and a deterministic function $g_\theta(z)$:

$$z \sim p_\theta(z), \qquad o = g_\theta(z),$$

where $g_\theta(z)$ has the compositional form

$$g_\theta^{(N)}(g_\theta^{(N-1)}(\cdots g_\theta^{(0)}(z)\cdots))$$

in which each successive layer operates on the output of the layer before (abbreviated notationally as $g_\theta^i(h_{i-1})$). The change of variables formula enables us to relate:

$$\log p(o) = \log p_\theta(g^{-1}(z)) + \sum_{i=0}^{N} |\det \frac{\partial h_i}{\partial h_{i-1}}|,$$

with $h_0 = o$ and $h_N = z$. Because p_θ is tractable, we need only for the determinant of each layer's Jacobian to be tractable to efficiently compute the density $\log p(o)$.

Borrowing on the early work in this field, we consider the following technique for $g_\theta^{(i+1)}(h_i)$ called an *affine coupling* which makes this determinant easy to compute:

$$h_i^{\text{left}}, h_i^{\text{right}} = \text{split}(h_i)$$
$$s_i = f_i(h_i^{\text{left}})$$
$$h_{i+1}^{\text{right}} = s_i \odot h_i^{\text{right}} + b_i(h_i^{\text{left}})$$
$$h_{i+1} = P_i \begin{bmatrix} h_i^{\text{left}} \\ h_{i+1}^{\text{right}} \end{bmatrix}$$

where $h_i \in \mathcal{R}^D$ is the layer input, $\odot$ is the element-wise product, f_i and b_i are arbitrary neural networks (not necessarily invertible), and P_i is a unimodular matrix which mixes elements between the two halves of h_i. Although this may seem intimidating, the computation is straightforward: using half of the layer's input we learn to produce an affine transformation to apply to the other half. This operation is invertible,

and the log-determinant of this layer is simply:

$$\log |\det \frac{\partial h_i}{\partial h_{i-1}}| = \log \sum_{j=0}^{D/2} |s_{ij}|.$$

The log-determinant of the entire encoding function is the sum of these terms for all layers i:

$$\log |\det \frac{\partial g_\theta^{-1}(o_t)}{\partial o_t}| = \log \sum_{i=0}^{N} \sum_{j=0}^{D/2} |s_{ij}|.$$

Taken together, this functional form enables us to define our decoding function $g_\theta(z)$, its exact inverse, and an efficient computation for the log-determinant of its Jacobian.

2.3 Background: Linear Latent Prior

Recall that in our primary objective function in addition to being able to learn $g_\theta(z)$, we must also be able to define a tractable computation for the distribution $p_\theta(z_t|z_{t-1}, \ldots z_0)$. In this section we introduce linear time-invariant systems as this density function. A natural model for the evolution of a vector-valued observation is that of a linear dynamic system:

$$x_t = Ax_{t-1} \qquad z_t = Cx_{t-1} + \gamma_{t-1} \qquad \gamma_{t-1} \sim \mathcal{N}(0, I)$$

Where x_t represents the hidden state, and z_t the observation at that hidden state. In this work, we assume that this system is *time-invariant*; however, we note that it is possible to extend this model to not only include time-varying dynamics, but also inputs, process noise over the hidden state, or noise distributions with different distributions, but omit them in this work for simplicity.

Linear Time Invariant (LTI) systems define a conditional distribution with tractable density over observations:

$$p_\theta(z_t|z_{t-1}, \ldots, z_0) = \mathcal{N}(CA^t x^*_0, I)$$

where x_0^* is the result of optimal latent state inference. For LTI systems, this is the result of a Kalman filter when conditioned on only past observations, and the Kalman smoother when conditioned on both past and future observations [Welch *et al.*, 1995]. Although many algorithms [Thrun *et al.*, 2005] exist to compute the optimal smoothing estimate x^* they can all be shown to produce the same least squares estimate:

$$x_0^* = \arg\max_{x_0} \left\| \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{T-1} \end{bmatrix} - \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{T-1} \end{bmatrix} x_0 \right\|_2^2$$
$$= \arg\max_{x_0} \| \mathcal{Z} - \mathcal{O}x_0 \|_2^2$$
$$= \mathcal{O}^+ \mathcal{Z}$$

The tractable density function is:

$$p_\theta(z_t|o_{t-1}, \ldots, o_0) = \mathcal{N}(CA^t \mathcal{O}^+ \mathcal{Z}, I)$$

where we use M^+ as the pseudoinverse of M. The efficiency and optimality of hidden state inference is one of the motivating factors behind our choice of a linear model for the latent evolution of embeddings.

Using a linear dynamic system as the transition model for our system introduces two major assumptions. The weaker assumption is the Markov property, that future observations are independent of past observations when conditioned on the hidden state. The stronger assumption is that the future hidden states are a linear mapping from past states and that this mapping remains constant through time. Although the linear dynamics prior may seem quite restrictive, LTI systems are surprisingly expressive and have been shown to model the latent dynamics of many high dimensional models [Brunton *et al.*, 2016; Lusch *et al.*, 2018]. A key theoretical insight in control literature proves the existence of an infinite dimensional linear operator, the Koopman operator, for some nonlinear projection of all nonlinear dynamic systems [Koopman, 1931]. When choosing a large latent hidden dimension, we are approximating this infinite-dimensional operator. So while the modeling assumption made by a linear dynamical prior is almost certainly not true, a large enough state space is a good approximation and will demonstrate the viability of ILE for difficult non-linear systems. Future work could explore options for more expressive yet tractable time-variant dynamic system models.

2.4 Invertible Linear Embedding

We now present our primary contribution: the invertible linear embedding.

Using an invertible neural network as our encoding and decoding function $g_\theta(o)$ and $g_\theta^{-1}(z)$, and an LTI dynamic system as described for the conditional distribution, we can derive our final loss function:

$$\mathcal{L} = -\log p_\theta(o_t \mid o_{t-1}, \ldots, o_0)$$

$$= -\log[\, p_\theta(g_\theta^{-1}(o_t) \mid o_{t-1}, \ldots, o_0)|\det \frac{\partial g_\theta^{-1}(o_t)}{\partial o_t}|]$$

$$= -\log\, p_\theta(z_t \mid o_{t-1}, \ldots, o_0) - \log|\det \frac{\partial z_t}{\partial o_t}|$$

$$= -\log\, \mathcal{N}(CA^t\mathcal{O}^+\mathcal{Z}, I) - \log|\det \frac{\partial z_t}{\partial o_t}|$$

Which results in:

$$\mathcal{L} = \frac{1}{2}\|\mathcal{Z} - \mathcal{O}\mathcal{O}^+\mathcal{Z}\|_2^2 - \log \sum_{i=0}^{N} \sum_{j=0}^{\frac{D}{2}} |s_{ij}| \qquad (2)$$

When minimized using sufficiently expressive $g_\theta(z)$, A, and C parameters, this loss function corresponds to exact maximum likelihood model of a video sequence which is assumed to have latent linear dynamics.

We can describe the function of these two terms intuitively. The first term (the *predictive error*) is the result of encoding each frame independently, solving for the best possible LTI dynamic system trajectory, and applying gradient descent to minimize any error. The more the embeddings behave as a linear system, the lower the predictive error. The second term (the *log-determinant*) encourages the embeddings to be large, preventing the first term from collapsing to easy-to-predict but useless trajectories such as $z_t = 0$. Although it may seem

like a strange regularization to "maximize the embedding values", the application of change of variables and strict invertibility ensures that this is the correct way to learn a mapping between our assumed latent model, and the true observations in image space.

2.5 Stability and Parameterization of A

Given the numerical instability induced from computing a least-squares solution in our training loop, the parameterization of the linear dynamic system is of critical concern. In particular, we must parameterize the learning method to maintain stable state transition matrices A. A stable discrete-time linear dynamic system is one where $\sigma(A) < 1$, so A^t does not explode for large t.

One feature of LTI systems that we can exploit to ensure training stability is that, for a given state-space parameterization A, C, there exist an infinite number of equivalent parameterizations that correspond to the same system and thus produce the same observation sequences. This can be explained intuitively: one can rotate the hidden state space by some transformation T, evolve the state in this transformed space before de-transforming observations with T^{-1}. In practice, this means we can consider any parameterization for A which has the eigenvalues of the true system if we learn a dense C matrix.

Because the stability of a linear dynamic system is characterized by the magnitude of the eigenvalues of A, we can choose T so it is easy to compute and restrict these values. If the true system $A^* = Q\Lambda Q^{-1}$ with a complex diagonal matrix of eigenvalues Λ, then we choose $T = Q$ implying that we learn $A = \Lambda$. This both decreases the number of learnable parameters in A while also making enforcing stability relatively trivial.

Jordan Normal Form

The primary issue with learning $A = \Lambda$ is that Λ as the eigenvalues of a real matrix will come in complex conjugate pairs[2]. However, Real Jordan Normal Form (JNF) [] offers a simple solution. By splitting the real and imaginary parts, we can construct an all-real matrix for which matrix multiplication simulates complex multiplication with this constraint. The following represents a 4×4 example:

$$A = \begin{bmatrix} \alpha_0 & \beta_0 & 0 & 0 \\ -\beta_0 & \alpha_0 & 0 & 0 \\ 0 & 0 & \alpha_1 & \beta_1 \\ 0 & 0 & -\beta_1 & \alpha_1 \end{bmatrix}$$

This form does have its drawbacks. In scenarios where any imaginary components are actually zero, then there should be an additional unique real component. Although somewhat inelegant, the negative impact of this scenario can be mitigated by simply increasing the dimensionality of A. Additionally if $\alpha_0 = \alpha_1$ and $\beta_0 = \beta_1$ true Jordan blocks should additionally have a one in the off-diagonal corresponding to eigenvalues with multiplicity greater than one. In practice this is not an issue as it is difficult to produce *exactly* identical eigenvalues.

---

[2]If we assumed that A^* was symmetric, the eigenvalues would have no imaginary components and we could instead simply learn a diagonal real matrix Λ.

Although there are many ways to ensure that the magnitude of each eigenvalue in the JNF does not exceed 1, we found the following reparameterization to be effective, using θ_α and θ_β as vectors of unconstrained real-valued parameters to produce the vectors of constrained real and imaginary components α and β:

$$\alpha = max((1 - \epsilon) - |\theta_\alpha|, 0)$$
$$\beta = max(1 - |\theta_\beta|, 0) * \sqrt{1 - \alpha^2}$$

where $\epsilon = 10^{-14}$. This particular transformation ensures that every unique real parameter pair $\theta_\alpha, \theta_\beta$ corresponds to a unique complex eigenvalue. The small epsilon subtraction ensures that we never compute $\sqrt{0}$. In our implementation, ϵ is chosen such that when we compute α and β with double precision, and then cast to single precision floating point we avoid $\sqrt{0}$ and allows $\alpha = 1$.

2.6 Addressing the Scale Ambiguity

When learning both the encoding function and the dynamic system parameters simultaneously, there is an ambiguity between the scale of the embedding and the scale of the dynamic system when the covariance is learned. As an illustrative example, consider the following system:

$$y_t = \gamma_t f_\theta(o_t) \qquad \hat{y}_t = \gamma_t C x_t$$

A scaling ambiguity occurs when we try to learn the covariance of the error in addition to the other parameters of our network, i.e when the predictive loss becomes:

$$\log p(y_t | y_{t-1}) \propto (y_t - \hat{y}_t)^T \Sigma (y_t - \hat{y}_t)$$
$$= (\gamma_t f_\theta(o_t) - \gamma_t C x_t)^T \Sigma (\gamma_t f_\theta(o_t) - \gamma C x_t)$$
$$= \gamma_t^2 (f_\theta(o_t) - C x_t)^T \Sigma (f_\theta(o_t) - C x_t)$$

The γ_t^2 term, which induces downward pressure on the embedding magnitudes when Σ is constant, can be absorbed as a learned Σ adjusts during training. This effectively removes its impact, but leaves behind the upward pressure on magnitudes from the $\log \gamma_t$ term, which will result in the system maximizing γ_t, rather than prediction error. In practice, this results in a runaway scale of the embeddings and numerical issues.

To address this we model the γ^{-1} as another layer in our invertible network which we simply adds another term to our loss function:

$$\mathcal{L} = \log p(\gamma_t^{-1} y_t | \cdot) + \log |\det \frac{\partial y_t}{\partial o_t}| - \log \gamma_t$$

In practice, we found that this adjustment improved training stability even when the covariance is held constant during training. Although γ_t could be learned, we used $\gamma_t = \frac{1}{N} \|y_t\|_1$. We also found that the L^1 norm performs better than the L^2 norm[3].

3 Experiments and Results

3.1 Datasets

We show results for both a synthetic and realistic dataset. The synthetic data, entitled Bouncing-MNIST, is generated using the Moving Symbols algorithm, a published benchmark designed to support the objective study of video prediction networks [Wang *et al.*, 2004; Szeto *et al.*, 2018]. Each video sequence samples an MNIST digit, assigns it an initial trajectory, and simulates elastic collisions with the image boundary.

The realistic sequences are sampled from UCF Sports Action [Rodriguez *et al.*, 2008; Soomro and Zamir, 2014]. This dataset contains video sequences of various sports such as diving, running, horseback riding, and golfing.

3.2 Network Topology

Our network is most similar to that used by [Kingma and Dhariwal, 2018], but without 1×1 convolutions, or the act-norm operation. We used 4 blocks of 10 affine-coupling layers each, where each block has an early connection out to the final embedding. Our non-invertible networks used at each step of flow were simple 3-layer networks of 3×3 convolution with two output channels for the affine transformation parameters and 512 channels in the center. For comparison, we implement the adversarial training algorithm of [Mathieu *et al.*, 2015], which is known for its sharp image quality.

3.3 Results

Method	First Frame		Fifth Frame	
	PSNR	SSIM	PSNR	SSIM
Invertible Linear Embedding	**23.5**	0.92	**17.4**	0.69
Adversarial Training	20.6	**0.95**	12.1	**0.83**
Last Input	17.3	0.76	14.5	0.67

Figure 4: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) scores, taking the mean over 100 held-out test sequences. We generate future frames $o_1, o_2, ..., o_5$ and calculate scores on o_1 and o_5 to measure both immediate and longer-horizon prediction quality. We again note that our approach does not explicitly minimize the mean squared error between predicted frames and ground truth.

We evaluate our algorithm by comparing against adversarial training in three ways: qualitatively through examples, with peak signal-to-noise ratio (PSNR), and with the structural similarity (SSIM) index [Wang *et al.*, 2004]. Statistical results are reported on the synthetic dataset.

Although adversarial training has a slight advantage in SSIM, the ILE algorithm outperforms it in PSNR. The difference is especially pronounced over a longer time horizon. Adversarial training maintains crisp shapes, yet lacks accurate motion projections over even moderate time horizons. After five frames it performs significantly worse than the naive baseline. ILE maintains a reasonable representation of the digit shape, and excels at motion projection over a long time horizon, even accurately predicting bounces off image boundaries. This suggests that the nonlinear dynamic system is being fit quite well.[4]

---

[3]Presumably because it better propagates small gradients in each dimension of y_t.

[4]The adversarial training PSNR scores are lower than those reported in [Mathieu *et al.*, 2015] because the synthetic dataset has much more motion than the UCF-101 dataset, which the original paper used as a benchmark. In our tests, digit velocity is up to 3 pixels/frame in each direction. However, the high velocity is in-

Figure 2: The Bouncing-MNIST dataset, modeling elastic collisions which preserve object shape.

Figure 3: The UCF Sports Action dataset, modeling the progression of a golf swing.

While adversarial training performs well on sequences where the motion is strictly linear, such as those pictured, it performs poorly in motion that is nonlinear in pixel space. For example when the digit bounces off a wall or when a golf club accelerates in the frame. In contrast, ILE models all motion sequences well, suggesting better generalization ability.

4 Directions for Future Work

Our work moves toward exact maximum likelihood optimization to improve performance in video prediction. We present here what we consider to be natural next steps, and the implications they might have.

Action-conditional and time-variant models. By extending the model of the hidden dynamical system to include an action u and linear mapping B it becomes possible to use ILE for model based reinforcement learning and optimal control. Additionally a simple extension to our model which may prove promising is to learn a time or state-conditional state-transition matrix A_t in lieu of the constant A presented. This particular extension could be done using any standard autoencoding neural network architecture as a JNF state transition matrix is diagonal and invertibility is not a requirement. Although time-varying linear dynamic systems are more difficult to analyze, they are models of much greater capacity and therefore could be better suited for difficult problems that would require infinite, or near-infinite dimensional state dimensions.

_____
tentional; a quality benchmark for video prediction should use sequences where motion is noticeable.

Scaling to larger frame dimensions. Invertible networks, perhaps as a direct result of the difficult task of modeling the entire unknown distribution over video frames, are large and difficult to train. In particular, memory usage in our model even for these relatively small frame sequences was a computational constraint. Architectural improvements such as those recently proposed by Grathwohl *et al.* could extend our results into images approaching modern video resolutions.

5 Conclusion

We have presented the invertible linear embedding, which provides exact maximum likelihood learning of video sequences. Our key contribution is to combine invertible networks with linear dynamical systems. While images sequences may lie on a complex probability manifold in high-dimensional space, an invertible network coupled with a change of variables learns how to properly map that manifold of probability to the well-behaved conditional Gaussian created by a linear dynamic system. By formulating this with a single learning objective, we arrive at an elegant joint optimization problem. The primary advantage of this approach is that we avoid making any assumptions about the distribution of the input domain.

In future work we believe even better qualitative performance can be had as more becomes known about optimization and training of invertible networks.

References

[Banijamali *et al.*, 2017] Ershad Banijamali, Rui Shu, Mohammad Ghavamzadeh, Hung Bui, and Ali Ghodsi. Ro-

bust locally-linear controllable embedding. *arXiv preprint arXiv:1710.05373*, 2017.

[Bengio *et al.*, 2013] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[Brunton *et al.*, 2016] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[Chiappa *et al.*, 2017] Silvia Chiappa, Sébastien Racaniere, Daan Wierstra, and Shakir Mohamed. Recurrent environment simulators. *arXiv preprint arXiv:1704.02254*, 2017.

[Dinh *et al.*, 2014] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[Dinh *et al.*, 2016] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[Grathwohl *et al.*, 2018] Will Grathwohl, Ricky TQ Chen, Jesse Betterncourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

[Janczak, 2004] Andrzej Janczak. *Identification of nonlinear systems using neural networks and polynomial models: a block-oriented approach*, volume 310. Springer Science & Business Media, 2004.

[Kingma and Dhariwal, 2018] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10236–10245, 2018.

[Koopman, 1931] Bernard O Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

[Lee *et al.*, 2018] Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *CoRR*, abs/1804.01523, 2018.

[Lotter *et al.*, 2016] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[Lusch *et al.*, 2018] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[Mathieu *et al.*, 2015] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

[Oh *et al.*, 2015] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, pages 2863–2871, 2015.

[Reda *et al.*, 2018] Fitsum A Reda, Guilin Liu, Kevin J Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. Sdc-net: Video prediction using spatially-displaced convolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 718–733, 2018.

[Rodriguez *et al.*, 2008] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. 2008.

[Soomro and Zamir, 2014] Khurram Soomro and Amir R Zamir. Action recognition in realistic sports videos. In *Computer vision in sports*, pages 181–208. Springer, 2014.

[Szeto *et al.*, 2018] Ryan Szeto, Simon Stent, German Ros, and Jason J. Corso. A dataset to evaluate the representations learned by video prediction models. In *International Conference on Learning Representations (Workshop Track)*, Apr 2018.

[Thrun *et al.*, 2005] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[Wang *et al.*, 2004] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[Watter *et al.*, 2015] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pages 2746–2754, 2015.

[Welch *et al.*, 1995] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995.