# Highlights

**Handling ill-conditioned omics data with deep probabilistic models.**

María Martínez-García, Pablo M. Olmos

- *Inference of the latent space driven by the label value.* The DBLR infers different low-dimensional latent distributions depending on the label value, forcing clustering in the latent space in an informative manner, thus capturing the underlying structure of the data.
- *Classification.* During inference, the model additionally learns a global vector of weights that allows to make predictions given the low-dimensional representation of the data.
- *Handling missing data.* As the DBLR is a probabilistic generative model, it can naturally handle partially missing observations during the training process, including not annotated observations as censored samples. In this article we cover the Missing at Random (MAR) case.
- *Regularization method to handle small datasets.* In order to handle small high-dimensional datasets, which usually entail overfitting problems, we introduced an additional regularization mechanism following a drop-out-like strategy that relies in the generative semi-supervised nature of the model.
- *Handling different data types.* We have defined and implemented different observation likelihood models that can be used to describe different data types. In particular, we show how to use the DBLR with binary and real-valued features.

# Handling ill-conditioned omics data with deep probabilistic models.

María Martínez-García[a,b,*], Pablo M. Olmos[a,b]

[a] *Signal Processing Group at Signal Theory and Communications Department, University Carlos III in Madrid (UC3M), Madrid, Spain*
[b] *Gregorio Marañón Health Research Institute (IiSGM), Madrid, Spain*

**Abstract**

The advent of high-throughput technologies has produced an increase in the dimensionality of omics datasets, which limits the application of machine learning methods due to the great unbalance between the number of observations and features. In this scenario, dimensionality reduction is essential to extract the relevant information within these datasets and project it in a low-dimensional space, and probabilistic latent space models are becoming popular given their capability to capture the underlying structure of the data as well as the uncertainty in the information. This article aims to provide a general classification and dimensionality reduction method based on deep latent space models that tackles two of the main problems that arise in omics datasets: the presence of missing data and the limited number of observations against the number of features. We propose a semi-supervised Bayesian latent space model that infers a low-dimensional embedding driven by the target label: the Deep Bayesian Logistic Regression (DBLR) model. During inference, the model also learns a global vector of weights that allows to make predictions given the low-dimensional embedding of the observations. Since this kind of datasets is prone to overfitting, we introduce an additional probabilistic regularization method based on the semi-supervised nature of the model. We compared the performance of the DBLR against several state-of-the-art methods for dimensionality reduction, both in synthetic and real datasets with different data types. The proposed model provides more informative low-dimensional representations, outperforms the baseline methods in classification and can naturally handle missing entries.

*Keywords:* Classification, Bayesian, dimensionality reduction, latent space model, missing data, VAE, omics, semi-supervised

*Please address correspondence to María Martínez-García
*Email addresses:* mmargarc@pa.uc3m.es (María Martínez-García), pamartin@ing.uc3m.es (Pablo M. Olmos)

## 1. Introduction

As data acquisition methods evolve, scientific datasets grow rapidly in scale and complexity. This is particularly critical in omics, where the advent of high-throughput technologies has produced an increase in the dimensionality of the data, causing an important unbalance between the number of features and observations. This hampers the application of classification models on these datasets, as they are often limited by the *curse of dimensionality* [1]. Therefore, dimensionality reduction methods can be essential for further analysis in this scenario [30]. By means of different approaches, they learn a low-dimensional representation or embedding of the original data, which can be used for different applications as data visualization, data exploration or data classification.

Feature selection has been widely applied in bioinformatics to discard irrelevant features and reduce dimensionality in large datasets applying different information criteria. As of today, state-of-the-art in bioinformatics is given by the minimal-redundancy-maximal-relevance (mRMR) criterion [23] or the Hilbert–Schmidt Independence Criterion (HSIC) [8, 3]. Although these methods can result very useful for biomarker discovery, they do not infer a low dimensional embedding that captures the information within the data, but rather they select a subset of input variables.

Alternatively to feature extraction, linear dimensionality reduction methods such as Principal Component Analys (PCA) [10] or Canonical Correlation Analysis (CCA) [11] are widely used as one of the first steps in high-dimensional data analysis pipelines in order to overcome the limitations introduced by the large number of features, as the increased risk of overfitting or intractable computation [12, 26, 14, 15]. However, linear methods can fail to capture the complexity of high-dimensional datasets, which often imply non-linear dependencies within the data. For this reason, non-linear dimensionality reduction methods as t-Stochastic Neighborhood Embedding) (t-SNE) [18] or Uniform Manifold Approximation and Projection (UMAP) [19] have gained ground. In fact, UMAP has been rapidly adopted as dimensionality reduction method for preprocessing in genetic studies [4].

In this context, probabilistic latent space models have become increasingly popular for high-dimensional data analysis in recent years. Given their flexibility and capability to capture the underlying structures of the data, they can provide remarkable results in a variety of scenarios, and have proven to be particularly useful, for example, in cancer research, where the exploration of the latent space allowed the identification of cancer subtypes [25, 2, 9, 29, 28]. Additionally, the architecture of these kind of models can be easily adapted to different problems and types of input data, which have led to the development of a variety of methods for specific purposes such as netAE [5] or scVI [17], where the authors adapt the loss functions and architectures to properly analyze single cell RNASeq data, trying to maintain the local and global structure to capture cell relationships. However, these models are designed to be applied in a specific setup with a specific type of data, as expert knowledge is introduced in their definition.

In this work, we propose a general semi-supervised dimensionality reduction and classification method based on latent space models: the Deep Bayesian Logistic Regression (DBLR) model. Its main features, and thus our main contributions, are:

- *Inference of the latent space driven by the label value.* The DBLR infers different low-dimensional latent distributions depending on the label value, forcing clustering in the latent space in an informative manner, thus capturing the underlying structure of the data.
- *Classification.* During inference, the model additionally learns a global vector of weights that allows to make predictions given the low-dimensional representation of the data.
- *Handling missing data.* As the DBLR is a probabilistic generative model, it can naturally handle partially missing observations during the training process, including not annotated observations as censored samples. In this article we cover the MAR case.
- *Regularization method to handle small datasets.* In order to handle small high-dimensional datasets, which usually entail overfitting problems, we introduced an additional regularization mechanism following a drop-out-like strategy that relies in the generative semi-supervised nature of the model.
- *Handling different data types.* We have defined and implemented different observation likelihood models that can be used to describe different data types. In particular, we show how to use the DBLR with binary and real-valued features.

The structure of the paper is as follows. In Section 2 (Methods) we briefly introduce the Variational Autoencoder (VAE), which is the base of the proposed method. Then, we define the DBLR and explain the inference process, as well as the regularization of the model and how we handle MAR data. In Sections 3 (Results) we describe the experiments, introducing the baseline methods and the datasets, and show the obtained results. Finally, in Section 4 (Discussion) we summarise these results and draw the final conclusions of this work.

## 2. Methods

The DBLR is a semi-supervised Bayesian Logistic Regression model that follows a VAE [16] approach, learning a low-dimensional latent representation of high-dimensional data in order to classify it. In this section, we first briefly introduce the vanilla VAE, as it is the base of our proposed model. Then, we will describe in detail its definition of the DBLR, including the inference process, regularization and handling of MAR data.

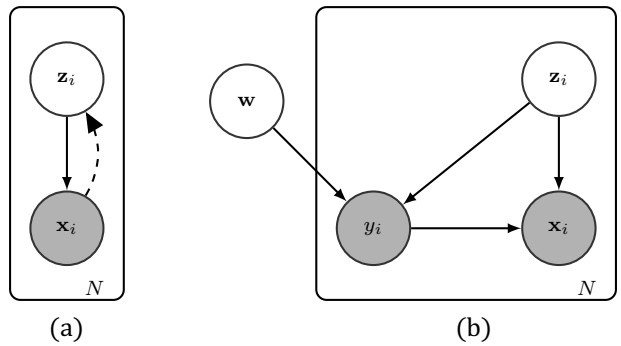


Figure 1: Graphical models. (a) Variational Autoencoder (VAE), (b) Deep Bayesian Logistic Regression model (DBLR)

### *2.1. VAE description*

Autoencoders are Neural Networks (NNs) that learn a compact representation of the input data in a nonlinear unsupervised way, combining input features while minimizing the reconstruction error. In VAEs, this latent representation is learnt by means of Variational Inference (VI), assuming the data is coming from an underlying probability distribution. The goal is to fit the model parameters and to approximate the posterior distribution $p(\mathbf{z}|\mathbf{x})$, which describes the low-dimensional latent (encoded) representation $\mathbf{z}$ given the original (decoded) input data $\mathbf{x}$. In the vanilla VAE, both the latent prior distribution $p(\mathbf{z})$ and the likelihood $p(\mathbf{x}|\mathbf{z})$ are assumed Gaussian

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathcal{I}), \tag{1}$$

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z})\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{z})), \tag{2}$$

where the mean vector $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z})$ and diagonal covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{z})$ are the outputs of two NNs (decoder) with parameters $\boldsymbol{\theta}$. Note the observations are i.i.d. under this model. Then, the marginal distribution $p(\mathbf{x})$ is given by

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} \tag{3}$$

As this integral is intractable, it is not possible to do exact inference. Instead, the posterior distribution is approximated by means of VI assuming a Gaussian variational family

$$q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{x})), \tag{4}$$

where the mean vector $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{z})$ and diagonal covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\phi}}(\mathbf{z})$ are the outputs of two NNs (encoder) with parameters $\boldsymbol{\phi}$.

The goal is to fit the parameters of the model to find the distribution that better approximates the posterior $p(\mathbf{z}|\mathbf{x})$ among the defined variational family by maximizing the Evidence Lower Bound (ELBO), which is given by

$$\log p(\mathbf{x}) \geq ELBO = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\big[p(\mathbf{x}|\mathbf{z})\big] - \mathcal{D}_{KL}\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big) \tag{5}$$

Using the negative ELBO as loss function for training, the likelihood of the input data (or reconstruction term) is maximized while keeping the approximated posterior close to the defined prior $p(\mathbf{z})$, as the Kullback-Leibler (KL) divergence in the right acts as a regularization term.

### *2.2. DBLR description*

The DBLR model generalizes the VAE in several ways. Assume a labelled $\{\mathbf{x}_i, y_i\}_{i=1}^N$ dataset where $\mathbf{x}$ is the D-dimensional feature vector and $y$ is the target. We denote these variables as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]^T$ and $\mathbf{y} = [y_1, y_2, ..., y_N]^T$. We note that, although we assume binary labels, the model naturally handles categorical targets. As it can be seen in the graphical model depicted in Figure 1, one of the main differences of the DBLR is the assumption of dependency between the latent distribution and the target variable. This implies that the underlying structure of the data $\mathbf{z} \in \mathbb{R}^{N\times K}$, responsible for generating the properties of the observed samples $\mathbf{x}$, also reflects the target variable. Thus, we force further clustering in the latent space in an informative manner, as the model captures the label information along the inference process [1]. Additionally, following the idea of a Logistic Regression (LR) model, the DBLR learns a global vector of weights $\mathbf{w} \in \mathbb{R}^{K+1}$ that allows to predict the label value given the low-dimensional representation of the input data. Table 1 summarizes the model variables.

| | |
|---|---|
| Input Data | $\mathbf{X} \in \mathbb{R}^{N\times D}$ or $\mathbf{X} \in \{0,1\}^{N\times D}$ |
| Target Variable / Label | $\mathbf{y} \in \{0,1\}^N$ |
| Low-dimensional Latent Variable | $\mathbf{Z} \in \mathbb{R}^{N\times K}, K \ll D$ |
| Global Weight Vector | $\mathbf{w} \in \mathbb{R}^{K+1}$ |

Table 1: Description of the model variables.

In the following sections we discuss the model in detail, including the inference process and the semi-supervised approach, used for handling not-annotated observations. We also introduce an additional regularization mechanism and explain how the model handles incomplete data.

#### *2.2.1. Prior distributions*

We assume multivariate Gaussian distributions with zero mean and identity covariance matrix as priors for both global weights $\mathbf{w}$ and latent variable $\mathbf{Z}$.

[1] This is common to other supervised VAE-like models such as [31, 32, 6, 21]

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \mathcal{I}) \tag{6}$$

$$p(\mathbf{Z}) = \prod_{i=1}^{N} p(\mathbf{z}_i), \quad p(\mathbf{z}_i) = \mathcal{N}(0, \mathcal{I}) \tag{7}$$

*2.2.2. Likelihoods*

Depending on the nature of the input data, two different cases are defined:

- *Real input case*

  For the real input case, $p(\mathbf{X}|\mathbf{Z}, \mathbf{y})$ is given by

$$\begin{gathered} p(\mathbf{X}|\mathbf{Z}, \mathbf{y}) = \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{z}_i, y_i) = \\ \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,x}}(\mathbf{z}_i), \beta\mathcal{I})^{y_i} \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,x}}(\mathbf{z}_i), \beta\mathcal{I})^{(1-y_i)} \end{gathered} \tag{8}$$

  where the mean vectors $\boldsymbol{\mu}$ are learnt by means of two NNs with input $\mathbf{z}_i$ and parameters $\boldsymbol{\eta}^0_{m,x}, \boldsymbol{\eta}^1_{m,x}$. Superscripts indicate the value of the label, while subscripts indicate that the parameters correspond to the mean estimation NN. For simplicity of the model, we have selected a fixed diagonal covariance matrix with value $\beta$, which is an hyperparatemer of the model.

- *Binary input case*

  For the binary input case, $p(\mathbf{X}|\mathbf{Z}, \mathbf{y})$ is given by

$$\begin{gathered} p(\mathbf{X}|\mathbf{Z}, \mathbf{y}) = \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{z}_i, y_i) = \\ \prod_{i=1}^{N} \Big( \prod_{j=1}^{D} Ber(\boldsymbol{\theta}_{\boldsymbol{\eta}^1_x}(\mathbf{z}_i)_j) \Big)^{y_i} \Big( \prod_{j=1}^{D} Ber(\boldsymbol{\theta}_{\boldsymbol{\eta}^0_x}(\mathbf{z}_i)_j) \Big)^{(1-y_i)} \end{gathered} \tag{9}$$

  where $\boldsymbol{\theta}_{\boldsymbol{\eta}_x}(\mathbf{z}_i)_j$ is the probability of feature $j$ to be 1. These probabilities are computed by means of two NNs with input $\mathbf{z}_i$ and parameters $\boldsymbol{\eta}^0_x, \boldsymbol{\eta}^1_x$. As in the previous case, superscripts indicate the value of the label.

  Observe that both in (8) and (9), the likelihood $p(\mathbf{x}_i|\mathbf{z}_i, y_i)$ critically depends on the target value, achieving a greater influence of $y$ through the learning process. This way, we force the model to learn different parameters (and thus different distributions) for data with different target values, inducing clustering in the latent space. We also note that we could have a mixture between binary and real valued features. In such case, the term

$p(\mathbf{x}_i|\mathbf{z}_i, y_i)$ would be the corresponding product of Gaussian and Bernoulli likelihoods.

Following the formulation of a LR model, the likelihood $p(\mathbf{y}|\mathbf{Z}, \mathbf{w})$ is defined as

$$p(\mathbf{y}|\mathbf{Z}, \mathbf{w}) = \prod_{i=1}^{N} p(y_i|\mathbf{z}_i, \mathbf{w}) = \prod_{i=1}^{N} \sigma(\mathbf{w}^T\mathbf{z}_i + w_0)^{y_i}(1 - \sigma(\mathbf{w}^T\mathbf{z}_i + w_0))^{(1-y_i)} \tag{10}$$

where $\sigma(\cdot)$ refers to the Sigmoid function. We introduce here a linear layer to keep the classification stage as simple as possible, forcing the model to infer highly separable latent representations to achieve a good classification performance. This way, the complexity relies in the generative model.

*2.2.3. Variational family and inference*

The posterior distribution is approximated via VI, using the following variational family:

$$q(\mathbf{Z}, \mathbf{w}|\mathcal{D}) = q(\mathbf{Z}|\mathbf{X})q(\mathbf{w}|\mathbf{Z}, \mathbf{y}) \tag{11}$$

The first term, $q(\mathbf{Z}|\mathbf{X})$, which is given by

$$q(\mathbf{Z}|\mathbf{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\eta}_{m,z}}(\mathbf{x}_i), \boldsymbol{\Sigma}_{\boldsymbol{\eta}_{c,z}}(\mathbf{x}_i)) \tag{12}$$

allows for inferring a latent representation of the observed data $\mathbf{X}$. The mean vector and the diagonal of the covariance matrix are learnt by means of two NNs with parameters $\boldsymbol{\eta}_{m,z}$ and $\boldsymbol{\eta}_{c,z}$ respectively. Subscripts $m$ and $c$ stand for *mean* and *covariance*.

The second term, given by (13) refers to the posterior distribution of the weight vector $\mathbf{w}$, which allows for classification. As it is a global variable, it is obtained as a product of the local contribution of each observed data point [22].

$$q(\mathbf{w}|\mathbf{Z}, \mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w) \propto \prod_{i=1}^{N} \mathcal{N}\Big(\boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,w}}(\mathbf{z}_i), \boldsymbol{\Sigma}_{\boldsymbol{\eta}^1_{c,w}}(\mathbf{z}_i)\Big)^{y_i} \mathcal{N}\Big(\boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,w}}(\mathbf{z}_i), \boldsymbol{\Sigma}_{\boldsymbol{\eta}^0_{c,w}}(\mathbf{z}_i)\Big)^{(1-y_i)} \tag{13}$$

Mean vectors $\boldsymbol{\mu}_{m,w}$ and the diagonal of the covariance matrix $\boldsymbol{\Sigma}_{m,w}$ are learnt by means of four NNs with input $\mathbf{z}_i$ and parameters $\boldsymbol{\eta}^1_{m,w}, \boldsymbol{\eta}^0_{m,w}, \boldsymbol{\eta}^1_{c,w}$ and $\boldsymbol{\eta}^0_{c,w}$. This way, we learn independent distributions or representations for the data with different label value.

Inference is done maximizing the Evidence Lower Bound (ELBO), which can be expressed as

$$
\begin{aligned}
ELBO = \int q(\mathbf{Z}, \mathbf{w}|\mathcal{D}) \log \frac{p(\mathbf{Z}, \mathbf{w}, \mathcal{D})}{q(\mathbf{Z}, \mathbf{w}|\mathcal{D})} d\mathbf{Z} d\mathbf{w} = \\
\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{y}|\mathbf{Z}, \mathbf{w})\Big] + \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\Big] \\
-\mathcal{D}_{KL}\big(q(\mathbf{w}|\mathbf{Z}, \mathbf{y})||p(\mathbf{w})\big) - \mathcal{D}_{KL}\big(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})\big)
\end{aligned} \tag{14}
$$

where the first two terms correspond to reconstruction of the label $\mathbf{y}$ and reconstruction of the input data $\mathbf{X}$, respectively. The two remaining terms correspond to the KL Divergence between the variational family and the prior distributions. For further details, please refer to the appendix.

### *2.2.4. Predictive distribution*

To obtain predictions for new data, we first forward the new observation $\mathbf{x}^*$ through the encoder in (12) to obtain the parameters of the latent distribution. We combine (12) and (13) to derive the predictive distribution

$$
\begin{aligned}
p(y^*|\mathbf{x}^*, \mathcal{D}) = \int p(y^*|\mathbf{z}^*, \mathbf{w}) q(\mathbf{z}^*|\mathbf{x}^*) q(\mathbf{w}|\mathbf{z}^*, \mathbf{y}, \mathcal{D}) d\mathbf{z}^* d\mathbf{w} \\
\approx \mathbb{E}_{q(\mathbf{z}^*,\mathbf{w}|\mathbf{x}^*,\mathcal{D})}\Big[p(y^*|\mathbf{z}^*, \mathbf{w})\Big]
\end{aligned} \tag{15}
$$

which we approximately evaluate using Monte Carlo sampling.

### *2.3. DBLR architecture*

The architecture of the model can be divided in three main blocks: encoder, decoder and weight estimation block. Figure 2 shows a visual scheme of this architecture.

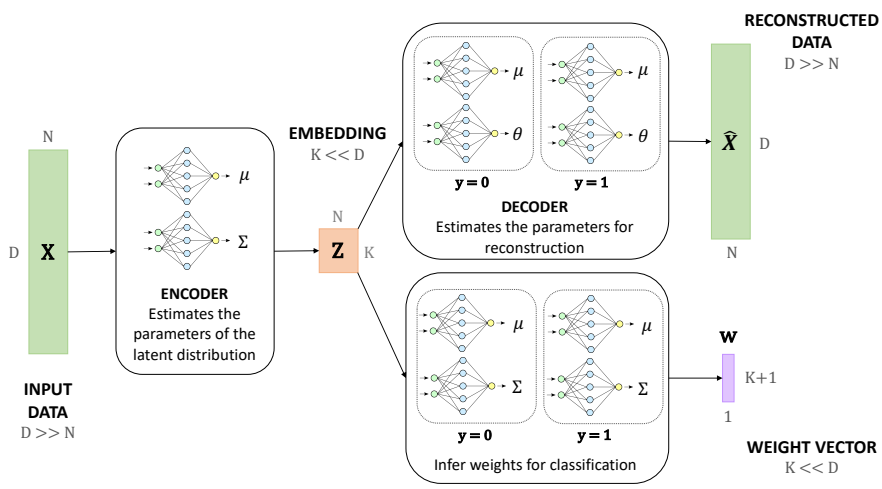


Figure 2: DBLR architecture.

- *Encoder*

  The encoder consists of two fully connected NNs. One estimates the mean vector, and the other estimates the diagonal of the covariance matrix of $q(\mathbf{Z}|\mathbf{X})$, which is given by (12)

- *Decoder*

  The decoder consists of two fully connected NNs: one to learn the parameters for observations with $y_i = 1$, and other to learn the parameter for observations with $y_i = 0$. Depending on the nature of the data (real or binary), this NN estimates different parameters for $p(\mathbf{X}|\mathbf{Z}, \mathbf{y})$. For the real input case, given by (8), it estimates a vector of means; for the binary input case, given by (9), it estimates a vector of probabilities.

- *Weight estimation block*

  This block consists of four fully connected NNs: two that learn the parameters of $q(\mathbf{w}|\mathbf{Z}, \mathbf{y})$ (13) for observations with $y_i = 1$, and other two for observations with $y_i = 0$. Since we approximate the weights by means of Gaussian distributions, we learn a vector of weights and the diagonal of a covariance matrix for each label value.

### *2.4. Semi-supervised approach*

As the presence of not-annotated observations or censored samples is a common problem in biological datasets, we define a semi-supervised approach to handle MAR data in the target variable $y$. When $y$ is not known, we can evaluate the reconstruction term $\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\big]$ in the ELBO loss (14) by first sampling from $p(\mathbf{y}|\mathbf{Z}, \mathbf{w})$, given by (10) [2]. In order to use gradient-based stochastic optimization, overcoming the inability to apply the reparameterization trick to discrete data, these samples are obtained by means of the Gumbel-Softmax trick [13]. Then, to evaluate the likelihood term $\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\big[\log p(\mathbf{y}|\mathbf{Z}, \mathbf{w})\big]$ we only use observed $y$ samples, i.e. we marginalize over the missing observations. This way, we only maximize the likelihood of the observed data.

### *2.5. Regularization*

Although the model is already regularized by means of the KL Divergence terms present in the ELBO, which constrain the latent representation of the data, the model can still overfit since the reconstruction terms dominate over the KL terms. Thus, it is essential to introduce additional regularization mechanisms to achieve a compromise between the flexibility of the model and its generalization capability. In our case, we define a drop-out-like strategy that relies on the semi-supervised nature of the DBLR.

[2] This is equivalent to assume an augmented variational family $q(\mathbf{Z}|\mathbf{X})q(\mathbf{w}|\mathbf{X}_o, \mathbf{y}_o)p(\mathbf{y}_m|\mathbf{Z}, \mathbf{w})$, where $o$ is the index set of observed targets and $m$ the set of not observed ones.

At each iteration of the gradient-descent optimization, we set a percentage $p$ of the target $\mathbf{y}$ at missing. Then, the target value for these observations is sampled from $p(\mathbf{y}|\mathbf{Z}, \mathbf{w})$ (10) by means of the Gumbel-Softmax trick, as the rest of the missing labels. Since this *censored* observations are selected at random, the model observes a different set of non-missing samples at each iteration of the learning process, preventing it from memorizing a given set of data.

### *2.6. Handling incomplete data*

We have already described how the DBLR deals with not annotated samples, as it takes part in both inference and regularization processes. However, given its formulation, the DBLR can also handle partially missing observations in the feature vectors $\mathbf{x}$.

If $\mathbf{x}$ is a vector with missing entries, to evaluate its projection over the $\mathbf{z}$ space using $q(\mathbf{z}_i|\mathbf{x}_i)$, we use the input-dropout trick described in [20], placing zeroes at missing entries in $\mathbf{X}$ and feeding the resulting vector $\tilde{\mathbf{X}}$. By placing zeroes, the output of the NNs that parameterize the distributions of the variational family are not affected by the missing entries. In Multilayer Perceptron (MLP) NN architectures, the outputs of the neurons are non-linear transformations of linear weighted sums of the inputs. Thus, the outputs and their derivatives do not depend on zero-valued entries. Then, the reconstruction terms in the ELBO loss (14) are only evaluated over observed attributes. Namely, we remove from $\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\big]$ the factors corresponding to missing attributes.

## 3. Results

We compared our proposed model against different methods in synthetic and real high-dimensional datasets. Experimental details are explained along the following sections.

### *3.1. Baseline methods*

In order to contrast the performance of our model against the state-of-the-art, we selected a set of different methods that are widely used for dimensionality reduction purposes in bioinformatics, making sure that we included models that use the target information. In particular, we compared our model against the following methods:

- *Supervised Uniform Manifold Approximation and Projection (S-UMAP)* [19]: it is a manifold learning algorithm constructed from a theoretical framework based in Riemannian geometry and algebraic topology. Although it was originally defined as an unsupervised dimensionality reduction method, it has been extended to include label information and thus obtain an embedding directly related with the target variable, as in our model.

- *Block Hilbert–Schmidt Independence Criterion (Block HSIC)*, [3]: it is a non-linear feature selection algorithm, which reduces the dimensionality of the input data based on the HSIC maximization between the high dimensional data and the resulting embedding. The *Block* version of the method reduces the memory complexity splitting the data in blocks. Note that this is a feature selection method, so it does not infer a low-dimensional representation of the data.

- *Principal Component Analysis (PCA)*, [10]: it solves the eigenvalue-eigenvector problem to generate a new reduced set of variables that attempts to explain the most variance of the data, while minimizing the residual variance.

- *Kernel Principal Component Analysis (KPCA)*, [27]: it is an extension of PCA that incorporates kernel methods. This enables the method to apply non-linear transformations to the data.

- *Canonical Correlation Analysis (CCA)*, [11]: it transforms the input variables to maximize the correlation between two datasets, which, in our case, correspond to the input features and the variables we want to predict.

- *Partial Least Squares (PLS)*, [7]: it generates a reduced set of components that attempts to explain as much as possible of the covariance between the input features and the target variable.

We used the Scikit-learn implementation for those methods available (PCA, KPCA, CCA and PLS). For UMAP and Block HSIC we used the implementation provided by the authors. [3] [4]

The goal of the proposed model is to reduce the dimensionality of the data inferring a latent distribution that captures its underlying structure while learning a vector of weights that enables classification of the observations. Thus, we compare the different methods in terms of the generated embeddings and in terms of classification performance.

For the classification experiment, we added a classification stage after the above mentioned methods by means of a LR including both L1 and L2 regularization. We decided to use a Logistic Regressor for this purpose to have a fair comparison between our model and the baselines, as the DBLR implements a Bayesian Logistic Regressor to describe the target variable and a non linear dimensionality reduction module. This way, we ensure the main differences in performance are mainly caused by the dimensionality reduction process and not by the characteristics of the classifier. However, it is worth mentioning that baseline methods were trained following a two-step approach where the latent representations were obtained prior to fitting the logistic regression model. In

[3] UMAP package documentation at `https://umap-learn.readthedocs.io/en/latest/basic_usage.html`

[4] Block HSIC implementation at `https://github.com/riken-aip/pyHSICLasso`

contrast, the DBLR optimizes the logistic regressor jointly with the latent representations.

For this experiment, we applied our model in two different manners. First, we trained the model over the train partition in a fully-supervised way, as the datasets do not present censored samples. We refer to these results as DBLR in the tables. Then, in order to exploit its semi-supervised nature, we trained the model over the full dataset setting test labels as missing. We refer to these results as DBLR SS in the tables (SS stands for Semi Supervised). However, it is important to remark that the semi-supervised capabilities of the model are also exploited in the DBLR version, since we are applying the previously defined regularization method based on internal censoring of targets at random.

Finally, we wanted to test the robustness of the model in the presence of partially-missing observations. For this purpose, we trained the model setting half of the training observations at missing for a given feature, with both samples and feature selected at random. Then, we compared the original observations against the imputation obtained by sampling from the model. Note this time we are introducing missing entries in $\mathbf{X}$, while during the regularization process we are introducing missing entries in $\mathbf{y}$. For simplicity, we use the cross-validated hyperparameters obtained in the previous experiment, since we are using the same data.

### *3.2. Datasets*

- *Synthetic data.*

  We generated synthetic data in order to compare the latent representations of different methods in a controlled setting. We start drawing samples from low-dimensional uniform mixture of Gaussian distributions with different mean vectors and covariance matrices, obtaining $\mathbf{Z} \in \mathbb{R}^{N \times K}$. Samples drawn from the same Gaussian are assigned the same label. Then, we increase the dimensionality of the data applying a linear transformation by means of a random Gaussian matrix $\mathbf{A} \in \mathbb{R}^{K \times D}$. Assuming a binary dataset, we obtain $\mathbf{X}$ as

  $$\mathbf{X} \sim Ber(\sigma(\mathbf{ZA} + b)) \tag{16}$$

  where $b \in \mathbb{R}^D$ is a random bias vector.

- *Real data. The Cancer Genome Atlas (TCGA) dataset.*

  Since the motivation behind the development of this method is to handle high dimensional datasets with limited number of observations, which are typical in omics, we analyzed four fat RNA-Seq datasets related to survival in different types of cancer. We used the tables preprocessed by [24], which contain RNA-Seq data extracted from the TCGA portal (`https://tcga-data.nci.nih.gov/tcga/`) and survival data extracted from the Genomic Data Commons (GDC) Portal (`https://portal.gdc.`

cancer.gov/). They provide data for 32 different types of cancer, but we selected a subset (Breast Invasive Carcinoma (BRCA), Brain Lower Grade Glioma (LGG), Lung Adenocarcinoma (LUAD), Head and Neck Squamous Cell Carcinoma (HNSC)) based on the number of observations available. We additionally standardized the variables to have mean zero and unit standard deviation.

Table 2: Description of the four TCGA datasets.

| Dataset | # Observations | # Features |
|---|---|---|
| BRCA (Breast Invasive Carcinoma) | 616 | 20221 |
| LGG (Brain Lower Grade Glioma) | 505 | 20223 |
| HNSC (Head and Neck Squamous Cell Carcinoma) | 473 | 20257 |
| LUAD (Lung Adenocarcinoma) | 434 | 20532 |

### *3.3. Hyperparameter tuning*

The hyperparameters of the different dimensionality reduction methods have been tuned by means of 5-fold cross validation. We have validated the dimensionality of the embedding (the low dimensional representation of the data) for all the models except for CCA and Partial Least Squares (PLS), where the number of components is limited to *min(number of samples, number of features, number of classes)*. As the target variables are binary, the number of components is limited to 2. More precisely, for each dataset we tried five different values for K, ranging from 2 to the total number of samples. In the case of our model, we have additionally validated the percentage $p$ of samples we set at missing for regularization, trying four different values $p = \{0, 0.1, 0.25, 0.5\}$. Note that by setting $p = 0$ we are not introducing regularization. For the classification stage we have validated the regularization parameter, which we denote by $C$ (for both L1 and L2 regularization), trying 10 different values that range from $C = 10^{-5}$ to $C = 10$.

All the networks have been fixed to a basic fully-connected configuration which consists on one hidden layer of dimension either 25 or 100. This is the only network parameter we have validated. Note that cross-validation of other network parameters (number of layers, activation functions, dropout layers...) can potentially improve the experimental results.

### *3.4. Experimental results*

In this section we present the results we have obtained applying the different baselines as well as our model to the datasets described above.

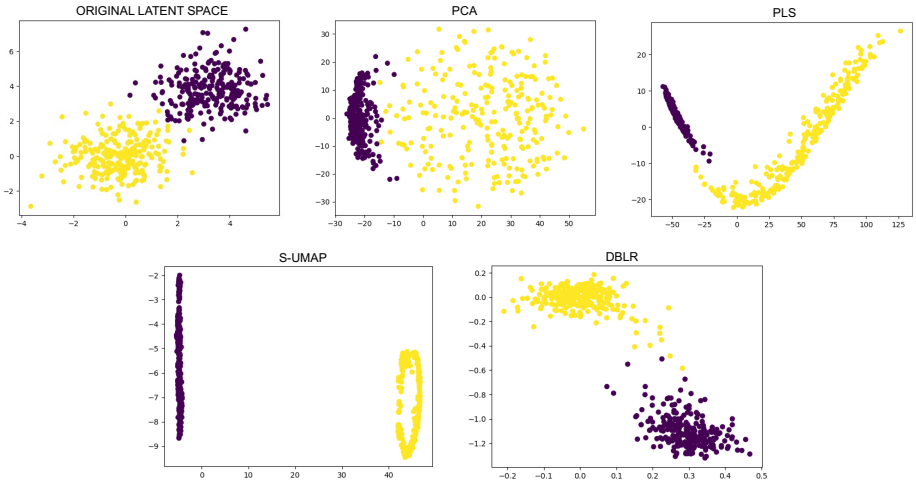


Figure 3: Low-dimensional embeddings generated by different methods against the original latent distribution for the binary input case ($N = 500, D = 10000, K = 2$).

*3.4.1. Low-dimensional embedding on synthetic data.*

First, we analyze the low-dimensional representations generated by different methods when applied to a binary synthetic dataset. Since we generated high dimensional data drawing samples from a low-dimensional latent distribution, we can compare the embeddings learnt by some baseline models and the DBLR against the original latent space. We generated datasets of 500 samples and 10000 features following the generative process described in (16). Figure 3 shows the embeddings obtained applying the different methods, which were configured to generate latent representations of dimension $K = 2$ for proper visualization.

Although all the methods generated highly separable low-dimensional representations that could be used for classification purposes, the obtained embeddings are quite different. While the DBLR preserves the natural structure of the data, the rest of the methods provide highly distorted and overfitted representations. The solution provided by the DBLR captures the original latent distribution of the data while enhancing the separation of the clusters, which reflects the label value.

*3.4.2. Classification on TCGA dataset.*

Now we analyze the classification performance of our model against the described baselines on real data using the Area Under the Receiver Operating Characteristics Curve (AUC). Since the number of observations is very limited in all the cases, we evaluate the performance of the different methods over 5 folds to avoid biases introduced by the partition of the data. For this reason, results are expressed in terms of mean and standard deviation. Notice that for each fold we treated train and test partitions independently.

Tables 3, 4, 6, 5 summarize the results obtained in classification for BRCA, LGG, LUAD and HNSC datasets, respectively. We show metrics obtained when applying the baseline methods as well as the DBLR in its basic and semi-supervised form, referred as DBLR and DBLR SS in the tables. We include in the caption of every table the DBLR validated parameters. Namely, the latent dimension $K$, the networks hidden dimensionality (25 or 100), and regularization parameter $p$.

| Model | AUC Train | AUC Test |
|---|---|---|
| PCA + LR (L1) | $0.8667 \pm 0.1645$ | $0.5640 \pm 0.0656$ |
| PCA + LR (L2) | $0.9077 \pm 0.0953$ | $0.5650 \pm 0.0672$ |
| KPCA + LR (L1) | $0.8674 \pm 0.1649$ | $0.5599 \pm 0.0687$ |
| KPCA + LR (L2) | $0.9156 \pm 0.0877$ | $0.5669 \pm 0.0682$ |
| CCA + LR (L1) | $1.0000 \pm 0.0000$ | $0.5689 \pm 0.0708$ |
| CCA + LR (L2) | $1.0000 \pm 0.0000$ | $0.5689 \pm 0.0708$ |
| PLS + LR (L1) | $0.8699 \pm 0.0257$ | $0.5903 \pm 0.0718$ |
| PLS + LR (L2) | $0.8699 \pm 0.0257$ | $0.5908 \pm 0.0712$ |
| S-UMAP + LR (L1) | $0.8922 \pm 0.0281$ | $0.4520 \pm 0.0579$ |
| S-UMAP + LR (L2) | $0.8916 \pm 0.0289$ | $0.4536 \pm 0.0578$ |
| HSIC + LR (L1) | $0.5000 \pm 0.0000$ | $0.5000 \pm 0.0000$ |
| HSIC + LR (L2) | $0.5818 \pm 0.0508$ | $0.5000 \pm 0.0000$ |
| DBLR | $0.8819 \pm 0.1575$ | $0.6159 \pm 0.0384$ |
| **DBLR SS** | $\mathbf{0.8847 \pm 0.1658}$ | $\mathbf{0.6183 \pm 0.0567}$ |

Table 3: 5-fold results for BRCA dataset. Mean values of the validated hyperparameters are $K = 394.40$, $p = 0.06$ and $hdim = 25$ for the DBLR, and $K = 222.40$, $p = 0.27$ and $hdim = 100$ for the semi-supervised DBLR.

Given the classification results, we can say that the DBLR outperforms all baseline methods in BRCA (Table 3), LGG (Table 4) and HNSC (Table 5) datasets, and achieves the second best result in LUAD (Table 6) dataset. Note that in the latter, although the mean test AUC obtained with the DBLR is slightly worse than the one provided by the HSIC, its standard deviation is lower, meaning that the results are more stable.

In the case of LGG and HNSC, the basic version of the model outperforms the semi-supervised approach, while for BRCA and LUAD, the semi-supervised version achieves the best performance. These results may suggest that not all the datasets benefit from the semi-supervised approach, and one possible explanation is the additional introduction of noise through not annotated samples (recall that we are handling test samples as the target was missing). However, some datasets may benefit from this approach, since the introduction of these not annotated observations can additionally regularize the training process and thus, improve generalization.

If we compare the performance of the different baselines, we can see that S-UMAP achieves the worst or one of the worst classification metrics for all the datasets in test. Linking these results with the ones obtained for synthetic data,

| Model | AUC Train | AUC Test |
|---|---|---|
| PCA + LR (L1) | $0.9729 \pm 0.0296$ | $0.7761 \pm 0.0539$ |
| PCA + LR (L2) | $0.9673 \pm 0.0258$ | $0.7758 \pm 0.0561$ |
| KPCA + LR (L1) | $0.9734 \pm 0.0286$ | $0.7740 \pm 0.0538$ |
| KPCA + LR (L2) | $0.9685 \pm 0.0264$ | $0.7753 \pm 0.0569$ |
| CCA + LR (L1) | $1.0000 \pm 0.0000$ | $0.7383 \pm 0.0787$ |
| CCA + LR (L2) | $1.0000 \pm 0.0000$ | $0.7383 \pm 0.0787$ |
| PLS + LR (L1) | $0.8495 \pm 0.0066$ | $0.7621 \pm 0.0887$ |
| PLS + LR (L2) | $0.8490 \pm 0.0068$ | $0.7598 \pm 0.0907$ |
| S-UMAP + LR (L1) | $0.9802 \pm 0.0161$ | $0.6404 \pm 0.0707$ |
| S-UMAP + LR (L2) | $0.9802 \pm 0.0163$ | $0.6470 \pm 0.0709$ |
| HSIC + LR (L1) | $0.8197 \pm 0.0318$ | $0.7638 \pm 0.0651$ |
| HSIC + LR (L2) | $0.7663 \pm 0.0693$ | $0.7569 \pm 0.0685$ |
| **DBLR** | $\mathbf{0.9172 \pm 0.0470}$ | $\mathbf{0.7981 \pm 0.0293}$ |
| DBLR SS | $0.8790 \pm 0.0921$ | $0.7294 \pm 0.0652$ |

Table 4: 5-fold results for LGG dataset. Mean values of the validated hyperparameters are $K = 303.20$, $p = 0.10$ and $hdim = 100$ for the DBLR, and $K = 263.20$, $p = 0.35$ and $hdim = 100$ for the semi-supervised DBLR.

| Model | AUC Train | AUC Test |
|---|---|---|
| PCA + LR (L1) | $0.7941 \pm 0.0909$ | $0.6231 \pm 0.0510$ |
| PCA + LR (L2) | $0.7539 \pm 0.0197$ | $0.6340 \pm 0.0440$ |
| KPCA + LR (L1) | $0.7946 \pm 0.0921$ | $0.6238 \pm 0.0507$ |
| KPCA + LR (L2) | $0.7540 \pm 0.0198$ | $0.6340 \pm 0.0440$ |
| CCA + LR (L1) | $0.8347 \pm 0.0121$ | $0.6279 \pm 0.0348$ |
| CCA + LR (L2) | $0.8343 \pm 0.0125$ | $0.6265 \pm 0.0355$ |
| PLS + LR (L1) | $0.8347 \pm 0.0121$ | $0.6279 \pm 0.0348$ |
| PLS + LR (L2) | $0.8343 \pm 0.0125$ | $0.6265 \pm 0.0355$ |
| S-UMAP + LR (L1) | $1.0000 \pm 0.0000$ | $0.6038 \pm 0.0948$ |
| S-UMAP + LR (L2) | $1.0000 \pm 0.0000$ | $0.5792 \pm 0.1092$ |
| HSIC + LR (L1) | $0.7462 \pm 0.1438$ | $0.5526 \pm 0.0333$ |
| HSIC + LR (L2) | $0.6204 \pm 0.0375$ | $0.5415 \pm 0.0393$ |
| **DBLR** | $\mathbf{0.7232 \pm 0.0884}$ | $\mathbf{0.6435 \pm 0.0365}$ |
| DBLR SS | $0.9156 \pm 0.0520$ | $0.6014 \pm 0.0522$ |

Table 5: 5-fold results for HNSC dataset. Mean values of the validated hyperparameters are $K = 265.40$, $p = 0.17$ and $hdim = 25$ for the DBLR, and $K = 284.20$, $p = 0.04$ and $hdim = 100$ for the semi-supervised DBLR.

we can hypothesise that this dimensionality reduction method tends to provide overfitted solutions that may not be suitable for further analysis, although it can be a powerful tool for data visualization.

In the case of the HSIC, the results vary importantly across the different datasets. As this is a feature selection method, results depend on the presence or absence of variables that are highly correlated with the target. The rest of the

| Model | AUC Train | AUC Test |
|---|---|---|
| PCA + LR (L1) | $0.6358 \pm 0.0257$ | $0.5853 \pm 0.0434$ |
| PCA + LR (L2) | $0.6842 \pm 0.0318$ | $0.6073 \pm 0.0543$ |
| KPCA + LR (L1) | $0.6358 \pm 0.0256$ | $0.5853 \pm 0.0433$ |
| KPCA + LR (L2) | $0.6843 \pm 0.0318$ | $0.6075 \pm 0.0541$ |
| CCA + LR (L1) | $1.0000 \pm 0.0000$ | $0.5521 \pm 0.0833$ |
| CCA + LR (L2) | $1.0000 \pm 0.0000$ | $0.5521 \pm 0.0833$ |
| PLS + LR (L1) | $0.8971 \pm 0.0118$ | $0.5805 \pm 0.0686$ |
| PLS + LR (L2) | $0.8971 \pm 0.0118$ | $0.5815 \pm 0.0691$ |
| S-UMAP + LR (L1) | $1.0000 \pm 0.0000$ | $0.5759 \pm 0.0521$ |
| S-UMAP + LR (L2) | $1.0000 \pm 0.0000$ | $0.6011 \pm 0.0458$ |
| HSIC + LR (L1) | $0.6832 \pm 0.1103$ | $0.6040 \pm 0.0690$ |
| **HSIC + LR (L2)** | $\mathbf{0.6359 \pm 0097}$ | $\mathbf{0.6316 \pm 0.0539}$ |
| DBLR | $0.6887 \pm 0.1307$ | $0.5994 \pm 0.0434$ |
| DBLR SS | $0.7867 \pm 0.1325$ | $0.6302 \pm 0.0308$ |

Table 6: 5-fold results for LUAD dataset. Mean values of the validated hyperparameters are $K = 19.20$, $p = 0.21$ and $hdim = 25$ for the DBLR, and $K = 225.80$, $p = 0.12$ and $hdim = 25$ for the semi-supervised DBLR.

methods can combine input features to extract new variables that capture the structure of the data, but in this case we can only select the ones that provide more information to the problem, but it may not be enough in all the scenarios.

For Kernel Principal Component Analysis (KPCA) and PCA, we obtained very similar results in all datasets. We trained the KPCA using a nonlinear kernel, particularly the Radial Basis Function (RBF) kernel. However, the parameter $\gamma$, which indicates the inverse of the standard deviation of the RBF kernel, was set to zero during hyperparameter tuning in all the cases, which is equivalent to use a linear kernel. This implies that both methods are applying a linear dimensionality reduction and thus, they achieve similar results.

It is important to confirm that our proposed method can achieve state-of-the-art classification performance while inferring a low-dimensional latent distribution of the data, capturing its underlying structure. In addition, given the probabilistic generative nature of the DBLR model, it can handle partially missing observations in the feature vectors and the target variable.

*3.4.3. Imputation of MAR data on The Cancer Genome Atlas (TCGA) dataset.*

Finally, we show that the DBLR can be trained with partially-missing observations, which can be imputed sampling from the model. Note that neither of the baseline methods can naturally deal with partially observed values or allow their probabilistic imputation. Figure 4 shows the imputation results obtained for a random feature in LGG dataset. Half the observations of that feature were set to missing in the train partition. Once the model was trained, we projected the samples over the latent space using $q(\mathbf{z}_i|\mathbf{x}_i)$ in (12) and then reconstructed the low-dimensional representation using $p(\mathbf{x}_i|\mathbf{z}_i, \mathbf{y}_i)$ in (8), obtaining an imputation of the missing values. We show the results for both the observed and

missing samples in train, as well as the results for the test partition, where we treated all the samples as missing entries. This way, we show that the model can impute missing values of new observations not used during training. Additionally, we compare our results with the ones obtained using a K-Nearest Neighbors (KNN) imputer [5] and sampling from a normal distribution with the empirical mean and variance of the observed training data.

The results show that the model is robust against missing entries, as it can still provide a good approximation of the distribution of the data. In fact, the distribution inferred by the DBLR is the one that better approximates the true data, capturing both the mode and the tails of the true density. The results show that the KNN imputer does not capture the tails of the distribution and the Normal distribution with the empirical mean and variance fails to capture the mode of the data.

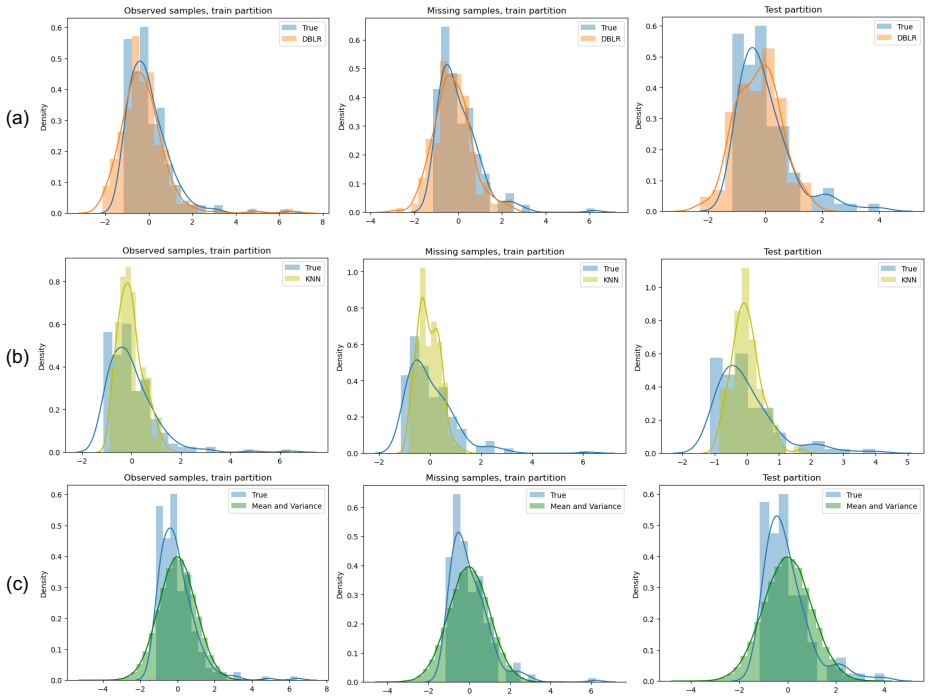


Figure 4: Empirical distribution of the data (represented in blue) against the inferred distributions obtained applying different methods. (a) Results using the DBLR. (b) Results using a KNN imputer with 5 neighbors. (c) Results using the empirical mean and variance of the observed training data.

[5] We used Scikit-learn implementation.

## 4. Discussion

In this work, we presented the DBLR, a semi-supervised bayesian latent space model that follows a VAE approach to infer a low-dimensional representation of high-dimensional input data. The inference process is driven by the label value, which allows to capture the underlying structure of the data and thus obtain a separable latent representation. Additionally, the model learns a global vector of weights to predict the label value given the low-dimensional embedding of the input data.

We showed in both synthetic and real databases that the model can infer informative embeddings, capturing the nature of the latent distribution of the data while enhancing separation with respect to the label value. This separation in the latent space allows the DBLR to obtain better classification results than other state-of-the-art methods given the low-dimensional embedding, as showed in the experiments.

In this context, it is important to remark that the proposed method is a probabilistic generative model. This implies that it infers probability densities we can draw samples from. Thus, it can handle missing entries in both the target variable and the input data, which is particularly relevant in clinical and life-science scenarios, where it is common to find censored samples and partially missing observations. We showed that the DBLR can be trained in the presence of missing values and still infer the distribution of the data (for both observed and missing entries).

The proposed model not only can handle MAR data, but it benefits from this capacity to additionally regularize the training process by means of a drop-out-like strategy: a percentage of the observed targets is set as missing randomly and their values are imputed sampling from the model. This way, the model learns from a different set of observed samples at each training epoch.

## 5. Funding

This work was supported by Spanish Ministerio de Ciencia e Innovación / Agencia Española de Investigación [grant numbers RTI2018-099655-B-10, PID2021-123182OB-I00]; by the Comunidad de Madrid [grant numbers IND2022/TIC-23550]; the European Union (FEDER and the European Research Council (ERC) through the European Unions Horizon 2020 research innovation program [grant number 714161]; and by the Instituto de Investigación Sanitaria Gregorio Marañón [grant Intramural].

## References

[1] Berisha, V., Krantsevich, C., Hahn, P.R., Hahn, S., Dasarathy, G., Turaga, P., Liss, J., 2021. Digital medicine and the curse of dimensionality. NPJ digital medicine 4, 1–8.

[2] Chen, R., Yang, L., Goodison, S., Sun, Y., 2020. Deep-learning approach to identifying cancer subtypes using high-dimensional genomic data. Bioinformatics 36, 1476–1483.

[3] Climente-González, H., Azencott, C.A., Kaski, S., Yamada, M., 2019. Block hsic lasso: model-free biomarker detection for ultra-high dimensional data. Bioinformatics 35, i427–i435.

[4] Diaz-Papkovich, A., Anderson-Trocmé, L., Gravel, S., 2021. A review of umap in population genetics. Journal of Human Genetics 66, 85–91.

[5] Dong, Z., Alterovitz, G., 2021. netae: semi-supervised dimensionality reduction of single-cell rna sequencing to facilitate cell labeling. Bioinformatics 37, 43–49.

[6] Feng, H.Z., Kong, K., Chen, M., Zhang, T., Zhu, M., Chen, W., 2021. Shot-vae: semi-supervised deep generative models with label-aware elbo approximations, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7413–7421.

[7] Geladi, P., Kowalski, B.R., 1986. Partial least-squares regression: a tutorial. Analytica chimica acta 185, 1–17.

[8] Gretton, A., Bousquet, O., Smola, A., Schölkopf, B., 2005. Measuring statistical dependence with hilbert-schmidt norms, in: International conference on algorithmic learning theory, Springer. pp. 63–77.

[9] Hira, M.T., Razzaque, M., Angione, C., Scrivens, J., Sawan, S., Sarker, M., 2021. Integrated multi-omics analysis of ovarian cancer using variational autoencoders. Scientific reports 11, 1–16.

[10] Hotelling, H., 1933. Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology 24, 498–520.

[11] Hotelling, H., 1992. Relations between two sets of variates, in: Breakthroughs in statistics. Springer, pp. 162–190.

[12] Hwang, B., Lee, J.H., Bang, D., 2018. Single-cell rna sequencing technologies and bioinformatics pipelines. Experimental & molecular medicine 50, 1–14.

[13] Jang, E., Gu, S., Poole, B., 2016. Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 .

[14] Kadkhoda, S., Darbeheshti, F., Rezaei, N., Azizi-Tabesh, G., Zolfaghari, F., Tavakolibazaz, S., Taslimi, R., Tavakkoly-Bazzaz, J., 2021. Investigation of circrna-mirna-mrna network in colorectal cancer using an integrative bioinformatics approach. Gastroenterology and Hepatology From Bed to Bench 14, 141.

[15] Ke, F., Kong, W., Wang, S., 2021. Identifying imaging genetics biomarkers of alzheimer's disease by multi-task sparse canonical correlation analysis and regression. Frontiers in Genetics 12.

[16] Kingma, D.P., Welling, M., 2013. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 .

[17] Lopez, R., Regier, J., Cole, M.B., Jordan, M.I., Yosef, N., 2018. Deep generative modeling for single-cell transcriptomics. Nature methods 15, 1053–1058.

[18] Van der Maaten, L., Hinton, G., 2008. Visualizing data using t-sne. Journal of machine learning research 9.

[19] McInnes, L., Healy, J., Melville, J., 2018. Umap: Uniform manifold approximation and projection for dimension reduction. URL: `https://arxiv.org/abs/1802.03426`, doi:`10.48550/ARXIV.1802.03426`.

[20] Nazabal, A., Olmos, P.M., Ghahramani, Z., Valera, I., 2020. Handling incomplete heterogeneous data using vaes. Pattern Recognition 107, 107501.

[21] Nie, W., Wang, Z., Patel, A.B., Baraniuk, R.G., 2020. An improved semi-supervised vae for learning disentangled representations. arXiv preprint arXiv:2006.07460 .

[22] Peis, I., Olmos, P.M., Artés-Rodríguez, A., 2020. Unsupervised learning of global factors in deep generative models. arXiv preprint arXiv:2012.08234 .

[23] Peng, H., Long, F., Ding, C., 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence 27, 1226–1238.

[24] Poirion, O.B., Jing, Z., Chaudhary, K., Huang, S., Garmire, L.X., 2021. Deepprog: an ensemble of deep-learning and machine-learning models for prognosis prediction using multi-omics data. Genome medicine 13, 1–15.

[25] Rashid, S., Shah, S., Bar-Joseph, Z., Pandya, R., 2021. Dhaka: variational autoencoder for unmasking tumor heterogeneity from single cell genomic data. Bioinformatics 37, 1535–1543.

[26] Rich-Griffin, C., Stechemesser, A., Finch, J., Lucas, E., Ott, S., Schäfer, P., 2020. Single-cell transcriptomics: a high-resolution avenue for plant functional genomics. Trends in plant science 25, 186–197.

[27] Schölkopf, B., Smola, A., Müller, K.R., 1998. Nonlinear component analysis as a kernel eigenvalue problem. Neural computation 10, 1299–1319.

[28] Wang, Z., Wang, Y., 2019. Extracting a biologically latent space of lung cancer epigenetics with variational autoencoders. BMC bioinformatics 20, 1–7.

[29] Way, G.P., Greene, C.S., 2018. Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders, in: PACIFIC SYMPOSIUM ON BIOCOMPUTING 2018: Proceedings of the Pacific Symposium, World Scientific. pp. 80–91.

[30] Yamada, R., Okada, D., Wang, J., Basak, T., Koyama, S., 2021. Interpretation of omics data analyses. Journal of Human Genetics 66, 93–102.

[31] Zemouri, R., 2020. Semi-supervised adversarial variational autoencoder. Machine Learning and Knowledge Extraction 2, 20.

[32] Zhang, Y., Lu, Z., 2019. Exploring semi-supervised variational autoencoders for biomedical relation extraction. Methods 166, 112–119.

## Appendix A. DBLR description

*Appendix A.1. Variational Family*

As it is described in the main document, the posterior distribution $q(\mathbf{Z}, \mathbf{w}|\mathcal{D})$ is approximated via VI, using the following variational family:

$$q(\mathbf{Z}, \mathbf{w}|\mathcal{D}) = q(\mathbf{Z}|\mathbf{X})q(\mathbf{w}|\mathbf{Z}, \mathbf{y}) \tag{A.1}$$

The second term, which refers to the posterior distribution of the weight vector $\mathbf{w}$, is given by

$$\begin{aligned} q(\mathbf{w}|\mathbf{Z}, \mathbf{y}) &= \mathcal{N}(\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w) \\ \propto \prod_{i=1}^{N} \mathcal{N}\Big(\boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,w}}(\mathbf{z}_i), \boldsymbol{\Sigma}_{\boldsymbol{\eta}^1_{c,w}}(\mathbf{z}_i)\Big)^{y_i} &\mathcal{N}\Big(\boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,w}}(\mathbf{z}_i), \boldsymbol{\Sigma}_{\boldsymbol{\eta}^0_{c,w}}(\mathbf{z}_i)\Big)^{(1-y_i)} \end{aligned} \tag{A.2}$$

where the mean vector is given by

$$\boldsymbol{\mu}_w = \Big(\sum_{i=1}^{N} \boldsymbol{\Sigma}_{\boldsymbol{\eta}^{y_i}_{c,w}}(\mathbf{z}_i)^{-1}\Big)^{-1} \Big(\sum_{i=1}^{N} \boldsymbol{\Sigma}_{\boldsymbol{\eta}^{y_i}_{c,w}}(\mathbf{z}_i)^{-1} \boldsymbol{\mu}_{\boldsymbol{\eta}^{y_i}_{m,w}}(\mathbf{z}_i)\Big) \tag{A.3}$$

and the covariance is computed as

$$\boldsymbol{\Sigma}_w = \Big(\sum_{i=1}^{N} \boldsymbol{\Sigma}_{\boldsymbol{\eta}^{y_i}_{c,w}}(\mathbf{z}_i)^{-1}\Big)^{-1} \tag{A.4}$$

Recall that, as it is a global variable, it is obtained as a product of the local contribution of each observed data point. The mean vectors $\boldsymbol{\mu}_{m,w}$ and the diagonal of the covariance matrix $\boldsymbol{\Sigma}_{m,w}$ are learnt by means of four neural networks with input $\mathbf{z}_i$ and parameters $\boldsymbol{\eta}^1_{m,w}, \boldsymbol{\eta}^0_{m,w}, \boldsymbol{\eta}^1_{c,w}$ and $\boldsymbol{\eta}^0_{c,w}$.

*Appendix A.2. ELBO*

Inference is done maximizing the Evidence Lower Bound (ELBO), which can be expressed as

$$\begin{aligned} ELBO = \int q(\mathbf{Z}, \mathbf{w}|\mathcal{D}) \log \frac{p(\mathbf{Z}, \mathbf{w}, \mathcal{D})}{q(\mathbf{Z}, \mathbf{w}|\mathcal{D})} d\mathbf{Z}d\mathbf{w} = \\ \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{y}|\mathbf{Z}, \mathbf{w})\Big] + \mathbb{E}_{q(\mathbf{Z},\mathbf{w}\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\Big] \\ -\mathcal{D}_{KL}\big(q(\mathbf{w}|\mathbf{Z}, \mathbf{y})||p(\mathbf{w})\big) - \mathcal{D}_{KL}\big(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})\big) \end{aligned} \tag{A.5}$$

The first term (A.6) corresponds to the reconstruction of the label $\mathbf{y}$, where $\sigma(\cdot)$ refers to the Sigmoid function.

$$\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{y}|\mathbf{Z},\mathbf{w})\Big] =$$
$$\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\sum_{i=1}^{N} y_i \log(\sigma(\mathbf{w}^T\mathbf{z}_i + w_0)) + (1-y_i)\log(1-\sigma(\mathbf{w}^T\mathbf{z}_i + w_0))\Big] \tag{A.6}$$

The second term corresponds to the reconstruction of the input data. Maximizing this term, we are inferring the low dimensional representation of the data that allows for the best possible reconstruction of $\mathbf{X}$. Depending on the nature of the data (*real* (A.7) or*binary* (A.8)), this term can be expressed as

- *Real input case*

$$\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z},\mathbf{y})\Big] =$$
$$-\frac{1}{2}\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\sum_{i=1}^{N} y_i\Big[\log\big(\det\big(2\pi\mathcal{I}\big)\big) + \big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,x}}(\mathbf{z}_i)\big)^T\big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,x}}(\mathbf{z}_i)\big)\Big]$$
$$+(1-y_i)\Big[\log\big(\det\big(2\pi\mathcal{I}\big)\big) + \big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,x}}(\mathbf{z}_i)\big)^T\big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,x}}(\mathbf{z}_i)\big)\Big]\Big] \tag{A.7}$$

- *Binary input case*

$$\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z},\mathbf{y})\Big] =$$
$$\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\sum_{i=1}^{N} y_i \sum_{j=1}^{D} x_{i,j}\log(\theta^1(\mathbf{z}_i)_j) + (1-x_{i,j})\log(1-\theta^1(\mathbf{z}_i)_j)$$
$$+(1-y_i)\sum_{j=1}^{D} x_{i,j}\log(\theta^0(\mathbf{z}_i)_j) + (1-x_{i,j})\log(1-\theta^0(\mathbf{z}_i)_j)\Big] \tag{A.8}$$

The two remaining terms allow for regularization. These correspond to the KL Divergence between the variational family and the prior distributions. Since the priors are Gaussian, the KL Divergences can be expressed in closed form as

$$\mathcal{D}_{KL}\big(q(\mathbf{w}|\mathbf{Z},\mathbf{y})||p(\mathbf{w})\big) =$$
$$\frac{1}{2}\sum_{j=1}^{K}\big(\gamma_{j,\boldsymbol{\eta}_{c,w}} + \boldsymbol{\mu}^2_{j,\boldsymbol{\eta}_{m,w}} - 1 - \log(\gamma_{j,\boldsymbol{\eta}_{c,w}})\big) \tag{A.9}$$

$$\mathcal{D}_{KL}\big(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})\big) =$$
$$\sum_{i=1}^{N}\frac{1}{2}\sum_{j=1}^{K}\big(\gamma_{i,j,\boldsymbol{\eta}_{c,z}} + \boldsymbol{\mu}^2_{i,j,\boldsymbol{\eta}_{m,z}} - 1 - \log(\gamma_{i,j,\boldsymbol{\eta}_{c,z}})\big), \tag{A.10}$$

where $\gamma$ refers to the diagonal of the covariance matrix $\mathbf{\Sigma}$.

Although the reconstruction term given by (A.7) and (A.8) may numerically dominate over the likelihood term (A.6) in the case of high-dimensional inputs, the model can still learn to make accurate predictions, as the target variable is indirectly used during inference and regularization. This means that prediction of label values is embedded in the computation of the rest of the ELBO terms, so the model can learn to properly predict despite the possible unbalance between the different likelihood terms.