Conditional Recurrent Flow: Conditional Generation of Longitudinal Samples with Applications to Neuroimaging

Seong Jae Hwang[1]
Univ. of Pittsburgh

Zirui Tao
Univ. of Wisconsin-Madison

Won Hwa Kim[2]
Univ. of Texas at Arlington

Vikas Singh[2]
Univ. of Wisconsin-Madison

Abstract

We develop a conditional generative model for longitudinal image datasets based on sequential invertible neural networks. Longitudinal image acquisitions are common in various scientific and biomedical studies where often each image sequence sample may also come together with various secondary (fixed or temporally dependent) measurements. The key goal is not only to estimate the parameters of a deep generative model for the given longitudinal data, but also to enable evaluation of how the temporal course of the generated longitudinal samples are influenced as a function of induced changes in the (secondary) temporal measurements (or events). Our proposed formulation incorporates recurrent subnetworks and temporal context gating, which provide a smooth transition in a temporal sequence of generated data that can be easily informed or modulated by secondary temporal conditioning variables. We show that the formulation works well despite the smaller sample sizes common in these applications. Our model is validated on two video datasets and a longitudinal Alzheimer's disease (AD) dataset for both quantitative and qualitative evaluations of the generated samples. Further, using our generated longitudinal image samples, we show that we can capture the pathological progressions in the brain that turn out to be consistent with the existing literature, and could facilitate various types of downstream statistical analysis.

1. Introduction

Consider a dataset of *longitudinal or temporal sequences* of data samples $\{\mathbf{x}^t\}_{i=1}^N$ where each sample $\mathbf{x}_i$ comes with *sequential covariates* $\{\mathbf{y}^t\}_{i=1}^N$, one for each time point t. In other words, we assume that for each sequential sample i, $\mathbf{x}_i^1, \cdots, \mathbf{x}_i^T = \{\mathbf{x}^t\}_i$, the sequential covariates $\mathbf{y}_i^1, \cdots, \mathbf{y}_i^T = \{\mathbf{y}^t\}_i$ provide some pertinent auxiliary information associated with that sequential sample. For example, in a neuroimaging study, if the sequential samples correspond to several longitudinal image scans of a participant over multiple years, the sequential covariate associated with each time point may be an assessment of disease severity or some other clinical measurement.

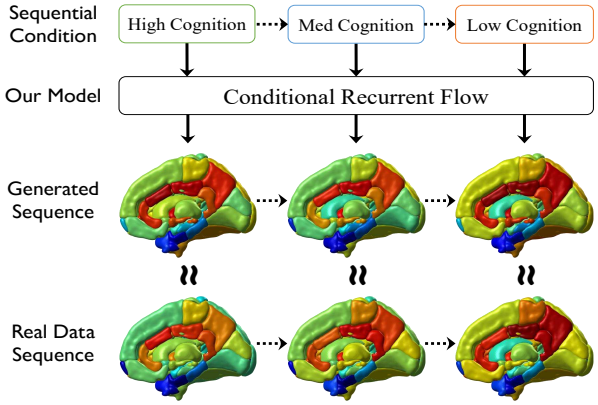


Figure 1: **Conditional sequence generation illustration. 1) Given:** a sequential condition of decreasing cognition (i.e., a memory test score sequence $\mathbf{y}_i^1 \rightarrow \mathbf{y}_i^2 \rightarrow \mathbf{y}_i^3$ indicating High→Medium→Low Cognition performance). **2) Model:** Conditional Recurrent Flow (CRow). **3) Generate:** a sequence of brain image progression $\mathbf{x}_i^1 \rightarrow \mathbf{x}_i^2 \rightarrow \mathbf{x}_i^3$ corresponding to the given cognition progression (i.e., brain regions with high (red) and low (blue) disease pathology). The Generated Sequence follows the trend of the Real Data Sequence (i.e., similar ($\approx$) to the real brain image progression) from the subjects with similarly decreasing cognition scores.

Our high level goal is to design conditional generative models for such sequential image data. In particular, we want a model which provides us a type of flexibility that is highly desirable in this setting. For instance, for a sample drawn from the distribution after the generative model has been estimated, we should be able to "adjust" the sequential covariates, say at a time point t, dynamically to influence the expected future predictions after t for *that* sample. It makes sense that for a heart rate sequence, the appropriate subsequence should be influenced by *when* the "violence" stimulus was introduced *as well as* the default heart rate pattern of the specific sample (participant) [2]. Notice that when $t = 1$, this construction is similar to conditional generative models where the "covariate" or condition $\mathbf{y}$ may simply denote an attribute that we may want to adjust for a sample: for example, increase the smile or age attribute for a face image sampled from the distribution as in [26].

We want our formulation to provide a modified set of $\mathbf{x}^t$s adaptively, if we adjust sequential covariates $\mathbf{y}^t$s for that

[1]: Work done while SJH was at the University of Wisconsin-Madison
[2]: Shared senior authorship

sample. If we know some important clinical information at some point during the study (say, at $t = 5$), this information should influence the future generation $\mathbf{x}^{t>5}$ conditioned both on this sequential covariate or event $\mathbf{y}^5$ as well as the past sequence of this sample $\mathbf{x}^{t<5}$. This will require *conditioning* on the corresponding sequential covariates at *each* time point t by accurately capturing the posterior distribution $p(\mathbf{x}^t|\mathbf{y}^t)$. Such *conditional sequence generation* needs a generative model for *sequential* data which can dynamically incorporate time-specific *sequential* covariates $\mathbf{y}^t$ of interest to adaptively modify sequences.

The setup above models a number of applications in medical imaging and computer vision that require generation of frame sequences conditioned on frame-level covariates. In neuroimaging, many longitudinal studies focus on identifying disease trajectories [3, 5, 28, 18]: for example, at what point in the future will specific regions in the brain exceed a threshold for brain atrophy? The future trend is invariably a function of clinical measures that a participant provides at each visit *as well as* the past trend of the subject. From a methodological standpoint, constructing a sequential generative model may appear feasible by appropriately augmenting the generation process using existing generative models. For example, one could simply concatenate the sequential measurements $\{\mathbf{x}^t\}$ as a single input for existing non-sequential conditional generative models such as conditional GANs [31, 19] and conditional variational autoencoders [38, 1]. We will see why this is not ideal shortly.

We find that for our application, an attractive alternative to discriminator-generator based GANs, is a family of neural networks called normalizing flow [36, 35, 10, 9] which involve *invertible networks* (i.e., reconstruct input from its output). What is particularly relevant is that such formulations work well for *conditionally* generating diverse samples with controllable degrees of freedom [4] – with an explicit mechanism to adjust the conditioning variable. But the reader will notice that while these models, in principle, can be used to approximate the posterior probability given an input of any dimension, concatenating a series of sequential inputs quickly blows up the size for these highly expressive models and renders them impractical to run, even on high-end GPU clusters. Even if we optimistically assume computational feasibility, variable length sequences cannot easily be adapted to these innately non-sequential generative models, especially for those that extend beyond the training sequence length. Also, data generated in this manner involve simply "concatenated" sequential data and do not consider the innate temporal relationships among the sequences, which is fundamental in recurrent models. For these reasons, adapting existing generative models, will involve setting up a generative model which is recursive for variable length inputs.

Given various potential downstream applications and the issues identified above with conditional sequential generation problem, we seek a model which (i) efficiently generates high dimensional sequence samples of *variable lengths* (ii) with dynamic time-specific conditions reflecting upstream observations (iii) with fast posterior probability estimation. We tackle the foregoing issues by introducing an invertible recurrent neural network, **CRow**, that includes **recurrent subnetwork** and **temporal context gating**. These modifications are critical in the following sense. *Invertibility* lets us precisely estimate the distribution of $p(\mathbf{x}^t|\mathbf{y}^t)$ in latent space. Introducing *recurrent subnetworks* and *temporal context gating* enables obtaining cues from previous time points $\mathbf{x}^{<t}$ to generate temporally sensible subsequent time points $\mathbf{x}^{\geq t}$. Specifically, our **contributions** are: **(A)** Our model generates conditional sequential samples $\{\mathbf{x}^t\}$ given sequential covariates $\{\mathbf{y}^t\}$ for $t = 1, \ldots, T$ time points where T can be arbitrarily long. Specifically, we allow this by posing the task as a *conditional sequence inverse problem* based on a conditional invertible neural network [4]. **(B)** Assessing the quality of the generated samples may not be trivial for certain modalities (e.g., non-visual features). With the specialized capability of the normalizing flow construction, our model estimates the posterior probabilities $p(\mathbf{x}^t|\mathbf{y}^t)$ of the generated sequences at each time point for potential downstream analyses involving uncertainty. **(C)** We demonstrate an interesting practical application of our model in a longitudinal neuroimaging dataset. We show that the generated longitudinal brain pathology trajectories (an illustration in Fig. 1) can lead to identifying specific regions in the brain which are statistically associated with Alzheimer's disease (AD).

2. Preliminary: Invertible Neural Networks

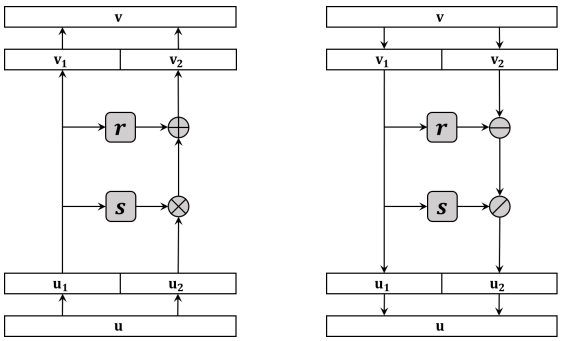
We first describe an *invertible neural network* (INN) which inverts an output back to its input for solving inverse problems (i.e., $\mathbf{z} = f(\mathbf{x}) \Leftrightarrow \mathbf{x} = f^{-1}(\mathbf{z})$). This becomes the building block of our method; thus, before we present our main model, let us briefly describe a specific type of invertible structure which was originally specialized for density estimation with neural network models.

2.1. Normalizing Flow

Estimating the density $p_{\mathbf{X}}(\mathbf{x})$ of sample $\mathbf{x}$ is a classical statistical problem in various fields including computer vision and machine learning in, e.g., uncertainty estimation [14, 15]. For tractable computation throughout the network, Bayesian adaptations are popular [34, 12, 33, 27, 23, 18], but these methods make assumptions on the prior distributions (e.g., exponential families).

A *normalizing flow* [36, 35] first learns a function $f(\cdot)$ which maps a sample $\mathbf{x}$ to a latent variable $\mathbf{z} = f(\mathbf{x})$ where $\mathbf{z}$ is from a standard normal distribution $\mathbf{Z}$. Then, with a change of variables formula, we estimate

$$p_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{Z}}(\mathbf{z})/|J_{\mathbf{X}}|, \quad |J_{\mathbf{X}}| = \left|\frac{\partial[\mathbf{x} = f^{-1}(\mathbf{z})]}{\partial \mathbf{z}}\right| \tag{1}$$



(a) Forward map (Eq. (2)) (b) Inverse map (Eq. (3))

Figure 2: Coupling layer in normalizing flow. Note the change of operation orders: $\mathbf{u} \rightarrow \mathbf{v}$ in forward and $\mathbf{v} \rightarrow \mathbf{u}$ in inverse.

where $|J_{\mathbf{X}}|$ is a Jacobian determinant. Thus, $f(\cdot)$ must be invertible, i.e., $\mathbf{x} = f^{-1}(\mathbf{z})$, and to use a neural network as $f(\cdot)$, a *coupling layer* structure was introduced in RealNVP [9, 10] for an easy inversion and efficient $|J_{\mathbf{X}}|$ computation as we describe next.

Forward map (Fig. 2a). Without loss of generality, in the context of network structures, we use an input $\mathbf{u} \in \mathbb{R}^d$ and an output $\mathbf{v} \in \mathbb{R}^d$ (i.e., $\mathbf{u} \rightarrow \mathbf{v}$). First, we split $\mathbf{u}$ into $\mathbf{u}_1 \in \mathbb{R}^{d_1}$ and $\mathbf{u}_2 \in \mathbb{R}^{d_2}$ where $d = d_1 + d_2$ (e.g., partition $\mathbf{u} \rightarrow [\mathbf{u}_1, \mathbf{u}_2]$). Then, we forward map $\mathbf{u}_1$ and $\mathbf{u}_2$ to $\mathbf{v}_1$ and $\mathbf{v}_2$ respectively:

$$\mathbf{v}_1 = \mathbf{u}_1, \quad \mathbf{v}_2 = \mathbf{u}_2 \otimes \exp(s(\mathbf{u}_1)) + r(\mathbf{u}_1) \tag{2}$$

where s and r are independent functions (i.e., subnetworks), and $\otimes$ and $+$ are element-wise product and addition respectively. Then, $\mathbf{v}_1$ and $\mathbf{v}_2$ construct $\mathbf{v}$ (e.g., $[\mathbf{v}_1, \mathbf{v}_2] \rightarrow \mathbf{v}$).

Inverse map (Fig. 2b). A straightforward arithmetic allows an exact inverse from $\mathbf{v}$ to $\mathbf{u}$ (i.e., $\mathbf{v} \rightarrow \mathbf{u}$):

$$\mathbf{u}_1 = \mathbf{v}_1, \quad \mathbf{u}_2 = (\mathbf{v}_2 - r(\mathbf{v}_1)) \oslash \exp(s(\mathbf{v}_1)) \tag{3}$$

where the subnetworks s and r are *identical* to those used in the forward map in Eq. (2), and $\oslash$ and $-$ are element-wise division and subtraction respectively. Note that the subnetworks are *not* explicitly inverted, thus any arbitrarily complex network can be utilized.

Also, the Jacobian matrix $J_{\mathbf{v}} = \partial \mathbf{v} / \partial \mathbf{u}$ is triangular so its determinant $|J_{\mathbf{v}}|$ is just the product of diagonal entries (i.e., $\prod_i \exp(s(\mathbf{u}_1))_i$) which is extremely easy to compute (we will discuss this further in Sec. 3.2.1).

To transform the "bypassed" split $\mathbf{u}_1$ (since $\mathbf{u}_1 = \mathbf{v}_1$), a *coupling block* consisting of two complementary coupling layers is constructed to transform both $\mathbf{u}_1$ and $\mathbf{u}_2$:

$$\begin{aligned} \mathbf{v}_1 &= \mathbf{u}_1 \otimes \exp(s_2(\mathbf{u}_2)) + r_2(\mathbf{u}_2) \\ \mathbf{v}_2 &= \mathbf{u}_2 \otimes \exp(s_1(\mathbf{v}_1)) + r_1(\mathbf{v}_1) \end{aligned} \tag{4}$$

and its inverse

$$\begin{aligned} \mathbf{u}_2 &= (\mathbf{v}_2 - r_1(\mathbf{v}_1)) \oslash \exp(s_1(\mathbf{v}_1)) \\ \mathbf{u}_1 &= (\mathbf{v}_1 - r_2(\mathbf{u}_2)) \oslash \exp(s_2(\mathbf{u}_2)). \end{aligned} \tag{5}$$

Such a series of transformations allow a more complex mapping which still comes with a chain of efficient Jacobian determinant computations, i.e., $\det(AB) = \det(A)\det(B)$ where A and B are the Jacobian matrices of two coupling layers. More details are included in the supplement.

Note that we have used (and will be using) $\mathbf{u}$ and $\mathbf{v}$ as generic input and output of an INN. Thus, specifically in the context of normalizing flow, by simply considering $\mathbf{u}$ and $\mathbf{v}$ to be $\mathbf{x}$ and $\mathbf{z}$ respectively, we can use a coupling layer based INN as a powerful *invertible* function $f(\cdot)$ to perform the normalizing flow described in Eq. (1).

3. Model Setup: Conditional Recurrent Flow

In this section, we describe our conditional sequence generation method called *Conditional Recurrent Flow* (CRow). We first describe a *conditional invertible neural network* (cINN) [4] which is one component of our model. Then, we explain how to incorporate temporal context gating and discuss the settings where CRow can be useful.

3.1. Conditional Sample Generation

Naturally, an inverse problem can be posed as a sample generation procedure by *sampling* a latent variable $\mathbf{z}$ and inverse mapping it to $\mathbf{x} = f^{-1}(\mathbf{z})$, thus *generating* a new sample $\mathbf{x}$. The concern is that we cannot specifically 'choose' to generate an $\mathbf{x}$ of interest since a latent variable $\mathbf{z}$ does not provide any interpretable associations with $\mathbf{x}$.

In other words, estimating the *conditional probability* $p(\mathbf{x}|\mathbf{y})$ is desirable since it represents an underlying phenomenon of the input $\mathbf{x} \in \mathbb{R}^d$ and covariate $\mathbf{y} \in \mathbb{R}^k$ (e.g., the probability of a specific brain imaging measure $\mathbf{x}$ of interest given a diagnosis $\mathbf{y}$). In fact, when we cast this problem into a normalizing flow, the goal becomes constructing an invertible network $f(\cdot)$ which maps a given input $\mathbf{x} \in \mathbb{R}^d$ to its corresponding covariate/label $\mathbf{y} \in \mathbb{R}^k$ and its latent variable $\mathbf{z} \in \mathbb{R}^m$ such that $[\mathbf{y}, \mathbf{z}] = f(\mathbf{x})$. The mapping must have an inverse for $\mathbf{x} = f^{-1}([\mathbf{y}, \mathbf{z}])$ to be recovered.

Specifically, when a flow-based model jointly encodes label and latent information (i.e., $[\mathbf{y}, \mathbf{z}] = \mathbf{v} = f(\mathbf{x})$ via Eq. (4)) while ensuring that $p(\mathbf{y})$ and $p(\mathbf{z})$ are independent, then the network becomes *conditionally* invertible (i.e., $\mathbf{x} = f^{-1}([\mathbf{y}, \mathbf{z}])$ conditioned on given $\mathbf{y}$). Such a network can be theoretically constructed through a bidirectional-type training [4], and this allows a conditional sampling $\mathbf{x} = f^{-1}([\mathbf{y}, \mathbf{z}])$ and the posterior estimation $p(\mathbf{x}|\mathbf{y})$.

Bidirectional training. This training process involves three losses: (1) $\mathcal{L}_{\mathbf{Z}}(p(\mathbf{y}, \mathbf{z}), p(\mathbf{y})p(\mathbf{z}))$ enforces $p(\mathbf{y})$ and $p(\mathbf{z})$ to be independent by making the network output $p(\mathbf{y}, \mathbf{z})$ to follow $p(\mathbf{y})p(\mathbf{z})$ which is true if and only if $p(\mathbf{y})$ and $p(\mathbf{z})$ are independent. (2) $\mathcal{L}_{\mathbf{Y}}(\mathbf{y}, \mathbf{y}_{\text{gt}})$ is the supervised label loss between our prediction $\mathbf{y}$ and the ground truth $\mathbf{y}_{\text{gt}}$. (3) $\mathcal{L}_{\mathbf{X}}(p(\mathbf{x}), p_{\mathbf{X}})$ improves the likelihood of the input $\mathbf{x}$ with respect to the prior $p_{\mathbf{X}}$. $\mathcal{L}_{\mathbf{Z}}$ and $\mathcal{L}_{\mathbf{X}}$ are based on a kernel-based moment matching measure Maximum Mean Discrepancy (MMD) [11, 44], also see appendix.

In practice, $\mathbf{x}$ and $[\mathbf{y}, \mathbf{z}]$ may not be of the same dimen-

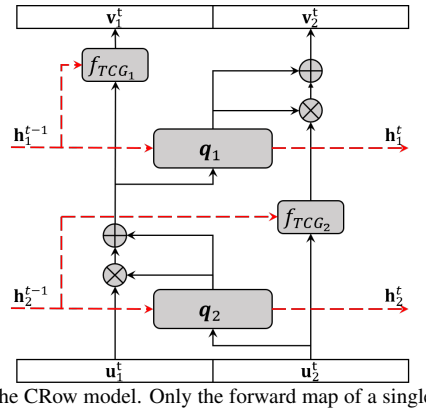


Figure 3: The CRow model. Only the forward map of a single block (two coupling layers) is shown for brevity. The inverse map involves a similar order of operations (analogous to Fig. 2a and Fig. 2b)

sions. To construct a square triangular Jacobian matrix, zero-padding both $\mathbf{x}$ and $[\mathbf{y}, \mathbf{z}]$ can alleviate this issue while also increasing the intermediate subnetwork dimensions for higher expressive power. Also, the forward mapping is essentially a prediction task that we encounter often in computer vision and machine learning, i.e., predicting $\mathbf{y} = f(\mathbf{x})$ or maximizing the likelihood $p(\mathbf{y}|\mathbf{x})$ without explicitly utilizing the latent $\mathbf{z}$. On the other hand, the inverse process of deriving $\mathbf{x} = f^{-1}(\mathbf{y})$, allows a more scientifically based analysis of the underlying phenomena, e.g., the interaction between brain ($\mathbf{x}$) and observed cognitive function ($\mathbf{y}$).

3.2. Conditional Recurrent Flow (CRow)

The existing normalizing flow type networks cannot explicitly incorporate sequential data which are now increasingly becoming important in various applications. Successful recurrent models such as gated recurrent unit (GRU) [6, 40] and long short-term memory (LSTM) [16, 37] explicitly focus on encoding the "memory" from the past and output proper state information for accurate sequential predictions given the past. Similarly, generated sample sequences must also follow sequentially sensible patterns or trajectories resembling likely sequences by encoding appropriate *temporal information* for the subsequent time points.

To overcome these issues, we introduce Conditional Recurrent Flow (CRow) model for conditional sequence generation. Given a sequence of input/output pairs $\{\mathbf{u}^t, \mathbf{v}^t\}$ for $t = 1, \dots, T$ time points, modeling the relationship between the variables across time needs to also account for the temporal characteristic of the sequence. Variants of recurrent neural networks (RNN) such as GRU and LSTM have been showing success in sequential problems, but they only enable forward mapping. We are specifically interested in an *invertible network which is also recurrent* such that given a *sequence* of inputs $\{\mathbf{u}^t\}$ (i.e., features $\{\mathbf{x}^t\}$) and their *sequence* of outputs $\{\mathbf{v}^t\}$ (i.e., covariates/labels and latent information $\{\mathbf{y}^t, \mathbf{z}^t\}$), we can model the invertible relationship between those sequences for posterior estimation and conditional sequence generation as illustrated in Fig. 1.

Without loss of generality, we can describe our model in terms of generic $\{\mathbf{u}^t\}$ and $\{\mathbf{v}^t\}$. We follow the coupling block described in Eq. (4) and Eq. (5) to setup a normalizing flow type invertible model. Then, we impose the recurrent nature on the model by allowing the model to learn and pass down a hidden state $\mathbf{h}^t$ to the next time point through the recurrent subnetworks. Specifically, we construct a *recurrent* subnetwork q which also contains a recurrent network (e.g., GRU) internally. This allows q to take the previous hidden state $\mathbf{h}^{t-1}$ and output the next hidden state $\mathbf{h}^t$ as $[\mathbf{q}, \mathbf{h}^t] = q(\mathbf{u}, \mathbf{h}^{t-1})$ where $\mathbf{q}$ is an element-wise transformation vector derived from $\mathbf{u}$ analogous to the output of a subnetwork $s(\mathbf{u})$ in Eq. (2). In previous coupling layers (i.e., Eq. (2)), two transformation vectors $\mathbf{s} = s(\cdot)$ and $\mathbf{r} = r(\cdot)$ were explicitly computed from two subnetworks for each layer. For CRow, we follow the structure of Glow [26] which computes a single vector $\mathbf{q} = q(\cdot)$ and splits it as $[\mathbf{s}, \mathbf{r}] = \mathbf{q}$. This allows us to use a single hidden state while concurrently learning $[\mathbf{s}, \mathbf{r}]$ which we denote as $\mathbf{s} = q_s(\cdot)$ and $\mathbf{r} = q_r(\cdot)$ to indicate the individual vectors. Thus, at each t with given $[\mathbf{u}_1^t, \mathbf{u}_2^t] = \mathbf{u}^t$ and $[\mathbf{v}_1^t, \mathbf{v}_2^t] = \mathbf{v}^t$,

$$\begin{aligned} \mathbf{v}_1^t &= \mathbf{u}_1^t \otimes \exp(q_{s_2}(\mathbf{u}_2^t, \mathbf{h}_2^{t-1})) + q_{r_2}(\mathbf{u}_2^t, \mathbf{h}_2^{t-1}) \\ \mathbf{v}_2^t &= \mathbf{u}_2^t \otimes \exp(q_{s_1}(\mathbf{v}_1^t, \mathbf{h}_1^{t-1})) + q_{r_1}(\mathbf{v}_1^t, \mathbf{h}_1^{t-1}) \end{aligned} \tag{6}$$

and the inverse is

$$\begin{aligned} \mathbf{u}_2^t &= (\mathbf{v}_2^t - q_{r_1}(\mathbf{v}_1^t, \mathbf{h}_1^{t-1})) \oslash \exp(q_{s_1}(\mathbf{v}_1^t, \mathbf{h}_1^{t-1})) \\ \mathbf{u}_1^t &= (\mathbf{v}_1^t - q_{r_2}(\mathbf{u}_2^t, \mathbf{h}_2^{t-1})) \oslash \exp(q_{s_2}(\mathbf{u}_2^t, \mathbf{h}_2^{t-1})). \end{aligned} \tag{7}$$

Note that the hidden states $\mathbf{h}_1^t$ and $\mathbf{h}_2^t$ generated from the recurrent network of the subnetworks are implicitly used within the subnetwork architecture (i.e., inputs to additional fully connected layers) and also passed to their corresponding recurrent network in the next time point as in Fig. 3.

3.2.1 Temporal Context Gating (TCG)

A standard (single) coupling layer transforms only a part of the input (i.e., $\mathbf{u}_1$ in Eq. (2)) by design which results in the determinant of a triangular Jacobian matrix $J_{\mathbf{v}}$:

$$|J_{\mathbf{v}}| = \left|\frac{\partial \mathbf{v}}{\partial \mathbf{u}}\right| = \begin{vmatrix} \frac{\partial \mathbf{v}_1}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{v}_1}{\partial \mathbf{u}_2} \\ \frac{\partial \mathbf{v}_2}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{v}_2}{\partial \mathbf{u}_2} \end{vmatrix} = \begin{vmatrix} I & 0 \\ \frac{\partial \mathbf{v}_2}{\partial \mathbf{u}_1} & diag(\exp s(\mathbf{u}_1)) \end{vmatrix} \tag{8}$$

thus $|J_{\mathbf{v}}| = \exp(\sum_i (s(\mathbf{u}_1))_i)$. This is a result from Eq. (2): (1) the element-wise operations on $\mathbf{u}_2$ for the diagonal submatrix of partial derivatives $\partial \mathbf{v}_2 / \partial \mathbf{u}_2 = diag(\exp s(\mathbf{u}_1))$, (2) the bypassing of $\mathbf{u}_1 = \mathbf{v}_1$ for $\partial \mathbf{v}_1 / \partial \mathbf{u}_1 = I$, and (3) $\partial \mathbf{v}_1 / \partial \mathbf{u}_2 = 0$. Ideally, transforming $\mathbf{u}_1$ would be beneficial. However, this is explicitly avoided in the coupling layer design since this should *not* involve $\mathbf{u}_1$ or $\mathbf{u}_2$ directly; otherwise, $J_{\mathbf{v}}$ will not be triangular.

Using $\mathbf{h}^t$ in CRow. In the case of CRow, it incorporates a hidden state $\mathbf{h}^{t-1}$ from the previous time point which is *neither* $\mathbf{u}$ nor $\mathbf{v}$. This is our *temporal information* which adjusts the mapping function $f(\cdot)$ to allow more accurate mapping depending on the previous time points of the sequence which is crucial for sequential modeling.

Specifically, we incorporate a *temporal context gating* $f_{\text{TCG}}(\alpha^t, \mathbf{h}^{t-1})$ using the temporal information $\mathbf{h}^{t-1}$ on a given input α^t at t as follows:

$$\begin{aligned} f_{\text{TCG}}(\alpha^t, \mathbf{h}^{t-1}) &= \alpha^t \otimes cgate(\mathbf{h}^{t-1}) \quad \text{(forward)} \\ f^{-1}_{\text{TCG}}(\alpha^t, \mathbf{h}^{t-1}) &= \alpha^t \oslash cgate(\mathbf{h}^{t-1}) \quad \text{(inverse)} \end{aligned} \tag{9}$$

where $cgate(\mathbf{h}^{t-1})$ can be any learnable function/network with a sigmoid function at the end. This is analogous to the context gating [30] in video analysis which scales the input α^t (since $cgate(\mathbf{h}^{t-1}) \in (0, 1)$) based on useful context, which in our setup is the temporal information $\mathbf{h}^{t-1}$.

Preserving the Jacobian structure. In the context of $|J_{\mathbf{v}}|$ computation in Eq. (8), we perform $f_{\text{TCG}}(\mathbf{u}_1, \mathbf{h}^{t-1}) = \mathbf{u}_1 \otimes cgate(\mathbf{h}^{t-1})$ (w.l.o.g., we omit t for $\mathbf{u}$ and $\mathbf{v}$). Importantly, we observe that this 'auxiliary' variable $\mathbf{h}^{t-1}$ could safely be used to transform $\mathbf{u}_1$ *without* altering the triangular structure of the Jacobian matrix for the following two reasons: (1) we still perform an element-wise operation $\mathbf{u}_1 \otimes cgate(\mathbf{h}^{t-1})$ resulting a diagonal submatrix for $\partial \mathbf{v}_1 / \partial \mathbf{u}_1$, and (2) $\partial \mathbf{v}_1 / \partial \mathbf{u}_2$ is still 0 since $\mathbf{u}_2$ is not involved in $f_{\text{TCG}}(\mathbf{u}_1, \mathbf{h}^{t-1})$. Thus, we now have

$$|J_{\mathbf{v}}| = \begin{vmatrix} \frac{\partial \mathbf{v}_1}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{v}_1}{\partial \mathbf{u}_2} \\ \frac{\partial \mathbf{v}_2}{\partial \mathbf{u}_1} & \frac{\partial \mathbf{v}_2}{\partial \mathbf{u}_2} \end{vmatrix} = \begin{vmatrix} diag(cgate(\mathbf{h}^{t-1})) & 0 \\ \frac{\partial \mathbf{v}_2}{\partial \mathbf{u}_1} & diag(\exp s(\mathbf{u}_1)) \end{vmatrix} \tag{10}$$

where $|J_{\mathbf{v}}| = [\prod_j cgate(\mathbf{h}^{t-1})_j] * [\exp(\sum_i (s(\mathbf{u}_1))_i)]$.

As seen in Fig. 3, we place f_{TCG} to transform the "bypassing" split (non-transforming partition) of each *layer* of a block (i.e., the "bypassing" partition $\mathbf{u}_2^t$ gets transformed by f_{TCG_2}). We specifically chose a gating mechanism for conservative adjustments so that the original information is preserved to a large degree through simple but learnable 'weighting'. The full forward and inverse steps involving f_{TCG} can easily be formulated by following Eq. (6) and Eq. (7) while respecting the order of operations seen in Fig. 3. See appendix for details.

3.3. How do we use CRow?

In essence, CRow aims to model an invertible mapping $[\{\mathbf{y}^t\}, \{\mathbf{z}^t\}] = f(\{\mathbf{x}^t\})$ between sequential/longitudinal measures $\{\mathbf{x}^t\}$ and their corresponding observations $\{\mathbf{y}^t\}$ with $\{\mathbf{z}^t\}$ encoding the latent information across $t = 1, \ldots, T$ time points. Once we train $f(\cdot)$, we can perform the following exemplary tasks:

(1) Conditional sequence generation: Given a series of observations of interest $\{\mathbf{y}^t\}$, we can sample $\{\mathbf{z}^t\}$ (each independently from a standard normal distribution) to generate $\{\mathbf{x}^t\} = f^{-1}([\{\mathbf{y}^t\}, \{\mathbf{z}^t\}])$. The advantage comes from how $\{\mathbf{y}^t\}$ can be flexibly constructed (either seen or unseen from the data) such as an arbitrary disease progression over time (see Fig. 1). Then, we randomly generate corresponding measures $\{\mathbf{x}^t\}$ to observe the corresponding longitudinal measures for both quantitative and qualitative analyses. Since the model is recurrent, the sequence length can be extended beyond the training data to model future trajectory.

A potential direction would be to use the generated sequences to directly enable common data analysis procedures (i.e., statistical analysis on synthetic data) and help evaluate scientific hypotheses.

(2) Sequential density estimation: Conversely, given $\{\mathbf{x}^t\}$, we can predict $\{\mathbf{y}^t\}$, and more importantly, estimate the density $p_{\mathbf{X}}(\{\mathbf{x}^t\})$ at each t. When $\{\mathbf{x}^t\}$ is generated from $\{\mathbf{y}^t\}$, the estimated density can indicate the 'integrity' of the generated sample (i.e., low $p_{\mathbf{X}}$ implies that the sequence is perhaps less common with respect to $\{\mathbf{y}^t\}$).

4. Experiments

We validate our framework in both a qualitative and quantitative manner with two sets of experiments: (1) two image sequence datasets and (2) a neuroimaging study.

4.1. Conditional Moving MNIST Generation

Moving Digit MNIST: We first test our model on a controlled Moving Digit MNIST dataset [39] of image sequences showing a hand-written digit from 0 to 9 moving in a path and bouncing off the boundary (see supplement for animations). This experiment qualitatively shows that the images in a generated sequence with specific conditions (i.e., image labels) are consistent across the sequence. Here, we specifically chose two digits (e.g., 0 and 1) to construct ~13K controlled sequences of frame length $T = 6$ where each frame of a sequence is an image of size 20 by 20 (vectorized as $\mathbf{x}^t \in \mathbb{R}^{400}$) and has a one-hot vector $\mathbf{y}^t \in \mathbb{R}^2$ of digit label at t indicating one of the two possible digits. We found this intuitive and interpretable assessment before experimenting with arguably less interpretable datasets (i.e., neuroimaging data we show later).

Training. Our model consists of three coupling blocks, each block shown in Fig. 3, where each subnetwork q contains one GRU cell and three layers of residual fully connected networks with ReLU activation. For each TCG (f_{TCG} in Fig. 3, Eq. (9)), the network $cgate(\cdot)$ is a single fully connected network with sigmoid activation. Each input frame $\mathbf{u}^t = \mathbf{x}^t$ is split into two halves $\mathbf{u}_1$ and $\mathbf{u}_2$. Models were trained on $T = 6$ time points, but further time points data can be generated since our model is recurrent. Each training sequence has a digit label sequence $\{\mathbf{y}^t\}$ for $t = 1, \ldots, 6$ where all $\mathbf{y}^t$ are "identical" in each sequence since the the same digit is shown throughout the sequence.

Generation. Now, we want to generate sequences show-

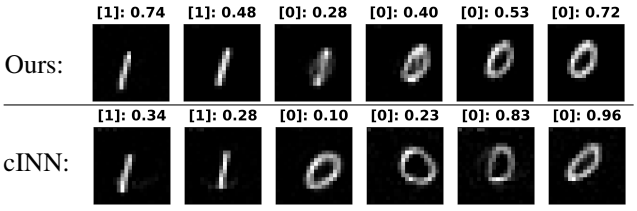


Figure 4: Examples of generated sequences given the changing condition 1→1→0→0→0→0 (top of each frame, **[digit label]: density**). Ours shows smooth transition while cINN shows temporally drastic transition.

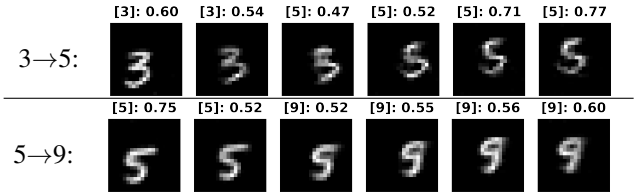


Figure 5: Examples of generated sequences using CRow.

ing digits gradually transform (e.g., changing from 1 to 0). We first specified sequential conditions (i.e., digit label) that change midway through the sequence (e.g., $\{\mathbf{y}^t\}$ sequence indicating digit labels 1→1→0→0→0→0). Then, we generated the corresponding sequences $\{\mathbf{x}^t\}$ and visually check if the changes across the frames look natural. Note that we trained *only* the image sequences with consistent digit labels. One demonstrative result is shown in Fig. 4 where we compare the generated image sequences with condition (i.e., digit label) changing from 1 to 0. Our result at the top of Fig. 4 shows gradual transition while cINN result does not show such temporally smooth and consistent behavior.

Density estimation. Our model quantifies its output confidence in the form of density (i.e., likelihood) shown at the top of each generated images in Fig. 4. Not only our model adjusts generation based on inputs, but it also outputs lower density at the frame showing the most drastic transformation as such patterns were not observed during the training, i.e., the likelihood decreases when then condition changes and then increase as the sequence goes. This means that our model simultaneously shows the conditional generation ability and estimates outputs' relative density given the training data seen. Different from other generative models, it allows conditional generation on sequential data while maintaining exact and efficient density estimation. More examples are shown in Fig. 5 (and appendix).

Moving Fashion MNIST: We also tested our model on a more challenging dataset called Moving Fashion MNIST [43] of moving apparel image sequences. The image sizes. frames lengths, and moving paths are identical to those of Moving Digit MNIST. An important difference is that they are *real images* of 10 types of apparels (i.e., T-shirt, Bag, etc. see supplement for the full list) instead of hand-written digits. The same models and training setups were used to generate the transforming sequences in a similar manner. In Fig. 6, we show the examples of various apparels successfully transforming to other types while moving. Compared to Moving Digit MNIST, capturing the smooth transformations of these apparel images are more challenging as the apparel shapes vary more in terms of shapes and sizes.

4.2. Longitudinal Neuroimaging Analysis

In this neuroimaging experiment, we evaluate if our conditionally generated samples actually exhibit statistically robust and clinically sound characteristics when trained with a longitudinal Alzheimer's disease (AD) brain imaging dataset. We generated a *sufficient* number of longitudinal

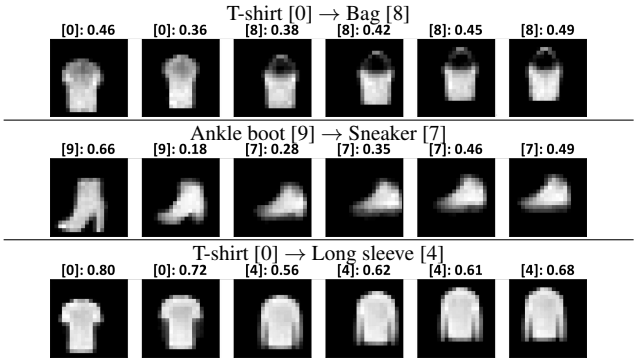


Figure 6: Examples of generated Moving Fashion MNIST sequences using CRow (apparel type [label index]). More examples are in the supplement.

brain imaging measures (i.e., $\{\mathbf{x}^t\}$) conditioned on various covariates (i.e., labels $\{\mathbf{y}^t\}$) associated with AD progression (e.g., memory). Thus, the generated brain imaging sequences should show the pathology progression consistent with the covariate progression (see Fig. 1 and Fig. 7 for illustrations). We then performed a statistical group analysis (i.e., healthy vs. disease progressions) to detect disease related features from the imaging measures. In the end, we expected that the brain regions of interests (ROIs) identified by the statistical group analysis are consistent with other AD literature with statistically stronger signal (i.e., lower p-value) than the results using the original training data.

Dataset. The Alzheimer's Disease Neuroimaging Initiative (ADNI) database (`adni.loni.usc.edu`) is one of the largest and still growing neuroimaging databases. Originated from ADNI, we use a longitudinal neuroimaging dataset called The Alzheimer's Disease Prediction of Longitudinal Evolution (TADPOLE) [29]. We used data from N=276 participants with $T = 3$ time points.

Input. For the longitudinal brain imaging sequence $\{\mathbf{x}^t\}$, we chose Florbetapir (AV45) Positron Emission Tomography (PET) scan measuring the level of *amyloid-beta* deposited in brain which has been a known type of pathology associated with Alzheimer's disease [42, 22]. The AV45 images were registered to a common brain template (MNI152) to derive the gray matter regions of interests (82 Desikan atlas ROIs [8], see appendix). Thus, each of the 82 ROIs ($\mathbf{x}^t \in \mathbb{R}^{82}$) holds an average Standard Uptake Value Ratio (SUVR) measure of AV45 where high AV45 implies more amyloid pathology in that region.

Condition. For the corresponding labels $\{\mathbf{y}^t\}$ for longitudinal conditions, we chose five covariates known to be tied to AD progression (normal to impaired range in square brackets): **(1)** *Diagnosis*: Normal/Control (CN), Mild Cognitive Impairment (MCI), and Alzheimer's Disease (AD) [CN→MCI→AD]. **(2)** *ADAS13*: Alzheimer's Disease Assessment Scale [0→85]. **(3)** *MMSE:* Mini Mental State Exam [0→30]. **(4)** *RAVLT-I:* Rey Auditory Verbal Learning Test - Immediate [0→75]. **(5)** *CDR:* Clinical Dementia Rating [0→18]. These assessments impose disease *progression*

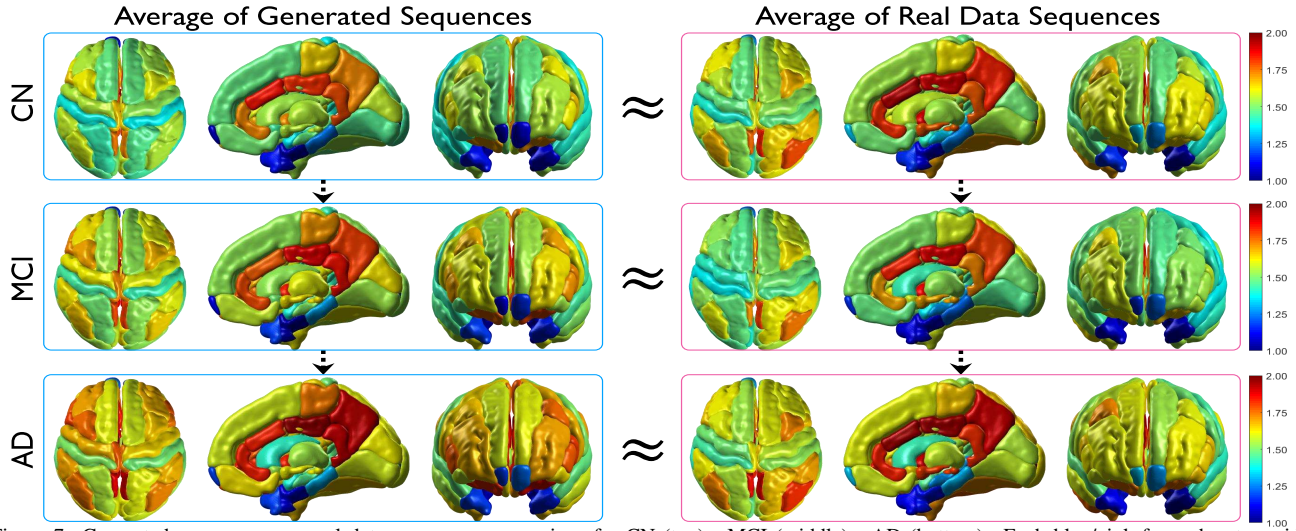


Figure 7: Generated sequences vs. real data sequences comparison for CN (top)→MCI (middle)→AD (bottom). Each blue/pink frame has top, side (interior of right hemisphere), and front views. **Left (blue frames):** The average of the 100 generated sequences conditioned on CN→MCI→AD. **Right (pink frames):** The average of the real samples with CN→MCI→AD in the dataset. Red/blue indicate high/low AV45. ROIs are expected to turn more red as CN→MCI→AD. The generated samples show magnitudes and sequential patterns similar (≈) to those of the real samples from the training data.

of samples. See supplement and [29] for details.

Analysis. We performed a statistical group analysis on each condition $\{\mathbf{y}^t\}$ independently with the following pipeline: *(1) Training:* First, we trained our model (the same subnetwork as Sec. 4.1) using the sequences of SUVR in 82 ROIs for $\{\mathbf{x}^t\}$ and the covariate ('label') sequences for $\{\mathbf{y}^t\}$. *(2) Conditional longitudinal sample generation:* Then, we generated longitudinal samples $\{\hat{\mathbf{x}}^t\}$ conditioned on two distinct longitudinal conditions: *Control* (healthy covariate sequence) versus *Progression* (worsening covariate sequence). Specifically, for each condition (e.g., Diagnosis), we generate N_1 samples of Control (e.g., $\{\hat{\mathbf{x}}_1^t\}_{i=1}^{N_1}$ conditioned on $\{\mathbf{y}^t\}_=$ CN→CN→CN) and N_2 samples of Progression ($\{\hat{\mathbf{x}}_2^t\}_{i=1}^{N_2}$ conditioned on $\{\mathbf{y}^t\}$ = CN→MCI→AD). Then, we perform a two sample t-test at $t = 3$ for each of 82 ROIs between $\{\hat{\mathbf{x}}_1^3\}_{i=1}^{N_1}$ and $\{\hat{\mathbf{x}}_2^3\}_{i=1}^{N_2}$ groups, and derive p-values to tell whether the pathology levels between the groups significantly differ in those ROIs.

Result 1: Control vs. Progression (Table 1, Top row block). We set the longitudinal conditions for each covariate based on its associated to healthy progression (e.g., low ADAS13 throughout) and disease progression (e.g., high ADAS13 related to eventual AD onset). We generated $N_1 = 100$ and $N_2 = 100$ samples for each group respectively. Then, we performed the above statistical group difference analysis under 4 setups: **(1)** Raw training data, **(2)** cINN [4], **(3)** Our model, and **(4)** Our model + TCG. With the raw data, the sample sizes of the desirable longitudinal conditions were extremely small for all setups, so no statistical significance was found after type-I error control. With cINN, we occasionally found few significant ROIs, but the non-sequential samples with only $t = 3$ could not generate realistic samples. With CRow we consistently found significant ROIs and detected the most number of ROIs (the ROIs for Diagnosis shown in Fig. 8) including many AD-specific regions reported in the aging literature such as hippocampus and amygdala [20, 22] (see appendix for the full list).

Result 2: Control vs. Early-progression (Table 1, Bottom row block). We setup a more challenging task where we generate samples which resemble the subjects that show slower progression of the disease (i.e., lower rate of covariate change over time). This case is especially important in

	# of Statistically Significant ROIs (# of ROIs after type-I error correction)				
Covariates	Diagnosis	ADAS13	MMSE	RAVLT-I	CDR-SB
Control	CN→CN→CN	10→10→10	30→30→30	70→70→70	0→0→0
Progression	CN→MCI→AD	10→20→30	30→26→22	70→50→30	0→5→10
cINN (N_1=100 / N_2= 100)	11 (4)	5 (2)	5 (0)	3 (0)	7 (0)
Ours (N_1=100 / N_2= 100)	25 (11)	24 (12)	19 (**2**)	15 (**2**)	18 (7)
Ours + TCG (N_1=100 / N_2= 100)	**28** (**12**)	**32** (**14**)	**31** (**2**)	**19** (**2**)	**25** (**9**)
Control	CN→CN→CN	10→10→10	30→30→30	70→70→70	0→0→0
Early-progression	CN→MCI→MCI	10→13→16	30→28→26	70→60→50	0→2→4
cINN (N_1=150 / N_2= 150)	2 (0)	2 (2)	2 (0)	0 (0)	1 (0)
Ours (N_1=150 / N_2= 150)	**6** (2)	6 (4)	11 (**4**)	**5** (**1**)	2 (0)
Ours + TCG (N_1=150 / N_2= 150)	**6** (**4**)	**8** (**5**)	**12** (**4**)	**5** (**1**)	**5** (**1**)

Table 1: Number of ROIs identified by statistical group analysis using the generated measures with respect to various covariates associated with AD at significance level $\alpha = 0.01$ (type-I error controlled result shown in parenthesis). Each column denotes sequences of disease progression represented by diagnosis/test scores. In all cases, using *CRow* with TCG yielded the most number of statistically significant ROIs.

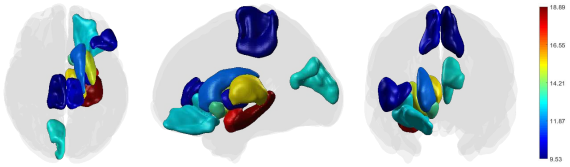


Figure 8: 12 Significant ROIs found between two Diagnosis groups (CN→CN→CN vs. CN→MCI→AD) at $t = 3$ using our model under 'Diagnosis' in Table 1. The colors denote the -log p-value. AD-related ROIs such as hippocampus, putamen, caudate, and amygdala are included.

AD when early detection leads to effective prevention. With $N_1 = 100$ and $N_2 = 100$ samples, no significant ROIs were found in all models. To improve the sensitivity, we generated $N_1 = 150$ and $N_2 = 150$ samples in all models and found several significant ROIs *only* with CRow related to an *early* AD progression such as hippocampus [13, 21, 24, 17] (full list in the appendix).

Statistical advantages. By generating realistic samples with CRow, we achieve the following advantages: **(1)** Increasing sample size makes the hypothesis test more sensitive and robust – rejecting the null when it is indeed false – leading to a lower type-II error. **(2)** Also, we do *not* simply detect spurious significant ROIs because (i) we control for type-I error via the *most conservative* Bonferroni multiple testing correction, and (ii) we additionally improve the statistical power of detecting the *true effects* (i.e., significant ROIs) that *at least* need to be detected with the raw data only. In Table 2, we show that the significant ROIs identified with the real data only are also detected through our framework with *improved* p-values from the Control vs. Progression experiment. These results on the generated data suggest that one can utilize CRow in a statistically meaningful manner *without neglecting the true signals* from important AD-specific ROIs [13, 32]. Note that the scientific validity of our findings requires further investigation on additional real data. These preliminary results, however, point to the promise of using such models to partly mitigate problems related to recruiting large number of participants for statistically identifying weak disease effects.

Generation assessments. In Fig. 7, we see the generated samples (Left) through CN→MCI→AD in three views of the ROIs and compare them to the real training samples (Right). We observe that the generated samples have similar AV45 loads through the ROIs, and more importantly, the progression pattern across the ROIs (i.e., ROIs turning more red indicating amyloid accumulation) follows that of the real sequence as well. We also quantified the similarities between the generated and real data sequences by computing effect size (Cohen's d [7]) which measures the difference between the two distributions (Table 3) showing that CRow generates the most realistic sequences.

	ROI	p-value	
		Real	CRow
Diagnosis	Left Amygdala	5.51E-03	**1.18E-06**
	Left Putamen	7.38E-03	**3.99E-05**
ADAS13	Left Inferior Temporal	3.34E-03	**7.93E-04**
	Left Middle Temporal	6.83E-03	**2.02E-03**
MMSE	Left Superior Parietal	7.13E-03	**1.52E-05**
	Left Supramarginal	6.75E-03	**8.20E-08**
RAVLT-I	Left Paracentral	9.16E-03	**8.09E-05**
CDR-SB	Left Hippocampus	4.01E-03	**3.36E-06**

Table 2: p-values in ROIs improve (get lower) with the sequences generated by CRow with increased sample size over using real sequence data.

Scientific remarks. Throughout our analyses, the significant ROIs found such as amygdala, putamen, temporal regions, hippocampus (e.g., shown in Fig. 8) and many others reported as AD-specific regions in the aging literature [13, 20, 21, 32, 41, 25]. This implies that the generated longitudinal sequences could resemble the underlying distribution of the real data which we may not be available with large enough sample sizes. The appendix includes additional details on the scientific interpretation of the results.

5. Conclusion

We design generative models for longitudinal datasets that can be modulated by secondary conditional variables. Our architecture is based on an invertible neural network that incorporates recurrent subnetworks and temporal context gating to pass information within a sequence generation, the network seeks to "learn" the conditional distribution of training data in a latent space and generate a sequence of samples whose longitudinal behavior can be modulated based on given conditions. We demonstrate experimental results using three datasets (2 moving videos, 1 neuroimaging) to evaluate longitudinal progression in sequentially generated samples. In neuroimaging problems which suffer from small sample sizes, our model can generate realistic samples which is promising.

Acknowledgments

Research supported by NIH (R01AG040396, R01EB022883, R01AG062336, R01AG059312), UW CPCP (U54AI117924), UW CIBM (T15LM007359) NSF CAREER Award (1252725), USDOT Research and Innovative Technology Administration (69A3551747134), and UTA Research Enhancement Program (REP).

	Cohen's d of Generated vs. Real of Progressions					Cohen's d of Generated vs. Real of Early-progressions				
Covariates	Diagnosis	ADAS13	MMSE	RAVLT-I	CDR-SB	Diagnosis	ADAS13	MMSE	RAVLT-I	CDR-SB
cINN	1.2551	1.5968	1.1498	1.8948	1.5516	1.0656	1.4985	0.9482	1.8435	1.4541
Ours	0.4193	0.5562	0.3485	0.7112	0.6456	0.3591	0.5612	0.2953	0.6133	0.6254
Ours + TCG	**0.2828**	**0.3915**	**0.1679**	**0.5889**	**0.3775**	**0.2341**	**0.5248**	**0.0902**	**0.5448**	**0.4998**

Table 3: Difference between the generated sequences and the real sequences at $t = 3$. Lower the effect size (Cohen's d), smaller the difference between the comparing distributions. In all settings, *CRow* with TCG generates the most realistic sequences with the smallest effect sizes.

References

[1] M Ehsan Abbasnejad, Anthony Dick, and Anton van den Hengel. Infinite variational autoencoder for semi-supervised learning. In *CVPR*, 2017. 2

[2] Solange Akselrod, David Gordon, F Andrew Ubel, et al. Power spectrum analysis of heart rate fluctuation: a quantitative probe of beat-to-beat cardiovascular control. *Science*, 213(4504):220–222, 1981. 1

[3] Gene E Alexander, Kewei Chen, Pietro Pietrini, et al. Longitudinal PET evaluation of cerebral metabolic decline in dementia: a potential outcome measure in Alzheimer's disease treatment studies. *American Journal of Psychiatry*, 159(5):738–745, 2002. 2

[4] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, et al. Analyzing Inverse Problems with Invertible Neural Networks. In *ICLR*, 2019. 2, 3, 7

[5] AD Baddeley, S Bressi, Sergio Della Sala, Robert Logie, and H Spinnler. The decline of working memory in Alzheimer's disease: A longitudinal study. *Brain*, 114(6):2521–2542, 1991. 2

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 4

[7] Jacob Cohen. *Statistical power analysis for the behavioral sciences*. Routledge, 2013. 8

[8] Rahul S Desikan, Florent Ségonne, Bruce Fischl, et al. An automated labeling system for subdividing the human cerebral cortex on mri scans into gyral based regions of interest. *NeuroImage*, 31(3):968–980, 2006. 6

[9] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 2, 3

[10] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016. 2, 3

[11] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015. 3

[12] Meire Fortunato, Charles Blundell, and Oriol Vinyals. Bayesian recurrent neural networks. *arXiv preprint arXiv:1704.02798*, 2017. 2

[13] NC Fox, EK Warrington, PA Freeborough, P Hartikainen, AM Kennedy, JM Stevens, and Martin N Rossor. Presymptomatic hippocampal atrophy in Alzheimer's disease: A longitudinal MRI study. *Brain*, 119(6):2001–2007, 1996. 8

[14] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with Bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015. 2

[15] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 2

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 4

[17] Seong Jae Hwang, Nagesh Adluru, Won Hwa Kim, Sterling C Johnson, Barbara B Bendlin, and Vikas Singh. Associations Between Positron Emission Tomography Amyloid Pathology and Diffusion Tensor Imaging Brain Connectivity in Pre-Clinical Alzheimer's Disease. *Brain connectivity*, 9. 8

[18] Seong Jae Hwang, Ronak Mehta, Hyunwoo J. Kim, Sterling C. Johnson, and Vikas Singh. Sampling-free Uncertainty Estimation in Gated Recurrent Units with Applications to Normative Modeling in Neuroimaging. In *UAI*, page 296, 2019. 2

[19] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *CVPR*, 2017. 2

[20] Kunlin Jin, Alyson L Peel, Xiao Ou Mao, Lin Xie, Barbara A Cottrell, David C Henshall, and David A Greenberg. Increased hippocampal neurogenesis in Alzheimer's disease. *Proceedings of the National Academy of Sciences*, 101(1):343–347, 2004. 7, 8

[21] Sterling C Johnson, Bradley T Christian, Ozioma C Okonkwo, et al. Amyloid burden and neural function in people at risk for Alzheimer's disease. *Neurobiology of aging*, 35(3):576–584, 2014. 8

[22] Abhinay D Joshi, Michael J Pontecorvo, Chrisopher M Clark, et al. Performance characteristics of amyloid PET with florbetapir F 18 in patients with Alzheimer's disease and cognitively normal subjects. *Journal of Nuclear Medicine*, 53(3):378–384, 2012. 6, 7

[23] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NIPS*, 2017. 2

[24] Won Hwa Kim, Annie M Racine, Nagesh Adluru, Seong Jae Hwang, et al. Cerebrospinal fluid biomarkers of neurofibrillary tangles and synaptic dysfunction are associated with longitudinal decline in white matter connectivity: A multi-resolution graph analysis. *NeuroImage: Clinical*, 21, 2019. 8

[25] Won Hwa Kim, Vikas Singh, Moo K Chung, et al. Multi-resolutional shape features via non-Euclidean wavelets: Applications to statistical analysis of cortical thickness. *NeuroImage*, 93:107–123, 2014. 8

[26] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018. 1, 4

[27] Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *NIPS*, 2015. 2

[28] Ramon Landin-Romero, Fiona Kumfor, Cristian E Leyton, Muireann Irish, John R Hodges, and Olivier Piguet. Disease-specific patterns of cortical and subcortical degeneration in a longitudinal study of Alzheimer's disease and behavioural-variant frontotemporal dementia. *NeuroImage*, 151:72–80, 2017. 2

[29] Razvan V Marinescu, Neil P Oxtoby, Alexandra L Young, et al. TADPOLE Challenge: Prediction of Longitudinal Evolution in Alzheimer's Disease. *arXiv preprint arXiv:1805.03909*, 2018. 6, 7

[30] Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with Context Gating for video classification. *arXiv preprint arXiv:1706.06905*, 2017. 5

[31] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[32] Rik Ossenkoppele, Marissa D Zwan, Nelleke Tolboom, et al. Amyloid burden and metabolic function in early-onset Alzheimer's disease: parietal lobe involvement. *Brain*, 135(7):2115–2125, 2012. 8

[33] George Papamakarios and Iain Murray. Fast ε-free inference of simulation models with Bayesian conditional density estimation. In *NIPS*, 2016. 2

[34] Rajesh Ranganath, Linpeng Tang, Laurent Charlin, and David Blei. Deep exponential families. In *AISTATS*, 2015. 2

[35] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015. 2

[36] Oren Rippel and Ryan Prescott Adams. High-dimensional probability estimation with deep density models. *arXiv preprint arXiv:1302.5125*, 2013. 2

[37] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Annual Conference of the International Speech Communication Association*, 2014. 4

[38] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *NIPS*, 2015. 2

[39] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using LSTMs. In *ICML*, 2015. 5

[40] Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*, 2015. 4

[41] Victor L Villemagne, Samantha Burnham, Pierrick Bourgeat, Belinda Brown, Kathryn A Ellis, Olivier Salvado, Cassandra Szoeke, S Lance Macaulay, Ralph Martins, Paul Maruff, et al. Amyloid β deposition, neurodegeneration, and cognitive decline in sporadic Alzheimer's disease: a prospective cohort study. *The Lancet Neurology*, 12(4):357–367, 2013. 8

[42] Dean F Wong, Paul B Rosenberg, Yun Zhou, et al. In vivo imaging of Amyloid deposition in Alzheimer's disease using the novel radioligand [18F] AV-45 (Florbetapir F 18). *Journal of nuclear medicine*, 51(6):913, 2010. 6

[43] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 6

[44] Hao Zhou, Vamsi K Ithapu, Sathya Narayanan Ravi, Grace Wahba, Sterling C. Johnson, and Vikas Singh. Hypothesis testing in unsupervised domain adaptation with applications in Alzheimer's disease. In *NIPS*, pages 2496–2504, 2016. 3