Deep Kalman Filters

Rahul G. Krishnan   **Uri Shalit**   **David Sontag**
Courant Institute of Mathematical Sciences
New York University

Abstract

Kalman Filters are one of the most influential models of time-varying phenomena. They admit an intuitive probabilistic interpretation, have a simple functional form, and enjoy widespread adoption in a variety of disciplines. Motivated by recent variational methods for learning deep generative models, we introduce a unified algorithm to efficiently learn a broad spectrum of Kalman filters. Of particular interest is the use of temporal generative models for counterfactual inference. We investigate the efficacy of such models for counterfactual inference, and to that end we introduce the "Healing MNIST" dataset where long-term structure, noise and actions are applied to sequences of digits. We show the efficacy of our method for modeling this dataset. We further show how our model can be used for counterfactual inference for patients, based on electronic health record data of 8,000 patients over 4.5 years.

1   Introduction

The compilation of Electronic Health Records (EHRs) is now the norm across hospitals in the United States. A patient record may be viewed as a sequence of diagnoses, surgeries, laboratory values and drugs prescribed over time. The wide availability of these records now allows us to apply machine learning techniques to answer medical questions: What is the best course of treatment for a patient? Between two drugs, can we determine which will save a patient? Can we find patients who are "similar" to each other? Our paper introduces new techniques for learning *causal* generative temporal models from noisy high-dimensional data, that we believe is the first step towards addressing these questions.

We seek to model the change of the patient's state over time. We do this by learning a representation of the patient that (1) evolves over time and (2) is sensitive to the effect of the actions taken by doctors. In particular, the approach we adopt is to learn a time-varying, generative model of patients.

Modelling temporal data is a well studied problem in machine learning. Models such as the Hidden Markov Models (HMM), Dynamic Bayesian Networks (DBN), and Recurrent Neural Networks (RNN) have been proposed to model the probability of sequences. Here, we consider a widely used probabilistic model: Kalman filters (Kalman, 1960). Classical Kalman filters are linear dynamical systems, that have enjoyed remarkable success in the last few decades. From their use in GPS to weather systems and speech recognition models, few other generative models of sequential data have enjoyed such widespread adoption across many domains.

In classical Kalman filters, the latent state evolution as well as the emission distribution and action effects are modelled as linear functions perturbed by Gaussian noise. However, for real world applications the use of linear transition and emission distribution limits the capacity to model complex phenomena, and modifications to the functional form of Kalman filters have been proposed. For example, the Extended Kalman Filter (Jazwinski, 2007) and the Unscented Kalman Filter (Wan *et al.* , 2000) are two different methods to learn temporal models with non-linear transition and emission distributions (see also Roweis & Ghahramani (2000) and Haykin (2004)). The addition

of non-linearities to the model makes learning more difficult. Raiko & Tornio (2009) explored ways of using linear approximations and non-linear dynamical factor analysis in order to overcome these difficulties. However, their methods do not handle long-range temporal interactions and scale quadratically with the latent dimension.

We show that recently developed techniques in variational inference (Rezende *et al.* , 2014; Kingma & Welling, 2013) can be adopted to learn a broad class of the Kalman filters that exist in the literature using a single algorithm. Furthermore, using deep neural networks, we can enhance Kalman filters with arbitrarily complex transition dynamics and emission distributions. We show that we can tractably learn such models by optimizing a bound on the likelihood of the data.

Kalman filters have been used extensively for optimal control, where the model attempts to capture how actions affect the observations, precipitating the task of choosing the best control signal towards a given objective. We use Kalman filters for a different yet closely related task: performing counterfactual inference. In the medical setting, counterfactual inference attempts to model the effect of an intervention such as a surgery or a drug, on an outcome, e.g. whether the patient survived. The hardness of this problem lies in the fact that typically, for a single patient, we only see one intervention-outcome pair (the patient cannot have taken and not taken the drug). The key point here is that by modelling the sequence of observations such as diagnoses and lab reports, as well as the interventions or actions (in the form of surgeries and drugs administered) across patients, we hope to learn the effect of interventions on a patient's future state.

We evaluate our model in two settings. First we introduce "Healing MNIST", a dataset of perturbed, noisy and rotated MNIST digits. We show our model captures both short- and long-range effects of actions performed on these digits. Second, we use EHR data from $8,000$ diabetic and pre-diabetic patients gathered over 4.5 years. We investigate various kinds of Kalman filters learned using our framework and use our model to learn the effect anti-diabetic medication has on patients.

The contributions of this paper are as follows:

- Develop a method for probabilistic generative modelling of sequences of complex observations, perturbed by non-linear actions, using deep neural nets as a building block. We derive a bound on the log-likelihood of sequential data and an algorithm to learn a broad class of Kalman filters.
- We evaluate the efficacy of different recognition distributions for inference and learning.
- We consider this model for use in counterfactual inference with emphasis on the medical setting. To the best of our knowledge, the use of continuous state space models has not been considered for this goal. On a synthetic setting we empirically validate that our model is able to capture patterns within a very noisy setting and model the effect of external actions. On real patient data we show that our model can successfully perform counterfactual inference to show the effect of anti-diabetic drugs on diabetic patients.

2 Background

Kalman Filters Assume we have a sequence of unobserved variables $z_1, \ldots, z_T \in \mathbb{R}^s$. For each unobserved variable z_t we have a corresponding *observation* $x_t \in \mathbb{R}^d$, and a corresponding *action* $u_t \in \mathbb{R}^c$, which is also observed. In the medical domain, the variables z_t might denote the true state of a patient, the observations x_t indicate known diagnoses and lab test results, and the actions u_t correspond to prescribed medications and medical procedures which aim to change the state of the patient. The classical Kalman filter models the observed sequence $x_1, \ldots x_T$ as follows:

$$z_t = G_t z_{t-1} + B_t u_{t-1} + \epsilon_t \text{ (action-transition)} , \qquad x_t = F_t z_t + \eta_t \text{ (observation)},$$

where $\epsilon_t \sim \mathcal{N}(0, \Sigma_t)$, $\eta_t \sim \mathcal{N}(0, \Gamma_t)$ are zero-mean i.i.d. normal random variables, with covariance matrices which may vary with t. This model assumes that the latent space evolves linearly, transformed at time t by the state-transition matrix $G_t \in \mathbb{R}^{s \times s}$. The effect of the control signal u_t is an additive linear transformation of the latent state obtained by adding the vector $B_t u_{t-1}$, where $B_t \in \mathbb{R}^{s \times c}$ is known as the *control-input model*. Finally, the observations are generated linearly from the latent state via the observation matrix $F_t \in \mathbb{R}^{d \times s}$.

In the following sections, we show how to replace all the linear transformations with non-linear transformations parameterized by neural nets. The upshot is that the non-linearity makes learning

much more challenging, as the posterior distribution $p(z_1, \ldots z_T | x_1, \ldots, x_T, u_1, \ldots, u_T)$ becomes intractable to compute.

Stochastic Backpropagation In order to overcome the intractability of posterior inference, we make use of recently introduced variational autoencoders (Rezende *et al.* , 2014; Kingma & Welling, 2013) to optimize a variational lower bound on the model log-likelihood. The key technical innovation is the introduction of a *recognition network*, a neural network which approximates the intractable posterior.

Let $p(x, z) = p_0(z)p_\theta(x|z)$ be a generative model for the set of observations x, where $p_0(z)$ is the prior on z and $p_\theta(x|z)$ is a generative model parameterized by θ. In a model such as the one we posit, the posterior distribution $p_\theta(z|x)$ is typically intractable. Using the well-known variational principle, we posit an approximate posterior distribution $q_\phi(z|x)$, also called a *recognition model* - see Figure 1a. We then obtain the following lower bound on the marginal likelihood:

$$
\begin{aligned}
\log p_\theta(x) &= \log \int_z \frac{q_\phi(z|x)}{q_\phi(z|x)} p_\theta(x|z)p_0(z)\mathrm{d}z \geq \int_z q_\phi(z|x) \log \frac{p_\theta(x|z)p_0(z)}{q_\phi(z|x)}\mathrm{d}z \\
&= \mathop{\mathbb{E}}_{q_\phi(z|x)} \left[\log p_\theta(x|z) \right] - \mathrm{KL}(\, q_\phi(z|x) || p_0(z)\,) = \mathcal{L}(x; (\theta, \phi)),
\end{aligned}
\tag{1}
$$

where the inequality is by Jensen's inequality. Variational autoencoders aim to maximize the lower bound using a parametric model q_ϕ conditioned on the input. Specifically, Rezende *et al.* (2014); Kingma & Welling (2013) both suggest using a neural net to parameterize q_ϕ, such that ϕ are the parameters of the neural net. The challenge in the resulting optimization problem is that the lower bound (1) includes an expectation w.r.t. q_ϕ, which implicitly depends on the network parameters ϕ. This difficulty is overcome by using *stochastic backpropagation*: assuming that the latent state is normally distributed $q_\phi(z|x) \sim \mathcal{N}\left(\mu_\phi(x), \Sigma_\phi(x)\right)$, a simple transformation allows one to obtain Monte Carlo estimates of the gradients of $\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]$ with respect to ϕ. The KL term in (1) can be estimated similarly since it is also an expectation. If we assume that the prior $p_0(z)$ is also normally distributed, the KL and its gradients may be obtained analytically.

Counterfactual Estimation Counterfactual estimation is the task of inferring the probability of a result given different circumstances than those empirically observed. For example, in the medical setting, one is often interested in questions such as "What would the patient's blood sugar level be had she taken a different medication?". Knowing the answers to such questions could lead to better and more efficient healthcare. We are interested in providing better answers to this type of questions, by leveraging the power of large-scale Electronic Health Records.

Pearl (2009) framed the problem of counterfactual estimation in the language of graphical models and *do*-calculus. If one knows the graphical model of the variables in question, then for some structures estimation of counterfactuals is possible by setting a variable of interest (e.g. medication prescribed) to a given value and performing inference on a derived sub-graph. In this work, we do not seek to learn the true underlying causal graph structure but rather seek to use *do*-calculus to observe the effect of interventions under a causal interpretation of the model we posit.

3 Related Work

The literature on sequential modeling and Kalman filters is vast and here we review some of the relevant work on the topic with particular emphasis on recent work in machine learning. We point the reader to Haykin (2004) for a summary of some approaches to learn Kalman filters.

Mirowski & LeCun (2009) model sequences using dynamic factor graphs with an EM-like procedure for energy minimization. Srivastava *et al.* (2015) consider unsupervised learning of video representations with LSTMs. They encode a sequence in a fixed length hidden representation of an LSTM-RNN and reconstruct the subsequent sequence based on this representation. Gregor *et al.* (2015) consider a temporal extension to variational autoencoders where independent latent variables perturb the hidden state of an RNN across time.

Langford *et al.* (2009) adopt a different approach to learn nonlinear dynamical systems using black-box classifiers. Their method relies on learning three sets of classifiers. The first is trained to construct a compact representation s_t to predict the x_{t+1} from x_t, the second uses s_{t-1} and x_{t-1} to

predict s_t. The third trains classifiers to use $s_{<t}$ to predict s_t and consequently x_t. In essence, the latent space s_t is constructed using these classifiers.

Gan *et al.* (2015) similarly learn a generative model by maximizing a lower bound on the likelihood of sequential data but do so in a model with discrete random variables.

Bayer & Osendorfer (2014) create a stochastic variant of Recurrent Neural Networks (RNNs) by making the hidden state of the RNN a function of stochastically sampled latent variables at every time step. Chung *et al.* (2015) model sequences of length T using T variational autoencoders. They use a single RNN that (1) shares parameters in the inference and generative network and (2) models the parameters of the prior and approximation to the posterior at time $t \in [1, \ldots T]$ as a deterministic function of the hidden state of the RNN. There are a few key differences between their work and ours. First, they do not model the effect of external actions on the data, and second, their choice of model ties together inference and sampling from the model whereas we consider decoupled generative and recognition networks. Finally, the time varying "memory" of their resulting generative model is both deterministic and stochastic whereas ours is entirely stochastic. i.e our model retains the Markov Property and other conditional independence statements held by Kalman filters.

Learning Kalman filters with Multi-Layer Perceptrons was considered by Raiko & Tornio (2009). They approximate the posterior using non-linear dynamic factor analysis (Valpola & Karhunen, 2002), which scales quadratically with the latent dimension. Recently, Watter *et al.* (2015) use temporal generative models for optimal control. While Watter *et al.* aim to learn a locally linear latent dimension within which to perform optimal control, our goal is different: we wish to model the data in order to perform counterfactual inference. Their training algorithm relies on approximating the bound on the likelihood by training on consecutive pairs of observations.

In general, control applications deal with domains where the effect of action is instantaneous, unlike in the medical setting. In addition, most control scenarios involve a setting such as controlling a robot arm where the control signal has a major effect on the observation; we contrast this with the medical setting where medication can often have a weak impact on the patient's state, compared with endogenous and environmental factors.

For a general introduction to estimating expected counterfactual effects over a population - see Morgan & Winship (2014); Höfler (2005); Rosenbaum (2002). For insightful work on counterfactual inference, in the context of a complex machine-learning and ad-placement system, see Bottou *et al.* (2013).

Recently, Velez (2013) use a partially observable Markov process for modeling diabetic patients over time, finding that the latent state corresponds to relevant lab test levels (specifically, A1c levels).

4 Model

Our goal is to fit a generative model to a sequence of observations and actions, motivated by the nature of patient health record data. We assume that the observations come from a latent state which evolves over time. We assume the observations are a noisy, non-linear function of this latent state. Finally, we also assume that we can observe actions, which affect the latent state in a possibly non-linear manner.

Denote the sequence of observations $\vec{x} = (x_1, \ldots, x_T)$ and actions $\vec{u} = (u_1, \ldots, u_{T-1})$, with corresponding latent states $\vec{z} = (z_1, \ldots, z_T)$. As previously, we assume that $x_t \in \mathbb{R}^d$, $u_t \in \mathbb{R}^c$, and $z_t \in \mathbb{R}^s$. The generative model for the deep Kalman filter is then given by:

$$
\begin{aligned}
z_1 &\sim \mathcal{N}(\mu_0; \Sigma_0) \\
z_t &\sim \mathcal{N}(G_\alpha(z_{t-1}, u_{t-1}, \Delta_t), S_\beta(z_{t-1}, u_{t-1}, \Delta_t)) \\
x_t &\sim \Pi(F_\kappa(z_t)).
\end{aligned}
\tag{2}
$$

That is, we assume that the distribution of the latent states is Normal, with a mean and covariance which are nonlinear functions of the previous latent state, the previous actions, and the time different

Δ_t between time $t-1$ and time t [1]. The observations x_t are distributed according to a distribution Π (e.g. a Bernoulli distribution if the data is binary) whose parameters are a function of the corresponding latent state z_t. Specifically, the functions $G_\alpha, S_\beta, F_\kappa$ are assumed to be parameterized by deep neural networks. We set $\mu_0 = 0, \Sigma_0 = I_d$, and therefore we have that $\theta = \{\alpha, \beta, \kappa\}$ are the parameters of the generative model. We use a diagonal covariance matrix $S_\beta(\cdot)$, and employ a log-parameterization, thus ensuring that the covariance matrix is positive-definite. The model is presented in Figure 1b, along with the recognition model q_ϕ which we outline in Section 5.

The key point here is that Eq. (2) subsumes a large family of linear and non-linear latent space models. By restricting the functional forms of $G_\alpha, S_\beta, F_\kappa$, we can train different kinds of Kalman filters within the framework we propose. For example, by setting $G_\alpha(z_{t-1}, u_{t-1}) = G_t z_{t-1} + B_t u_{t-1}, S_\beta = \Sigma_t, F_\kappa = F_t z_t$ where G_t, B_t, Σ_t, F_t are matrices, we obtain classical Kalman filters. In the past, modifications to the Kalman filter typically introduced a new learning algorithm and heuristics to approximate the posterior more accurately. In contrast, within the framework we propose any parametric differentiable function can be substituted in for one of $G_\alpha, S_\beta, F_\kappa$. Learning any such model can be done using backpropagation as will be detailed in the next section.
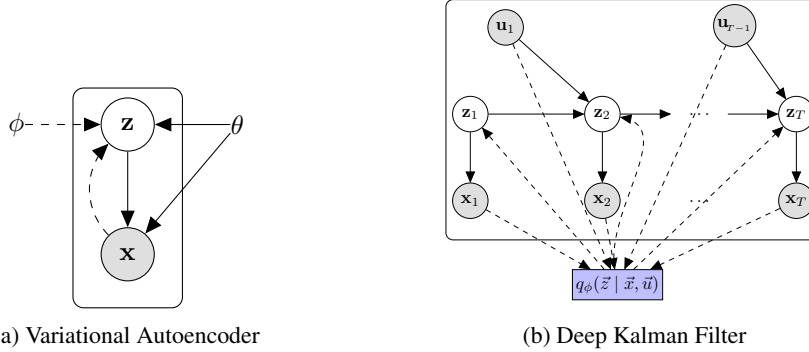

(a) Variational Autoencoder   (b) Deep Kalman Filter

Figure 1: (a): Learning static generative models. Solid lines denote the generative model $p_0(z)p_\theta(x|z)$, dashed lines denote the variational approximation $q_\phi(z|x)$ to the intractable posterior $p(z|x)$. The variational parameters ϕ are learned jointly with the generative model parameters θ. (b): Learning in a Deep Kalman Filter. A parametric approximation to $p_\theta(\vec{z}|\vec{x})$, denoted $q_\phi(\vec{z}|\vec{x}, \vec{u})$, is used to perform inference during learning.

5 Learning using Stochastic Backpropagation

5.1 Maximizing a Lower Bound

We aim to fit the generative model parameters θ which maximize the conditional likelihood of the data given the external actions, i.e we desire $\max_\theta \log p_\theta(x_1 \ldots, x_T | u_1 \ldots u_{T-1})$. Using the variational principle, we apply the lower bound on the log-likelihood of the observations $\vec{x}$ derived in Eq. (1). We consider the extension of the Eq. (1) to the temporal setting where we use the following factorization of the prior:

$$q_\phi(\vec{z}|\vec{x}, \vec{u}) = \prod_{t=1}^{T} q(z_t | z_{t-1}, x_t, \ldots, x_T, \vec{u}) \tag{3}$$

We motivate this structured factorization of q_ϕ in Section 5.2. We condition the variational approximation not just on the inputs $\vec{x}$ but also on the actions $\vec{u}$.

Our goal is to derive a lower bound to the conditional log-likelihood in a form that will factorize easily and make learning more amenable. The lower bound in Eq. (1) has an analytic form of the KL term only for the simplest of transition models G_α, S_β. Resorting to sampling for estimating the gradient of the KL term results in very high variance. Below we show another way to factorize the KL term which results in more stable gradients, by using the Markov property of our model.

---

[1]More precisely, this is a *semi-Markov* model, and we assume that the time intervals are modelled separately. In our experiments we consider homogeneous time intervals.

Algorithm 1 Learning Deep Kalman Filters

---

while $notConverged()$ **do**
  $\vec{x} \leftarrow sampleMiniBatch()$
  Perform inference and estimate likelihood:
  1. $\hat{z} \sim q_\phi(\vec{z}|\vec{x}, \vec{u})$
  2. $\hat{x} \sim p_\theta(\vec{x}|\hat{z})$
  3. Compute $\nabla_\theta \mathcal{L}$ and $\nabla_\phi \mathcal{L}$ (Differentiating (5))
  4. Update θ, ϕ using ADAM
end while

We have for the conditional log-likelihood (see Supplemental Section A for a more detailed derivation):

$$\log p_\theta(\vec{x}|\vec{u}) \geq \int_{\vec{z}} q_\phi(\vec{z}|\vec{x}, \vec{u}) \log \frac{p_0(\vec{z}|\vec{u}) p_\theta(\vec{x}|\vec{z}, \vec{u})}{q_\phi(\vec{z}|\vec{x}, \vec{u})} d\vec{z}$$

$$= \mathop{\mathbb{E}}_{q_\phi(\vec{z}|\vec{x}, \vec{u})} [\log p_\theta(\vec{x}|\vec{z}, \vec{u})] - \mathrm{KL}(q_\phi(\vec{z}|\vec{x}, \vec{u}) || p_0(\vec{z}|\vec{u}))$$

$$\overset{(using\ x_t \perp\!\!\!\perp x_{\neg t}|\vec{z})}{=} \sum_{t=1}^{T} \mathop{\mathbb{E}}_{z_t \sim q_\phi(z_t|\vec{x}, \vec{u})} [\log p_\theta(x_t|z_t, u_{t-1})] - \mathrm{KL}(q_\phi(\vec{z}|\vec{x}, \vec{u}) || p_0(\vec{z}|\vec{u})) = \mathcal{L}(x; (\theta, \phi)).$$

The KL divergence can be factorized as:

$$\mathrm{KL}(q_\phi(\vec{z}|\vec{x}, \vec{u}) || p_0(\vec{z})) \tag{4}$$

$$= \int_{z_1} \ldots \int_{z_T} q_\phi(z_1|\vec{x}, \vec{u}) \ldots q_\phi(z_T|z_{T-1}, \vec{x}, \vec{u}) \log \frac{p_0(z_1, \cdots, z_T)}{q_\phi(z_1|\vec{x}, \vec{u}) \ldots q_\phi(z_T|z_{T-1}, \vec{x}, \vec{u})} \mathrm{d}\vec{z}$$

$$\overset{(Factorization\ of\ p(\vec{z}))}{=} \mathrm{KL}(q_\phi(z_1|\vec{x}, \vec{u}) || p_0(z_1))$$

$$+ \sum_{t=2}^{T} \mathop{\mathbb{E}}_{z_{t-1} \sim q_\phi(z_{t-1}|\vec{x}, \vec{u})} [\mathrm{KL}(q_\phi(z_t|z_{t-1}, \vec{x}, \vec{u}) || p_0(z_t|z_{t-1}, u_{t-1}))].$$

This yields:

$$\log p_\theta(\vec{x}|\vec{u}) \geq \mathcal{L}(x; (\theta, \phi)) =$$

$$\sum_{t=1}^{T} \mathop{\mathbb{E}}_{q_\phi(z_t|\vec{x}, \vec{u})} [\log p_\theta(x_t|z_t)] - \mathrm{KL}(q_\phi(z_1|\vec{x}, \vec{u}) || p_0(z_1))$$

$$- \sum_{t=2}^{T} \mathop{\mathbb{E}}_{q_\phi(z_{t-1}|\vec{x}, \vec{u})} [\mathrm{KL}(q_\phi(z_t|z_{t-1}, \vec{x}, \vec{u}) || p_0(z_t|z_{t-1}, u_{t-1}))]. \tag{5}$$

Equation (5) is differentiable in the parameters of the model (θ, ϕ), and we can apply backpropagation for updating θ, and stochastic backpropagation for estimating the gradient w.r.t. ϕ of the expectation terms w.r.t. $q_\phi(z_t)$. Algorithm 1 depicts the learning algorithm. It can be viewed as a four stage process. The first stage is inference of $\vec{z}$ from an input $\vec{x}, \vec{u}$ by the recognition network q_ϕ. The second stage is having the generative model p_θ reconstruct the input using the current estimates of the posterior. The third stage involves estimating gradients of the likelihood with respect to θ and ϕ, and the fourth stage involves updating parameters of the model. Gradients are typically averaged across stochastically sampled mini-batches of the training set.

5.2 On the choice of the Optimal Variational Model

For time varying data, there exist many choices for the recognition network. We consider four variational models of increasing complexity. Each model conditions on a different subset of the observations through the use of Multi-Layer Perceptrons (MLP) and Recurrent Neural Nets (RNN) (As implemented in Zaremba & Sutskever (2014)):

- **q-INDEP**: $q(z_t|x_t, u_t)$ parameterized by an MLP

- **q-LR**: $q(z_t|x_{t-1}, x_t, x_{t+1}, u_{t-1}, u_t, u_{t+1})$ parameterized by an MLP

- **q-RNN**: $q(z_t|x_1, \ldots, x_t, u_1, \ldots u_t)$ parameterized by a RNN

- **q-BRNN**: $q(z_t|x_1, \ldots, x_T, u_1, \ldots, u_T)$ parameterized by a bi-directional RNN

In the experimental section we compare the performance of these four models on a challenging sequence reconstruction task.

An interesting question is whether the Markov properties of the model can enable better design of approximations to the posterior.

Theorem 5.1. *For the graphical model depicted in Figure 1b, the posterior factorizes as:*

$$p(\vec{z}|\vec{x}, \vec{u}) = p(z_1|\vec{x}, \vec{u}) \prod_{t=2}^{T} p(z_t|z_{t-1}, x_t, \ldots, x_T, u_{t-1}, \ldots, u_{T-1})$$

Proof. We use the independence statements implied by the generative model in Figure 1b to note that $p(\vec{z}|\vec{x}, \vec{u})$, the true posterior, factorizes as:

$$p(\vec{z}|\vec{x}, \vec{u}) = p(z_1|\vec{x}, \vec{u}) \prod_{t=2}^{T} p(z_t|z_{t-1}, \vec{x}, \vec{u})$$

Now, we notice that $z_t \perp\!\!\!\perp x_1, \ldots, x_{t-1}|z_{t-1}$ and $z_t \perp\!\!\!\perp u_1 \ldots, u_{t-2}|z_{t-1}$, yielding:

$$p(\vec{z}|\vec{x}, \vec{u}) = p(z_1|\vec{x}, \vec{u}) \prod_{t=2}^{T} p(z_t|z_{t-1}, x_t, \ldots, x_T, u_{t-1}, \ldots, u_{T-1})$$

$\square$

The significance of Theorem 5.1 is twofold. First, it tells us how we can use the Markov structure of our graphical model to simplify the posterior that any $q_\phi(\vec{z})$ must approximate. Second, it yields insight on how to design approximations to the true posterior. Indeed this motivated the factorization of q_ϕ in Eq. 3. Furthermore, instead of using a bi-directional RNN to approximate $p(z_t|\vec{x}, \vec{u})$ by summarizing both the past and the future $(x_1, \ldots, x_T)$, one can approximate the same posterior distribution using a single RNN that summarizes the future $(x_t, \ldots, x_T)$ as long as one also conditions on the previous latent state (z_{t-1}). Here, z_{t-1} serves as a summary of $x_1, \ldots, x_{t-1}$.

For the stochastic backpropagation model, the variational lower bound is tight if and only if $\mathrm{KL}(q_\phi(z|x)||p_\theta(z|x)) = 0$. In that case, we have that $\mathcal{L}(x; (\theta, \phi)) = \log p_\theta(x)$, and the optimization objective (5) reduces to a maximum likelihood objective. In the stochastic backpropagation literature, the variational distribution $q_\phi(z|x)$ is usually Gaussian and therefore cannot be expected to be equal to $p_\theta(z|x)$. An interesting question is whether using the idea of the universality of normalizing flows (Tabak *et al.*, 2010; Rezende & Mohamed, 2015) one can transform $q_\phi(z|x)$ to be equal (or arbitrarily close) to $p_\theta(z|x)$ and thus attain equality in the lower bound. Such a result leads to a consistency result for the learned model, stemming from the consistency of maximum likelihood.

5.3 Counterfactual Inference

Having learned a generative temporal model, we can use the model to perform counterfactual inference. Formally, consider a scenario where we are interested in evaluating the effect of an intervention at time t. We can perform inference on the set of observations: $\{x_1, \ldots, x_t, u_1, \ldots, u_{t-1}\}$ using the learned q_ϕ. This gives us an estimate z_t. At this point, we can apply u_t (the action intended for the patient) as well as $\tilde{u}_t$ (the action to be contrasted against). We can forward sample from this latent state in order to contrast the expected effect of different actions.

6 Experimental Section

We implement and train models in Torch (Collobert *et al.* , 2011) using ADAM (Kingma & Ba, 2014) with a learning rate of 0.001 to perform gradient ascent. Our code is implemented to parameterize $\log S_\beta$ during learning. We use a two-layer Long-Short Term Memory Recurrent Neural Net (LSTM-RNN, Zaremba & Sutskever (2014)) for sequential variational models. We regularize models during training (1) using dropout (Srivastava *et al.* , 2014) with a noise of 0.1 to the input of the recognition model (2) through the addition of small random uniform noise (on the order of a tenth of the maximal value) to the actions.

Comparing recognition models We experiment with four choices of variational models of increasing complexity: **q-INDEP** where $q(z_t|x_t)$ is parameterized by an MLP, **q-LR** where $q(z_t|x_{t-1}, x_t, x_{t+1})$ is parameterized by an MLP, **q-RNN** where $q(z_t|x_1, \ldots, x_t)$ is parameterized by an RNN, and **q-BRNN** where $q(z_t|x_1, \ldots, x_T)$ is parameterized by a bi-directional RNN.

6.1 Healing MNIST

Healthcare data exhibits diverse structural properties. Surgeries and drugs vary in their effect as a function of patient age, gender, ethnicity and comorbidities. Laboratory measurements are often noisy, and diagnoses may be tentative, redundant or delayed. In insurance health claims data, the situation is further complicated by arcane, institutional specific practices that determine how decisions made by healthcare professions are repurposed into codes used for reimbursements.

To mimic learning under such harsh conditions, we consider a synthetic dataset derived from the MNIST Handwritten Digits (LeCun & Cortes, 2010). We select several digits and create a synthetic dataset where rotations are performed to the digits. The rotations are encoded as the actions ($\vec{u}$) and the rotated images as the observations ($\vec{x}$). This realizes a sequence of rotated images. To each such generated training sequence, exactly one sequence of three consecutive squares is superimposed with the top-left corner of the images in a random starting location, and add up to 20% bit-flip noise. We consider two experiments: **Small Healing MNIST**, using a single example of the digit 1 and digit 5, and **Large Healing MNIST** where 100 different digits (one hundred 5's and one hundred 1's) are used. The training set comprises approximately 40000 sequences of length five for **Small Healing MNIST**, and 140000 sequences of length five for **Large Healing MNIST**. The large dataset represents the temporal evolution of two distinct subpopulations of patients (of size 100 each). The squares within the sequences are intended to be analogous to seasonal flu or other ailments that a patient could exhibit that are independent of the actions and which last several timesteps.

The challenges present within this dataset are numerous. (1) Image data is intrinsically high dimensional and much recent work has focused on learning patterns from it. It represents a setting where the posterior is complex and often requires highly non-linear models in order to approximate it. The additions of rotated images to the training data adds more complexity to the task. (2) In order to learn through random noise that is this high, one needs to have a model of sequences capable of performing "filtering". Models that rely on predicting the next image based on the previous one (Goroshin *et al.* , 2015; Memisevic, 2013) may not suffice to learn the structure of digits in the presence of large noise and rotation. Furthermore, long-range patterns - e.g. the three consecutive blocks in the upper-left corner - that exist in the data are beyond the scope of such models.

We learn models using the four recognition models described in Section 5. Figure 2a shows examples of training sequences (marked **TS**) from **Large Healing MNIST** provided to the model, and their corresponding reconstructions (marked **R**). The reconstructions are performed by feeding the input sequence into the learned recognition network, and then sampling from the resulting posterior distribution. Recall that the model posits $\vec{x}$ drawn from an independent Bernoulli distribution whose mean parameters (denoted mean probabilities) we visualize. We discuss results in more detail below.

Small Healing MNIST: Comparing Recognition Models

We evaluated the impact of different variational models on learning by examining test log-likelihood and by visualizing the samples generated by the models. Since **q-RNN** and **q-BRNN** have more parameters by virtue of their internal structure, we added layers to the **q-INDEP** network (6 layers) and **q-LR** network (4 layers) until training on the dataset was infeasible - i.e. did not make any

Small Rotation Right

Small Rotation Left

Large Rotation Right

Large Rotation Left

No Rotation

(a) Reconstruction during training (b) Samples: Different rotations (c) Inference on unseen digits
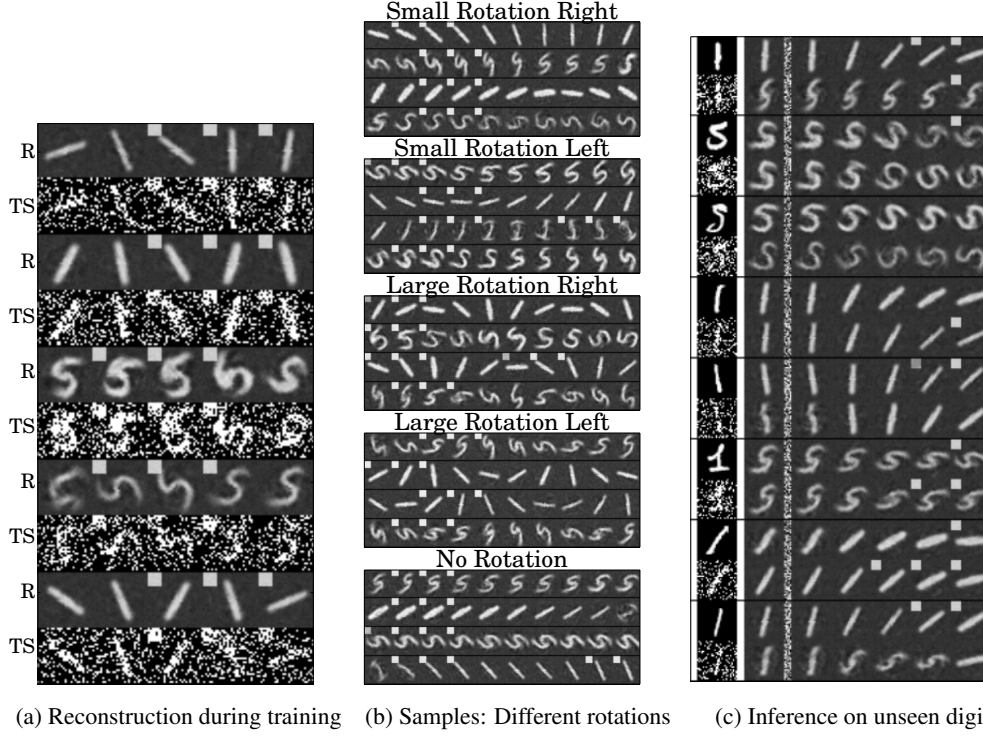
Figure 2: **Large Healing MNIST**. (a) Pairs of Training Sequences (TS) and Mean Probabilities of Reconstructions (R) shown above. (b) Mean probabilities sampled from the model under different, constant rotations. (c) Counterfactual Reasoning. We reconstruct variants of the digits $5, 1$ *not* present in the training set, with (bottom) and without (top) bit-flip noise. We infer a sequence of 1 timestep and display the reconstructions from the posterior. We then keep the latent state and perform forward sampling and reconstruction from the generative model under a constant right rotation.

q-BRNN

q-RNN

q-LR

q-INDEP

(a) Samples from models trained with different q_ϕ

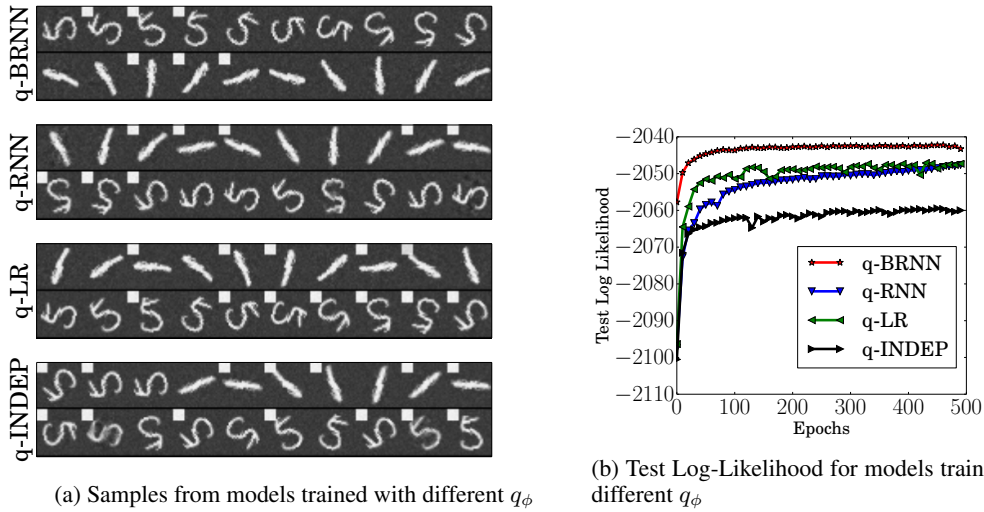(b) Test Log-Likelihood for models trained with different q_ϕ

Figure 3: **Small Healing MNIST**: (a) Mean probabilities sampled under different variational models with a constant, large rotation applied to the right. (b) Test log-likelihood under different recognition models.

9

gains after more than 100 epochs of training[2]. Figure 3b depicts the log-likelihood under the test set (we estimate test log-likelihood using importance sampling based on the recognition network - see Supplemental Section A). Figure 3a depicts pairs of sequence samples obtained under each of the variational models.

It is unsurprising that **q-BRNN** outperforms the other variants. In a manner similar to the Forward-Backward algorithm, the Bi-Directional RNN summarizes the past and the future at every timestep to form the most effective approximation to the posterior distribution of z_t.

It is also not surprising that **q-INDEP** performs poorly, both in the quality of the samples and in held out log-likelihood. Given the sequential nature of the data, the posterior for z_t is poorly approximated when only given access to x_t. The samples capture the effect of rotation but not the squares.

Although the test log-likelihood was better for **q-LR** than for **q-RNN**, the samples obtained were worse off - in particular, they did not capture the three consecutive block structure. The key difference between **q-LR** and **q-RNN** is the fact that the former has no memory that it carries across time. This facet will likely be more relevant in sequences where there are multiple patterns and the recognition network needs to remember the order in which the patters are generated.

Finally, we note that both **q-RNN** and **q-BRNN** learn generative models with plausible samples.

Large Healing MNIST

Figure 2a depicts pairs of training sequences, and the mean probabilities obtained after reconstruction as learning progresses. There are instances (first and third from the top) where the noise level is too high for the structure of the digit to be recognized from the training data alone. The reconstructions also shows that the model learns different styles of the digits (corresponding to variances within individual patients).

Figure 2b depicts samples from the model under varying degrees of rotation (corresponding to the intensity of a treatment for example). Here again, the model shows that it is capable of learning variations within the digit as well as realizing the effect of an action and its intensity. This is a simple form of counterfactual reasoning that can be performed by the model, since none of the samples on display are within the training set.

Figure 2c answers two questions. The first is what happens when we ask the model to reconstruct on data that looks similar to the training distribution but not the same. The input image is on the left (with a clean and noisy version of the digit displayed) and the following sample represent the reconstruction by the variational model of a sequence created from the input images. Following this, we forward sample from the model using the inferred latent representation under a constant action.

To this end, we consider digits of the same class (i.e. $1, 5$) but of a different style than the model was trained on. This idea has parallels within the medical setting where one asks about the course of action for a new patient. On this unseen patient, the model would infer a latent state similar to a patient that exists in the training set. This facet of the model motivates further investigation into the model's capabilities as a metric for patient similarity. To simulate the medical question: What would happen if the doctor prescribed the drug "rotate right mildly"? We forward sample from the model under this action.

In most cases, noisy or not, the patient's reconstruction matches a close estimate of a digit found in the training set (close in log-likelihood since this is the criterion the emission distribution was trained on). The final four rows depict scenarios in which the noise level is too high and we don't show the model enough sequences to make accurate inferences about the digit.

6.2 Generative Models of Medical Data

We learn a generative model on healthcare claims data from a major health insurance provider. We look into the effect of anti-diabetic drugs on a population of 8000 diabetic and pre-diabetic patients.

---

[2]In particular, we found that adding layers to the variational model helped but only up to a certain point. Beyond that, learning the model was infeasible.
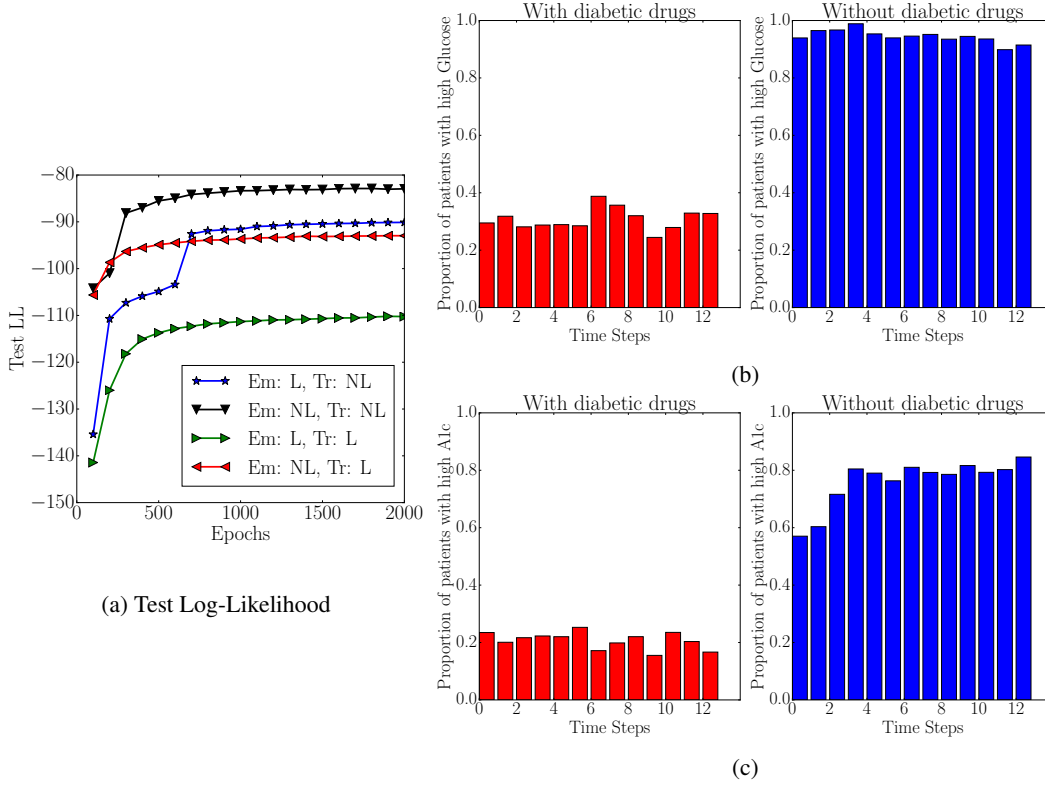
Figure 4: Results of disease progression modeling for 8000 diabetic and pre-diabetic patients. (a) Test log-likelihood under different model variants. Em(ission) denotes F_κ, Tr(ansition) denotes G_α under Linear (L) and Non-Linear (NL) functions. We learn a fixed diagonal S_β. (b) Proportion of patients inferred to have high (top quantile) glucose level with and without anti-diabetic drugs, starting from the time of first Metformin prescription. (c) Proportion of patients inferred to have high (above 8%) A1c level with and without anti-diabetic drugs, starting from the time of first Metformin prescription. Both (b) and (c) are created using the model trained with non-linear emission and transition functions.

The (binary) observations of interest here are: A1c level (hemoglobin A1c, a type of protein commonly used in the medical literature to indicate level of diabetes where high A1c level are an indicator for diabetes) and glucose (the amount of a patient's blood sugar). We bin glucose into quantiles and A1c into medically meaningful bins. The observations also include age, gender and ICD-9 diagnosis codes depicting various comorbidities of diabetes such as congestive heart failure, chronic kidney disease and obesity.

For actions, we consider prescriptions of nine diabetic drugs including Metformin and Insulin, where Metformin is the most commonly prescribed first-line anti-diabetic drug. For each patient, we group their data over four and half years into three months intervals.

We aim to assess the effect of anti-diabetic drugs on a patient's A1c and glucose levels. To that end, we ask a counterfactual question: how would the patient's A1c and glucose levels be had they not received the prescribed medications as observed in the dataset.

A complication in trying to perform counterfactual inference for the A1c and glucose levels is that these quantities are not always measured for each patient at each time step. Moreover, patients who are suspected of being diabetic are tested much more often for their A1c and glucose levels, compared with healthy patients, creating a confounding factor, since diabetic patients tend to have higher A1c and glucose levels. To overcome this we add an observation variable called "lab indicator", denoted x^{ind}, which indicates whether the respective lab test, either A1c or glucose, was taken regardless of its outcome. We condition the time t lab indicator observation, x_t^{ind}, on the latent state z_t, and we condition the A1c and glucose value observations on both the latent state and the lab indicator observation. That way, once the model is trained we can perform counterfactual inference by using the *do*-operator on the lab indicator: setting it to 1 and ignoring its dependence on the latent state. See Figure 5 for an illustration of the model.
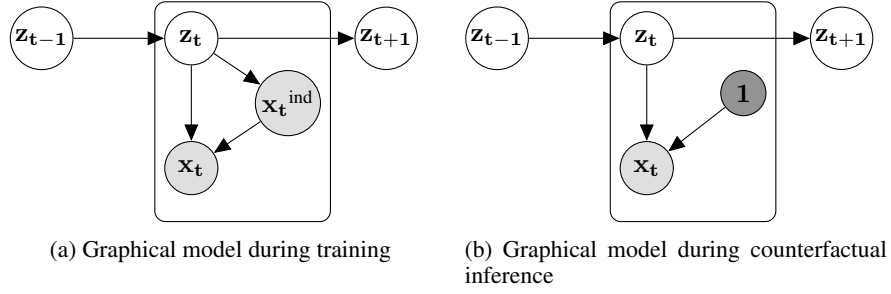
(a) Graphical model during training (b) Graphical model during counterfactual inference

Figure 5: (a) Generative model with lab indicator variable, focusing on time step t. (b) For counterfactual inference we set x_t^{ind} to 1, implementing Pearl's *do*-operator

We train the model on a dataset of 8000 patients. We use **q-BRNN** as the recognition model.

Variants of Kalman Filters: Figure 4(a) depicts the test log likelihood under variants of the graphical model depicted in Figure 1b. **Em**(ission) denotes F_κ, the emission function, and **Tr**(ansition) denotes G_α, the transition function of the mean. We learn a fixed diagonal covariance matrix (S_β). See Eq. (2) for the role these quantities play in the generative model. Linear (L) denotes a linear relationship between entities, and Non-Linear (NL) denotes a non-linear one parameterized by a two-layer neural network. Early in training, a non-linear emission function suffices to achieve a high test log likelihood though as training progresses the models with non-linear transition functions dominate.

Counterfactual Inference: We use a model with non-linear transition and non-linear emission functions. We look at patients whose first prescribed anti-diabetic drug was Metformin, the most common first-line anti-diabetic drug, and who have at least six months of data before the first Metformin prescription. This leaves us with 800 patients for whom we ask the counterfactual question. For these patients, we infer the latent state up to the time t_0 of first Metformin prescription. After t_0 we perform forward sampling under two conditions: the "**with**" condition is using the patient's true medication prescriptions; the "**without**" condition is removing the medication prescriptions, simulating a patient who receives no anti-diabetic medication. In both cases we set the lab indicator variable x_t^{ind} to be 1, so we can observe the A1c and glucose lab values. We then compare the inferred A1c and glucose lab values between the "**with**" and "**without**" conditions after the time of first Metformin prescription. Figure 4 presents the results, where we track the proportion of patients with high glucose level (glucose in the top quantile) and high A1c levels (A1c above 8%), starting from the time of first Metformin prescription. It is evident that patients who do not receive the anti-diabetic drugs are much more prone to having high glucose and A1c levels.

7 Discussion

We show promising results that nonlinear-state space models can be effective models for counterfactual analysis. The parametric posterior can be used to approximate the latent state of unseen data. We can forward sample from the model under different actions and observe their consequent effect. Beyond counterfactual inference, the model represents a natural way to embed patients into latent space making it possible to ask questions about patient similarity. Another avenue of work is understanding whether the latent variable space encodes identifiable characteristics of a patient and whether the evolution of the latent space corresponds to known disease trajectories.

There exists interesting avenues of future work for our model in a multitude of areas. A natural question that arises, particularly with models trained on the Healing MNIST is the quality of temporal and spatial invariance in the learned filters. Unsupervised learning of videos is another domain where our model holds promise. Approaches such as (Srivastava *et al.* , 2015) model video sequences using LSTMs with deterministic transition operators. The effect of stochasticity in the latent space is an interesting one to explore.

Acknowledgements

References

Bayer, Justin, & Osendorfer, Christian. 2014. Learning Stochastic Recurrent Networks. *arXiv preprint arXiv:1411.7610*.

Bottou, Léon, Peters, Jonas, Quinonero-Candela, Joaquin, Charles, Denis X, Chickering, D Max, Portugaly, Elon, Ray, Dipankar, Simard, Patrice, & Snelson, Ed. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, **14**(1), 3207–3260.

Chung, Junyoung, Kastner, Kyle, Dinh, Laurent, Goel, Kratarth, Courville, Aaron, & Bengio, Yoshua. 2015. A Recurrent Latent Variable Model for Sequential Data. *arXiv preprint arXiv:1506.02216*.

Collobert, Ronan, Kavukcuoglu, Koray, & Farabet, Clément. 2011. Torch7: A matlab-like environment for machine learning. *In: BigLearn, NIPS Workshop*.

Gan, Zhe, Li, Chunyuan, Henao, Ricardo, Carlson, David, & Carin, Lawrence. 2015. Deep Temporal Sigmoid Belief Networks for Sequence Modeling.

Goroshin, Ross, Mathieu, Michael, & LeCun, Yann. 2015. Learning to Linearize under Uncertainty. *arXiv preprint arXiv:1506.03011*.

Gregor, Karol, Danihelka, Ivo, Graves, Alex, Rezende, Danilo Jimenez, & Wierstra, Daan. 2015. DRAW: A Recurrent Neural Network For Image Generation. *In: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*.

Haykin, Simon. 2004. *Kalman filtering and neural networks*. Vol. 47. John Wiley & Sons.

Höfler, M. 2005. Causal inference based on counterfactuals. *BMC medical research methodology*, **5**(1), 28.

Jazwinski, Andrew H. 2007. *Stochastic processes and filtering theory*. Courier Corporation.

Kalman, Rudolph Emil. 1960. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, **82**(1), 35–45.

Kingma, Diederik, & Ba, Jimmy. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, Diederik P, & Welling, Max. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Langford, John, Salakhutdinov, Ruslan, & Zhang, Tong. 2009. Learning Nonlinear Dynamic Models. *Pages 593–600 of: Proceedings of the 26th Annual International Conference on Machine Learning*. ACM.

LeCun, Yann, & Cortes, Corinna. 2010. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*.

Memisevic, Roland. 2013. Learning to Relate Images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **35**(8), 1829–1846.

Mirowski, Piotr, & LeCun, Yann. 2009. Dynamic factor graphs for time series modeling. *Pages 128–143 of: Machine Learning and Knowledge Discovery in Databases*. Springer.

Morgan, Stephen L, & Winship, Christopher. 2014. *Counterfactuals and causal inference*. Cambridge University Press.

Pearl, Judea. 2009. *Causality*. Cambridge university press.

Raiko, Tapani, & Tornio, Matti. 2009. Variational Bayesian learning of nonlinear hidden state-space models for model predictive control. *Neurocomputing*, **72**(16), 3704–3712.

Rezende, Danilo Jimenez, & Mohamed, Shakir. 2015. Variational Inference with Normalizing Flows. *arXiv preprint arXiv:1505.05770*.

Rezende, Danilo Jimenez, Mohamed, Shakir, & Wierstra, Daan. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Rosenbaum, Paul R. 2002. *Observational studies*. Springer.

Roweis, Sam, & Ghahramani, Zoubin. 2000. An EM algorithm for identification of nonlinear dynamical systems.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, & Salakhutdinov, Ruslan. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, **15**(1), 1929–1958.

Srivastava, Nitish, Mansimov, Elman, & Salakhutdinov, Ruslan. 2015. Unsupervised learning of video representations using LSTMs. *arXiv preprint arXiv:1502.04681*.

Tabak, Esteban G, Vanden-Eijnden, Eric, *et al.* . 2010. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, **8**(1), 217–233.

Valpola, Harri, & Karhunen, Juha. 2002. An unsupervised ensemble learning method for nonlinear dynamic state-space models. *Neural computation*, **14**(11), 2647–2692.

Velez, Finale Doshi. 2013. Partially-Observable Markov Decision Processes as Dynamical Causal Models.

Wan, Eric, Van Der Merwe, Ronell, *et al.* . 2000. The unscented Kalman filter for nonlinear estimation. *Pages 153–158 of: Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*. IEEE.

Watter, Manuel, Springenberg, Jost Tobias, Boedecker, Joschka, & Riedmiller, Martin. 2015. Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images. *arXiv preprint arXiv:1506.07365*.

Zaremba, Wojciech, & Sutskever, Ilya. 2014. Learning to Execute. *arXiv preprint arXiv:1410.4615*.

A Lower Bound on Likelihood

In the following we omit the dependence of q on $\vec{x}$ and $\vec{u}$, and omit the subscript ϕ. We can show that the KL divergence between the approximation to the posterior and the prior simplifies as:

$$\text{KL}(q(z_1, \ldots, z_T)||p(z_1, \ldots, z_T)) = \int_{z_1} \ldots \int_{z_T} q(z_1) \ldots q(z_T|z_{T-1}) \log \frac{p(z_1, z_2, \ldots, z_T)}{q(z_1) \ldots q(z_T|z_{T-1})} \tag{6}$$

(Factorization of the variational distribution)

$$= \int_{z_1} \ldots \int_{z_T} q(z_1) \ldots q(z_T|z_{T-1}) \log \frac{p(z_1)p(z_2|z_1, u_1) \ldots p(z_T|z_{T-1}, u_{T-1})}{q(z_1) \ldots q(z_T|z_{T-1})}$$

(Factorization of the prior)

$$= \int_{z_1} \ldots \int_{z_T} q(z_1) \ldots q(z_T|z_{T-1}) \log \frac{p(z_1)}{q(z_1)} + \sum_{t=2}^{T} \int_{z_1} \ldots \int_{z_T} q(z_1) \ldots q(z_T|z_{T-1}) \log \frac{p(z_t|z_{t-1})}{q(z_t|z_{t-1})}$$

$$= \int_{z_1} q(z_1) \log \frac{p(z_1)}{q(z_1)} + \sum_{t=2}^{T} \int_{z_{t-1}} \int_{z_t} q(z_t) \log \frac{p(z_t|z_{t-1})}{q(z_t|z_{t-1})}$$

(Each expectation over z_t is constant for $t \notin \{t, t-1\}$)

$$= \text{KL}(q(z_1)||p(z_1)) + \sum_{t=2}^{T-1} \underset{q(z_{t-1})}{\mathbb{E}} [\text{KL}(q(z_t|z_{t-1})||p(z_t|z_{t-1}, u_{t-1}))]$$

For evaluating the marginal likelihood on the test set, we can use the following Monte-Carlo estimate:

$$p(\vec{x}) \cong \frac{1}{S} \sum_{s=1}^{S} \frac{p(\vec{x}|\vec{z}^{(s)})p(\vec{z}^{(s)})}{q(\vec{z}^{(s)}|\vec{x})} \quad \vec{z}^{(s)} \sim q(\vec{z}|\vec{x}) \tag{7}$$

This may be derived in a manner akin to the one depicted in Appendix E (Rezende *et al.* , 2014) or Appendix D (Kingma & Welling, 2013).

The log likelihood on the test set is computed using:

$$\log p(\vec{x}) \cong \log \frac{1}{S} \sum_{s=1}^{S} \exp \log \left[\frac{p(\vec{x}|\vec{z}^{(s)})p(\vec{z}^{(s)})}{q(\vec{z}^{(s)}|\vec{x})} \right] \tag{8}$$

(8) may be computed in a numerically stable manner using the log-sum-exp trick.

B KL divergence between Prior and Posterior

Maximum likelihood learning requires us to compute:

$$KL(q(z_1, \ldots, z_T)||p(z_1, \ldots, z_T)) = KL(q(z_1)||p(z_1)) + \sum_{t=2}^{T-1} \underset{q(z_{t-1})}{\mathbb{E}} [KL(q(z_t)||p(z_t|z_{t-1}, u_{t-1}))] \tag{9}$$

The KL divergence between two multivariate Gaussians q, p with respective means and covariances $\mu_q, \Sigma_q, \mu_p, \Sigma_p$ can be written as:

$$KL(q||p) = \frac{1}{2} \left(\underbrace{\log \frac{|\Sigma_p|}{|\Sigma_q|}}_{(a)} - D + \underbrace{\text{Tr}(\Sigma_p^{-1}\Sigma_q)}_{(b)} + \underbrace{(\mu_p - \mu_q)^T \Sigma_p^{-1}(\mu_p - \mu_q)}_{(c)} \right) \tag{10}$$

The choice of q and p is suggestive. using (9) & (10), we can derive a closed form for the KL divergence between $q(z_1 \ldots z_T)$ and $p(z_1 \ldots z_T)$.

μ_q, Σ_q are the outputs of the variational model. Our functional form for μ_p, Σ_p is based on our generative and can be summarized as:

$$\mu_{p1} = 0 \qquad \Sigma_{p1} = \mathbb{1} \qquad \mu_{pt} = G(z_{t-1}, u_{t-1}) = G_{t-1} \qquad \Sigma_{pt} = \Delta\vec{\sigma}$$

Here, Σ_{pt} is assumed to be a learned diagonal matrix and Δ a scalar parameter.

Term (a) For $t = 1$, we have:

$$\log \frac{|\Sigma_{p1}|}{|\Sigma_{q1}|} = \log|\Sigma_{p1}| - \log|\Sigma_{q1}| = -\log|\Sigma_{q1}| \tag{11}$$

For $t > 1$, we have:

$$\log \frac{|\Sigma_{pt}|}{|\Sigma_{qt}|} = \log|\Sigma_{pt}| - \log|\Sigma_{qt}| = D\log(\Delta) + \log|\vec{\sigma}| - \log|\Sigma_{qt}| \tag{12}$$

Term (b) For $t = 1$, we have:

$$\text{Tr}(\Sigma_{p1}^{-1}\Sigma_{q1}) = \text{Tr}(\Sigma_{q1}) \tag{13}$$

For $t > 1$, we have:

$$\text{Tr}(\Sigma_{pt}^{-1}\Sigma_{qt}) = \frac{1}{\Delta} \text{Tr}(\text{diag}(\vec{\sigma})^{-1}\Sigma_{qt}) \tag{14}$$

Term (c) For $t = 1$, we have:

$$(\mu_{p1} - \mu_{q1})^T \Sigma_{p1}^{-1} (\mu_{p1} - \mu_{q1}) = ||\mu_{q1}||^2 \tag{15}$$

For $t > 1$, we have:

$$(\mu_{pt} - \mu_{qt})^T \Sigma_{pt}^{-1} (\mu_{pt} - \mu_{qt}) = \Delta(G_{t-1} - \mu_{qt})^T \text{diag}(\vec{\sigma})^{-1} (G_{t-1} - \mu_{qt}) \tag{16}$$

Rewriting (9) using (11), (12), (13), (14), (15), (16), we get:

$$
\begin{aligned}
KL(q(z_1, \ldots, z_T)||p(z_1, \ldots, z_T)) = \frac{1}{2} &\left((T-1)D\log(\Delta)\log|\vec{\sigma}| - \sum_{t=1}^{T} \log|\Sigma_{qt}| \right. \\
&+ \text{Tr}(\Sigma_{q1}) + \frac{1}{\Delta} \sum_{t=2}^{T} \text{Tr}(\text{diag}(\vec{\sigma})^{-1}\Sigma_{qt}) + ||\mu_{q1}||^2 \\
&\left. + \Delta \sum_{t=2}^{T} \mathop{\mathbb{E}}_{z_{t-1}} \left[(G_{t-1} - \mu_{qt})^T \text{diag}(\vec{\sigma})^{-1} (G_{t-1} - \mu_{qt}) \right] \right)
\end{aligned}
\tag{17}
$$

We can now take gradients with respect to $\mu_{qt}, \Sigma_{qt}, G(z_{t-1}, u_{t-1})$ in (17).
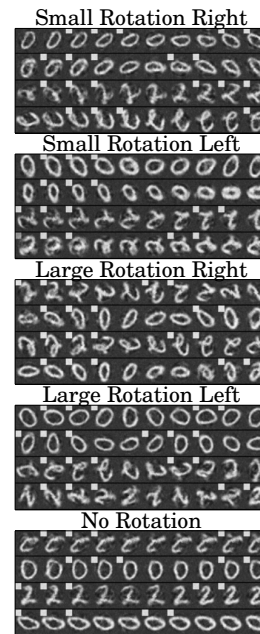
C Additional Experimental Results

We consider a variant of **Large Healing MNIST** trained on 100 different kinds of 0, 2s each.

Figure 6a and 6b depict the reconstructions and samples from a model trained on the digits 0 and 2.
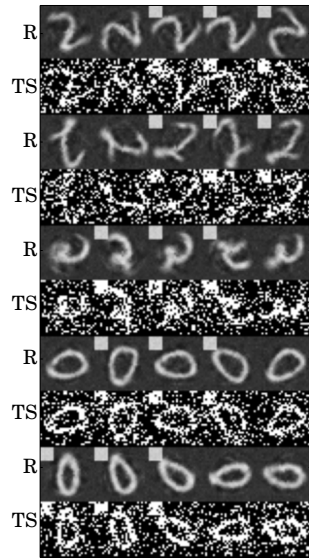
(a) **Large Healing MNIST** (0,2):Pairs of Training Sequences (TS) and mean probabilities of Reconstructions (R) shown above.

(b) **Large Healing MNIST** (0,2): Mean probabilities sampled under different, constant rotations.