Probabilistic Graphical Models and Deep Generative Models

not-MIWAE: Deep Generative Modelling with Missing Not at Random Data and a Supervised MNAR Extension (sup-not-MIWAE)

Amine Maazizi
ENSTA Paris; ENS Paris-Saclay, France
amine.maazizi@ensta.fr
amine.maazizi@ip-paris.fr

Adam Gassem
ENSTA Paris; ENS Paris-Saclay, France
adam.gassem@ensta.fr
adam.gassem@ip-paris.fr

Ewerthon Melzani
ENSTA Paris; ENS Paris-Saclay, France
ewerthon.melzani@ensta.fr
ewerthon.melzani@ip-paris.fr

Feb 28, 2026 – v1

Abstract

We study deep latent variable models under Missing Not At Random (MNAR) covariate mechanisms, where the missingness pattern depends on unobserved values and thus cannot be ignored during likelihood-based training. We first reproduce the *not-MIWAE* framework, which jointly models the data distribution and the missingness process and is trained via importance-weighted variational inference. We then contribute (i) an optimal-transport viewpoint on classical single imputation, recasting common loss-minimizing point estimators as Wasserstein projections onto Dirac measures and clarifying the associated collapse of posterior uncertainty; (ii) a supervised MNAR extension, *sup-not-MIWAE*, obtained by a direct probabilistic integration of not-MIWAE and supMIWAE without domain-specific architectural modifications; and (iii) an extended experimental evaluation including high-dimensional image data, adapting the MNAR clipping experiment to CelebA and reporting quantitative and qualitative results. We release a unified PyTorch implementation of MIWAE, not-MIWAE, supMIWAE, and sup-not-MIWAE as open-source software.

Keywords: MNAR, missing data, deep generative models, variational inference, importance sampling, IWAE/MIWAE, imputation, optimal transport, supervised learning.

Reproducibility: Code is available at GitHub and as an installable package at PyPI.

1 Introduction

1.1 Presentation of the article

This work reproduces and extends the framework introduced in *"not-MIWAE: Deep Generative Modelling with Missing Not at Random Data"* [1] . The authors address the challenge of performing likelihood-based inference in Deep Latent Variable Models (DLVMs) when data is Missing Not At Random (MNAR). The paper proposes a method to explicitly model the missing data mechanism jointly with the data distribution, utilizing importance-weighted variational inference to maximize a lower bound of the joint likelihood.

1.2 Contribution

Beyond reviewing and reproducing the not-MIWAE framework, this work makes the following contributions.

First, we introduce an optimal-transport interpretation of imputation under MNAR, recasting classical loss-based point estimators as Wasserstein projections onto Dirac measures. This perspective clarifies the implicit collapse of posterior uncertainty induced by single imputation and provides a principled motivation for distributional imputation strategies that preserve higher-order uncertainty.

Second, we propose a supervised MNAR extension, **sup-not-MIWAE** (an extension of supMIWAE [2]), obtained by a direct and faithful probabilistic integration of not-MIWAE and supMIWAE within the same modeling and inference setting as not-MIWAE, deliberately avoiding domain-specific architectural or training modifications. This formulation is not intended to outperform specialized domain-tailored approaches, but rather to provide a transparent supervised MNAR baseline that remains directly comparable to the original models.

Third, we reproduce and extend the original experimental study by evaluating not-MIWAE on high-dimensional image data, adapting the MNAR clipping experiment to the CelebA dataset and reporting both quantitative and qualitative results that demonstrate effective imputation and recovery of the underlying missingness mechanism.

Finally, we provide unified PyTorch implementations of MIWAE, not-MIWAE, supMIWAE, and sup-not-MIWAE, released as open-source code on GitHub and distributed as an installable Python package via PyPI (discussed mainly in appendix B) , enabling reproducible experimentation and facilitating further research.

1.3 Related Work

MIWAE [3] extends importance-weighted variational objectives to missing-data settings under ignorable mechanisms (MCAR/MAR), enabling robust likelihood-based learning and imputation. not-MIWAE [1] addresses MNAR by explicitly modeling the missingness process and optimizing a lower bound on the joint likelihood of observed values and masks. For supervised learning with missing covariates, supMIWAE [2] marginalizes predictions over the missing-feature posterior rather than relying on single imputation. Our work is positioned as a reproduction and consolidation of these frameworks, with two extensions: an optimal-transport interpretation of single imputation under MNAR, and a faithful supervised MNAR baseline (sup-not-MIWAE) that preserves the original probabilistic modeling assumptions for direct comparability.

1.4 Notation and Hypotheses

Notation

We adopt the notation used in the paper. We assume a dataset $\boldsymbol{X} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)^\top$ containing n i.i.d. copies of a random variable $\boldsymbol{x}$ in a p-dimensional feature space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$.

For any sample x, the values are partitioned into an observed part $\boldsymbol{x}^o$ and a missing part $\boldsymbol{x}^m$, such that $\boldsymbol{x} = (\boldsymbol{x}^o, \boldsymbol{x}^m)$. The missingness pattern is defined by a binary mask variable $\boldsymbol{s} \in \{0,1\}^p$, defined as:

$$s_j = \begin{cases} 1 & \text{if } x_j \text{ is observed,} \\ 0 & \text{if } x_j \text{ is missing.} \end{cases}$$

The full probabilistic model involves two sets of parameters: θ for the data generating process $p_\theta(x)$, and ϕ for the missing data mechanism $p_\phi(\boldsymbol{s}|\boldsymbol{x})$.

Hypotheses

The probabilistic framework categorizes the missing data mechanism based on the conditional distribution of the mask $\boldsymbol{s}$:

- **MCAR (Missing Completely at Random):** The missingness does not depend on the data values, i.e., $p_\phi(\boldsymbol{s}|\boldsymbol{x}) = p_\phi(\boldsymbol{s})$.
- **MAR (Missing at Random):** The missingness depends only on the observed data, i.e., $p_\phi(\boldsymbol{s}|\boldsymbol{x}) = p_\phi(s|\boldsymbol{x}^o)$.
- **MNAR (Missing Not at Random):** The missingness, in our case, depends on both the observed and the missing data, i.e., $p_\phi(\boldsymbol{s}|\boldsymbol{x})$ cannot be simplified and depends on $\boldsymbol{x}^m$.

1.5 The Main Problem

The central problem addressed in the paper is performing inference when the MNAR assumption holds. To maximize the likelihood, we must consider the joint distribution of the observed data $\boldsymbol{x}^o$ and the mask s, which is obtained by integrating out the missing data $\boldsymbol{x}^m$:

$$p_{\theta,\phi}(\boldsymbol{x}^o, s) = \int \underbrace{p_\theta(\boldsymbol{x}^o, \boldsymbol{x}^m)}_{\text{Data Model}} \underbrace{p_\phi(s|\boldsymbol{x}^o, \boldsymbol{x}^m)}_{\text{Missing Model}} d\boldsymbol{x}^m \tag{1}$$

In the MCAR and MAR cases, the missing model $p_\phi(\boldsymbol{s}|\dots)$ does not depend on the variable of integration $\boldsymbol{x}^m$. Consequently, it can be moved outside the integral, allowing the likelihood to factorize:

$$p_{\theta,\phi}(\boldsymbol{x}^o, s) \propto p_\theta(\boldsymbol{x}^o)$$

However, in the **MNAR** case, the term $p_\phi(\boldsymbol{s}|\boldsymbol{x}^o, \boldsymbol{x}^m)$ depends explicitly on $\boldsymbol{x}^m$. The likelihood in euqation 1does not factorize, meaning the parameters of the data generating process (θ) and the missing data mechanism (ϕ) are tied together. Ignoring this coupling leads to biased estimates, which necessitates the joint modeling approach proposed by the authors.

2 Model

2.1 Derivation of the MNAR Log-Likelihood

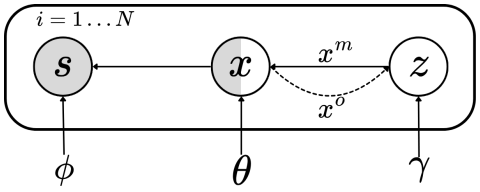

Figure 1: Graphical Model

The paper considers Deep Latent Variable Models (DLVMs) where the data x is generated from a latent variable z. To derive the likelihood function used for optimization, we start with the full joint distribution of all variables (latent, observed, missing, and mask) and marginalize out the unobserved variables.

$$p(\boldsymbol{x}^o, \boldsymbol{x}^m, \boldsymbol{s}, \boldsymbol{z}) = p(\boldsymbol{z}) \cdot p_\theta(\boldsymbol{x}^o, \boldsymbol{x}^m|\boldsymbol{z}) \cdot p_\phi(\boldsymbol{s}|\boldsymbol{x}^o, \boldsymbol{x}^m)$$

The core assumption in this framework is *conditional independence*: given the latent code $\boldsymbol{z}$, the features of $\boldsymbol{x}$ are independent of one another. Mathematically, this implies that the observation model factorizes as $p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \prod_j p_\theta(x_j|\boldsymbol{z})$. This assumption is critical because it allows us to neatly split the probability of the full data into observed and missing components: $p_\theta(\boldsymbol{x}|\boldsymbol{z}) = p_\theta(\boldsymbol{x}^o|\boldsymbol{z})p_\theta(\boldsymbol{z})$.

To obtain the log-likelihood of the observed quantities $p(\boldsymbol{x}^o, s)$, we integrate out the unobserved variables z and $\boldsymbol{x}^m$:

$$\mathcal{L}(\theta, \phi) = \log \iint p_\phi(s|\boldsymbol{x}^o, \boldsymbol{x}^m) p_\theta(\boldsymbol{x}^o|\boldsymbol{z}) p_\theta(\boldsymbol{x}^m|\boldsymbol{z}) p(\boldsymbol{z}) \; d\boldsymbol{z} \, d\boldsymbol{x}^m \tag{2}$$

This integral highlights the difficulty of MNAR: the missing data $\boldsymbol{x}^m$ connects the missing mechanism p_ϕ and the generative model p_θ, preventing simplification.

2.2 Variational Lower Bound and Importance Sampling

To make the maximization of the log-likelihood tractable, the authors employ an importance sampling approach. We introduce a variational distribution $q_\gamma(\boldsymbol{z}|\boldsymbol{x}^o)$ to approximate the true posterior of the latent variables.

$$p_{\theta,\phi}(\boldsymbol{x}^o, \boldsymbol{s}) = \iint \frac{p_\phi(s|\boldsymbol{x}^o, \boldsymbol{x}^m) p_\theta(\boldsymbol{x}^o|\boldsymbol{z}) p(\boldsymbol{z})}{q_\gamma(\boldsymbol{z}|\boldsymbol{x}^o)} \cdot \underbrace{q_\gamma(\boldsymbol{z}|\boldsymbol{x}^o) p_\theta(\boldsymbol{x}^m|\boldsymbol{z})}_{\text{Joint Proposal}} \, d\boldsymbol{z} \, d\boldsymbol{x}^m$$

Specifically, we sample $\boldsymbol{z}$ from the inference network and then sample $\boldsymbol{x}^m$ from the generative model conditioned on that $\boldsymbol{z}$.

Therefore, the integral can be rewritten as an expectation:

$$p_{\theta,\phi}(\boldsymbol{x}^o, \boldsymbol{s}) = \mathbb{E}_{\substack{\boldsymbol{z} \sim q_\gamma(\boldsymbol{z}|\boldsymbol{x}^o) \\ \boldsymbol{x}^m \sim p_\theta(\boldsymbol{x}^m|\boldsymbol{z})}} \left[\frac{p_\phi(\boldsymbol{s}|\boldsymbol{x}^o, \boldsymbol{x}^m) p_\theta(\boldsymbol{x}^o|\boldsymbol{z}) p(\boldsymbol{z})}{q_\gamma(z|\boldsymbol{x}^o)} \right] \tag{3}$$

To optimize this quantity, the expectation is approximated using a Monte Carlo estimate with K importance samples. This leads to the objective function $\mathcal{L}_K(\theta, \phi, \gamma)$ used in the paper:

$$\mathcal{L}_K(\theta, \phi, \gamma) = \sum_{i=1}^{n} \mathbb{E} \left[\log \frac{1}{K} \sum_{k=1}^{K} w_{ki} \right] \tag{4}$$

where the importance weights w_{ki} are given by:

$$w_{ki} = \frac{p_\phi(s_i|\boldsymbol{x}_i^o, \boldsymbol{x}_{ki}^m) p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki}) p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)}$$

and $(\boldsymbol{z}_{ki}, \boldsymbol{x}_{ki}^m)$ are K i.i.d. samples drawn from the proposal $q_\gamma(\boldsymbol{z}|\boldsymbol{x}_i^o)p_\theta(\boldsymbol{x}^m|\boldsymbol{z})$. Jensen's inequality guarantees that this objective is a lower bound on the true likelihood. Furthermore, the sequence of bounds $\mathcal{L}_K$ converges monotonically to the true log-likelihood $\ell(\theta, \phi)$ as $K \to \infty$ (proof in appendix A).

2.3 Imputation

Once the model parameters are learned, the generative model can be employed to impute missing values. As formally derived in Appendix B of [1], imputation can be cast as a statistical decision problem where the goal is to select an

estimator $\hat{\boldsymbol{x}}^m$ that minimizes the expected risk under the posterior distribution $p_{\theta,\phi}(\boldsymbol{x}^m|\boldsymbol{x}^o,\boldsymbol{s})$:

$$\hat{\boldsymbol{x}}^m = \arg\min_{\hat{\boldsymbol{x}}^m} \mathbb{E}_{\boldsymbol{x}^m}\left[L(\boldsymbol{x}^m,\hat{\boldsymbol{x}}^m)|\boldsymbol{x}^o,\boldsymbol{s}\right]. \tag{5}$$

The choice of the loss function L dictates the form of the optimal estimator.

Squared Loss When L corresponds to the squared error, the risk minimizer is the conditional mean. This expectation can be estimated via Self-Normalized Importance Sampling (SNIS) [3]. The estimate is given by:

$$\hat{\boldsymbol{x}}^m = \mathbb{E}[\boldsymbol{x}^m|\boldsymbol{x}^o,\boldsymbol{s}] \approx \sum_{k=1}^{K} \alpha_k \,\mathbb{E}[\boldsymbol{x}^m|\boldsymbol{z}_k], \quad \text{with } \alpha_k = \frac{w_k}{\sum_{j=1}^{K} w_j} \tag{6}$$

where w_k are the importance weights derived previously and $\mathbb{E}[\boldsymbol{x}^m|\boldsymbol{z}_k]$ represents the mean of the decoder distribution.

Absolute Loss Alternatively, to prioritize robustness against outliers, we may minimize the absolute error. The optimal estimator in this case is the conditional median. This is computed by estimating the cumulative distribution function (CDF) for each missing feature j:

$$F_j(x_j) = \mathbb{E}[\mathbb{1}_{x_j^m \le x_j}|\boldsymbol{x}^o,\boldsymbol{s}] \approx \sum_{k=1}^{K} \alpha_k F_{x_j|\boldsymbol{z}_k}(x_j), \tag{7}$$

where $F_{x_j|\boldsymbol{z}_k}$ is the CDF of the observation model (often available in closed-form). The imputed value $\hat{x}_j^m$ is then obtained by solving $F_j(x_j) = 0.5$.

2.3.1 Geometric Interpretation and Limitations

From an Optimal Transport perspective, classical imputation strategies can be reinterpreted not merely as loss minimization, but as a variational approximation problem. We seek a surrogate distribution q that approximates the complex posterior $\nu = p_{\theta,\phi}(\cdot|\boldsymbol{x}^o,\boldsymbol{s})$ by minimizing the Wasserstein-p distance:

$$\min_{q\in\mathcal{Q}} W_p^p(q,\nu) = \min_{q\in\mathcal{Q}} \inf_{\gamma\in\Pi(q,\nu)} \int ||u-v||^p \, d\gamma(u,v). \tag{8}$$

Standard point estimators implicitly constrain the variational family $\mathcal{Q}$ to the set of Dirac measures, i.e., $q = \delta_{\hat{\boldsymbol{x}}^m}$. Under this strict constraint, the transport plan is unique, and the Wasserstein distance collapses to the expected risk:

$$W_p^p(\delta_{\hat{\boldsymbol{x}}^m},\nu) = \mathbb{E}_{\boldsymbol{x}^m\sim\nu}\left[||\hat{\boldsymbol{x}}^m - \boldsymbol{x}^m||^p\right]. \tag{9}$$

Consequently, minimizing W_2 recovers the conditional mean (Squared Loss), while minimizing W_1 recovers the conditional median (Absolute Loss).

Limitation and Generalization: This geometric view exposes a fundamental limitation of standard single imputation: by enforcing q to be a Dirac mass, we collapse the entire posterior uncertainty into a point estimate (instead of considering the estimation as a possible distribution) more robust generalization-avoiding this information loss would involve relaxing the constraint on $\mathcal{Q}$ to allow for parametric distributions (e.g., Gaussians or mixtures) rather than point fixed masses. This would allow the imputed model to capture higher-order moments and preserve the aleatoric uncertainty inherent in the missing data.

2.4 Using Prior Information

Effectively recovering the full data distribution in an MNAR setting requires assumptions or prior knowledge about the missingness mechanism, as general MNAR models may lead to inconsistent estimation (Molenberghs et al., 2008 [4]).

The missing model $p_\phi(\boldsymbol{s}|\boldsymbol{x})$ effectively acts as a classifier that predicts the mask s based on the full data x. It is typically modeled as a Bernoulli distribution:

$$p_\phi(\boldsymbol{s}|\boldsymbol{x}) = \prod_{j=1}^{p} \pi_{\phi,j}(\boldsymbol{x})^{s_j}(1-\pi_{\phi,j}(\boldsymbol{x}))^{1-s_j}$$

where $\pi_{\phi,j}(\boldsymbol{x})$ is the probability that feature j is observed. The flexibility of this mapping allows for the incorporation of prior knowledge:

- **Agnostic/General:** π_ϕ can be a generic neural network if the mechanism is unknown.

- **Self-Masking:** If we know missingness depends only on the value itself (e.g., self-censoring), we can restrict the model to $\pi_{\phi,j}(\boldsymbol{x}) = \sigma(ax_j + b)$, significantly reducing the search space and improving identifiability.

By explicitly designing the missing model to reflect known constraints, the joint optimization can better disentangle the data distribution parameters θ from the missingness parameters ϕ as shown in the different experiments4.1.

3 A Supervised Extension: sup-not-MIWAE

As noted in [1], the not-MIWAE model can be extended to supervised learning settings with covariates missing not at random by adapting the methodology introduced in [2]. While not-MIWAE was originally proposed for density estimation and imputation under Missing Not At Random (MNAR) mechanisms, many practical applications instead require *supervised learning*, where the objective is to predict an outcome variable y (for classification or regression) from partially observed covariates x. In the following, we present a natural supervised extension of not-MIWAE, inspired by the *Supervised MIWAE (supMIWAE)* framework originally developed under the MAR assumption. Recent work by [5] addresses a closely related objective in the context of MNAR multivariate time series classification, proposing a conceptually aligned but practically more elaborate framework with domain-specific architectural and training modifications; in contrast, our work focuses on a simpler and more faithful probabilistic integration of not-MIWAE and supMIWAE, evaluated in settings comparable to those of the original foundational models.

3.1 Motivation

The key idea behind supMIWAE is to minimize the expected prediction error by *marginalizing over missing features* instead of relying on single imputation. This is achieved by jointly modeling the covariates x and the labels y, and by computing predictions of the form

$$p(y|\boldsymbol{x}^o) = \int p(y|\boldsymbol{x}^o, \boldsymbol{x}^m)\, p(\boldsymbol{x}^m|\boldsymbol{x}^o)\, d\boldsymbol{x}^m.$$

Under MAR, the missingness mechanism is ignorable, and the marginal $p(\boldsymbol{x}^m|\boldsymbol{x}^o)$ can be learned using MIWAE.

However, under MNAR, the conditional distribution of the missing values depends on the missingness pattern itself. As shown in previous sections, this requires explicit modeling of the mask variable s. Consequently, extending supMIWAE to MNAR settings requires combining the supervised marginalization principle with the joint modeling strategy of not-MIWAE tackled in Section 2. We refer to this extension as **sup-not-MIWAE**.

3.2 Method

We obtain a supervised extension of not-MIWAE by attaching a predictor head to the same joint model that couples the data-generating process and the missingness mechanism. Concretely, we keep the not-MIWAE MNAR likelihood $p_\theta(x \mid z)\, p_\phi(s \mid x)$ and add a discriminative term $p_\varphi(y \mid x)$ so that labels are predicted from the *complete* covariates (observed and missing), while inference remains conditioned only on x^o. The resulting graphical model (Fig. 2) augments not-MIWAE with a supervised node y depending on x:

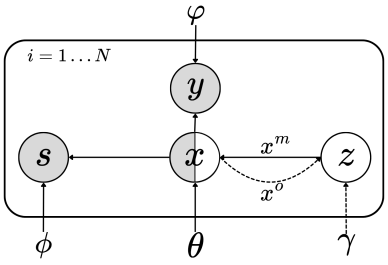

Figure 2: Graphical model of sup-not-MIWAE

Given (x_i^o, s_i, y_i), the supervised MNAR marginal couples prediction and mask modeling through the missing values:

$$p(x_i^o, y_i, s_i) = \int p_\theta(x_i^o, x_i^m)\, p_\varphi(y_i \mid x_i^o, x_i^m)\, p_\phi(s_i \mid x_i^o, x_i^m)\, dx_i^m.$$

Using the same importance-weighted variational strategy as in not-MIWAE, with $z_{ki} \sim q_\gamma(z \mid x_i^o)$ and $x_{ki}^m \sim p_\theta(x^m \mid z_{ki})$, we maximize:

$$\mathcal{L}_K^{\text{sup}} = \sum_{i=1}^{n} \mathbb{E}\left[\log \frac{1}{K} \sum_{k=1}^{K} w_{ki}^{\text{sup}}\right], \qquad w_{ki}^{\text{sup}} = \frac{p_\phi(s_i \mid x_i^o, x_{ki}^m)\, p_\varphi(y_i \mid x_i^o, x_{ki}^m)\, p_\theta(x_i^o \mid z_{ki})\, p(z_{ki})}{q_\gamma(z_{ki} \mid x_i^o)}.$$

3.3 Prediction via Marginalization

At test time, predictions are obtained by marginalizing over the missing features:

$$p(y|\boldsymbol{x}^o, \boldsymbol{s}) = \int p_\varphi(y|\boldsymbol{x}^o, \boldsymbol{x}^m)\, p_{\theta,\phi}(\boldsymbol{x}^m|\boldsymbol{x}^o, \boldsymbol{s})\, d\boldsymbol{x}^m.$$

In practice, this integral is approximated using self-normalized importance sampling:

$$p(y|\boldsymbol{x}^o, \boldsymbol{s}) \approx \sum_{k=1}^{K} \alpha_k\, p_\varphi(y|\boldsymbol{x}^o, \boldsymbol{x}_k^m), \quad \alpha_k = \frac{w_k^{\text{sup}}}{\sum_{j=1}^{K} w_j^{\text{sup}}}.$$

This mirrors the prediction rule of supMIWAE, but with importance weights that explicitly account for the MNAR missingness mechanism.

4 Experiments and Results

We applied the model to various MNAR scenarios (selfmasking missing process) to evaluate its efficacy in recovering missing values compared to standard baselines including Mean imputation, MICE (Multivariate Imputation by Chained Equations), and the original MIWAE (Missing Data Importance Weighted Autoencoder), which assumes MAR/MCAR.

The primary metric for evaluation is the Root Mean Squared Error (RMSE) calculated on the missing values. Since the MNAR mechanism is introduced synthetically in our experiments, we have access to the ground truth for these missing values, allowing for a precise calculation of imputation accuracy.

4.1 Results

Table 1: Imputation RMSE on UCI datasets under MNAR self-masking. Lower is better.

Model	**Banknote** (61.8%)	**Concrete** (50.3%)	**Breast** (43.6%)	**White** (44.6%)	**CelebA** (Images)
Mean	1.546	1.329	1.548	1.460	0.466
KNN	1.543	1.366	1.290	1.410	0.350
MICE	1.540	1.247	1.130	1.290	–
MIWAE	1.243	1.148	1.113	1.310	–
not-MIWAE variants					
Self-masking (Unknown parameters)	**0.770**	**1.099**	**0.836**	**0.950**	**0.093**
Linear	1.023	1.149	1.018	1.040	–
Non-linear	1.133	1.128	0.938	1.080	–

Table 1 presents the quantitative results. Standard imputation methods (Mean, KNN, MICE) and the MAR-assuming MIWAE exhibit significantly higher error rates, demonstrating their inability to correct for the bias introduced by the MNAR mechanism. In contrast, the *not-MIWAE* model equipped with a self-masking missing mechanism consistently achieves the lowest RMSE across all datasets, outperforming the baselines and other not-MIWAE variants (linear/non-linear), highlighting the value of incorporating structural prior knowledge about the missing process. The CelebA result further confirms this advantage in high-dimensional image data, where the model achieves a remarkably low RMSE of 0.093 (using CNN based VAE).

4.2 Clipping in CelebA Images

To evaluate the model's capability in handling MNAR mechanisms in high-dimensional and complex data, we reproduced the clipping experiment described in the original paper. However, we replaced the SVHN dataset with the CelebA dataset, consisting of celebrity portraits which we resized to 32×32 pixels. This task is arguably more challenging due to the greater complexity and variation in face images compared to house numbers.

We simulated a self-masking missing mechanism that emulates the "clipping" phenomenon, where pixel intensities above a certain threshold are censored. The missing mask s_{ij} for pixel x_{ij} was Bernoulli sampled according to the probability:

$$P(s_{ij} = 1|x_{ij}) = \frac{1}{1 + e^{-\text{logits}}}, \quad \text{logits} = W(x_{ij} - b) \tag{10}$$

We used parameters $W = -50$ and $b = 0.75$.

We trained the not-MIWAE model using a "known" missing process configuration, where the functional form of the self-masking mechanism (logistic regression) was provided to the model, but the parameters were initialized randomly. The model successfully recovered the parameters of the missing mechanism, converging to values very close to the ground truth W and b. This indicates that the joint optimization effectively disentangled the data distribution from the missingness mechanism. Figure 3 illustrates the qualitative results of the experiment.

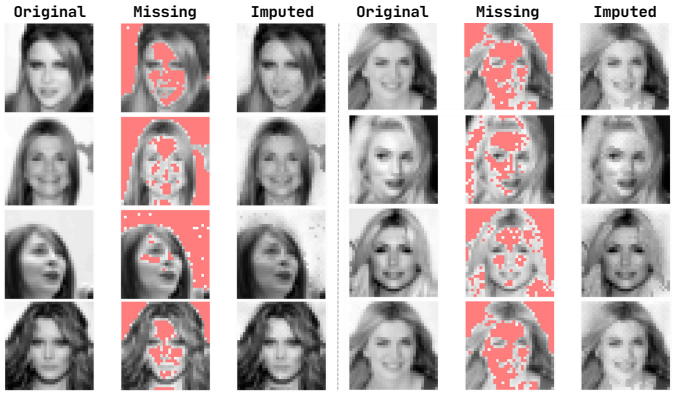

Figure 3: Visual results on CelebA dataset (32×32). Left: Original images. Middle column: Images with simulated MNAR clipping ($b = 0.75$). Right column: Imputations generated by not-MIWAE.

4.3 Supervised Learning under MNAR

We evaluate supervised learning under MNAR covariates using synthetic self-masking, comparing MNAR-aware models to standard imputation baselines across classification and regression tasks.

For classification, we use the Covertype dataset with 20,000 samples. For regression, we use the UCI Wine Quality (Red) dataset with 1,599 sample. In both cases, approximately 30% MNAR missingness is introduced via self-masking. Following [2], we employ limited-capacity predictive models to isolate the effect of uncertainty handling under missing data.

Model	Accuracy
Mean + Model*[1]	0.5405
MICE + Model*	0.5425
Two-Stage (not-MIWAE + Model*)	0.5445
Sup-not-MIWAE	**0.5690**
Oracle (no missing data)	0.7123

Table 2: Classification on Covertype under MNAR missingness.

Model	Test RMSE
Mean + Model*	0.6966
MICE + Model*	0.6713
Two-Stage (not-MIWAE + Model*)	**0.6503**
Sup-not-MIWAE	0.6718
Oracle (no missing data)	0.6173

Table 3: Regression on Wine Quality (Red) under MNAR missingness.

Across both tasks, Sup-not-MIWAE consistently outperforms naive imputation baselines, confirming the benefit of explicitly modeling the MNAR mechanism and marginalizing over missing covariates during prediction. In classification, Sup-not-MIWAE achieves the best performance among MNAR-trained models in the limited-capacity regime; this behavior is consistent with prior observations that marginalization-based approaches are particularly effective when the discriminative learner cannot absorb imputation uncertainty [2]. In regression, the two-stage pipeline achieves the lowest RMSE, reflecting the fact that accurate point imputations can be sufficient when paired with a strong regressor; nevertheless, Sup-not-MIWAE performed comparably to the two-stage pipeline.

5 Conclusion

In this work, we successfully reproduced the not-MIWAE framework and introduced **sup-not-MIWAE**, extending the approach to supervised learning under MNAR mechanisms. Our experiments on UCI datasets and high-dimensional CelebA images confirm that explicitly modeling the missingness mechanism significantly reduces imputation error compared to MAR-based baselines.

5.1 Limitations and Critical Analysis

Despite these successes, our analysis highlights several critical limitations inherent to the framework. First, there is a fundamental **flexibility trade-off**. The not-MIWAE relies on joint optimization of a deep generative data model (p_θ) and a missingness model (p_ϕ). We observed that simultaneously employing highly flexible deep neural

[1] *Model denotes a logistic regression classifier for classification tasks and a ridge regression model for regression tasks.

networks for both components can cause the model to be "led astray," failing to disentangle the data structure from the missingness artifacts due to identifiability issues. Consequently, successful deployment often requires constraining one of the components (e.g., using a linear missingness model or a simpler PPCA data model), which limits the method's universality.

Second, this identifiability issue creates a strong **dependence on prior domain knowledge**. Unlike "black-box" deep learning approaches, not-MIWAE requires the practitioner to correctly hypothesize the functional form of the missing mechanism (e.g., self-masking) to achieve state-of-the-art results. Third, the reliance on the **reparameterization trick** in the data space restricts the method primarily to continuous data distributions (e.g., Gaussian, Student-t). Extending this to discrete or mixed data types requires continuous relaxations (e.g., Gumbel-Softmax), which introduces approximation errors and hyperparameter complexity.

Finally, the approach is **computationally intensive**, especially for high-dimensional data. The requirement to generate and process multiple importance samples for every data point significantly increases memory usage and training time. For instance, accurate evaluation is resource-heavy: the imputation RMSE is estimated using 10,000 importance samples and the mean and standard errors are found over 5 runs, often necessitating hundreds of thousands of training iterations to achieve convergence.

5.2 Future Directions: A Distributional Perspective

A significant direction for future research involves revisiting the nature of imputation itself. Throughout this work, we treated imputed values as Dirac masses—point estimates minimizing a specific loss. As discussed in Section 2, this collapses the posterior uncertainty.

We propose extending this theory to a **distributional framework** potentially grounded in Optimal Transport. Rather than assigning a single deterministic value to a missing pixel, future models should aim to output a probability measure (e.g., a Gaussian or a mixture). This could be achieved via a *Hybrid Approach*, where observed points are modeled as Dirac measures and missing points as parametric distributions. This would allow the model to quantify the aleatoric uncertainty inherent to the MNAR process, providing confidence intervals alongside imputations—a crucial feature for high-stakes decision-making.

Acknowledgements

We thank our instructors for detailed feedback on earlier versions of this manuscript. We also thank the maintainers of the referenced open-source libraries and datasets used in our experiments.

References

[1] Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. not-miwae: Deep generative modelling with missing not at random data. In *International Conference on Learning Representations (ICLR)*, 2021.

[2] Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. How to deal with missing data in supervised deep learning? In *International Conference on Learning Representations (ICLR)*, 2022. Earlier version appeared as a workshop submission (Artemiss, 2020).

[3] Pierre-Alexandre Mattei and Jes Frellsen. Leveraging the exact likelihood of deep latent variable models. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pages 3859–3870, 2018.

[4] Geert Molenberghs and Michael G. Kenward. *Missing Data in Clinical Studies*. Wiley, 2007.

[5] Seunghyun Kim, Hyunsu Kim, Eunggu Yun, Hwangrae Lee, Jaehun Lee, and Juho Lee. Probabilistic imputation for time-series classification with missing data. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 16654–16667, 2023.

[6] Justin Domke and Daniel R. Sheldon. Importance weighting and variational inference. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, 2018.

[7] Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations (ICLR)*, 2016.

A Proof of Convergence and Bounds for $\mathcal{L}_K$

This section details the theoretical properties of the lower bound objective $\mathcal{L}_K(\theta, \phi, \gamma)$. The objective is derived from importance-weighted variational inference and is computed by drawing K samples from the proposal distribution. We define the importance weights w_{ki} as:

$$w_{ki} = \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)} \tag{11}$$

The objective is given by $\mathcal{L}_K = \sum_{i=1}^n \mathbb{E}[\log \frac{1}{K} \sum_{k=1}^K w_{ki}]$. We prove the following three properties:

1. **Upper Bound Check:** $\log p_{\theta,\phi}(\boldsymbol{x}^o, \boldsymbol{s}) \geq \mathcal{L}_K$,
   Using Jensen's inequality (concave case) gives that

$$\mathcal{L}_K = \sum_{i=1}^n \mathbb{E}\left[\log \frac{1}{K} \sum_{k=1}^K \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)}\right] \tag{12}$$

$$\leq \sum_{i=1}^n \log \mathbb{E}\left[\frac{1}{K} \sum_{k=1}^K \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)}\right] = \sum_{i=1}^n \log p_{\theta,\phi}(\boldsymbol{x}_i^o,\, \boldsymbol{s}_i). \tag{13}$$

2. **Monotonicity:** $\mathcal{L}_K \geq \mathcal{L}_L$ for $K \geq L$
   Let $I \subset \{1, \ldots, K\}$ with $|I| = L$ be a uniformly distributed subset of distinct indices from $\{1, \ldots, K\}$. We will use the following simple observation: $\mathbb{E}_{I=\{k_1,\ldots,k_L\}}\left[\frac{a_{k_1}+\cdots+a_{k_L}}{L}\right] = \frac{a_1+\cdots+a_K}{K}$ for any sequence of numbers $a_1, \ldots, a_K$. Using this observation and Jensen's inequality, we get

$$\mathcal{L}_K = \sum_{i=1}^n \mathbb{E}_{(\boldsymbol{x}_{1i}^m, \boldsymbol{z}_{1i}),\ldots,(\boldsymbol{x}_{Ki}^m, \boldsymbol{z}_{Ki})}\left[\log \frac{1}{K} \sum_{k=1}^K \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)}\right] \tag{14}$$

$$= \sum_{i=1}^n \mathbb{E}_{(\boldsymbol{x}_{1i}^m, \boldsymbol{z}_{1i}),\ldots,(\boldsymbol{x}_{Ki}^m, \boldsymbol{z}_{Ki})}\left[\log \mathbb{E}_{I=\{k_1,\ldots,k_L\}}\left\{\frac{1}{L} \sum_{l=1}^L \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{k_l i}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{k_l i})\, p(\boldsymbol{z}_{k_l i})}{q_\gamma(\boldsymbol{z}_{k_l i}|\boldsymbol{x}_i^o)}\right\}\right] \tag{15}$$

$$\geq \sum_{i=1}^n \mathbb{E}_{(\boldsymbol{x}_{1i}^m, \boldsymbol{z}_{1i}),\ldots,(\boldsymbol{x}_{Ki}^m, \boldsymbol{z}_{Ki})}\left[\mathbb{E}_{I=\{k_1,\ldots,k_L\}}\left\{\log \frac{1}{L} \sum_{l=1}^L \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{k_l i}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{k_l i})\, p(\boldsymbol{z}_{k_l i})}{q_\gamma(\boldsymbol{z}_{k_l i}|\boldsymbol{x}_i^o)}\right\}\right] \tag{16}$$

$$= \sum_{i=1}^n \mathbb{E}_{(\boldsymbol{x}_{1i}^m, \boldsymbol{z}_{1i}),\ldots,(\boldsymbol{x}_{Li}^m, \boldsymbol{z}_{Li})}\left[\log \frac{1}{L} \sum_{l=1}^L \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{li}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{li})\, p(\boldsymbol{z}_{li})}{q_\gamma(\boldsymbol{z}_{li}|\boldsymbol{x}_i^o)}\right] = \mathcal{L}_L \tag{17}$$

3. **Convergence:** $\lim_{K\to\infty} \mathcal{L}_K = \log p_{\theta,\phi}(\boldsymbol{x}^o, \boldsymbol{s})$
   Consider the random variable M_{Ki} defined as the sample mean:

$$M_{Ki} = \frac{1}{K} \sum_{k=1}^K \frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)} \tag{18}$$

   To apply the Strong Law of Large Numbers (SLLN), we analyze the term inside the summation. If this term is in L^1, meaning its absolute expectation is finite:

$$\mathbb{E}\left|\frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)}\right| < \infty \tag{19}$$

   then, by the SLLN, the sample mean M_{Ki} converges almost surely (a.s.) to its true expected value as $K \to \infty$:

$$\lim_{K\to\infty} M_{Ki} \xrightarrow{a.s.} \mathbb{E}\left[\frac{p_\phi(\boldsymbol{s}_i|\boldsymbol{x}_i^o,\, \boldsymbol{x}_{ki}^m)\, p_\theta(\boldsymbol{x}_i^o|\boldsymbol{z}_{ki})\, p(\boldsymbol{z}_{ki})}{q_\gamma(\boldsymbol{z}_{ki}|\boldsymbol{x}_i^o)}\right] \tag{20}$$

$$= p_{\theta,\phi}(\boldsymbol{x}_i^o, \boldsymbol{s_i}) \tag{21}$$

   Hence, $\mathcal{L}_K = \sum_{i=1}^n \mathbb{E}[\log M_{Ki}]$ converges to $\log p_{\theta,\phi}(\boldsymbol{x}^o, \boldsymbol{s})$ as $K \to \infty$. Regarding the precise convergence rate, we refer to the analysis presented in **Appendix D** of Ipsen et al. (2021) [1], which applies Theorem 3 of Domke & Sheldon (2018) [6]. Furthermore, the general importance-weighted framework and the monotonicity property (Item 2) are based on the original IWAE formulation by Burda et al. (2016) [7].

B Implementation Details

We developed a unified PyTorch library, `notmiwae-pytorch`, which implements MIWAE, not-MIWAE, and our proposed extension sup-not-MIWAE. The package is open-source and available on PyPI.

B.1 Software Architecture and Missing Processes

A key design goal was to decouple the *data generation process* from the *missing data mechanism*. We structured the project around three core modules: `base.py` (components), `notmiwae.py` (model logic), and `trainer.py` (optimization).

B.1.1 Modular Missing Processes ('base.py')

We implemented an abstract `BaseMissingProcess` class. This polymorphism allows users to easily inherit from the class and create custom missing processes without altering the core VAE training logic. We provide four distinct implementations:

- **Self-Masking (`selfmasking`):** Models the missingness probability of feature d based solely on its own value:

  $$\text{logit}(p(s_d = 1|x)) = -W_d(x_d - b_d)$$

  Application: Generic sensor failures where extreme values are missing, but the direction (high vs. low) is unknown.

- **Self-Masking with Known Signs (`selfmasking_known_signs`):** A constrained variant where W_d is forced to be positive or negative via a `signs` parameter.
  - *Sign +1.0:* High values are missing. *Application:* Sensor saturation or image pixel clipping (used in our CelebA experiments).
  - *Sign -1.0:* Low values are missing. *Application:* Values falling below a limit of detection (LOD).

- **Linear Process (`linear`):** Maps all features x to the missingness logits via a linear transformation ($Wx + b$). *Application:* Captures cross-feature dependencies (e.g., variable A affecting the missingness of variable B).

- **Non-linear Process (`nonlinear`):** Uses a Multilayer Perceptron (MLP) to map inputs to missingness probabilities. *Application:* Captures complex, non-monotonic missingness patterns, though requires more data to avoid overfitting.

B.1.2 Optimization Loop ('trainer.py')

We implemented a robust training loop that handles:

- **Gradient Clipping:** Essential for stabilizing the training of the Student-t decoder, preventing exploding gradients during the learning of degrees of freedom.

- **TensorBoard Logging:** Real-time tracking of ELBO components (separating $\log p(x|z)$ from $\log p(s|x)$) to diagnose whether the model is focusing on reconstruction or correctly recovering the missing mechanism.

B.2 Core Logic: Importance Weighting

The mathematical core of the paper is the calculation of importance weights that account for the missingness mechanism $p(s|x)$. Below is the critical PyTorch implementation:

```
def _compute_importance_weights(self, x, s, z, ...):
    # Expand inputs for K importance samples
    # x_mixed combines sampled missing values with observed data
    x_mixed = x_sample * (1 - s_expanded) + x_expanded * s_expanded

    # 1. Log-likelihood of data p(x|z) (masked)
    log_p_x_given_z = (s_expanded * p_x_given_z.log_prob(x_expanded)).sum(-1)

    # 2. Log-likelihood of missing mechanism p(s|x)
    # Crucial: The missing process sees the imputed x_mixed
    miss_logits = self.missing_model(x_mixed)
    log_p_s_given_x = Bernoulli(logits=miss_logits).log_prob(s_expanded).sum(-1)

    # 3. Prior and Approximate Posterior
    log_p_z = prior.log_prob(z).sum(-1)
```

```
    log_q_z_given_x = Normal(q_mu, q_std).log_prob(z).sum(-1)

    # 4. Total Importance Weights for not-MIWAE
    log_w = log_p_x_given_z + log_p_s_given_x + log_p_z - log_q_z_given_x

    return log_w
```

Listing 1: Importance-weight computation used in not-MIWAE.

B.3 Challenges and Lessons Learned

Debugging Stochastic Models Validating the implementation was non-trivial due to the stochastic nature of the VAE. Specifically, verifying the `selfmasking_known_signs` process required rigorous unit tests. We had to ensure that the custom gradients propagated correctly through the sampling step and that the learnable parameters respected the sign constraints (e.g., weights remaining positive for saturation modeling) during backpropagation.

Scaling to High-Dimensional Data (CelebA) Transitioning from UCI tabular datasets to the CelebA image dataset necessitated a strided CNN architecture. This introduced significant computational overhead, particularly regarding GPU memory, as importance sampling expands input tensors by a factor of K (number of samples), leading to rapid VRAM consumption. Consequently, we restricted the image resolution to 32×32 pixels; increasing the resolution further would drastically increase complexity and training time, rendering the importance-weighted training regime computationally prohibitive.

Modern PyTorch Practices Reproducing a paper originally written in TensorFlow v1 required translating static graph concepts into dynamic PyTorch modules. We learned to structure the project for distribution (using `pyproject.toml` for PyPI) and utilized `torch.distributions` for efficient reparameterization.

C Experimental Setup

C.1 Data Preparation and MNAR Generation

For all datasets, we applied standard scaling (zero mean, unit variance)[1] to the observed features. Missing values were introduced synthetically to simulate a Missing Not At Random (MNAR) mechanism.

MNAR Self-Masking Process We simulated self-masking missingness where the probability of a feature being missing depends on its own value. Following the implementation in our codebase, the missingness probability p_{miss} for feature d and sample i is given by:

$$p_{miss,i,d} = \sigma(W_d(x_{i,d} - b_d))$$

where $\sigma(\cdot)$ is the sigmoid function, and W_d and b_d are the weight and bias parameters for feature d.

The binary mask $M_{i,d}$ (where 1 indicates observed and 0 indicates missing) is sampled as:

$$M_{i,d} \sim 1 - \text{Bernoulli}(p_{miss,i,d})$$

For our experiments, the parameters W and b were sampled randomly for each feature to create diverse missingness profiles. Before passing data to the encoder, all missing entries were imputed with **zero** (the mean value after scaling)[1].

C.2 Model Architectures

C.2.1 UCI Datasets (Tabular)

For the UCI datasets (Banknote, Concrete, etc.), we utilized Multilayer Perceptrons (MLPs). The encoder maps the input to a latent space of dimension $z_{dim} = 50$, and the decoder maps it back to the data space. We used a hidden dimension of $h = 128$.

Table 4: UCI Encoder Architecture

Layer (Type)
Input $\in \mathbb{R}^D$
Linear($D \to h$)
Tanh Activation
Linear($h \to h$)
Tanh Activation
Outputs:
μ: Linear($h \to z_{dim}$)
$\log \sigma$: Linear($h \to z_{dim}$) (clamped)

Table 5: UCI Decoder Architecture

Layer (Type)
Latent $z \in \mathbb{R}^{z_{dim}}$
Linear($z_{dim} \to h$)
Tanh Activation
Linear($h \to h$)
Tanh Activation
Outputs:
μ_x: Linear($h \to D$)
σ_x: Softplus(Linear($h \to D$))

C.2.2 CelebA Dataset (Images)

For the CelebA dataset (32×32 images), we adopted the Convolutional Neural Network (CNN) architecture described in the original not-MIWAE paper[1].

Table 6: CelebA Encoder (CNN)

Layer (Output Size)
Input ($32 \times 32 \times 1$)
Conv2D ($16 \times 16 \times 64$)
Conv2D ($8 \times 8 \times 128$)
Conv2D ($4 \times 4 \times 256$)
Reshape (4096)
Outputs:
μ: Dense(20)
$\log \sigma$: Dense(20)

Table 7: CelebA Decoder (CNN)

Layer (Output Size)
Latent z (20)
Dense (4096)
Reshape ($4 \times 4 \times 256$)
ConvTranspose ($8 \times 8 \times 256$)
ConvTranspose ($16 \times 16 \times 128$)
Outputs:
μ_x: ConvTranspose ($32 \times 32 \times 1$)
$\log \sigma_x$: ConvTranspose ($32 \times 32 \times 1$)

C.3 Training Hyperparameters

All models were implemented in PyTorch and optimized using the following hyperparameters:

- **Optimizer:** AdamW with 1Cycle learning rate scheduler.
- **Learning Rate:** Initial learning rate set to 1×10^{-4}.
- **Batch Size:**
  - UCI Datasets: 256
  - CelebA Dataset: 64
- **Importance Samples (K):**
  - Training: 100 samples per data point.
  - Evaluation (RMSE Estimation): 10,000 samples per data point.
- **Training Duration:** 100 epochs.