
Identifiable VAEs for Neural Latent Discovery

Zijing Wu

Department of Computational Science and Engineering
Georgia Institute of Technology
zwu381@gatech.edu

Anqi Wu

Department of Computational Science and Engineering
Georgia Institute of Technology
anqiwu@gatech.edu

Abstract

Advances in neural recording enable us to record hundreds of neurons simultaneously, placing an increasing demand of developing appropriate computational statistical tools to analyze high-dimensional neural data. In neural latent analysis, where the latent representation of the raw neural data is extracted to understand the internal state of the biological neural network, variational auto-encoder (VAE) and its variations are emerging as a set of promising approaches. Recently, pi-VAE (Zhou, 2020) was proposed to address the identifiability issue of applying Latent Variable Models to neural spike data. However, pi-VAE's dependency on external label information and failure to meet identifiability requirements due to its discrete observation model have limited its applicability. In this paper, we present a novel method called dynamical identifiable VAE (di-VAE) to address these limitations. Our approach proposes a continuous relaxation of the multivariate Poisson distribution, eliminating the need for external labels and introducing temporal dynamics through the use of history latent information as the conditional variable u . We systematically explore different di-VAE configurations using a rat hippocampus place cell dataset and further validate our chosen implementation on the Neural Latents Benchmark datasets. The results demonstrate competitive qualitative and quantitative performance.

1 Introduction

Advances in neural recording enable us to record hundreds of neurons simultaneously (Stevenson and Kording (2011)), which places an increasing demand for developing appropriate computational statistical tools to analyze the neural data. These neural datasets are usually high-dimensional, which means the data are large matrices, and at each time point it is a data vector consisting of dozens or hundreds of elements. The high-dimensionality makes it difficult to interpret the data as we cannot visualize them without meaningfully projecting the data into a 2D or 3D space. The low-dimensional projection, which is unobserved, is called the latent representation of the original data.

In many brain areas, it has been found that the activity of large populations of neurons can often be accounted for by the dynamics of a small number of latent factors. Thus, we may think that the latent representation represents the internal state of the biological neural network and would yield insights into brain computation if the true latent representation is known.

In neural latent discovery, where the latent representation of the raw neural data is extracted via computational statistical methods, variational auto-encoder (VAE) (Kingma and Welling (2013)) and

its variations are emerging as a set of promising approaches. Variational auto-encoder is a Bayesian generative model. It has a decoder which decodes, or say, generates the original data from a set of unobserved latent factors, while it also has an encoder which encodes the original data and gets the factorized latent distribution, from which we obtain the sampled latent with variational inference. The optimization of the model is by maximizing the evidence lower bound (ELBO) of the model Kingma and Welling (2013).

Aiming to make scientific discovery, we want to ensure the model can theoretically learn the true latent of the neural population activity when the model is fully optimized. This desired property is called the identifiability of the model Bengio et al. (2013). However, Khemakhem et al. (2020a) proves that any nonlinear generative models, such as VAE, is essentially not identifiable without a prior conditioned by some additional variables. Inspiringly, the results in Khemakhem et al. (2020a) help define identifiable VAE and its sufficient conditions. Building upon this, later work introduced different VAE models that are identifiable. (Khemakhem et al. (2020a), Khemakhem et al. (2020b), Zhou and Wei (2020), Mita et al. (2021)). But these models all use external labels as the conditioning variable for the latent prior. Neither do they consider conditioning the latent prior with the previous latent, which should be a promising approach given the richness of information contained in the previous latent. As there is no guarantee the external labels are available together with the neural recording in an experiment, it calls for the need of models with less or no supervision and ability to achieve comparable or better performance compared to previous supervised methods.

To address the need of identifiable VAE with pure unsupervision, we first proposed dynamical identifiable VAE, a generative model trying to combine two separate lines of work - the identifiable VAE (Khemakhem et al. (2020a), Khemakhem et al. (2020b), Zhou and Wei (2020), Mita et al. (2021)), which tries to address the identifiability issue of VAE models, and dynamical VAE Girin et al. (2020), a unified framework of VAE variations incorporate temporal dependencies in the model latent states and the input data sequence. We systematically explored different implementations of dynamical identifiable VAE with a rat hippocampus place cell dataset.

Further experimentation with pi-VAE indicates its non-identifiability. Theoretically, pi-VAE requires a specific proof of identifiability when the conditional variable u is discrete, prompting a new and vital research direction to resolve this issue. In response, we introduced the IGR-iVAE model, which is identifiable and assumes a continuous noise model, thus avoiding these theoretical constraints. Identifiability in this context extends to the generative process, including the observational model. A fixed observational model can restrict the model's applicability to diverse data types. According to Khemakhem et al. (2020a), the observation model must align with the ground truth, underscoring the significance of a flexible observation noise distribution.

Our models can benefit science and people by providing a more accurate and efficient method for analyzing complex, time-dependent datasets. In their usual implementations, VAEs are not identifiable, making it difficult to accurately analyze and interpret the data. An Di-VAE, on the other hand, can produce a unique latent representation for each input, allowing for more accurate analysis and interpretation of the data. This can be particularly useful in fields such as biology and medicine, where analyzing time-dependent data is crucial for understanding dynamic systems and improving patient care. Furthermore, an Di-VAE can be trained on large-scale datasets, making it a scalable and efficient tool for analyzing complex data. In this thesis, we propose a novel approach for training an Di-VAE and demonstrate its effectiveness on a range of real-world datasets.

In terms of benefits of the model, our proposed models could benefit neural science and the field of brain-computer interfaces by providing a more accurate way to analyze neural activity data. Traditional VAEs are not identifiable, making it difficult to accurately interpret the latent representations they produce. This can limit their usefulness in the study of complex neural activity patterns. IGR-iVAE, on the other hand, can produce unique latent representations for each input, allowing for more accurate analysis and interpretation of neural data. Furthermore, it can incorporate implementation in Di-VAE so that it could also be applied to settings without experiment labels, thus, can operate without supervision on demand. This can be particularly useful in the field of brain-computer interfaces. The models could also help to advance our understanding of the brain and its complex dynamics, leading to potential advancement in the field of neuroscience.

2 Literature Review and Background

With the fast development in machine learning Richards et al. (2019) and the growing ability to record many neurons simultaneously Stevenson and Kording (2011), computational analysis of neural population activity is becoming one of the key methods in modern neuroscience. From empirical studies, we learn that the neurons do not fire independently but coordinate with each other Ebitz and Hayden (2021). This high-dimensional neural population activity can be nonlinearly projected onto a low-dimensional latent space that characterizes the internal state of the neural activity. To model the probabilistic distribution of the latent variable, a good baseline could be established with variational auto-encoder (VAE) Kingma and Welling (2013), which consists of an encoder and a decoder. It first encodes the input data into a latent probabilistic distribution then tries to reconstruct the input data by decoding the sampled latent. One advantage of VAE is its power to capture the covariation of the neural population activity given that the encoder could be parametrized with an artificial neural network with sufficiently large parameter space. But the latent from a vanilla VAE's latent space is often entangled, hindering meaningful interpretation. (Higgins et al. (2017), Zhou and Wei (2020), Liu et al. (2021)) Besides interpretability, VAE could also suffer from lack of identifiability, a term borrowed from representation learning Bengio et al. (2013) describes the model property that demands a one-to-one correspondence between parameter space and output space. Recently, Khemakhem et al. (2020a) provides the first rigorous proof of the identifiability theorem for a broad set of deep generative models including VAE. The theorem grants the model identifiability only under several assumptions that can be easily satisfied. In short, Identifiability ensures the model encodes the true latent when it is optimally trained. This is important for yielding scientific insights based on the discovered neural latent.

Based on the identifiability theorem, there has been an ongoing line of research striving to address the issue of interpretability and identifiability with a new set of models coined as identifiable VAEs (Khemakhem et al. (2020a), Khemakhem et al. (2020b), Mita et al. (2021)). The results of these models demonstrate enhanced ability in learning disentangled latent representation in both synthetic and real world datasets, thus inspiring promising application in neural latent analysis.

2.1 Identifiability

Definition 1. Suppose a statistical model is parameterized by Θ . Then we say the model is identifiable if there exists a one-to-one mapping $\Theta \rightarrow \mathcal{P}_\Theta$.

Identifiability (of parameters Θ) is a desired property for correctly estimating the true model parameter θ^* that generates the ground true distribution. Take 1D Gaussian for example (for which μ, σ are identifiable actually), it would intuitively feels troublesome that different pairs of (μ, σ) can parameterize the same Gaussian. Practically, without identifiability, it becomes impossible to use MLE to estimate the θ^* . Thus, identifiability of model parameters is a necessary condition for MLE of the ground true model parameters at the least.

Note that for a real life dataset, we never truly have access to the true underlying distribution p_{θ^*} . But thanks to the equivalence between [[maximum likelihood estimation and minimizing the KL divergence]], we are able to approximate it through maximum likelihood estimation, and by the identifiability definition, approximating the true model parameters θ^* .

2.2 Identifiability for deep LVM

Deep latent variable model A deep latent variable model is given by

$$p_\theta(x, z) = p_\theta(x|z)p_\theta(z)$$

where θ is the model parameters, $p_\theta(x|z)$ is a distribution often parameterized by a neural network called the decoder, and $p_\theta(z)$ is a prior distribution over the latent variables z . We are able to get the marginal of x by

$$p_\theta(x) = \int p_\theta(x|z)dz$$

Putting VAE in the context of deep LVM and statistical model The generative model of VAE (variational autoencoder) can be seen as a subclass of deep LVM, which thus assumes the form of

$$p_\theta(x, z) = p_\theta(x|z)p_\theta(z)$$

To perform inference, traditionally one has to solve

$$p_{\theta}(z|x) = \frac{p_{\theta}(x, z)}{p(x)}$$

which is intractable due to the intractability of $p_{\theta}(x, z)$. That's when VAE's variational inference comes to help, which we will not discuss here.

Note that when we see it as a statistical model $(p_{\theta}(x), \theta)$, the input z can be seen as a parameter in θ , which we can inferenced. Thus, during inference, we assign $p(z)$ a *prior*, usually Gaussian; given observed data x , we will calculate $p_{\phi}(z|x)$, the *posterior*, where ϕ is usually parameterized by neural networks in a VAE. In the model loss function, we use the forward Kullback-Leibler divergence to minimize the difference between the posterior and prior.

3 Di-VAE - iVAE with Latent Dynamics

Notations We denote $\mathbf{x} \in \mathbb{R}^n$ as the observation. In our experimental applications, \mathbf{x} would be the spike counts within a small time span, recorded from n channels. We then denote $\mathbf{z} \in \mathbb{R}^m$ ($m \ll n$) as the underlying unobserved latent variables for \mathbf{x} .

We denote $\mathbf{l} \in \mathbb{R}^d$ as the concatenated vector of external labels, which are task variables recorded along with the neural recording. Also, we denote $\mathbf{v} \in \mathbb{R}^{2mp}$ as the internal state random variables. In our case, \mathbf{v} consists of the inferred parameters of the multivariate normal distribution of the latent at the previous p time points, which have $2mp$ counts totally. We also call both \mathbf{l} and \mathbf{v} the *input prior* as they are both treated as the given variables for obtaining the conditioned latent prior $p(\mathbf{z}|\mathbf{l})$ and $p(\mathbf{z}|\mathbf{v})$. To unify the notation when we describe the model, we denote \mathbf{u} as the *input prior*, and hence, we can also denote the conditioned latent prior with $p(\mathbf{z}|\mathbf{u})$

Model modes Adapting the terming from DVAE Girin et al. (2020), we call the model is in *ex-driven mode* if only external input is used, *in-driven mode* if only the history latent is used, or *mix-driven mode* if a certain combination of the external input and history latent is taken. The differences among the three modes lie at 1) what we use as the *input prior* and 2) how we combine the *input prior* to obtain the latent prior distribution, which is simply an isometric standard Gaussian in a vanilla VAE. In its *ex-driven mode*, the RNN cell is exactly the original pi-VAE. In the other two modes, the model becomes recurrent when we introduce previous latent states to inference at the current time point. When we unroll the inference model, at each time step, the RNN cell is a modified pi-VAE.

3.1 Generative model

Unlike VAE, which uses an isometric standard Gaussian as the latent prior distribution, our model uses the conditioned latent prior $p_{\mathbf{T}, \lambda}(\mathbf{z}|\mathbf{u})$. The generative model thus can be formulated as

$$p_{\theta}(\mathbf{x}, \mathbf{z}|\mathbf{u}) = p_{\theta}(\mathbf{x}|\mathbf{z})p_{\mathbf{T}, \lambda}(\mathbf{z}|\mathbf{u}), \quad (1)$$

where $p_{\mathbf{T}, \lambda}(\mathbf{z}|\mathbf{u})$, is assumed to be conditionally independent, where each element $\mathbf{z}_i \in \mathbf{z}$ has an exponential family distribution given \mathbf{u} , as described in (Khemakhem, 2020),

$$p_{\mathbf{T}, \lambda}(\mathbf{z}|\mathbf{u}) = \prod_{i=1}^m p(\mathbf{z}_i|\mathbf{u}) \prod_{i=1}^m \frac{Q_i(\mathbf{z}_i)}{Z_i(\mathbf{u})} \exp \left[\sum_{j=1}^k T_{i,j}(\mathbf{z}_i) \lambda_{i,j}(\mathbf{u}) \right], \quad (2)$$

where Q_i is the base measure, $\mathbf{T}_i = (T_{i,1} \dots, T_{i,k})$ are the sufficient statistics, $Z_i(\mathbf{u})$ is the normalizing factor, $\lambda_i = (\lambda_{i,1}(\mathbf{u}), \dots, \lambda_{i,k}(\mathbf{u}))$ are the natural parameters, and k is the pre-defined number of sufficient statistics.

3.2 Inference algorithm

The inference is a modification of VAE (Kingma & Welling, 2013), The inference algorithm simultaneously learns the deep generative model and the approximate posterior $q(\mathbf{z}|\mathbf{x}, \mathbf{u})$ by maximizing $\mathcal{L}(\boldsymbol{\theta}, \phi)$,

$$\log p(\mathbf{x}|\mathbf{u}) \geq \mathcal{L}(\boldsymbol{\theta}, \phi) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x}, \mathbf{u})}[\log(p(\mathbf{x}, \mathbf{z}|\mathbf{u}) - q(\mathbf{z}|\mathbf{x}, \mathbf{u}))] \quad (3)$$

We decompose the approximate posterior as the product of the two conditional distributions.

$$q(\mathbf{z}|\mathbf{x}, \mathbf{u}) \propto q_\phi(\mathbf{z}|\mathbf{x})p_{\mathbf{T},\lambda}(\mathbf{z}|\mathbf{u}), \quad (4)$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ is assumed to be conditionally independent exponential family distribution, i.e. $q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{i=1}^m q(\mathbf{z}_i|\mathbf{x})$, and is parameterized by a neural network, while $p_{\mathbf{T},\lambda}(\mathbf{z}|\mathbf{u})$ is defined in equation 2.

We modeled both $q_\phi(\mathbf{z}|\mathbf{x})$, $p_{\mathbf{T},\lambda}(\mathbf{z}|\mathbf{u})$ as independent Gaussian distribution. Then

$$\mathbf{z}_t|\mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_{x,t}, \text{diag}(\boldsymbol{\sigma}_{x,t})), \text{ where } [\boldsymbol{\mu}_{x,t}, \text{diag}(\boldsymbol{\sigma}_{x,t})] = \varphi_x(x_t), \quad (5)$$

where $\boldsymbol{\mu}_{x,t}$, $\text{diag}(\boldsymbol{\sigma}_{x,t})$ denotes the parameters of the prior distribution conditioned by x_t , φ_x can be any highly flexible function such as a neural network. In practice, we substitute the variance with log variance for numerical stability. For $q_\phi(\mathbf{z}_t|\mathbf{x}_t)$, we use two separate but equally initialized neural networks to inference the mean and variance respectively.

Ex-driven mode (pi-VAE) If the model is in ex-driven mode, $p_{\mathbf{T},\lambda}(\mathbf{z}|\mathbf{u})$ is approximated by a latent distribution conditioned by a set of external labels, which are fed as the *input prior*,

$$\mathbf{z}_t|\mathbf{u}_t = \mathbf{z}_t|\mathbf{l}_t \sim \mathcal{N}(\boldsymbol{\mu}_{u,t}, \text{diag}(\boldsymbol{\sigma}_{u,t})), \text{ where } [\boldsymbol{\mu}_{u,t}, \text{diag}(\boldsymbol{\sigma}_{u,t})] = [\boldsymbol{\mu}_{l,t}, \text{diag}(\boldsymbol{\sigma}_{l,t})] = \varphi_u(\mathbf{l}_t) \quad (6)$$

where $\boldsymbol{\mu}_{*t}$, $\text{diag}(\boldsymbol{\sigma}_{*t})$ denotes the parameters of the prior distribution conditioned by $*_t$, φ_u can be any highly flexible function such as a neural network.

In-driven mode (implementation 3) If the model is in in-driven mode, $p_{\mathbf{T},\lambda}(\mathbf{z}|\mathbf{u})$ is approximated by a latent distribution conditioned by the history latent, which is fed as the *input prior*,

$$\begin{aligned} \mathbf{z}_t|\mathbf{u}_t = \mathbf{z}_t|\mathbf{v}_t &\sim \mathcal{N}(\boldsymbol{\mu}_{u,t}, \text{diag}(\boldsymbol{\sigma}_{u,t})), \text{ where} \\ [\boldsymbol{\mu}_{u,t}, \text{diag}(\boldsymbol{\sigma}_{u,t})] &= [\boldsymbol{\mu}_{v,t}, \text{diag}(\boldsymbol{\sigma}_{v,t})] = \varphi_u(\text{param}(q(\mathbf{z}_{t-p}|\mathbf{x}_{t-p}\mathbf{v}_{t-p}), \dots \\ & q(\mathbf{z}_{t-2}|\mathbf{x}_{t-2}\mathbf{v}_{t-2}), q(\mathbf{z}_{t-1}|\mathbf{x}_{t-1}\mathbf{v}_{t-1}))), \end{aligned} \quad (7)$$

where $\boldsymbol{\mu}_{*t}$, $\text{diag}(\boldsymbol{\sigma}_{*t})$ denotes the parameters of the prior distribution conditioned by $*_t$, φ_u can be any highly flexible function such as a neural network, and $\text{param}(\cdot)$ denotes all the parameters of the conditional multivariate Gaussian distributions.

Mix-driven mode (Ex. combine time and history latent) If the model is in mix-driven mode, $p_{\mathbf{T},\lambda}(\mathbf{z}|\mathbf{u})$ is approximated by a latent distribution conditioned by both the external labels and the history latent. Specifically, $u_t = [\mathbf{l}_t, \mathbf{v}_t]$ and

$$\mathbf{z}_t|\mathbf{u}_t = \mathbf{z}_t|\mathbf{l}_t, \mathbf{v}_t \sim \mathcal{N}(\boldsymbol{\mu}_{u,t}, \text{diag}(\boldsymbol{\sigma}_{u,t})). \quad (8)$$

We have different schemes for obtaining $\boldsymbol{\mu}_{u,t}$, $\text{diag}(\boldsymbol{\sigma}_{u,t})$. At first, we tried

$$\boldsymbol{\mu}_{u,t} = \boldsymbol{\mu}_{l,t} + \boldsymbol{\mu}_{v,t} \quad (9)$$

$$\boldsymbol{\sigma}_{u,t} = \boldsymbol{\sigma}_{l,t} + \boldsymbol{\sigma}_{v,t}, \quad (10)$$

where $[\boldsymbol{\mu}_{l,t}, \boldsymbol{\sigma}_{l,t}]$ and $[\boldsymbol{\mu}_{v,t}, \boldsymbol{\sigma}_{v,t}]$ are obtained in the same way respectively as in the two previous discussed modes.

For future exploration, we could try more statistical intuitive combination method such as calculating the conditional prior as a product of distributions,

$$q(\mathbf{z}_t|\mathbf{u}_t) \propto q(\mathbf{z}_t|\mathbf{l}_t)q(\mathbf{z}_t|\mathbf{v}_t), \quad (11)$$

where $[\boldsymbol{\mu}_{l,t}, \boldsymbol{\sigma}_{l,t}]$ and $[\boldsymbol{\mu}_{v,t}, \boldsymbol{\sigma}_{v,t}]$ are obtained in the same way respectively as in the two previous discussed modes.

4 IGR-iVAE: iVAE with Continuous Relaxation of Data Distribution

Let N be the number of neurons recorded in the high-dimensional data, K be the number of discrete categories in the discrete distribution to be relaxed into a continuous distribution. Given a discrete distribution over $S = \{1, 2, \dots, K\}$, we use one-hot encoding of length K to represent the discrete distribution. Then S can be interpreted as the vertices of the $(K - 1)$ simplex, $\Delta^{(K-1)} = \{x \in \mathbb{R}^{K-1} : x_k > 0 \text{ and } \sum_{k=1}^K x_k = 1\}$. Note that it is equivalent to $S^{(K-1)} = \{x \in \mathbb{R}^{K-1} : x_k > 0 \text{ and } \sum_{k=1}^{K-1} x_k < 1\}$, since the last coordinate can be inferred from the other coordinates by $x_K = 1 - \sum_{k=1}^{K-1} x_k$. It is more desirable to place the continuous relaxation over $S^{(K-1)}$ since it allows us to correctly calculate the Jacobians of Eq. 28. (See Appendix of Potapczynski et al. (2020) for more details.)

In order to construct an injective mapping $f(z) = x$, the mapped random variable x needs to be continuous given that latent variable z is continuous. Otherwise, the injectivity of f would be hard to satisfied. One way to solve the problem is to find a continuous relaxation for the discrete Poisson distribution which is used as the observation model.

$$\text{Poisson}(\lambda) \stackrel{d}{\approx} \text{IGR}(\tau) \quad (12)$$

where $\lambda \in (0, \infty)^N$, and τ is the temperature hyperparameter for IGR.

4.1 Generative model of IGR-iVAE

4.1.1 Notations

Let \mathbf{x} be the data which is discretely distributed, x be the one-hot encoding of the data.

4.2 The decoder architecture

In our model, the decoder mapping function is a composite function $\mathbf{g} \circ f$ which maps from the latent space to the (one-hot encoded) data space. The function f is implemented by normalizing flow (GIN decoder as in Zhou and Wei (2020)). The function \mathbf{g} is defined as

$$\mathbf{g}(y) = \begin{cases} g_1(\vec{y}_1) \\ g_2(\vec{y}_1) \\ \dots \\ g_N(\vec{y}_N) \end{cases} \quad (13)$$

where $\vec{y}_i \in \mathbb{R}^{K-1}$ and g_i is an invertible function that maps from \mathbb{R}^{K-1} to S^{K-1} of our choice (softmax₊₊). Hence $\mathbf{g} : \mathbb{R}^{N(K-1)} \rightarrow (S^{K-1})^N$ is also invertible.

4.3 Derive ELBO

The inference of VAE is done by optimizing $\mathcal{L}(\theta, \phi)$, the evidence lower bound (ELBO), which in its canonical form is written as:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi(z|x)}[\log p(x|z)] - \mathbb{D}_{KL}(q_\phi(z|x)||p(z)) \quad (14)$$

In pi-vae (Zhou and Wei (2020)), the model employs an additional variable u to condition the prior in order to fulfill some of the conditions of the identifiability theorem (Khemakhem et al. (2020a)). The ELBO becomes:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi(z|x,u)}[\log p_f(x|z)] - \mathbb{D}_{KL}(q_\phi(z|x,u)||p_{T,\lambda}(z|u)) \quad (15)$$

in which they obtain $q_\phi(z|x,u)$, named as the "full posterior" by them, by calculating the product of the two conditioned Gaussian pdfs,

$$q_\phi(z|x,u) \propto q_\phi(z|x)p_{T,\lambda}(z|u) \quad (16)$$

(The above should go to a background section, maybe also with the identifiability theorem)

Our model (igr-ivae)'s ELBO can assume the same formulation as pi-vae's and by taking the negative of the ELBO, the loss function is defined as

$$-\mathcal{L}(\theta, \phi) = \underbrace{-\mathbb{E}_{q_\phi(z|x,u)}[\log p_\theta(x|z)]}_{\text{NLL loss}} + \underbrace{\mathbb{D}_{KL}(q_\phi(z|x,u)||p_{T,\lambda}(z|u))}_{\text{KL loss}} \quad (17)$$

The KL loss is to be calculated as the sum of the KL divergence of each z^i from the full posterior to the prior. To derive an analytical form of the log density of x , we employ the fact that $x = g(y)$ is an invertible mapping and use the change of variable theorem,

$$p_\theta(x) = p_{\theta-\{\tau\}}(g^{-1}(x))|\det \mathcal{J}_{g^{-1}}| \quad (18)$$

$$= p_{\theta-\{\tau\}}(y)|\det \mathcal{J}_{g^{-1}}| \quad (19)$$

$$= p_{\theta-\{\tau\}}(y)|\det \mathcal{J}_g|^{-1} \quad (20)$$

which gives us

$$\log p_\theta(x) = \log p_\theta(y) - \log |\det \mathcal{J}_g| \quad (21)$$

The first term is the log likelihood of Gaussian given data $g^{-1}(x)$, and the second term is the log determinant of the Jacobian of g . With the inverse g^{-1} and the log Jacobian derived in the Supplement (Eq. 29 and Eq. 44), The log density then becomes

$$\log p_\theta(x) = -\frac{1}{2} \sum_{i=1}^{K-1} \left(\frac{y_i - \mu_i}{\sigma_i} \right)^2 + \frac{1}{2} \log \sigma_i^2 \quad (22)$$

$$+ (K-1) \log \tau - \sum_{i=1}^{K-1} \log x_i - \log \left(1 - \sum_{j=1}^{K-1} x_j \right) \quad (23)$$

Note that the third term is a constant as K and τ are two hyperparameters. In practice, if we keep the rest of the log Jacobian terms, the 0^+ values in x_i for $i \neq j$ where $x_j = 1^-$ would cause numerical issues. Properly handling them is crucial to the success of optimization.

We exploit the fact only the first $K-1$ dimensions of x are considered in calculation and the very one-hot encoding nature of x .

The quantity $\sum_{i=1}^{K-1} x_i$ is the same for all $\mathbf{x} < K-1$ and only differs when $\mathbf{x} = K$. Thus, we can set $K = \max(\{\mathbf{x} + 2 : \mathbf{x} \in \mathbb{D}\})$ (\mathbb{D} is the dataset in count). With this, we ensures that no sample $\mathbf{x} = K$ is drawn during optimization and can address the numerical problem by removing the then-constant third and fourth terms during optimization (For example, if the maximum spike count is 5, then including the zero spike category, the first 6 categories would have non-zero numbers of samples. Now if we set $K = 5 + 2 = 7$, having the last category as an additional one with no actual sample is in this category. then for all samples x in the dataset, the sum $\sum_{i=1}^{K-1} x_i$ is the same for some constant)

Thus, the final derivation of NLL loss thus equivalently assumes the following form,

$$-\mathbb{E}_{q_\phi(z|x,u)}[\log p_\theta(x|z)] = \mathbb{E}_{q_\phi(z|x,u)} \left[\sum_{j=1}^N \frac{1}{2\sigma_{x_j}^2} \sum_{i=1}^{K-1} (g^{-1}(x_j) - \mu_{ji})^2 \right] \quad (24)$$

$$= \mathbb{E}_{q_\phi(z|x,u)} \left[\sum_{j=1}^N \frac{1}{2\sigma_{x_j}^2} \sum_{i=1}^{K-1} (y_{ji} - f(z)_{ji})^2 \right] \quad (25)$$

where σ_{x_j} are constant hyperparameters depending x (TODO: think about why this common practice when optimizing Gaussian NLL is okay), and instead of inverse mapping, to prevent numerical issues, the data y is obtained through an offline optimization of $\text{MSE}(g(y), x)$.

5 Experiments on Rat hippocampus CA1 data

Currently, the project has two main objectives: 1) to evaluate the performance of Di-VAE on the rat hippocampus CA1 data and 2) to evaluate the performance of IGR-iVAE on the rat hippocampus CA1 data. We will detail the results of the experiments in this section. These are the two model implementation ideas we have tested. Note that the implementations are orthogonal to each other and can be potentially combined to form a more powerful model. The experiments with a combined model will be carried out in the future directions of this work.

5.1 Results with Di-VAE

To get the reconstruction assessment of the models, we computed the peristimulus time histogram (PSTH) of a thresholded model output firing rate and the ground truth. So we have 8 result PSTH's, which are for the 7 models of consideration (zhou vae, vanilla vae, zhou pivae, pivae indriven, pivae exdriven by time, pivae exdriven by time, location, and direction, and pivae mixdriven with time) and the ground truth. We will detail how we 1) thresholded the model output and 2) compute the PSTH shortly later. Then for each model, we took the RMSE of the model PSTH and the ground truth's. We averaged the distance across channels and obtained the quantitative evaluation of the model.

Since the model outputs are real numbers instead of integers that represent the firing rate, we need a method to threshold the outputs to get the spikes. We proposed a cutoff determination procedure - output firing rates larger than the cutoff would be 1 spike/second or 0 otherwise. We used the percentage of 1 spike counts in ground truth as the cutoff ratio. We approximated the percentage using the average of the data. The cutoff for each model is $100 * \text{the cutoff ratio percentile of the output}$. The cutoff ratio of the rat data is 0.044. The cutoffs range from 0.198 to 0.298.

Due to the neural firing mechanism of place cells, we will use a modified version of PSTH to represent the tuning curves of the neurons. This x axis of this modified PSTH is the equally divided location bins along the 1.6m track the rat traversed, while the y axis is the firing rate in unit spike/second. If we consider one location bin for the PSTH of one channel, at each trial, we divide this spike count by the time traversed to get the spike rate. We took the average across trials to get the PSTH.

We compare the channel-wise averaged distances between the model PSTH and the ground truth among the models in Figure 2. The unsupervised models (indriven pivae and mixdriven pivae) have equal or even better performance compared with other models, which uses external labels. We further compare the latent representations for all the models tested. The results are shown in Table 1.

5.2 Results with IGR-iVAE

We also tested the IGR-iVAE model on the rat hippocampus CA1 data. We plotted the latent representations of IGR-iVAE run on the Rat hippocampus CA1 with four random seeds. The results are shown in Fig. 3. The latent representations are well separated between two different directions and evolve smoothly along location shift. The performance is consistent in all four runs with different random seeds. The experiments also requires quantitative evaluations to further investigate IGR-iVAE's performance. There is an ongoing effort and the results will be demonstrated in the final manuscript of this work.

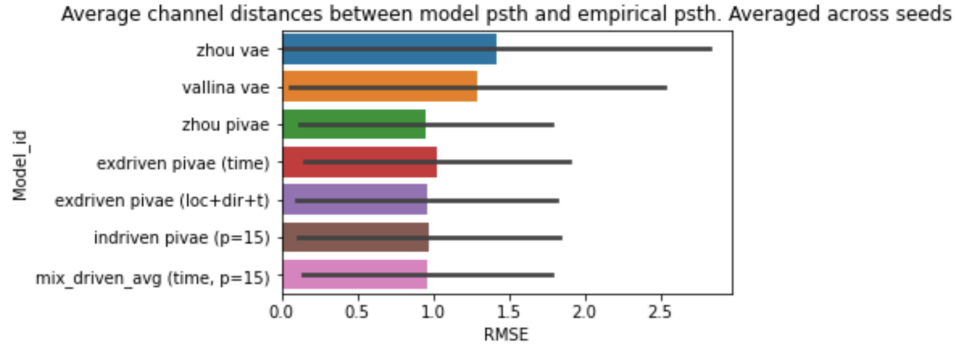


Figure 1

vae zhou, 2020	vallina vae	pi-vae zhou, 2020	pi-vae ex-driven with location+time+direction	pi-vae ex-driven with time	pi-vae in-driven (p=15)	pivae mix-driven with time (p=15)

Table 1

6 Discussion

The research project started with the idea of incorporating history latents as the input conditioning variable u in the pi-VAE model framework, achieving unsupervision in learning latent dynamics in this way. Based on the hypothesis that it would leverage the structural information in the latent dynamics to maintain its performance even without any supervision from experiment labels as u , we implemented the Di-VAE model. Specifically, the latent of pi-VAE does not assume any dynamics. Our model (Di-VAE) incorporates latent dynamics by inputting history latent as the conditioning variable u . This contribution orthogonally adds on the first and leads to better latent learning and downstream decoding performance.

For pi-VAE, further experiments suggests that it is actually non-identifiable. On the theoretical end, as we pointed out, it assumes a bespoke proof for identifiability in the case of the conditional variable u is discrete. Thus, it spawned a new and important research direction to address the identifiability issue in the pi-VAE model. Hence, we proposed a new model, IGR-iVAE, which is identifiable. Our model (IGR-iVAE) assumes a continuous noise model and thus free of such theoretical difficulties. Identifiability is defined up to a generative process, which includes the observational model. If the observational model is fixed, it limits the applicability of the identifiable model to different data. Also note that in Khemakhem et al. (2020a), it says the observation model has to be the same as the ground truth. Thus, having a flexible observation noise distribution is important.

This thesis documents the progress of the research project, which is still ongoing. Future experiments will be conducted to fully evaluate the quantitative performance of the IGR-iVAE models. Furthermore, we will explore the potential of combining the Di-VAE and IGR-iVAE models to form a better iVAE with or without supervised labels. Qualitative and quantitative analysis will also be applied to other datasets to further analyse the models' performance.

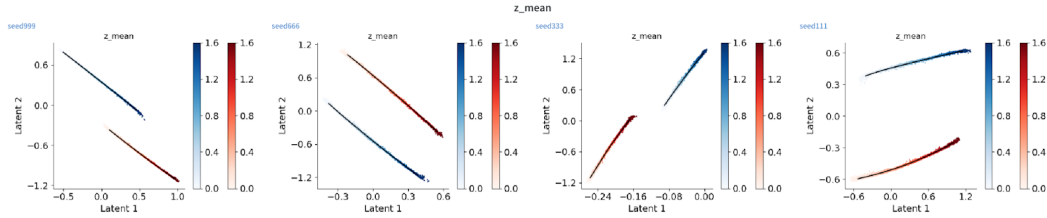


Figure 3

References

- Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.
- Dinh, L., Krueger, D., and Bengio, Y. (2014). Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*.
- Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*.
- Ebitz, R. B. and Hayden, B. Y. (2021). The population doctrine in cognitive neuroscience. *Neuron*, 109(19):3055–3068.
- Girin, L., Leglaive, S., Bie, X., Diard, J., Hueber, T., and Alameda-Pineda, X. (2020). Dynamical variational autoencoders: A comprehensive review. *arXiv preprint arXiv:2008.12595*.
- Higgins, I., Matthey, L., Pal, A., Burgess, C. P., Glorot, X., Botvinick, M. M., Mohamed, S., and Lerchner, A. (2017). beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3.
- Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. (2020a). Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR.
- Khemakhem, I., Monti, R., Kingma, D., and Hyvarinen, A. (2020b). Ice-beem: Identifiable conditional energy-based deep models based on nonlinear ica. *Advances in Neural Information Processing Systems*, 33:12768–12778.
- Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- Liu, R., Azabou, M., Dabagia, M., Lin, C.-H., Gheshlaghi Azar, M., Hengen, K., Valko, M., and Dyer, E. (2021). Drop, swap, and generate: A self-supervised approach for generating neural activity. *Advances in neural information processing systems*, 34:10587–10599.
- Mita, G., Filippone, M., and Michiardi, P. (2021). An identifiable double vae for disentangled representations. In *International Conference on Machine Learning*, pages 7769–7779. PMLR.
- Potapczynski, A., Loaiza-Ganem, G., and Cunningham, J. P. (2020). Invertible gaussian reparameterization: Revisiting the gumbel-softmax. *Advances in Neural Information Processing Systems*, 33:12311–12321.
- Richards, B. A., Lillicrap, T. P., Beaudoin, P., Bengio, Y., Bogacz, R., Christensen, A., Clopath, C., Costa, R. P., de Berker, A., Ganguli, S., et al. (2019). A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770.
- Sorrenson, P., Rother, C., and Köthe, U. (2020). Disentanglement by nonlinear ica with general incompressible-flow networks (gin). *arXiv preprint arXiv:2001.04872*.
- Stevenson, I. H. and Kording, K. P. (2011). How advances in neural recording affect data analysis. *Nature neuroscience*, 14(2):139–142.

Zhou, D. and Wei, X.-X. (2020). Learning identifiable and interpretable latent models of high-dimensional neural activity using pi-vae. *Advances in Neural Information Processing Systems*, 33:7234–7247.

7 Supplementary Material

7.1 Closed-form expression of likelihood

7.1.1 Base distribution p_z

We have the multidimensional relaxation scheme. Let $l \leq N$ be the latent dimensionality, $z \in \mathbb{R}^l$ be the latent variable, and $p \in \mathbb{R}^{N-l}$ be the zero padding. Note that z are independent Gaussians, so the base distribution p_z can be seen as a product of independent Gaussians embedded in \mathbb{R}^N . So we can write down the density of p_z as

$$p_z(z) = \prod_{i=1}^l p_{z_i}(z_i) \prod_{i=l+1}^N \mathbb{I}(z_i = 0) \quad (26)$$

where \mathbb{I} is the indicator function. Then the log density of p_z is

$$\log p_z(z) = \sum_{i=1}^l \log p_{z_i}(z_i) + \sum_{i=l+1}^N \log \mathbb{I}(z_i = 0) \quad (27)$$

7.1.2 Derive g^{-1} and \mathcal{J}_g when g is softmax $_{++}$

Proposition 1. Let $g : \mathbb{R}^{K-1} \rightarrow S^{K-1}$ be the softmax $_{++}$ function given by Eq. 15 in Potapczynski et al. (2020):

$$s_i = g(y_i, \tau)_i = \frac{\exp(y_i/\tau)}{\sum_{k=1}^{K-1} \exp(y_k/\tau) + \delta}, \quad (28)$$

where $\tau \in \mathbb{R}^+$ is the temperature and $\delta > 0$. Let g_i, y_i be the i -th element of the input and output vectors. Then g is invertible and the inverse $g^{-1} : S^{K-1} \rightarrow \mathbb{R}^{K-1}$ is given by:

$$y_i = \tau(\ln s_i + \ln \delta - \ln(1 - C)), \quad C = \sum_i^{K-1} s_i \quad (29)$$

Proof. To prove the invertibility of g , we need to show that g is one-to-one and onto. It is clear to see that g is onto based on the fact that exponential function is a monotonically increasing function defined on \mathbb{R} and its range is \mathbb{R}^+ . So it suffices to show that g is one-to-one, which is to show that for any element $s \in S^{K-1}$, there exists a unique $y \in \mathbb{R}^K$ such that $g(y) = s$.

Let $\tau = 1$ so that $y_i/\tau = y_i$. This is for avoiding clustering in the proof and the result is still valid for any $\tau > 0$. So we have

$$s_i = \frac{\exp(y_i)}{\sum_{k=1}^{K-1} \exp(y_k) + \delta} \quad (30)$$

$$\ln s_i = \ln y_i - \ln \left(\sum_{k=1}^{K-1} \exp(y_k) + \delta \right) \quad (31)$$

$$\sum_{k=1}^{K-1} \exp(y_k) + \delta = \exp(y_i - \ln s_i) \quad (32)$$

$$\sum_{k=1}^{K-1} \exp(y_k) + \delta = \frac{\exp(y_i)}{s_i} \quad (33)$$

$$s_i \sum_{k=1}^{K-1} \exp(y_k) + s_i \delta = \exp(y_i) \quad (34)$$

Summing Eq. 34 over i gives

$$\sum_{i=1}^{K-1} s_i \sum_{k=1}^{K-1} \exp(y_k) + \delta \sum_{k=1}^{K-1} s_k = \sum_{i=1}^{K-1} \exp(y_i) \quad (35)$$

$$\sum_{k=1}^{K-1} \exp(y_k) \left(-1 + \sum_{k=1}^{K-1} s_k \right) + \delta \sum_{k=1}^{K-1} s_k = 0 \quad (36)$$

$$\sum_{k=1}^{K-1} \exp(y_k) = \frac{\delta \sum_{k=1}^{K-1} s_k}{1 - \sum_{k=1}^{K-1} s_k} \quad (37)$$

$$(38)$$

We can substitute the LHS of Eq. 37 into Eq. 30 and get

$$\exp(y_i) = s_i \left(\frac{\delta \sum_{k=1}^{K-1} s_k}{1 - \sum_{k=1}^{K-1} s_k} + \delta \right) \quad (39)$$

$$y_i = \ln s_i + \ln \left(\frac{\delta C}{1 - C} + \delta \right), \quad C = \sum_{k=1}^{K-1} s_k \quad (40)$$

$$y_i = \ln s_i + \ln \left(\frac{\delta}{1 - C} \right) \quad (41)$$

$$y_i = \ln s_i + \ln \delta - \ln(1 - C) \quad (42)$$

Substitute in $y_i = y_i/\tau$ and we get Eq. 29. Thus, g is one-to-one.

Hence, g is invertible and the inverse g^{-1} is given by Eq. 29.

□

\mathcal{J}_g is given by Eq. 26 in Potapczynski et al. (2020):

$$|\det \mathcal{J}_g(y, \tau)| = \frac{1}{\tau^{K-1}} \left(\prod_{i=1}^{K-1} g_i \right) \left(1 - \sum_{j=1}^{K-1} g_j \right) \quad (43)$$

and

$$\log |\det \mathcal{J}_g(y, \tau)| = -(K-1) \log \tau + \sum_{i=1}^{K-1} \log g_i + \log \left(1 - \sum_{j=1}^{K-1} g_j \right) \quad (44)$$

7.1.3 IGR stacking for multidimensional continuous relaxaiton

We show o implementation of aplying IGR for multidimensional continuous relaxation. So for the i -th random variable to be relaxed, we apply IGR_i with respective IGR parameters λ^i , which is outputed from the normalizing flow.

Let $g_i = \text{IGR}^i$. Then $x_i = g_i(\lambda^i)$. Note that each relaxation is independent to the others. Thus, the multidimensional relaxation g is still invertible.

The Jacobian of g is a matrix with block matrices \mathcal{J}_{g_i} on the diagonal. Hence, the Jacobian determinant of g is given by the product of the Jacobians of each IGR, which can be proved by the cofactor formula in calculating determinant.

7.1.4 Derive f^{-1} and \mathcal{J}_f

The mapping function f is implemented by a General Incompressible-flow Network (GIN) Sorrenson et al. (2020), which shares the flexibility of RealNVP Dinh et al. (2016) with a similar model structure while enjoying the volume-preserving properties of the NICE framework Sorrenson et al. (2020)

GIN is a stack of invertible coupling blocks, each of which consists of 2 coupling layers and a permutation layer. We can denote the coupling layer with f_i . For the convenience of notation, we remove the subscript when defining a coupling layer.

General coupling layer (Notations inherited from Dinh et al. (2014)) Let $x \in \mathbb{R}^D$. I_1, I_2 be a partition of $\llbracket 1, D \rrbracket$ such that $|I_1| = d$. Let x_{I_1}, x_{I_2} be the corresponding split of x and m a function defined on \mathbb{R}^d . let the function defined by a coupling layer be f and g be the coupling law. A coupling layer f is defined as:

$$y_{I_1} = x_{I_1} \quad (45)$$

$$y_{I_2} = g(x_{I_2}, m(x_{I_1})) \quad (46)$$

The inverse of the coupling layer is defined as:

$$x_{I_1} = y_{I_1} \quad (47)$$

$$x_{I_2} = g^{-1}(y_{I_2}, m(y_{I_1})) \quad (48)$$

Suppose $I_1 = \llbracket 1, D \rrbracket, I_2 = \llbracket 1, D \rrbracket$. It is easy to see the Jacobian determinant of the coupling layer is:

$$\frac{\partial y}{\partial x} = \begin{bmatrix} I_d & 0 \\ \frac{\partial y_{I_D}}{x_{I_d}} & \frac{\partial y_{I_D}}{x_{I_D}} \end{bmatrix} \quad (49)$$

Thus, the Jacobian determinant is $\frac{\partial y}{\partial x} = \det\left(\frac{\partial y_{I_D}}{x_{I_D}}\right)$.

Affine coupling layer An example of a coupling layer is the affine coupling layer (Dinh et al. (2014)), which uses the transformation $g(a, b) = a \odot b_1 + b_2, b_1 \neq 0, m : \mathbb{R}^d \rightarrow \mathbb{R}^{D-d} \times \mathbb{R}^{D-d}$ as the coupling law.

In this case, $\frac{\partial y_{I_D}}{x_{I_D}}$ is a diagonal matrix the Jacobian determinant is

$$\mathcal{J}_f = \det\left(\frac{\partial y_{I_D}}{x_{I_D}}\right) = \prod_{i \in I_2} b_i \quad (50)$$

and

$$\log \mathcal{J}_f = \sum_{i \in I_2} \log b_i \quad (51)$$

GIN coupling layer A GIN coupling layer is a special case of affine coupling layers. It inherits all the properties and is defined as:

$$y_{I_1} = x_{I_1} \quad (52)$$

$$y_{I_2} = x_{I_2} \odot \exp(s(x_{I_1})) + t(x_{I_1}), \quad (53)$$

where s, t are parameterized by neural networks. We can see log Jacobian of the coupling layer is the sum of the scaling function $s(x_{I_1})$:

$$\log \mathcal{J}_f = \sum_{i \in I_2} \log \exp(s(x_{I_1}))_i = \sum_{i \in I_2} s(x_{I_1})_i = 1 \quad (54)$$

if we set the last component of s to be the negative sum of $s(x_{I_1-1})$. Note that we can do this since $s(x_{I_1})$ is parameters of the coupling layer.

The inverse of a GIN coupling layer can be instantiated from Eq. 48:

$$x_{I_1} = y_{I_1} \quad (55)$$

$$x_{I_2} = \frac{y_{I_2} - t(x_{I_1})}{\exp(s(x_{I_1}))} \quad (56)$$

Thus, the inverse f^{-1} is given by Eq. 55 and 56 and the log Jacobian of f is the unity.

7.2 Materials

The materials used in this research include a rat hippocampus dataset as well as several software tools. The programming language used in this research is Python, and for typesetting the thesis we use Latex. The research also relies on Pytorch, Tensorflow, Numpy, and Matplotlib for numerical computations and data visualization. The PACE computing cluster was used for running the experiments and Visual Studio Code was used as the code editor.

Table 1: Materials used in Research

Material	Description
Dataset	Rat hippocampus dataset
Software	Python, Latex
	Pytorch, Tensorflow, Numpy, Matplotlib
	PACE computing cluster
	Visual Studio Code