Probabilistic Deep Models for Satellite-Driven Crop Type Classification

Xin Cai

Contents

1 Research Aims & Objectives

Variability in Satellite Image Time Series (SITS) due to various reasons, especially the inherent volatility influenced by geographical and climatic/meteorological conditions, causes significant challenges for building crop type classification systems based on SITS. For example, crop classification models based on deep neural networks have difficulty in generalizing to test data collected in geographical regions or in years different from the one used for training [1], consequently largely restricting its applicability. The hypothesis is the lack of effective mechanisms to identify the underlying generative process that is flexible enough to explain the observed variability caused by geographical and temporal changes, which calls for probabilistic models or generative classifiers. Additionally, probabilistic modelling and inference is a long-standing endeavour in the machine learning community and plays a pivotal role in modern pattern recognition. For example, latent variable models and the Expectation-Maximization (EM) algorithm [2] have been the backbone of statistical modelling for decades.

However, there has been little exploration in probabilistic modelling for crop type classification using SITS, not to mention deep probabilistic models [1]. To the best of my knowledge, the existing probabilistic crop classification models still are built upon Probabilistic Graphical Models (PGMs), such as Markov Random Fields [3] or Hidden Markov Models [4, 5]. Therefore, my research attempts to transfer the success of deep generative models in other research areas to SITS analysis, especially for crop type classification. To be more concrete, deep generative models that will be reviewed in Sec. 2.2, including Normalizing Flows (NFs), Variational AutoEncoders (VAEs), and Dynamical Variational AutoEncoders (DVAEs), will be explored for improving classification accuracy and data efficiency compared to the current non-probabilistic deep crop classification models that will be reviewed in Sec. 2.1, where the latter objective, i.e., data efficiency, is a natural byproduct of using generative models as NFs and VAEs have proven to be effective in semi-supervised or unsupervised learning [6–12]. The structure of PhD thesis is outlined in Fig. 3. The three main chapters constituting the methodological part of the thesis will closely revolve around applying the latest advances in NFs, VAEs and DVAEs for achieving the proposed two objectives for crop type classification, where necessary modifications or innovations will be made to accommodate the characteristics of crop phenology reflected through different types of space-borne sensors such as multi-spectral information in Sentinel-2 data.

To sum up, the contributions of my PhD research will be as follows:

- Applying deep generative models, i.e., NFs, VAEs, and DVAEs, to process SITS for improving accuracy in crop type classification;
- Extending the proposed deep generative models to perform semi-supervised learning while maintaining the incurred accuracy drop at an insignificant level, i.e., improving data efficiency for crop type classification;
- Conducting extensive experiments to verify the effectiveness of proposed deep generative models (i.e., ablation studies[2]) on publicly available benchmark datasets (Sec. 3) and make comparisons to the state-of-the-art deep crop classification models.

[1] Despite the potential abuse of terminology, deep probabilistic models and deep generative models will be used interchangeably in the report.

[2] Ablation studies is the terminology in Deep Learning literature, which refers to independently assessing the effectiveness of proposed various components.

2 Literature Review

2.1 SITS-based Crop Analytics

Crop type mapping using Satellite Image Time Series (SITS) has received considerable attention because of its huge potential in crop phenology study, crop yield prediction, facilitating informed decisions on subsidy grants, and food security estimation. With the increasing availability of satellite data, significant research efforts have been devoted to developing automated tools for efficient analysis. Traditionally, manually crafted features, such as vegetation indices [13], combined with machine learning algorithms, such as Random Forest classifier [14] and Support Vector Machine [15], have been the norm for crop classification. Recently, deep learning-based models have dominated the research field by adapting neural architectures developed in other domains, especially in computer vision and natural language processing. The existing approaches generally fall into two categories, 1) pure time-series models, and 2) the combination of spatial and temporal encoders. The former one is focused on capturing temporal dynamics by using sequential neural models, such as Recurrent Neural Networks [16, 17], Temporal Convolution Neural Networks [18, 19], and self-attention models [20]. The latter one argues that extraction of spatial features using neural networks is superior to simple statistics, mainly following the paradigm of obtaining spatial embeddings with spatial encoders and then processing these embeddings sequentially with temporal encoders. Along this line of research, recent work [21] has pointed out that convolutions are not well-suited for extracting spatial features from satellite data for crop classification due to the highly irregular boundaries of parcel fields and limited texture patterns available. Consequently, the researchers have proposed to use Pixel-Set Encoder [21] to obtain learnable statistical descriptors, which is inspired by advances in 3D point cloud processing [22]. The resulted spatial embeddings are then processed by temporal encoders using the self-attention mechanism [23], leveraging its power for capturing long-range dependencies. Furthermore, the lightweight temporal attention module has been proposed in [24] by dividing feature vectors into different groups with specialized attention weights calculated for each group. This module can be considered as a strengthened extension of group convolution, which is the core computing unit of ResNeXt [25].

2.2 Deep Generative Models

In the literature, deep generative models can be roughly divided into two groups: likelihood-based and likelihood-free models, as shown in Fig. 1. Likelihood-free models aim to make generated samples resemble the realistic data as close as possible without explicitly assuming the form of the likelihood function, such as the recently prevalent Generative Adversarial Networks (GANs) [26] which minimize the discrepancy between the generating and the true data distributions through accurately classifying synthetic or realistic samples. The cost of simulation required for parameter learning of likelihood-free models is generally prohibitive. Energy-Based models (EBMs) characterize the data distribution with an energy function, e.g., the Boltzmann/Gibbs distribution, whose configuration forces realistic samples to reside in the low-energy area. The optimization of EBMs relies on Markov Chain Monte Carlo (MCMC) techniques to sidestep the intractable computation of the normalizing constant/partition function, which is known to be computationally demanding due to the long mixing time of MCMC, especially in the high-dimensional space. It should be noted that there has been rapid progress in recent years concerning improving the computational efficiency of either likelihood-free models [27] or EBMs [28]. But these two kinds of approaches are excluded from my current

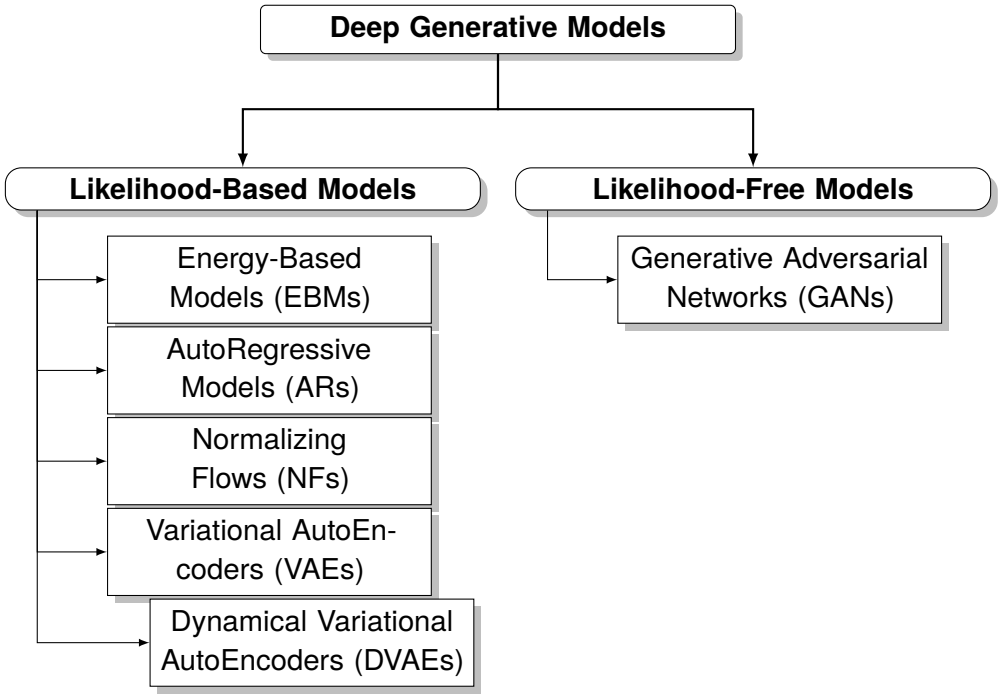

Figure 1: The taxonomy of deep generative models.

stage of research because of the limited energy and time. Instead, my research will be focused on AutoRegressive Models (ARs), Normalizing Flows (NFs), Variational AutoEncoders (VAEs), and its extension to sequential latent variable models, i.e., the Dynamical Variational AutoEncoders (DVAEs), which will be detailed in the following sections. ARs are built upon the simple idea of decomposing the probability density into a series of conditionals by the product rule. The relevant advances mainly concentrate on architectural designs and practical implementations [29–31], and therefore ARs are not reviewed specifically in this report. Also, these methods are intimately connected with each other, and such connections will be discussed where appropriate.

2.2.1 Normalizing Flows

Normalizing Flows (NFs) are a family of likelihood-based generative models, characterized by employing a series of bijective transformations to transform a simple base probability measure to an arbitrarily complicated one. Several merits of NFs include exact log-likelihood evaluation, efficient inference and sampling procedures, and universal representational power in theory, resulting in a considerable research interest in theoretical and practical development of NFs. The central idea of NFs can be formulated as in Eq. (1), which is the well-known change of variables formula:

$$\begin{aligned} \mathbf{x} &= T(\mathbf{u}) \quad where \quad \mathbf{u} \sim p_u(\mathbf{u}) \\ p_x(\mathbf{x}) &= p_u(\mathbf{u}) \left|\det J_T(\mathbf{u})\right|^{-1} \end{aligned} \tag{1}$$

where $\mathbf{x}$ and $\mathbf{u}$ are both D dimensional random vectors, the transformation T must be invertible and differentiable, and $J_T(\mathbf{u})$ denotes Jacobian of the transformation T as shown in Eq. (2):

$$J_T(\mathbf{u}) = \begin{bmatrix} \frac{\partial T_1}{\partial u_1} & \cdots & \frac{\partial T_1}{\partial u_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial T_D}{\partial u_1} & \cdots & \frac{\partial T_D}{\partial u_D} \end{bmatrix} \tag{2}$$

The Jacobian determinant $\det J_T(\mathbf{u})$ quantifies the contracted or expanded volume induced by the transformation T in a small neighborhood of $\mathbf{u}$. More importantly, the chain of invertible and differentiable functions remains invertible and differentiable, which is critical to constructing flexible probability distributions and conforms well to the architectural design principle of deep neural networks. The Jacobian determinant of the composition of a series of invertible and differentiable functions $T_1, T_2, \ldots, T_k$ is characterized by Eq. (3):

$$\begin{aligned} (T_k \circ \cdots \circ T_2 \circ T_1)^{-1} &= T_1^{-1} \circ T_2^{-1} \circ \cdots \circ T_k^{-1} \\ \det J_{T_k \circ \cdots \circ T_2 \circ T_1}(\mathbf{u}) &= \det J_{T_k}(T_{k-1}(\mathbf{u})) \cdots \det J_{T_2}(T_1(\mathbf{u})) \cdot \det J_{T_1}(\mathbf{u}) \end{aligned} \tag{3}$$

The universal representation capability of NFs has been proved in [32, 33], which lays the theoretical foundation of utilizing NFs to model arbitrarily complex probability distributions, e.g., by minimizing some kind of divergence or discrepancy measures as shown in Eq. (4):

$$\begin{aligned} \mathcal{L}(\theta) &= D_{\mathrm{KL}}\left(p_x^*(\mathbf{x}) \| p_x(\mathbf{x};\theta)\right) \\ &= -\mathbb{E}_{p_x^*(\mathbf{x})}\left[\log p_u\left(T^{-1}(\mathbf{x};\phi);\psi\right) + \log|\det J_{T^{-1}}(\mathbf{x};\phi)|\right] + const. \end{aligned} \tag{4}$$

where $p_x^*(\mathbf{x})$ denotes the target distribution and the model $p_x(\mathbf{x};\theta)$ is parameterized by $\theta = \{\phi, \psi\}$, where, in particular, ϕ are the parameters of T and ψ are the parameters of $p_u(\mathbf{u})$.

The practical challenges of constructing NFs are concerned with striking a balance between the expressive power and the efficient computations of forward (T) and inverse (T^{-1}) flows and the associated Jacobian determinants, directly influencing the computational cost of the sampling and density evaluation processes and therefore constituting the central theme of the majority of work in this research field. In particular, the computation complexity of the Jacobian determinant is normally $O\left(D^3\right)$, which is prohibitive for most real-world applications where data is high-dimensional. It is highly desirable to keep the computation complexity of Jacobian determinants growing linearly with the input dimensionality with specifically tailored neural architectures.

Autoregressive Flows The design principle of autoregressive flows is to ensure bijective transformations have a triangular Jacobian, and consequently the log-determinant can be trivially calculated by the sum of the diagonal elements on the log scale. Generally, the autoregressive flows can be described as in Eq. (5), which follows the notational convention in [33, 34]:

$$z_i' = \tau(z_i; \mathbf{h}_i) \quad where \quad \mathbf{h}_i = c_i(\mathbf{z}_{<i}) \tag{5}$$

where τ is termed the (invertible) transformer and c_i denotes an autoregressive conditioner which can be implemented as an arbitrary complex neural net as long as it takes as input with subscripts less than i to ensure a triangular Jacobian, and thus the log-determinant can be readily evaluated according to Eq. (6).

$$\log|\det T_\phi(\mathrm{z})| = \sum_{i=1}^{D} \log\left|\frac{\partial \tau}{\partial z_i}(z_i; \mathrm{h}_i)\right| \tag{6}$$

Research efforts have been devoted to devise expressive transformers and conditioners while keeping their computations efficient or at least tractable. A simple form of the transformer is using the location-scale transformation as shown in Eq. (7), called affine autoregressive flows:

$$\tau(z_i; \mathbf{h}_i) = \sigma_i \cdot z_i + \mu_i \quad where \quad \mathbf{h}_i = \{\sigma_i, \mu_i\} \quad and \quad \sigma_i \neq 0 \tag{7}$$

The glaring weakness of affine autoregressive flows is the restricted expressiveness, which can be compensated, however, by stacking multiple such base flows. Therefore, affine autoregressive flows are popular in the literature, such as NICE [35], RealNVP [36], IAF [37], and MAF [38], because of its simplicity and fast computation. Given the fact that the conic combination and composition of monotonic functions remain monotonic, restricting Multi-Layer Perceptrons (MLPs) to have strictly positive weights and monotonic activation functions therefore can approximate arbitrarily well any monotonic function. Following this principle, Huang et al. [34] put forward deep sigmoidal flows (DSF) and deep dense sigmoidal flows (DDSF) as the instantiation of the transformer, and Ho et al. [39] similarly proposed to use the Cumulative Distribution Function (CDF) of mixture of logistic distributions as the transformer. Monotonic transformers are guaranteed theoretically to be universal density approximators, sharing the same spirit of the inverse transform sampling. However, computation efficiency has been sacrificed for the expressiveness. Specifically, the calculation of the log-determinant of the Jacobian needs more careful treatment as a result of multiple hidden layers in MLPs and inverse functions cannot be solved analytically, meaning that inversion need to be achieved using numerical methods such as bisection search. To remedy the issues caused by no closed-form invertible functions, spline-based transformers have been proposed, consisting of simple spline functions that can be inverted analytically in each segment, such as quadratic splines [40], cubic splines [41], and rational-quadratic splines [42].

Regarding the implementation of the conditioner, as mentioned previously, $c_i(\mathbf{z}_{<i})$ can be implemented as an arbitrary complex neural network provided that the autoregressive structure is respected. In practice, however, two issues need to be taken into account: 1) sharing parameters between different subnetworks $c_i(\mathbf{z}_{<i})$ to save computational cost, and 2) fast computation of forward and/or inverse flows. One common strategy is to use the masking mechanism, giving rise to the so-called masked autoregressive flows. Generally, connections in neural networks are removed to force the autoregressive constraints to be obeyed, i.e., there must be no path from z_i to $(\mathbf{h}_1, \ldots, \mathbf{h}_i)$, usually by multiplying the weight matrix by a binarized mask matrix. Germain et al. [43] proposed a general framework dubbed MADE originally for decomposing any probability distribution into a set of conditionals by the product rule, where the autoregressive constraints need to be satisfied for the decomposition to be valid. Roughly speaking, MADE assigns each node in any hidden layer a degree which stipulates the maximum number of input nodes to which it can be connected and each node is only allowed to connect to nodes with lower or equal degrees. Masked or causal convolutions have also been widely used for constructing autoregressive neural nets, such as PixelCNN [29] and WaveNet [30], where convolution filters are masked to prevent from accessing future observations in a specified ordering. Masked autoregressive flows are efficient for forward computations. Specifically, $\mathbf{z}'$ can be calculated

in parallel via $z_i' = \tau\left(z_i; \mathbf{h}_i\right)$ as the conditioning features $(\mathbf{h}_1, \ldots, \mathbf{h}_D)$ can be obtained in a single forward propagation. However, the inverse procedure is inherently sequential because $\mathbf{h}_i$ needed for computing $z_i = \tau^{-1}\left(z_i'; \mathbf{h}_i\right)$ cannot be obtained until all $(z_1, \ldots, z_i)$ become available, which means that the inversion cannot be parallelized and is D times more computationally expensive than its forward pass. Consequently, masked autoregressive flows and the dual inverse autoregressive flows [37] are limited to applications where either forward or inverse evaluations are needed.

In order to overcome difficulties caused by the imbalanced computational cost, Song et al. [44] proposed a numerical inversion method in contrast to the previously mentioned analytic inversion methods. To be more specific, Song et al. [44] proposed a fixed-point iterative algorithm for computing inverse values approximately as shown in Eq. (8).

$$\begin{aligned} g\left(\mathbf{z}; \alpha, \mathbf{z}'\right) &= \mathbf{z} - \alpha \operatorname{diag}\left(J_{T_\phi}\left(\mathbf{z}\right)\right)^{-1}\left(T_\phi\left(\mathbf{z}\right) - \mathbf{z}'\right) \\ \mathbf{z}_{k+1} &= g\left(\mathbf{z}_k; \alpha, \mathbf{z}'\right) \end{aligned} \tag{8}$$

where $0 < \alpha < 2$ to ensure the spectral radius of the Jacobian of $g\left(\mathbf{z}; \alpha, \mathbf{z}'\right)$ is strictly less than 1 and therefore $g\left(\mathbf{z}; \alpha, \mathbf{z}'\right)$ is a contraction mapping. Therefore, as long as the number of iterations required for the local convergence is less than the dimensionality D, this fixed-point iterative algorithm can be much more efficient than computing the inverse of masked autoregressive flows exactly. Besides, coupling layers [35, 36] are another kind of popular implementations of the conditioner that can be equally fast for either sampling or density evaluation. Generally, the implementation of coupling layers divides $\mathbf{z}$ into two parts $\mathbf{z} = (\mathbf{z}_{\leq d}, \mathbf{z}_{>d})$ such that the first part $\mathbf{z}_{\leq d}$ is kept identical and only the second part is transformed conditioned upon the first part, as shown in Eq. (9):

$$\begin{aligned} \mathbf{z}_{\leq d}' &= \mathbf{z}_{\leq d} \\ \left(\mathbf{h}_{d+1}, \ldots, \mathbf{h}_D\right) &= \mathrm{NN}\left(\mathbf{z}_{\leq d}\right) \\ \mathbf{z}_i' &= \tau\left(\mathbf{z}_i; \mathbf{h}_i\right) \quad where \quad i > d \end{aligned} \tag{9}$$

where d is a partition index commonly set equal to $\lfloor \frac{D}{2} \rfloor$, and the inverse transformation can be readily computed as in Eq. (10):

$$\begin{aligned} \mathbf{z}_{\leq d} &= \mathbf{z}_{\leq d}' \\ \left(\mathbf{h}_{d+1}, \ldots, \mathbf{h}_D\right) &= \mathrm{NN}\left(\mathbf{z}_{\leq d}\right) \\ \mathbf{z}_i &= \tau^{-1}\left(\mathbf{z}_i'; \mathbf{h}_i\right) \quad where \quad i > d \end{aligned} \tag{10}$$

It is easy to see that both forward and inverse computations of coupling layers can be parallelized but at the price of reduced expressive power. When stacking multiple coupling layers, it is preferable to permute each element in $\mathbf{z}$ accordingly such that every split of $\mathbf{z}$ can be transformed alternately. As the permutation is an invertible operator with absolute Jacobian determinant equal to 1, it can be seamlessly integrated with other invertible and differentiable transformations in constructing deep flows.

Linear Flows Linear flows can be considered a generalization of the permutation operation and defined as in Eq. (11)

$$\mathbf{z}' = \mathbf{W}\mathbf{z} \tag{11}$$

where $\mathbf{W}$ is a $D \times D$ invertible matrix and the Jacobian determinant is simply $\det\mathbf{W}$. A naïve implementation of linear flows is not practical considering that $\mathbf{W}$ is not guaranteed to be invertible when getting updated and computing the inverse matrix and Jacobian determinant would have $O\left(D^3\right)$ computational complexity. The primary solution is to use matrix decomposition to factorize $\mathbf{W}$ into products of special matrices with easier control over invertibility and lower computational cost when computing the inverse and Jacobian determinant. Kingma et al. [45] proposed to use LU decomposition to parameterize $\mathbf{W}$ in their work Glow as shown in Eq (12):

$$\mathbf{W} = \mathbf{PL}\left(\mathbf{U} + \operatorname{diag}\left(\mathbf{s}\right)\right) \tag{12}$$

where $\mathbf{P}$ is a permutation matrix, $\mathbf{L}$ is a lower triangular matrix with ones on the diagonal, and $\mathbf{U}$ is an upper triangular matrix whose diagonal elements are zero. The invertibility can be guaranteed by setting elements in the vector $\mathbf{s}$ to be nonzero, and the log-determinant of Jacobian can be readily computed as $\log|\det\mathbf{W}| = \sum|s_i|$ in $O\left(D\right)$.

Constraining $\mathbf{W}$ to be orthogonal $\mathbf{Q}$ is another kind of strategy to resolve those challenges mentioned above, yielding orthogonal flows, given that the absolute determinant of any orthogonal matrix is one and $\mathbf{Q}^{-1} = \mathbf{Q}^{\top}$. But the parameterization of orthogonal matrix poses new challenges. Tomczak et al. [46] proposed to use Householder transformations to parameterize orthogonal matrices as shown in Eq. (13):

$$\begin{aligned} \mathbf{Q} &\overset{d}{=} \prod_{k=1}^{K} \mathbf{H}_k \\ \mathbf{H}_k &= \mathbf{I} - 2\frac{\mathbf{v}_k^{\top}\mathbf{v}_k}{\|\mathbf{v}_k\|^2} \end{aligned} \tag{13}$$

Each Householder transformation can be computed in time $O\left(D\right)$ and thus the total complexity is $O\left(KD\right)$. Golinski et al. [47] proposed to use the exponential map and the Cayley transformation to construct orthogonal matrices as shown in Eq. (14) and (15), respectively:

$$\mathbf{Q} \overset{d}{=} \exp\left(\mathbf{A}\right) = \sum_{i=0}^{\infty} \frac{\mathbf{A}^k}{k!} \tag{14}$$

$$\mathbf{Q} \overset{d}{=} \left(\mathbf{I} - \mathbf{A}\right)\left(\mathbf{I} + \mathbf{A}\right)^{-1} \tag{15}$$

where $\mathbf{A}$ is skew-symmetric, i.e., $\mathbf{A}^{\top} = -\mathbf{A}$. The computational complexity of either the exponential map or the Cayley map is $O\left(D^3\right)$, which means such methods scale poorly with the dimensionality.

Residual Flows Residual flows can be thought of as an extension of the concept of residual learning proposed by He et al. [48] to NFs, which can be formulated as in Eq. (16):

$$\mathbf{z}' = \mathbf{z} + g_{\phi}\left(\mathbf{z}\right) \tag{16}$$

where $g_\phi(\mathbf{z})$ needs to be carefully constructed to make the whole residual transformations to be invertible. Generally, there exist two different lines of research to tackle this challenge: 1) the matrix determinant lemma-based methods, and 2) the contractive mapping-based methods. The matrix determinant lemma is shown in Eq. (17):

$$\det\left(\mathbf{A}+\mathbf{V}\mathbf{W}^{\top}\right)=\det\left(\mathbf{I}+\mathbf{W}^{\top}\mathbf{A}^{-1}\mathbf{V}\right)\det\left(\mathbf{A}\right) \tag{17}$$

where $\mathbf{A}$ is an invertible matrix of size $D\times D$, and $\mathbf{V}$, $\mathbf{M}$ are matrices of size $D\times M$. The matrix determinant lemma provides a feasible or even computationally efficient manner to compute the Jacobian determinant of the residual transformation provided that the determinant and inverse of $\mathbf{A}$ is tractable and usually M is supposed to be less than D. Rezende et al. [49] proposed two simple implementations built upon the matrix determinant lemma: planar and radial flows. Planar flows can be described as in Eq. (18):

$$\mathbf{z}'=\mathbf{z}+\mathbf{v}\sigma\left(\mathbf{w}^{\top}\mathbf{z}+b\right) \tag{18}$$

where $\mathbf{v}\in\mathbb{R}^D$, $\mathbf{w}\in\mathbb{R}^D$, $b\in\mathbb{R}$, and σ denotes a differentiable activation function. The corresponding Jacobian determinant is shown in Eq. (19):

$$\det J_{T_\phi}(\mathbf{z})=1+\sigma'\left(\mathbf{w}^{\top}\mathbf{z}+b\right)\mathbf{w}^{\top}\mathbf{v} \tag{19}$$

which is the result of direct application of the matrix determinant lemma and has computational complexity of $O(D)$. Also, parameter values of $\mathbf{w}$ and $\mathbf{v}$ need to restricted such that $\mathbf{w}^{\top}\mathbf{v}>-1/\sup_x\sigma'(x)$ to ensure invertibility. Planar flows can be considered as a single hidden layer MLP with a single hidden node, and therefore the expressive power is rather limited. Similarly, radial flows can be constructed in accordance with the Eq. (20):

$$\mathbf{z}'=\mathbf{z}+\frac{\beta}{\alpha+r(\mathbf{z})}(\mathbf{z}-\mathbf{z}_0)\quad where\quad r(\mathbf{z})=\|\mathbf{z}-\mathbf{z}_0\| \tag{20}$$

where $\alpha\in(0,\infty)$, $\beta\in\mathbb{R}$, and $\|\cdot\|$ denotes the Euclidean norm. Invertibility is guaranteed by setting $\beta>-\alpha$. The Jacobian determinant can be computed as follows in Eq. (21):

$$\det J_{T_\phi}(\mathbf{z})=\left(1+\frac{\alpha\beta}{(\alpha+r(\mathbf{z}))^2}\right)\left(1+\frac{\beta}{\alpha+r(\mathbf{z})}\right)^{D-1} \tag{21}$$

which also is efficient to compute in time $O(D)$ but with restricted expressiveness. Van den Berg et al. [50] extended planar flows to a single layer MLP with M hidden units, known as Sylvester flows, which can be expressed as in Eq. (22):

$$\mathbf{z}'=\mathbf{z}+\mathbf{V}\sigma\left(\mathbf{W}^{\top}\mathbf{z}+\mathbf{b}\right) \tag{22}$$

where $\mathbf{V}\in\mathbb{R}^{D\times M}$, $\mathbf{W}\in\mathbb{R}^{D\times M}$, $\mathbf{b}\in\mathbb{R}^M$, and the differentiable activation function σ is applied in an element-wise fashion. The corresponding Jacobian determinant is calculated as in Eq. (23):

$$\det J_{T_\phi}(\mathbf{z}) = \det\left(\mathbf{I} + \mathbf{S}(\mathbf{z})\,\mathbf{W}^\top \mathbf{V}\right) \tag{23}$$

where $\mathbf{S}(\mathbf{z})$ is an $M \times M$ diagonal matrix with diagonal elements equal to $\sigma'(\mathbf{W}^\top \mathbf{z} + \mathbf{b})$. To ensure the transformation defined in Eq. (22) to be invertible, Van den Berg et al. [50] proposed to parameterize $\mathbf{V} = \mathbf{QU}$ and $\mathbf{W} = \mathbf{QL}$, where $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_M]$ consists of a set of orthornormal vectors $\mathbf{q}_i \in \mathbb{R}^D$, and $\mathbf{U}$, $\mathbf{L}$ are respective upper and lower triangular matrices of size $M \times M$ with diagonal entries satisfying the requirement $\mathbf{U}_{ii}\mathbf{L}_{ii} > -1/\|\sigma'\|_\infty$. By following the above parameterization, Eq. (22) and (23) are converted to the following two equations (24) and (25), respectively,

$$\mathbf{z}' = \mathbf{z} + \mathbf{QU}\sigma\left(\mathbf{L}^\top \mathbf{Q}^\top \mathbf{z} + \mathbf{b}\right) \tag{24}$$

$$\det J_{T_\phi}(\mathbf{z}) = \det\left(\mathbf{I} + \mathbf{S}(\mathbf{z})\,\mathbf{L}^\top \mathbf{U}\right) = \prod_{i=1}^{D}\left(1 + \mathbf{S}(\mathbf{z})_{ii}\,\mathbf{L}_{ii}\mathbf{U}_{ii}\right) \tag{25}$$

In addition to being invertible, the computational complexity of the Jacobian determinant is further reduced to $O(D)$. However, there is no analytic form for the inverse of planar, radial or Sylvester flows, which makes them not suitable for direct density estimation as shown in Eq. (4) and is also the reason why these techniques have been proposed to increase posterior expressiveness in variational inference.

The second kind of approaches attempts to parameterize neural nets such that $g_\phi(\mathbf{z})$ in Eq. (16) is a contraction mapping and so is true for $\mathbf{z}' - g_\phi(\mathbf{z})$. As stated by the Banach fixed-point theorem, there is a unique fixed-point such that $\mathbf{z}^* = \mathbf{z}' - g_\phi(\mathbf{z}^*)$ and the inverse can be found using the fixed-point iteration as shown in Eq. (26):

$$\mathbf{z}_{k+1} = \mathbf{z}' - g_\phi(\mathbf{z}_k) \tag{26}$$

Building contraction maps or equivalently ensuring the Lipschitz constant of neural network layers to be less than one is highly challenging. Behrmann et al. [51] proposed to use spectral normalization [52] to regularize weight matrices of linear or convolutional layers given that the Lipschitz constant of any linear function is its spectral norm. Besides, the evaluation of the Jacobian determinant has a time cost of $O(D^3)$, which is impractical when scaling to high-dimensional data, and instead needs to be evaluated by stochastic estimators [51].

Other Practical Considerations Flow-based neural nets are generally composed of stacking multiple transformation functions introduced above which can meet the essential requirements: invertibility, differentiability and tractable or efficient computation of Jacobian determinants. In this report, such transformation functions are referred to as atomic flow layers. Apart from these atomic flow layers, however, constructing deep flows necessitates other auxiliary techniques to ease optimization. Firstly, normalization techniques have been demonstrated to be critical to training deep neural nets. However, the existing normalization layers may fail to meet those essential requirements. In other words, normalization layers in flow-based models require additional treatment that can make them to be effective atomic flow layers. Fortunately, batch normalization [53] is directly applicable as it is a special case of the location-scale transformation as shown in Eq. (27):

$$\mathrm{BN}(\mathbf{z}) = \boldsymbol{\alpha} \odot \frac{\mathbf{z} - \hat{\boldsymbol{\mu}}}{\sqrt{\hat{\boldsymbol{\sigma}}^2 + \boldsymbol{\epsilon}}} + \boldsymbol{\beta} \tag{27}$$

where $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\sigma}}^2$ are batch statistics and therefore can be treated as constants. The Jacobian determinant is also readily to compute as shown in Eq. (28):

$$\det J_{BN}(\mathbf{z}) = \prod_{i=1}^{D} \frac{\alpha_i}{\sqrt{\hat{\sigma}_i^2 + \epsilon_i}} \tag{28}$$

As an immediate consequence of Eq. (3), interleaving batch normalization layers with any other atomic flow layers $T_{k+1} \circ \mathrm{BN} \circ T_k$ can be treated an invertible and differentiable transformation as a whole, whose Jacobian determinant is tractable.

Given that input and output of atomic flow layers have the same dimensionality as a consequence of invertibility, constructing deep flows would encounter computational bottleneck with high-dimensional real-world data. Dinh et al. [36] proposed to remove a small subset of dimensions of intermediate representations $\mathbf{z}_k$ every certain number of steps of flows so that the removed elements are excluded from the subsequent transformations, which can be regarded as skip-connections that map those removed elements to the final representation directly and therefore is termed multiscale architecture.

2.2.2 Variational AutoEncoders

Variational AutoEncoders (VAEs) provide a unified framework for approximate posterior inference and log-density estimation which can be optimized through stochastic gradient descent (SGD), giving rise to a scalable and computationally efficient method for joint inference and learning of Deep Latent Variable Models (DLVMs). Generally, the objective function of VAEs can be formulated as in Eq. (29):

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= \log \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \\
\mathcal{L}_{\theta,\phi}(\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})\right)
\end{aligned} \tag{29}$$

where $\mathcal{L}_{\theta,\phi}(\mathbf{x})$ is referred to as individual-datapoint evidence lower bound (ELBO), which consists of the probabilistic encoder/recognition model $q_\phi(\mathbf{z}|\mathbf{x})$, the probabilistic decoder/generative model $p_\theta(\mathbf{x}|\mathbf{z})$, and a regularization term, i.e., Kullback–Leibler (KL) divergence between the approximate posterior and prior distributions.

Posterior inference and learning often are computationally prohibitive or intractable. However, the intractability problem can be satisfactorily resolved in the VAE framework by employing the reparameterization trick, i.e., reparameterizing latent variables $\mathbf{z}$ as a deterministic and differentiable function of an externalized randomness source:

$$\begin{aligned}
\boldsymbol{\epsilon} &\sim p\left(\boldsymbol{\epsilon}\right) \\
\mathbf{z} &= \text{Encoder}_{\phi}\left(\mathbf{x}, \boldsymbol{\epsilon}\right) \\
\mathcal{L}_{\theta,\phi}\left(\mathbf{x}\right) &= \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\log p_{\theta}\left(\mathbf{x}, \mathbf{z}\right) - \log q_{\phi}\left(\mathbf{z}|\mathbf{x}\right)\right] \\
&\approx \log p_{\theta}\left(\mathbf{x}, \mathbf{z}\right) - \log q_{\phi}\left(\mathbf{z}|\mathbf{x}\right) = \tilde{\mathcal{L}}_{\theta,\phi}\left(\mathbf{x}\right)
\end{aligned} \tag{30}$$

The Eq. (30) implies that the reparameterization trick enables the gradients of the ELBO $\nabla\mathcal{L}_{\theta,\phi}\left(\mathbf{x}\right)$ w.r.t both generative model and variational parameters θ and ϕ to be calculated with Monte Carlo estimator $\nabla\tilde{\mathcal{L}}_{\theta,\phi}\left(\mathbf{x}\right)$, which is an unbiased estimator of the exact gradients of the ELBO. SGD-based optimization of ELBO by using the reparameterization trick is referred to as the Stochastic Gradient Variational Bayes (SGVB) estimator [54].

In contrast to traditional Variational Inference (VI) where each data point has a separate variational distribution, the parameters of the recognition model in VAEs are shared among all input variables, which is referred to as amortized VI. Also, there are no oversimplified assumptions on variational distributions parameterized by deep neural networks, e.g., independence between latent variables assumed in mean-field VI, which results in more flexible and increased expressiveness of the approximate posterior. The most significant contribution of VAEs is the reparameterization trick, which allows obtaining low-variance gradient estimators of ELBO and therefore leads to efficient optimization using automatic differentiation in modern deep learning libraries. All these advantages combined together leads to a recent upsurge of interest in VAEs, which marries Bayesian Networks and Deep Learning. VAEs since its inception have also been successfully applied to a wide variety of areas from generative modelling, semi-supervised learning to unsupervised representation learning.

Multi-Sample Stochastic Lower Bounds & Stochastic Gradient Estimators Reinterpreting VAEs from the lens of importance sampling, i.e., the approximate posterior can be considered the proposal distribution in importance sampling, leads to the extension of the vanilla VAE objective (single-sample) to multi-sample Monte Carlo Objective (MCO). Burda et al. [55] introduced IWAE lower bound as shown in Eq. (31), which has proven to be a tightened variational lower bound as the number of samples increases, which, however, comes at the cost of rendering naive gradient estimators (Eq. (32)) troublesome. More specifically, increasing the number of samples would impede the learning procedure and thus deteriorate model performance due to the high variance of naive gradient estimators w.r.t. variational parameters in particular, which has been shown theoretically and empirically in [56, 57].

$$\mathcal{L}_{K}\left(\mathbf{x}\right) = \mathbb{E}_{\mathbf{z_1},\cdots,\mathbf{z_K}\sim q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\log \frac{1}{K}\sum_{i=1}^{K}\frac{p_{\theta}\left(\mathbf{x}, \mathbf{z}_i\right)}{q_{\phi}\left(\mathbf{z}_i|\mathbf{x}\right)}\right] \leq \log p_{\theta}\left(\mathbf{x}\right) \tag{31}$$

$$\begin{aligned}
w_i &= \frac{p_{\theta}\left(\mathbf{x}, \mathbf{z}_i\right)}{q_{\phi}\left(\mathbf{z}_i|\mathbf{x}\right)} \\
\tilde{w}_i &= \frac{w_i}{\sum_{i=1}^{K} w_i} \\
\nabla_{\theta,\phi}\mathcal{L}_{K}\left(\mathbf{x}\right) &= \mathbb{E}_{\epsilon_{1:K}}\left[\sum_{i=1}^{K}\tilde{w}_i \log w_i\right]
\end{aligned} \tag{32}$$

Therefore, various variance reduction techniques have been proposed to obtain more informative gradient estimators with lower variance for the multi-sample MCO. The mainstream solutions [58–61] leverage control variates from literature of importance sampling to reduce variance, which can be roughly considered as subtracting zero expectation quantities, called baselines, that are positively correlated with the magnitude of the learning signal and perhaps dependent on input data. VIMCO [59] is a general framework for unbiased gradient estimations of multi-sample MCO, which realizes effective variance reduction through evaluating the magnitude of the learning signal using the other $K-1$ samples in a minibatch, therefore obviating the need to learn additional parameters to compute baselines such as done in NVIL [58]. Roeder et al. [60] identified the source of high variance arising from the lurking score function term $\nabla_\phi \log q_\phi(\mathbf{z}|\mathbf{x})$ by inspecting the total derivative of ELBO and therefore proposed to remove it from gradient computations, i.e., forcing the updates of variational parameters ϕ only rely on latent variables $\mathbf{z}$, which is referred to as STL estimator. However, completely removing the score function term from gradient estimations makes the STL estimator biased. Consequently, Tucker et al. [61] proposed to apply twice the reparameterization trick to construct an unbiased estimator called DReG, leading to the automatic cancellation of the score function term, which is broadly applicable to a range of gradient estimators, such as IWAE, RWS [62], and JVI [63]. It is worth mentioning that RWS was proposed to train Helmholtz machines [64] and Deep Belief Networks (DBNs) [65] which are deep directed graphical models and bear a close resemblance to VAEs. RWS features a decoupled training procedure of alternating updates of parameters between the generative and the recognition model, and thus does not optimize a unified objective such as ELBO. Despite this deficiency, it has been considered as an attractive alternative in practice [57].

Discrete Latent Variables Due to the jumping discontinuities of discrete latent variables, incorporating stochastic layers composed of discrete random variables into VAEs brings additional challenges for constructing appropriate gradient estimators. For the gradient estimators mentioned previously, apart from those solely relying on score function estimators, such as VIMCO and RWS, which would suffer from the high variance issues, other gradient estimators, such as STL and DReG, are not directly applicable to discrete distributions, thereby requiring reparameterizable continuous relaxations. Although as far as mixture distributions are concerned, marginalizing over discrete/membership variables [8, 60] can be regarded as an alternative to sidestep directly dealing with discrete latent variables, it would be computationally demanding when the dimensions of discrete variables increase significantly. Jang et al. [66] and Maddison et al. [67] independently discovered reparameterization methods that can approximate discrete distributions in the limiting case, which are referred to as Gumbel-Softmax and Concrete distributions, respectively, as shown in Eq. (33):

$$X_k = \frac{\exp\left(\left(\log \alpha_k + G_k\right)/\lambda\right)}{\sum_{i=1}^{n} \exp\left(\left(\log \alpha_i + G_i\right)/\lambda\right)} \tag{33}$$

$$p_{\alpha,\lambda}(x) = (n-1)!\lambda^{n-1} \prod_{k=1}^{n} \frac{\alpha_k x_k^{-\lambda-1}}{\sum_{i=1}^{n} \alpha_i x_i^{-\lambda}} \tag{34}$$

where $X \in \Delta^{n-1} = \left\{x \in \mathbb{R}^n \mid x_k \in [0,1], \sum_{i=1}^{n} x_i = 1\right\}$ is a probability simplex, the temperature parameter $\lambda \in (0,\infty)$, (unnormalized)parameters of the categorical distribution $\alpha_k \in (0,\infty)$, and i.i.d. gumbel noise $G_k \sim \text{Gumbel}(0,1)$. The corresponding probability distribution implied by Eq. (33) is shown in Eq. (34). The property of Concrete random variables $\mathbb{P}\left(\lim_{\lambda\to 0} X_k = 1\right) = \frac{\alpha_k}{\sum_{i=1}^{K} \alpha_i}$ guarantees that when the temperature λ approaches 0, the samples

of the Concrete distribution can well-approximate those drawn from $\arg\max(\log \alpha_k + G_k) \sim \text{Cat}\left(\boldsymbol{\pi} \mid \pi_i = \frac{\alpha_i}{\sum_{i=1}^{K} \alpha_i}\right)$, which is the well-known Gumbel-Max trick [68]. As pointed out by Potapczynski et al. [69], the necessary conditions for constructing reparameterizable continuous relaxations of discrete distributions are a reparameterizable continuous distribution defined on the probability simplex, i.e., defined as transforming parameter-independent random noise, and concentrating the probability mass on the vertices of the simplex when the temperature parameter approaches 0. Consequently, Potapczynski et al. [69] proposed to transform Gaussian noise onto a simplex by invertible transformations, referred to as Invertible Gaussian Reparamterization (IGR), which should satisfy two requirements: 1) efficient computation of the Jacobian determinant, and 2) placing the probability mass around vertices as the temperature λ reaches 0. IGR is more flexible than Gumbel-Softmax or Concrete Random Variables but at the cost of losing interpretability of parameters.

Posterior Collapse Almost without exception DLVMs are notorious for the optimization issues. In particular, the objective of VAEs tends to get stuck in the local optimum where approximate posteriors are driven towards to match the uninformative priors and consequently the latent variables remain inactive during the training procedure, which is referred to as posterior collapse or the KL vanishing problem. The problem would be exacerbated when the recognition model is coupled with a powerful decoder, such as autogressive models, causing difficulties especially for applying VAEs for sequential data. Numerous approaches have been proposed to mitigate the posterior collapse problem. Bowman et al. [70] proposed to gradually increase the weight of the KL term such that in the initial stage of training the recognition model is not adversely affected by the prior distribution. The monotonically increasing scheduling scheme of the KL cost has been further extended to a cyclical annealing schedule by Fu et al. [71] where the provided theoretical analysis has suggested that increasing the weight of the KL cost would reduce mutual information between observations and latent codes and therefore repeatedly setting the weight to zero is beneficial for meaningful latent representation learning. Kingma et al. [37] proposed a slightly modified VAE objective where the substituted KL term can enforce at least a minimum amount of λ nats information to be encoded in a subset of latent dimensions, called Free Bits objective. Another popular solution is to weaken the capacity of the decoder, e.g., limiting the receptive field of conditional distributions used in autoregressive models [72]. In the same spirit, Lucas et al. [73] proposed to augment the vanilla VAE objective with an auxiliary reconstruction term which can guide the global structure information encoded in latent codes and the remaining local variations captured by autoregressive models. Dieng et al. [74] proposed to augment the decoder with skip connections between observations $\mathbf{x}$ and latent codes $\mathbf{z}$ which can promote increased mutual information and therefore prevent the KL vanishing problem. He et al. [75] has shown that a novel training procedure where the updates of the parameters of the recognition model are more frequent than those of the generative model is beneficial for preventing posterior collapse based on empirical observations that the approximate posterior usually lags far behind the true posterior on synthetic datasets. Li et al. [76] has conducted extensive experiments on the existing solutions aimed at tackling the posterior collapse problem and found that the combination of pre-training inference network with the AutoEncoder objective and Free Bits VAE objective can yield better results.

Expressive Posteriors & Flexible Priors Given that the ELBO can be rewritten as in Eq. (35), the gap between the ELBO $\mathcal{L}_{\theta,\phi}(\mathbf{x})$ and the marginal log-likelihood $\log p_\theta(\mathbf{x})$ is reduced as the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ is closer to the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$ in terms of the KL divergence.

$$\log p_\theta(\mathbf{x}) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right]}_{=\mathcal{L}_{\theta,\phi}(\mathbf{x})} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})}\right]}_{=D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})\right)} \tag{35}$$

This motivates improving the expressive power of the approximate posterior, i.e., building more complex variational families, in order to improve the model performance. One general method is NFs as introduced previously, such as planar flows [49], radial flows [49], Sylvester flows [50], Householder flows [46], and inverse autoregressive flows [37], which have been proposed in the context of SVI for constructing flexible posteriors. The procedure can be roughly described as in Eq. (36):

$$\begin{aligned} \mathbf{z}_0 &\sim q_0(\mathbf{z}|\mathbf{x}) \\ \mathbf{z}_K &= T_K \circ \cdots \circ T_2 \circ T_1(\mathbf{z}_0) \\ \mathbf{z}_K &\sim q_K(\mathbf{z}|\mathbf{x}) \end{aligned} \tag{36}$$

where $q_0(\mathbf{z}|\mathbf{x})$ is a simple base distribution, such as a spherical Gaussian, and T_i are invertible and differentiable transformations with efficiently computable Jacobian determinants. Consequently, the ELBO in Eq. (29) can be reformulated as in Eq. (37):

$$\mathcal{L}_{\theta,\phi}(\mathbf{x}) = \mathbb{E}_{q_0}\left[\log q_0(\mathbf{z}|\mathbf{x}) - \log p_\theta(\mathbf{x},\mathbf{z})\right] - \mathbb{E}_{q_0}\left[\sum_{k=1}^{K} \log\left|\det\left(\frac{\partial T_k(\mathbf{z}_{k-1};\phi_k)}{\partial \mathbf{z}_{k-1}}\right)\right|\right] \tag{37}$$

The other kind of approaches to increase the flexibility of approximate posteriors is to introduce auxiliary latent variables. Given that such auxiliary latent variables are usually arranged in a hierarchical structure, the detailed discussions are deferred to the next subsection.

Additionally, there has been a parallel line of research on increasing the flexibility of priors as too simplistic priors may cause overregularization and consequently uninformative latent representations. Tomczak et al. [77] proved that the optimal prior should match the aggregated posterior, i.e., $p^*_\lambda(\mathbf{z}) = \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)$, which is consistent with similar findings in other work [78, 79]. Built upon this observation, Tomczak et al. [77] proposed the variational mixture of posteriors prior (VampPrior) as shown in Eq. (38) to approximate the aggregated posterior:

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k) \tag{38}$$

where $\mathbf{u}_k$, referred to as the pseudo input, is a vector of the same dimensionality as the input and can be updated through backpropagation. Also, NFs can be employed to build flexible priors in a similar manner to improve expressiveness of posteriors. Ding et al. [80] proposed to use RealNVP [36] to augment the prior distributions and combine the vanilla VAE objective with IWAE objective [56] to facilitate learning of prior-related parameters. In contrast to previous methods, Van den Oord et al. [81] proposed Vector Quantized VAE (VQ-VAE) where a codebook is used to represent a discretized prior distribution and can be learned in a way similar to K-means. To overcome one of the drawbacks of VQ-VAE that the quantization process is deterministic,

Sønderby et al. [82] further extended VQ-VAE to its probabilistic version by using VIMCO [59] and Gumbel-Softmax [66, 67] gradient estimators and demonstrated that VIMCO struggles with high dimensions of the discrete latent space.

Multiple Stochastic Latent Layers & Hierarchical VAEs All the previous discussions concerning VAEs have so far been restricted to a single layer of stochastic variables. Incorporating multiple stochastic latent layers into the VAE framework, i.e., building deep hierarchical VAE models, is notoriously challenging mainly as a result of optimization issues, and consequently the posterior collapse problem mentioned above would be exacerbated, i.e., posteriors of the vast majority of latent variables would collapse to a trivial prior, which can be seen from the following VAE objective for multiple stochastic latent layers (39):

$$\mathcal{L}_{\theta,\phi}\left(\mathbf{x}\right)=\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\frac{p_\theta\left(\mathbf{x},\mathbf{z}_{<L}|\mathbf{z}_L\right)}{q_\phi\left(\mathbf{z}_{<L}|\mathbf{x}\right)}\right]-\mathbb{E}_{q_\phi(\mathbf{z}_{<L}|\mathbf{x})}\left[D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{z}_L|\mathbf{z}_{<L}\right)\|p_\theta\left(\mathbf{z}_L\right)\right)\right] \tag{39}$$

when $p_\theta\left(\mathbf{x},\mathbf{z}_{<L}\right)$ is powerful enough, latent variables at the top layer tend to be ignored. The optimization issues have restricted the study of VAEs to shallow models discussed previously. Sønderby et al. [83] are among the first to explore bidirectional inference in building deep VAEs, resulting in Ladder VAE (LVAE). Specifically, approximate posteriors are derived as a combination of information from both the bottom-up inference and top-down generative processes, which can also be regarded as iteratively correcting posterior distributions with information carried from priors as the top-down path used for approximate posterior inference shares parameters with the generative model. Maaløe et al. [84] proposed Bidirectional-Inference Variational Autoencoder (BIVA), which can be considered a generalization of LVAE by introducing skip-connections into generative models and splitting latent variables at each level of the hierarchy into two groups such that one of which can be used to construct stochastic bottom-up inference path. Vahdat et al. [85] proposed to build deep hierarchical VAE models from the perspective of architectural design and use residual parameterization of posteriors and spectral normalization [52] to ease optimization. Razavi et al. [86] extended VQ-VAE to its deep hierarchical variant by simply maintaining different codebooks for different levels in the hierarchy such that low-level codebooks are conditioned on higher-level codebooks. Williams et al. [87] proposed Hierarchical Quantized Autoencoders (HQA) where the probabilistic VQ-VAE, referred to as stochastic quantization in this work [87], is combined with a hierarchical latent structure and they further illustrated layer-wise quantization can resolve mode-covering behaviour in the data space, therefore facilitating generating realistic samples.

2.2.3 Dynamical Variational Auto-Encoders

VAEs that have been discussed so far do not take temporal correlations in data into consideration, i.e., sequence vectors are treated independently with each other, which would undermine the modelling power when applied to process sequential data. Naturally, the extension of static VAEs to use latent dynamics to capture temporal dependencies in the observed data by integrating Recurrent Neural Networks (RNNs) or State Space Models (SSMs) into the VAE framework yields a class of powerful probabilistic sequential latent variable models, which is referred to as Dynamical Variational Auto-Encoders (DVAEs) [88]. RNNs and SSMs are powerful deterministic and stochastic sequential models, respectively, which lends themselves to be the natural choices for latent temporal modelling in DVAEs. Generally, DVAEs are temporally replicated VAEs at different time indices, also termed per timestep VAEs. The temporal structure in the generative model of DVAEs can be characterized by the following Eq. (40):

$$
\begin{aligned}
p_\theta\left(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T}\right) &= \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_t, \mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}\right) \\
&= \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t}, \mathbf{u}_{1:t}\right) p_\theta\left(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}\right)
\end{aligned} \tag{40}
$$

where $\mathbf{x}_{1:T}$ are observed data sequences, $\mathbf{z}_{1:T}$ are latent random vectors/state sequences, and $\mathbf{u}_{1:T}$ are control input sequences, which is the terminology inherited from SSMs. The causal dependence along the temporal dimension is clearly reflected in the factorization. It should be noted that the Eq. (40) gives the general form of generative distributions, which can be further simplified by additionally making conditional independence assumptions. The general form of the inference network is given in Eq. (41):

$$
q_\phi\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right) = \prod_{t=1}^{T} q_\phi\left(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right) \tag{41}
$$

It can be seen from the Eq. (41) that the inference process is generally non-causal, i.e., it depends on past, present and future observations and control signals, which corresponds to the concept of smoothing in control theory. The sophisticated temporal dependencies between latent variables distributed along the temporal axis would complicate the inference process and therefore usually need to be simplified by exploiting the conditional independence, i.e., the D-separatedness [89], in graphical models. The ELBO for VAEs can be naturally extended to DVAEs as shown in Eq. (42), which is firstly presented in the work [88]:

$$
\begin{aligned}
\mathcal{L}_{\theta,\phi}\left(\mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right) &= \mathbb{E}_{q_\phi\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right)}\left[\log p_\theta\left(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T}\right) - \log q_\phi\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right)\right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{q_\phi\left(\mathbf{z}_{1:t} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right)}\left[\log p_\theta\left(\mathbf{x}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t}, \mathbf{u}_{1:t}\right)\right] \\
&\quad - \sum_{t=1}^{T} \mathbb{E}_{q_\phi\left(\mathbf{z}_{1:t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right)}\Big[D_{\mathrm{KL}}\left(q_\phi\left(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right) \| \right. \\
&\qquad \left. p_\theta\left(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}\right)\right)\Big]
\end{aligned} \tag{42}
$$

As the reparameterization trick remains valid for Eq. (42), the intractable expectations can be approximated by Monte Carlo estimates where samples from $q_\phi\left(\mathbf{z}_{1:t} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right)$ are obtained recursively by the ancestral sampling.

The general framework introduced above for DVAEs greatly facilitates the analysis and comparison of the existing approaches on combining RNNs or SSMs with VAEs. Deep Kalman Filters (DKF) [90] and Deep Markov Models (DMM) [91] proposed by Krishnan et al. can be viewed as imposing a first-order Markovian structure on latent variables in VAEs and therefore is a generalization of linear dynamical systems, where the generative and inference models can be further simplified by the Markov assumption, yielding the following two Eq. (43) and (44):

$$p_\theta\left(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T}\right) = \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_t | \mathbf{z}_t\right) p_\theta\left(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_t\right) \tag{43}$$

$$q_\phi\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}\right) = \prod_{t=1}^{T} q_\phi\left(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t:T}\right) \tag{44}$$

Fraccaro et al. [92] proposed a unified framework, called Kalman Variational Autoencoder (KVAE), for joint optimization of VAEs and Linear Gaussian SSMs (LGSSMs) by firstly mapping high-dimensional observations to a low-dimensional manifold using a shared VAE across timesteps and then learning temporal dynamics in the lower-dimensional space by LGSSMs with analytical solutions provided by classic Kalman Filters.

In addition to the efforts of integrating SSMs with VAEs, extending RNNs to sequential latent variable models has been popular in the literature. STOchastic Recurrent Network (STORN) proposed by Bayer et al. [93] is among the first attempts to introduce stochasticity into deterministic RNNs by allowing the recursive updates of hidden representations $\mathbf{h}_t$ in RNNs to take as additional input random variables $\mathbf{z}_t$ which are independent between different time indices, i.e., $\mathbf{h}_t = f_h\left(\mathbf{x}_{1:t-1}, \mathbf{z}_{1:t}\right)$. Although $\mathbf{h}_t$ has been turned into random variables, the entanglement of $\mathbf{h}_t$ and $\mathbf{z}_t$ has established a fully connected graphical model, which complicates the inference process, and the reduction made in the work [93] $q_\phi\left(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:T}\right) = q_\phi\left(\mathbf{z}_t | \mathbf{x}_{1:t}\right)$ is not well-justified, hence leading to inferior performance. The Variational Recurrent Neural Network (VRNN) proposed by Chung et al. [94] model temporal dependencies both in observed data and latent states with a single RNN, i.e., the per timestep generative and inference process in VAEs are tied through a shared RNN. Unlike the i.i.d. assumption made in STORN $p_\theta\left(\mathbf{z}_{1:T}\right) = \prod_{t=1}^{T} p_\theta\left(\mathbf{z}_t\right)$ [93], VRNN introduces the temporal structure into latent random variables, i.e., $p_\theta\left(\mathbf{z}_{1:T}\right) = \prod_{t=1}^{T} p_\theta\left(\mathbf{z}_t | \mathbf{h}_t = f_h\left(\mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}\right)\right)$, resulting in increased flexibility in priors. The drawback of VRNN is similar to that of STORN, posterior inference is simplified without respecting the dependencies of random variables in the true posterior, i.e., artificially imposing conditional independence, due to the shared RNN. The Stochastic Recurrent Neural Network (SRNN) proposed by Fraccaro et al. [95] is distinct from STORN and VRNN in a clear separation of deterministic and stochastic layers by stacking a layer of SSM on top of a layer of RNN. Despite the first-order Markovian structure assumed in the SSM, the state transitions of SSM are nonlinear and conditioned upon hidden representations of a RNN, which makes the SSM layer can capture more complex temporal dynamics. More importantly, as pointed out by Girin et al. [88], SRNN is one of the earliest work where the approximate posterior inference fully respects the dependencies in the true posterior, i.e., $q_\phi\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}\right) = \prod_{t=1}^{T} q_\phi\left(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}\right)$, where the future observations $\mathbf{x}_{t:T}$ are summarized through a backward RNN. The recurrent VAE (RVAE) was proposed by Leglaive et al. [96] for speech and audio spectrogram modelling. By omitting the application-specific details of RVAE, it can be considered as an effective remedy for STORN in the sense that the approximate posterior inference is consistent with dependence relations in the true posterior, i.e., $q_\phi\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}\right) = \prod_{t=1}^{T} q_\phi\left(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{t:T}\right)$, where the past latent states $\mathbf{z}_{1:t-1}$ and the future observations $\mathbf{x}_{t:T}$ are summarized by a forward and backward RNN, respectively. The Disentagled Sequential Autoencoder (DSAE) proposed by Li et al. [97] explicitly introduces a sequence-level latent state $\mathbf{v}$ which would guide the learning of temporal dynamics through time-dependent latent variables $\mathbf{z}_t$, therefore realizing a decomposition of contents and dynamics. The generative and inference model of DSAE can be described as in Eq. (45) and (46), respectively:

$$p_\theta\left(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}, \mathbf{v}\right) = p_\theta\left(\mathbf{v}\right) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_t|\mathbf{z}_t, \mathbf{v}\right) p_\theta\left(\mathbf{z}_t|\mathbf{z}_{1:t-1}\right) \tag{45}$$

$$q_\phi\left(\mathbf{z}_{1:T}, \mathbf{v}|\mathbf{x}_{1:T}\right) = q_\phi\left(\mathbf{v}|\mathbf{x}_{1:T}\right) \prod_{t=1}^{T} q_\phi\left(\mathbf{z}_t|\mathbf{z}_{1:t-1}, \mathbf{v}, \mathbf{x}_{t:T}\right) \tag{46}$$

It can be seen from the Eq. (45) that DSAE is a generalization of DKF in the sense that dependencies in latent variables $\mathbf{z}_t$ are no longer restricted to the first-order Markovian assumption. But it should be pointed out that the inference process proposed by Li et al. [97] does not follow the Eq. (46) which is the corrected form by Girin et al. [88].

3 Benchmark Datasets

Table 1: An overview of publicly available benchmark datasets for crop type mapping. "TS" and "ST" in the column "Format" stand for *Time Series* and *Spatio-Temporal*, respectively. "S1", "S2", "PF", "M", and "T" in the column "Modality" denote *Sentinel-1, Sentinel-2, Planet Fusion, Meteorological Data, and Topographic Data*, respectively.

Source	Area of Focus	Format	Modality	# Fields	Temporal Density	Temporal Shift
BREIZHCROPS [98]	France	TS	S2	6.1×10^5	$\geq$ 5 days	✓
TimeSen2Crop [99]	Austria	TS	S2	1.1×10^6	$\geq$ 5 days	✓
DENETHOR [1]	Germany	ST	S1+S2+PF	4.5×10^3	Daily	✓
EUROCROPS [100]	EU Member States	TS	S2	8×10^5	$\geq$ 5 days	✓
CropHarvest [101]	Global	TS	S1+S2+M+T	9×10^4	Monthly	✓

The exploitation of raw satellite data for Machine Learning (ML) researchers and practitioners who lack a strong background in Earth Observation (EO) and Remote Sensing (RS) has been impeded by the high barriers to entry, such as specialized geospatial analysis software (e.g., Google Earth Engine(GEE)), data pre-processing techniques (e.g., cloud removal, atmospheric correction, and radiometric calibration), and peculiarities of satellite data (e.g., multi-spectral bands in Sentinel-2 and polarizations in Synthetic Aperture Radar (SAR) data). Recently, there have been great efforts in RS community devoted to providing ML analysis-ready benchmark datasets to lower the barriers to entry and therefore foster collaborations between RS and ML researchers and practitioners.

Specifically, the existing publicly available and ML analysis-ready benchmark datasets for crop type mapping are summarized in Tab. 1, which fall into two general categories: 1) time-series, and 2) spatio-temporal data format. The former one is obtained by mean-aggregating reflectance values in each band and parcel field where geometry information or boundaries of crop parcels are usually accessible based on the premise that temporal evolution of crop phenology is critical and sufficient for distinguishing different crop types. But Kondmann et al. [1] argued that the discarded spatial information may be beneficial for improving classification performance and therefore presented the first benchmark dataset called DENETHOR, which includes Planet Fusion, Sentinel-1, and Sentinel-2 satellite data with daily temporal frequency and high-quality field geometry information at 3m spatial resolution. The extremely high resolution in both temporal and spatial dimensions significantly restricts the geographical size that can be covered due to computational concerns. Consequently, the number of labelled parcel fields in other datasets listed in Tab. 1 is at least one order of magnitude larger than that of DENETHOR. Except for the DENETHOR, other recently released datasets all adopted the time-series format but with respective characteristic features. BREIZHCROPS [98] presented Sentinel-2 top-of-atmosphere (Level 1C) as well as bottom-of-atmosphere (Level 2A) time series, where the latter is atmospherically corrected. TimeSen2Crop [99] employed LSTM to improve the reliability of labelled units by only keeping those with high predicted confidence given that the ground-truth labels are obtained through farmers' self-declarations. EUROCROPS [100] collected agricultural parcel information from 13 of all 27 European Union (EU) member states, aiming at achieving the maximal regional coverage in EU, and introduced a novel taxonomy

scheme called HCAT-ID. But, at the current stage, they only released a demo dataset for a first study, which only includes crop data from Austria, Slovenia, and Denmark. CropHarvest [101] is a benchmark dataset that covers geographically diverse regions across the globe by assembling 20 datasets, part of which has already been made publicly available before such as DENETHOR and the remaining part is collected from NASA Harvest. In addition to Sentinel-1 and Sentinel-2 data, meteorological and topographic data have also been provided. In conjunction with the crop benchmark dataset, an ML friendly API and the code used to acquire the satellite data from GEE have also been open-sourced. Last but not least, all datasets listed in Tab. 1 have considered the temporal shift effect, i.e., the out-of-year generalization, because crop type maps need to be updated regularly due to crop rotation practices. But the performance drop of existing crop classification systems as a result of testing on data collected in years different from the one used for training can be substantial, e.g., approximately 12 percentage points drop in accuracy has been reported in DENETHOR [1].

4 Work to Date

Annual work progress is summarized as follows:

- Conducted a literature review on crop type classification, SITS analysis, deep generative models, deep semi-/un- supervised learning, and deep neural architecture designs;
- Acquired an intermediate level of knowledge and skill in performing parallel training of deep neural nets on SLURM-based High Performance Computing (HPC) facilities;
- Built a pipeline to process raw SITS for crop type classification;
- Participated in AI4Food Security Challenge and submitted final solutions;
- Submitted a manuscript entitled "Revisiting VAE Objective for Improving Satellite Image Time Series Classification[3]" to ECML PKDD 2022.

[3]This is the link to the preprint on Arxiv.

5 Plan of Future Research Activities

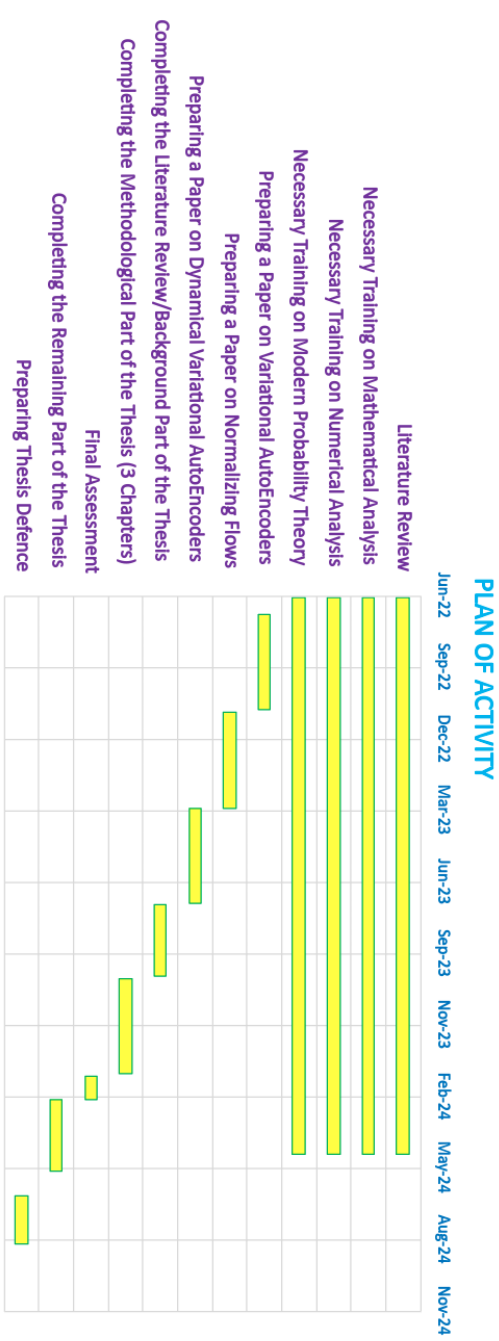

Figure 2: Gantt chart of planned activities for the remaining period of study.

6 Thesis Outline

NFs and VAEs are two popular generative modelling frameworks, which have experienced rapid progress in recent years with increasingly powerful capabilities to capture highly complex probability distributions of real-world data, as seen in Sec. 2.2. DVAEs as the extension of VAEs to sequential latent variable modelling have also been demonstrated effective for capturing complex temporal dynamics in sequential data, such as audio and video signals. However, these three kinds of generative modelling frameworks have not been explored so far for crop type classification. Therefore, the primary focus of my PhD research is to showcase the potential of applying deep generative models to improve classification accuracy and data efficiency for crop type classification with SITS data, which will form the three main chapters of my PhD thesis. The general structure of my PhD thesis is outlined as follows.

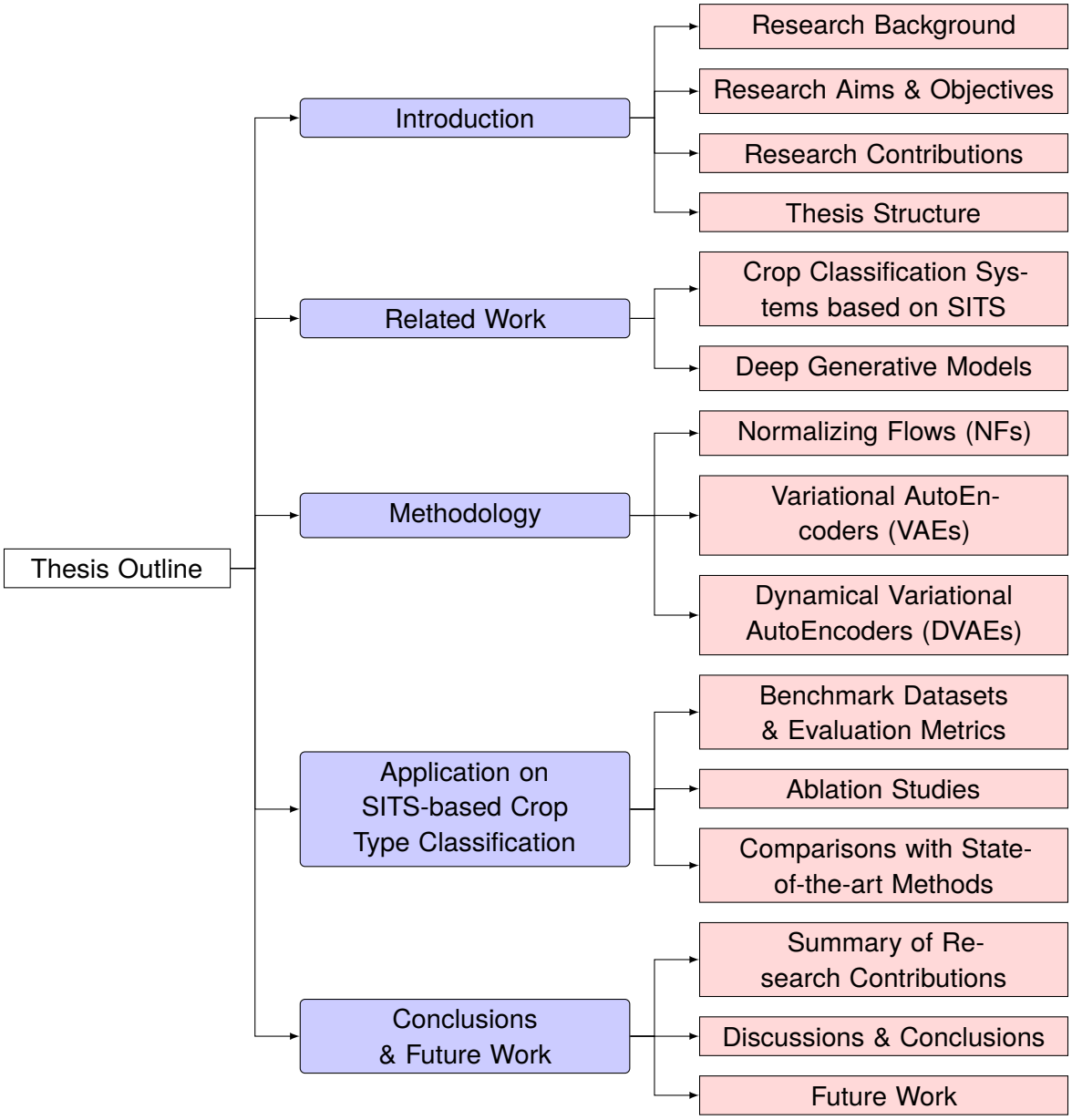

Figure 3: The outline of thesis structure.

References

[1] L. Kondmann, A. Toker, M. Rußwurm, A. Camero, D. Peressuti, G. Milcinski, P.-P. Mathieu, N. Longépé, T. Davis, G. Marchisio *et al.*, "Denethor: The dynamicearthnet dataset for harmonized, inter-operable, analysis-ready, daily crop monitoring from space," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[2] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[3] A. H. S. Solberg, T. Taxt, and A. K. Jain, "A markov random field model for classification of multisource satellite imagery," *IEEE transactions on geoscience and remote sensing*, vol. 34, no. 1, pp. 100–113, 1996.

[4] P. Leite, R. Feitosa, A. Formaggio, G. Da Costa, K. Pakzad, and S. IDA, "Hidden markov models applied in agricultural crops classification," *Proceeding of GEOBIA (GEOgraphic Object-Based Image Analysis for the 21St Century)*, 2008.

[5] S. Siachalou, G. Mallinis, and M. Tsakiri-Strati, "A hidden markov models approach for crop classification: Linking crop phenology to time series of multi-sensor remote sensing data," *Remote Sensing*, vol. 7, no. 4, pp. 3633–3650, 2015.

[6] A. Atanov, A. Volokhova, A. Ashukha, I. Sosnovik, and D. P. Vetrov, "Semi-conditional normalizing flows for semi-supervised learning," *CoRR*, vol. abs/1905.00505, 2019.

[7] P. Izmailov, P. Kirichenko, M. Finzi, and A. G. Wilson, "Semi-supervised learning with normalizing flows," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4615–4630.

[8] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling, "Semi-supervised learning with deep generative models," *Advances in neural information processing systems*, vol. 27, 2014.

[9] X. Cheng, W. Xu, T. Wang, W. Chu, W. Huang, K. Chen, and J. Hu, "Variational semi-supervised aspect-term sentiment analysis via transformer," in *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019*. Association for Computational Linguistics, 2019, pp. 961–969.

[10] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*. ijcai.org, 2017, pp. 1965–1972.

[11] L. Maaløe, M. Fraccaro, and O. Winther, "Semi-supervised generation with cluster-aware generative models," *CoRR*, vol. abs/1704.00637, 2017.

[12] B. Paige, J.-W. van de Meent, A. Desmaison, N. Goodman, P. Kohli, F. Wood, P. Torr *et al.*, "Learning disentangled representations with semi-supervised deep generative models," *Advances in neural information processing systems*, vol. 30, 2017.

[13] P. Hao, Y. Zhan, L. Wang, Z. Niu, and M. Shakir, "Feature selection of time series modis data for early crop classification using random forest: A case study in kansas, usa," *Remote Sensing*, vol. 7, no. 5, pp. 5347–5369, 2015.

[14] S. Valero, D. Morin, J. Inglada, G. Sepulcre, M. Arias, O. Hagolle, G. Dedieu, S. Bontemps, P. Defourny, and B. Koetz, "Production of a dynamic cropland mask by processing remote sensing image series at high temporal and spatial resolutions," *Remote Sensing*, vol. 8, no. 1, p. 55, 2016.

[15] R. Devadas, R. Denham, and M. Pringle, "Support vector machine classification of object-based data for crop mapping, using multi-temporal landsat imagery," *International archives of the photogrammetry, remote sensing and spatial information sciences*, vol. 39, no. 1, pp. 185–190, 2012.

[16] M. Rußwurm and M. Körner, "Multi-temporal land cover classification with sequential recurrent encoders," *ISPRS International Journal of Geo-Information*, vol. 7, no. 4, p. 129, 2018.

[17] M. Rußwurm and M. Korner, "Temporal vegetation modelling using long short-term memory networks for crop identification from medium-resolution multi-spectral satellite images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 11–19.

[18] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean, "Inceptiontime: Finding alexnet for time series classification," *Data Mining and Knowledge Discovery*, vol. 34, no. 6, pp. 1936–1962, 2020.

[19] C. Pelletier, G. I. Webb, and F. Petitjean, "Temporal convolutional neural network for the classification of satellite image time series," *Remote Sensing*, vol. 11, no. 5, p. 523, 2019.

[20] M. Rußwurm and M. Körner, "Self-attention for raw optical satellite time series classification," *ISPRS journal of photogrammetry and remote sensing*, vol. 169, pp. 421–435, 2020.

[21] V. S. F. Garnot, L. Landrieu, S. Giordano, and N. Chehata, "Satellite image time series classification with pixel-set encoders and temporal self-attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 325–12 334.

[22] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[24] V. S. F. Garnot and L. Landrieu, "Lightweight temporal self-attention for classifying satellite images time series," in *International Workshop on Advanced Analytics and Learning on Temporal Data*. Springer, 2020, pp. 171–181.

[25] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[26] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[27] G. Papamakarios, "Neural density estimation and likelihood-free inference," *arXiv preprint arXiv:1910.13233*, 2019.

[28] Y. Song and D. P. Kingma, "How to train your energy-based models," *arXiv preprint arXiv:2101.03288*, 2021.

[29] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves *et al.*, "Conditional image generation with pixelcnn decoders," *Advances in neural information processing systems*, vol. 29, 2016.

[30] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," in *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*. ISCA, 2016, p. 125.

[31] X. Chen, N. Mishra, M. Rohaninejad, and P. Abbeel, "Pixelsnail: An improved autoregressive generative model," in *International Conference on Machine Learning*. PMLR, 2018, pp. 864–872.

[32] V. Bogachev, A. V. Kolesnikov, and K. Medvedev, "Triangular transformations of measures," *Sbornik Mathematics*, vol. 196, pp. 309–335, 2004.

[33] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *Journal of Machine Learning Research*, vol. 22, no. 57, pp. 1–64, 2021.

[34] C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville, "Neural autoregressive flows," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2078–2087.

[35] L. Dinh, D. Krueger, and Y. Bengio, "NICE: non-linear independent components estimation," in *3rd International Conference on Learning Representations, ICLR 2015, Workshop Track Proceedings*, 2015.

[36] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[37] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Improved variational inference with inverse autoregressive flow," *Advances in neural information processing systems*, vol. 29, 2016.

[38] G. Papamakarios, T. Pavlakou, and I. Murray, "Masked autoregressive flow for density estimation," *Advances in neural information processing systems*, vol. 30, 2017.

[39] J. Ho, X. Chen, A. Srinivas, Y. Duan, and P. Abbeel, "Flow++: Improving flow-based generative models with variational dequantization and architecture design," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2722–2730.

[40] T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák, "Neural importance sampling," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–19, 2019.

[41] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Cubic-spline flows," *arXiv preprint arXiv:1906.02145*, 2019.

[42] ——, "Neural spline flows," *Advances in neural information processing systems*, vol. 32, 2019.

[43] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *International Conference on Machine Learning*. PMLR, 2015, pp. 881–889.

[44] Y. Song, C. Meng, and S. Ermon, "Mintnet: Building invertible neural networks with masked convolutions," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[45] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," *Advances in neural information processing systems*, vol. 31, 2018.

[46] J. M. Tomczak and M. Welling, "Improving variational auto-encoders using householder flow," *arXiv preprint arXiv:1611.09630*, 2016.

[47] A. Golinski, M. Lezcano-Casado, and T. Rainforth, "Improving normalizing flows via better orthogonal parameterizations," in *ICML Workshop on Invertible Neural Networks and Normalizing Flows*, 2019.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[49] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538.

[50] R. van den Berg, L. Hasenclever, J. M. Tomczak, and M. Welling, "Sylvester normalizing flows for variational inference," in *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018*. AUAI Press, 2018, pp. 393–402.

[51] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen, "Invertible residual networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 573–582.

[52] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[54] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

[55] Y. Burda, R. B. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," in *4th International Conference on Learning Representations, ICLR 2016*, 2016.

[56] T. Rainforth, A. Kosiorek, T. A. Le, C. Maddison, M. Igl, F. Wood, and Y. W. Teh, "Tighter variational bounds are not necessarily better," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4277–4285.

[57] T. A. Le, A. R. Kosiorek, N. Siddharth, Y. W. Teh, and F. Wood, "Revisiting reweighted wake-sleep for models with stochastic control flow," in *Uncertainty in Artificial Intelligence*. PMLR, 2020, pp. 1039–1049.

[58] A. Mnih and K. Gregor, "Neural variational inference and learning in belief networks," in *International Conference on Machine Learning*. PMLR, 2014, pp. 1791–1799.

[59] A. Mnih and D. Rezende, "Variational inference for monte carlo objectives," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2188–2196.

[60] G. Roeder, Y. Wu, and D. K. Duvenaud, "Sticking the landing: Simple, lower-variance gradient estimators for variational inference," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[61] G. Tucker, D. Lawson, S. Gu, and C. J. Maddison, "Doubly reparameterized gradient estimators for monte carlo objectives," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[62] J. Bornschein and Y. Bengio, "Reweighted wake-sleep," in *3rd International Conference on Learning Representations, ICLR 2015*, 2015. [Online]. Available: http://arxiv.org/abs/1406.2751

[63] S. Nowozin, "Debiasing evidence approximations: On importance-weighted autoencoders and jackknife variational inference," in *International conference on learning representations*, 2018.

[64] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel, "The helmholtz machine," *Neural computation*, vol. 7, no. 5, pp. 889–904, 1995.

[65] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[66] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[67] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[68] C. J. Maddison, D. Tarlow, and T. Minka, "A* sampling," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[69] A. Potapczynski, G. Loaiza-Ganem, and J. P. Cunningham, "Invertible gaussian reparameterization: Revisiting the gumbel-softmax," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 311–12 321, 2020.

[70] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Józefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016*. ACL, 2016, pp. 10–21.

[71] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin, "Cyclical annealing schedule: A simple approach to mitigating KL vanishing," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019,*. Association for Computational Linguistics, 2019, pp. 240–250.

[72] X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel, "Variational lossy autoencoder," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[73] T. Lucas and J. Verbeek, "Auxiliary guided autoregressive variational autoencoders," in *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018*. Springer, 2018, pp. 443–458.

[74] A. B. Dieng, Y. Kim, A. M. Rush, and D. M. Blei, "Avoiding latent variable collapse with generative skip models," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 2397–2405.

[75] J. He, D. Spokoyny, G. Neubig, and T. Berg-Kirkpatrick, "Lagging inference networks and posterior collapse in variational autoencoders," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[76] B. Li, J. He, G. Neubig, T. Berg-Kirkpatrick, and Y. Yang, "A surprisingly effective fix for deep latent variable modeling of text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*. Association for Computational Linguistics, 2019, pp. 3601–3612.

[77] J. Tomczak and M. Welling, "Vae with a vampprior," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 1214–1223.

[78] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[79] M. D. Hoffman and M. J. Johnson, "Elbo surgery: yet another way to carve up the variational evidence lower bound," in *Workshop in Advances in Approximate Bayesian Inference, NIPS*, vol. 1, no. 2, 2016.

[80] X. Ding and K. Gimpel, "Flowprior: Learning expressive priors for latent variable sentence models," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 3242–3258.

[81] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[82] C. K. Sønderby, B. Poole, and A. Mnih, "Continuous relaxation training of discrete latent variable image models," in *Beysian DeepLearning workshop, NIPS*, vol. 201, 2017.

[83] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders," *Advances in neural information processing systems*, vol. 29, 2016.

[84] L. Maaløe, M. Fraccaro, V. Liévin, and O. Winther, "Biva: A very deep hierarchy of latent variables for generative modeling," *Advances in neural information processing systems*, vol. 32, 2019.

[85] A. Vahdat and J. Kautz, "Nvae: A deep hierarchical variational autoencoder," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 667–19 679, 2020.

[86] A. Razavi, A. Van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with vq-vae-2," *Advances in neural information processing systems*, vol. 32, 2019.

[87] W. Williams, S. Ringer, T. Ash, D. MacLeod, J. Dougherty, and J. Hughes, "Hierarchical quantized autoencoders," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4524–4535, 2020.

[88] L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda, "Dynamical variational autoencoders: A comprehensive review," *Found. Trends Mach. Learn.*, vol. 15, no. 1-2, pp. 1–175, 2021.

[89] D. Geiger, T. Verma, and J. Pearl, "Identifying independence in bayesian networks," *Networks*, vol. 20, no. 5, pp. 507–534, 1990.

[90] R. G. Krishnan, U. Shalit, and D. Sontag, "Deep kalman filters," *arXiv preprint arXiv:1511.05121*, 2015.

[91] R. Krishnan, U. Shalit, and D. Sontag, "Structured inference networks for nonlinear state space models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[92] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther, "A disentangled recognition and nonlinear dynamics model for unsupervised learning," *Advances in neural information processing systems*, vol. 30, 2017.

[93] J. Bayer and C. Osendorfer, "Learning stochastic recurrent networks," *arXiv preprint arXiv:1411.7610*, 2014.

[94] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," *Advances in neural information processing systems*, vol. 28, 2015.

[95] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," *Advances in neural information processing systems*, vol. 29, 2016.

[96] S. Leglaive, X. Alameda-Pineda, L. Girin, and R. Horaud, "A recurrent variational autoencoder for speech enhancement," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 371–375.

[97] L. Yingzhen and S. Mandt, "Disentangled sequential autoencoder," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5670–5679.

[98] M. Rußwurm, C. Pelletier, M. Zollner, S. Lefèvre, and M. Körner, "Breizhcrops: A time series dataset for crop type mapping," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences ISPRS (2020)*, 2020.

[99] G. Weikmann, C. Paris, and L. Bruzzone, "Timesen2crop: A million labeled samples dataset of sentinel 2 image time series for crop-type classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 4699–4708, 2021.

[100] M. Schneider, A. Broszeit, and M. Körner, "Eurocrops: A pan-european dataset for time series crop type classification," *arXiv preprint arXiv:2106.08151*, 2021.

[101] G. Tseng, I. Zvonkov, C. L. Nakalembe, and H. Kerner, "Cropharvest: A global dataset for crop-type classification," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.