Supplementary Material

Handling ill-conditioned omics data with deep probabilistic models.

María Martínez-García and Pablo M. Olmos

This document contains supplementary material related to the definition of the Deep Bayesian Logistic Regression (DBLR) model, as well as additional experiments to analyze the performance of the model.

1 DBLR description

1.1 Evidence Lower Bound (ELBO)

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

$$\begin{gathered} ELBO = \int q(\mathbf{Z}, \mathbf{w}|\mathcal{D}) \log \frac{p(\mathbf{Z}, \mathbf{w}, \mathcal{D})}{q(\mathbf{Z}, \mathbf{w}|\mathcal{D})} d\mathbf{Z} d\mathbf{w} = \\ \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{y}|\mathbf{Z}, \mathbf{w})\Big] + \mathbb{E}_{q(\mathbf{Z},\mathbf{w}\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\Big] \\ -\mathcal{D}_{KL}\big(q(\mathbf{w}|\mathbf{Z}, \mathbf{y})||p(\mathbf{w})\big) - \mathcal{D}_{KL}\big(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})\big) \end{gathered} \tag{1}$$

The first term (2) corresponds to the reconstruction of the label $\mathbf{y}$, where $\sigma(\cdot)$ refers to the Sigmoid function.

$$\begin{gathered} \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{y}|\mathbf{Z}, \mathbf{w})\Big] = \\ \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\sum_{i=1}^{N} y_i \log(\sigma(\mathbf{w}^T\mathbf{z}_i + w_0)) + (1 - y_i)\log(1 - \sigma(\mathbf{w}^T\mathbf{z}_i + w_0))\Big] \end{gathered} \tag{2}$$

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* (3) or*binary* (4)), this term can be expressed as

- *Real input case*

$$\begin{gathered} \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\Big] = \\ -\frac{1}{2}\mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\sum_{i=1}^{N} y_i\Big[\log\big(\det\big(2\pi\mathcal{I}\big)\big) + \big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,x}}(\mathbf{z}_i)\big)^T\big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^1_{m,x}}(\mathbf{z}_i)\big)\Big] \\ +(1 - y_i)\Big[\log\big(\det\big(2\pi\mathcal{I}\big)\big) + \big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,x}}(\mathbf{z}_i)\big)^T\big(\mathbf{x}_i - \boldsymbol{\mu}_{\boldsymbol{\eta}^0_{m,x}}(\mathbf{z}_i)\big)\Big]\Big] \end{gathered} \tag{3}$$

- *Binary input case*

$$\begin{gathered} \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{y})\Big] = \\ \mathbb{E}_{q(\mathbf{Z},\mathbf{w}|\mathcal{D})}\Big[\sum_{i=1}^{N} y_i \sum_{j=1}^{D} x_{i,j}\log(\theta^1(\mathbf{z}_i)_j) + (1 - x_{i,j})\log(1 - \theta^1(\mathbf{z}_i)_j) \\ +(1 - y_i)\sum_{j=1}^{D} x_{i,j}\log(\theta^0(\mathbf{z}_i)_j) + (1 - x_{i,j})\log(1 - \theta^0(\mathbf{z}_i)_j)\Big] \end{gathered} \tag{4}$$

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

$$\mathcal{D}_{KL}\big(q(\mathbf{w}|\mathbf{Z},\mathbf{y})||p(\mathbf{w})\big) = \frac{1}{2}\sum_{j=1}^{K}\big(\gamma_{j,\boldsymbol{\eta}_{c,w}} + \boldsymbol{\mu}^2_{j,\boldsymbol{\eta}_{m,w}} - 1 - \log(\gamma_{j,\boldsymbol{\eta}_{c,w}})\big) \tag{5}$$

$$\mathcal{D}_{KL}\big(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})\big) = \sum_{i=1}^{N}\frac{1}{2}\sum_{j=1}^{K}\big(\gamma_{i,j,\boldsymbol{\eta}_{c,z}} + \boldsymbol{\mu}^2_{i,j,\boldsymbol{\eta}_{m,z}} - 1 - \log(\gamma_{i,j,\boldsymbol{\eta}_{c,z}})\big), \tag{6}$$

where γ refers to the diagonal of the covariance matrix Σ.

Although the reconstruction term given by (3) and (4) may numerically dominate over the likelihood term (2) 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 the prediction of the label values is embedded in the computation of the rest of the ELBO terms, so the model can learn to correctly predict despite the possible unbalance between the different likelihood terms.

2 Additional experiments

2.1 Benefits of dimensionality reduction

We conducted two experiments to demonstrate the benefits of dimensionality reduction. The first experiment involved training a Multilayer Perceptron (MLP) classifier in the four high-dimensional data sets (Breast Invasive Carcinoma (BRCA), Head and Neck Squamous Cell Carcinoma (HNSC), Brain Lower Grade Glioma (LGG), and Lung Adenocarcinoma (LUAD)) used in the primary manuscript, without applying any dimensionality reduction techniques. Subsequently, we applied various linear and nonlinear methods (Principal Component Analys (PCA), Canonical Correlation Analysis (CCA), and Uniform Manifold Approximation and Projection (UMAP)) to reduce the dimensionality of these data sets and retrain the classifier. We utilized a MLP classifier with a hidden layer of dimension 100 and ReLU activation layers. In the case of PCA, the number of components was adjusted by cross-validation. The same 5 folds utilized in the primary experiments were used for each data set, with the results presented in terms of the mean and standard deviation across the 5 folds in tables 1, 2, 3 and 4.

Model	**AUC Train**	**AUC Test**
MLPClassifier	0.7180 ± 0.1456	0.4706 ± 0.0330
PCA + MLPClassifier	0.6927 ± 0.1336	0.5154 ± 0.0280
CCA + MLPClassifier	0.9277 ± 0.0430	0.5380 ± 0.0292
UMAP + MLPClassifier	1.0000 ± 0.0000	0.4994 ± 0.0437

Table 1: 5-fold results for BRCA dataset.

Model	**AUC Train**	**AUC Test**
MLPClassifier	0.9198 ± 0.0887	0.6048 ± 0.0856
PCA + MLPClassifier	0.8098 ± 0.1397	0.6236 ± 0.0856
CCA + MLPClassifier	1.0000 ± 0.0000	0.6916 ± 0.0515
UMAP + MLPClassifier	0.8808 ± 0.2132	0.5385 ± 0.0457

Table 2: 5-fold results for LGG dataset.

The general trend observed is that classifiers trained using the reduced feature space exhibit better generalization, with larger Area Under the Receiver Operating Characteristics Curve (AUC) values for test partitions. Moreover, this experiment reaffirms one of the hypotheses proposed in the original manuscript, where we mentioned that UMAP may not be well suited for this type of analysis, as it tends to overfit the

training partition, impeding generalization. In this experiment, the results obtained with UMAP were worse than those obtained with simpler methods such as PCA or CCA.

Model	AUC Train	AUC Test
MLPClassifier	0.8356 ± 0.1182	0.5384 ± 0.0488
PCA + MLPClassifier	0.8037 ± 0.1895	0.5583 ± 0.0576
CCA + MLPClassifier	1.0000 ± 0.0000	0.5466 ± 0.0984
UMAP + MLPClassifier	1.0000 ± 0.0000	0.5553 ± 0.0490

Table 3: 5-fold results for HNSC dataset.

Model	AUC Train	AUC Test
MLPClassifier	0.9334 ± 0.0799	0.5464 ± 0.0231
PCA + MLPClassifier	0.7540 ± 0.1460	0.5829 ± 0.0481
CCA + MLPClassifier	1.0000 ± 0.0000	0.5737 ± 0.0367
UMAP + MLPClassifier	0.9840 ± 0.0235	0.5407 ± 0.0222

Table 4: 5-fold results for LUAD dataset.

In the second experiment, we trained a Logistic Regression model with L1 and L2 regularization on the same four high-dimensional data sets used in the first experiment, without reducing the number of features. This allowed us to compare the performance of the model trained on the raw data sets with that of the model trained with different dimensionality reduction techniques (results presented in the main manuscript). The regularization hyperparameter was adjusted via cross-validation in both models. Tables 5, 6, 7, 8 summarize the obtained results across 5 folds in terms of mean and standard deviation. Again, the results showed that classifiers trained with the reduced feature space exhibit better generalization.

In light of the results obtained, we can conclude that the introduction of dimensionality reduction methods improves the generalization capability of the classifiers in this scenario.

Model	AUC Train	AUC Test
LR L1	0.5406 ± 0.0910	0.5201 ± 0.0450
LR L2	0.5420 ± 0.0939	0.5188 ± 0.0421

Table 5: 5-fold results for BRCA dataset.

Model	AUC Train	AUC Test
LR L1	0.5799 ± 0.1095	0.5456 ± 0.0640
LR L2	0.5410 ± 0.0917	0.5188 ± 0.0421

Table 6: 5-fold results for LGG dataset.

Model	AUC Train	AUC Test
LR L1	0.5813 ± 0.0008	0.5000 ± 0.0000
LR L2	0.5814 ± 0.0008	0.5000 ± 0.0000

Table 7: 5-fold results for HNSC dataset.

Model	AUC Train	AUC Test
LR L1	0.6405 ± 0.0012	0.5000 ± 0.0000
LR L2	0.6405 ± 0.0010	0.5000 ± 0.0000

Table 8: 5-fold results for LUAD dataset.

2.2 Ablation study of hyperparameters p and K

We performed an ablation study of the parameters p and K to analyze their influence on the model classification performance. The purpose of the parameter p is to determine the percentage of the target $\mathbf{y}$ that is randomly set as missing in each iteration of the gradient descent optimization. The parameter K determines the dimensionality of the latent space.

First, we trained the model with different values of the parameter p (from 0 to 0.5), keeping the dimension of the latent space ($K = 250$) and the train/test partitions fixed. The model was trained three times for each configuration, and the training with the highest AUC score in the test partition was selected to account for possible differences due to the initial conditions.

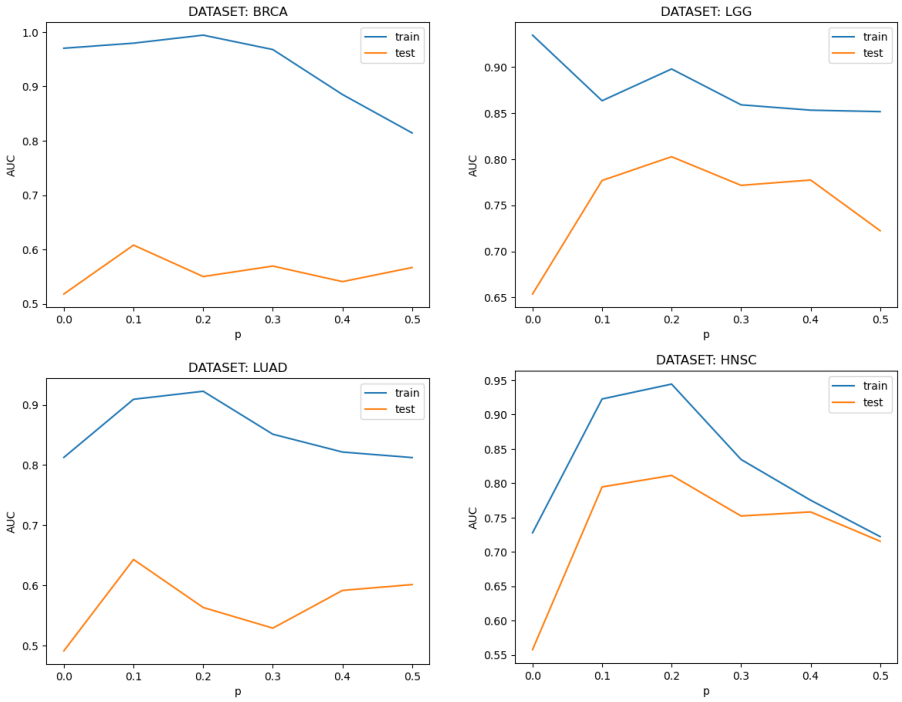

Figure 1: Ablation study for the parameter p across four different real datasets.

In this experiment, we observed that the model's generalization improved when the proposed drop-out strategy was applied. The model achieved the best results with values of p between 0.1 and 0.2 for all data sets. For larger values of p, the performance of the model was degraded due to the extra noise introduced during training. Figure 1 shows the AUC obtained in train and test partitions for the different configurations of the model in four real data sets, namely BRCA, HNSC, LGG, and LUAD data sets.

Then we trained the model with different values of the parameter K (from 5 to 500), keeping the drop-out probability ($p = 0.1$ in light of the results of the previous experiment) and the train/test partitions fixed. The model was again trained three times for each configuration to account for possible differences due to the initial conditions. The results are shown in Figure 2.

We observed that, in general, increasing the dimension of the latent space resulted in an improvement of the performance in the train partition but not necessarily in a better generalization of the model. In fact, in the cases of LGG and LUAD, for example, the AUC in the test partition remained stable while the AUC

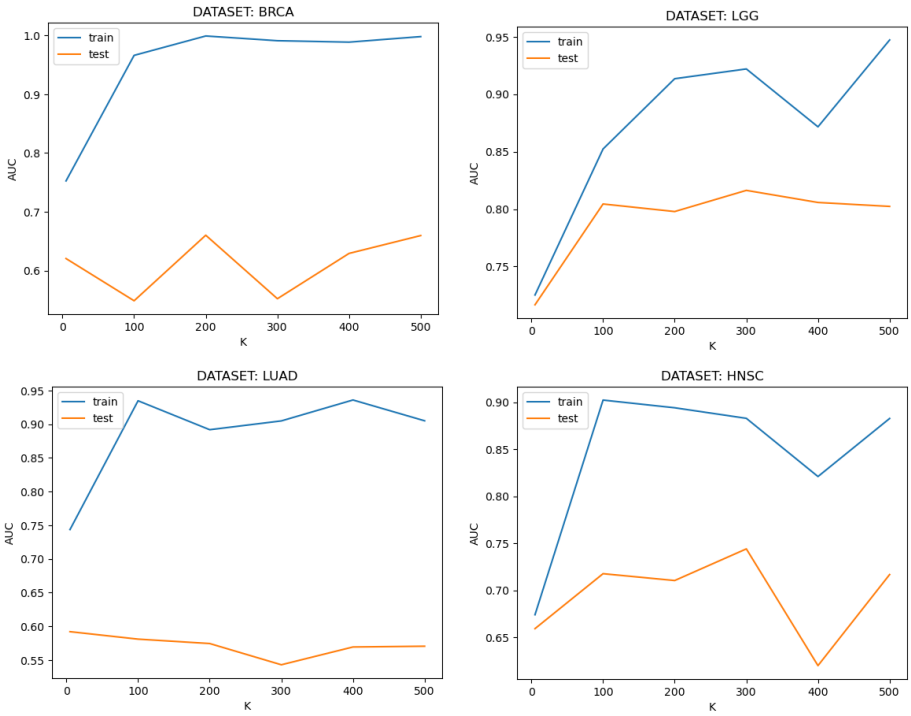

Figure 2: Ablation study for the parameter K across four different real datasets.

in the train partition increased. These results suggest that the model is capable of effectively capturing the essential information with a reduced dimensionality in the latent space. Increasing the dimensionality of this projection does not necessarily result in a better classification performance, but instead in more overfitted solutions.

In light of these results, we confirm that the selected configurations for the experiments in the main manuscript are correct, as the selected values for p and K agree with the results of the ablation study.

2.3 Ablation study of the DBLR's tolerance to missing labels

We conducted an ablation study of the DBLR's tolerance to missing labels. In this case, the unlabeled samples were prefixed and never revealed during training. In addition, the DBLR was still using the dropout regularization ($p = 0.1$) for the labeled observations. Figure 3 shows the results obtained for different percentages (from 5% to 30%) of unlabeled samples in four real data sets, namely BRCA, LGG, LUAD, and HNSC. The results are presented in terms of mean and standard deviation across 5 folds. They show that the DBLR remains robust in the presence of missing labels, as performance did not degrade when we increased the percentage of unlabeled samples.

2.4 Evaluation of the imputation method

We have extended the evaluation of the imputation performance of our proposed method through the use of both Quantile-Quantile (Q-Q) plots and goodness-of-fit tests, with the aim of providing a comprehensive comparison between the DBLR and baseline methods.

According to the main experiment, half of the observations of a randomly selected feature were intentionally set to missing in the training partition. After training the model, we projected the samples into the latent space using $q(\mathbf{z}_i|\mathbf{x}_i)$ (eq. 12) and reconstructed the low-dimensional representation utilizing $p(\mathbf{x}_i|\mathbf{z}_i, y_i)$ (eq. 8) to obtain an imputation of the missing values. Our results show both the observed and missing samples in the training set, as well as those in the test partition, where all samples were treated as missing entries.

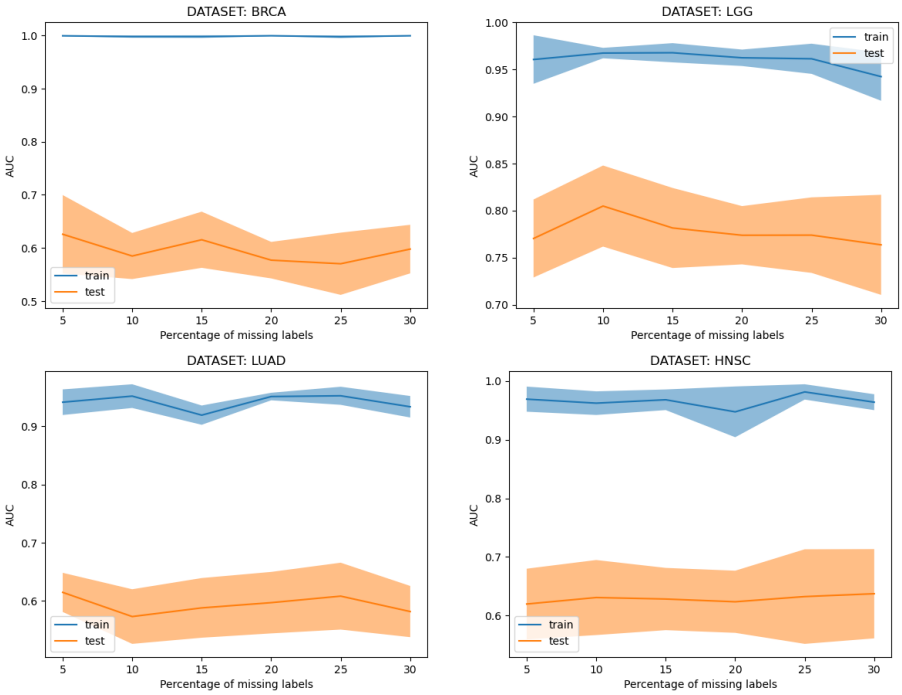

Figure 3: Ablation study of the DBLR's tolerance to missing labels across four real datasets.

In doing so, we demonstrate the model's ability to impute missing values for new observations not included in the training dataset.

Q-Q plots are a graphical technique utilized to analyze and compare two probability distributions plotting their quantiles against each other. If the two distributions are equal, the points on the Q-Q plot should lie along the line $y = x$. We generated Q-Q plots for the different data partitions and imputation methods. The results obtained, shown in Figure 4 indicate that the DBLR model is more effective in approximating the true data even for new observations not seen during the training phase, as demonstrated by the larger number of samples lying along the straight line for every partition compared to the baseline methods.

In addition to the graphical representations presented (that is, the histograms of the distributions and the Q-Q plots), we conducted a quantitative evaluation of the performance of the imputation methods. We first employed the Kolmogorov-Smirnov (K-S) test, a nonparametric measure of the similarity between two continuous probability distributions, to compare the distribution of the true data and the imputed data. Specifically, we used the two-sample K-S test, as our objective was to compare the distribution of two different samples. Table 11 summarizes the results obtained. At a 95% confidence level, the DBLR was the only method for which the null hypothesis, which postulates that the distribution of the imputed data is equivalent to that of the true data, was not rejected. This indicates that the proposed method is capable of better aligning the imputed data with the true distribution.

Finally, we employed the A-D test, a variation of the K-S test, which gives greater emphasis to the tails of the distributions. While the K-S test is more sensitive to discrepancies near the central region of the distribution, the A-D test is more sensitive to variations present in the tails. Table 10 summarizes the results obtained. Again, at a 95% confidence level, the DBLR was the only method for which the null hypothesis was not rejected. These results were obtained using the Python package `scipy.stats`. It is important to note that the implementation of the A-D test involves flooring the p-values at 0.1% and capping them at 25%, which is why the exact p-values are not presented for all experiments.

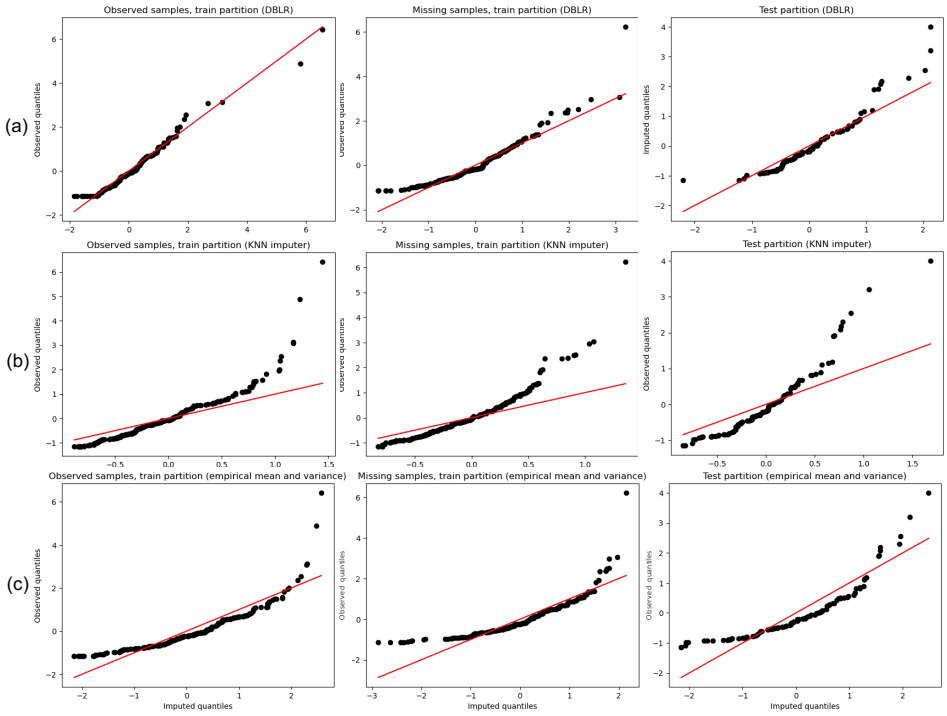

Figure 4: Q-Q plots the true data against the imputation applying different methods. (a) Results using the DBLR. (b) Results using a K-Nearest Neighbors (KNN) imputer with 5 neighbors. (c) Results using the empirical mean and variance of the observed training data.

Model	Observed Train	Missing Train	Test
DBLR	statistic = 0.0792 p-value = 0.5517	statistic = 0.0940 p-value = 0.3339	statistic = 0.1287 p-value = 0.3742
KNN imputer	statistic = 0.1930 p-value = 0.0010	statistic = 0.2030 p-value = 0.0005	statistic = 0.2475 p-value = 0.0040
Empirical mean and variance	statistic = 0.1930 p-value = 0.0010	statistic = 0.0940 p-value = 0.0411	statistic = 0.1287 p-value = 0.0040

Table 9: K-S results comparing the true data against the imputation applying different methods, namely the DBLR, a KNN imputer with 5 neighbors, and a Gaussian distribution with the empirical mean and variance of the observed training data.

Model	Observed Train	Missing Train	Test
DBLR	statistic = 0.3292 p-value = 0.2447	statistic = 1.5366 p-value = 0.0750	statistic = -0.1146 p-value > 0.25
KNN imputer	statistic = 9.0640 p-value < 0.001	statistic = 13.2474 p-value = ¡ 0.001	statistic = 7.4444 p-value < 0.001
Empirical mean and variance	statistic = 4.779 p-value = 0.0041	statistic = 2.9010 p-value = 0.0213	statistic = 2.1705 p-value = 0.0415

Table 10: Anderson-Darling (A-D) results comparing the true data against the imputation applying different methods, namely the DBLR, a KNN imputer with 5 neighbors, and a Gaussian distribution with the empirical mean and variance of the observed training data.

2.5 Time consumption

We have measured the training time consumption of the baseline methods used to compare the performance of the DBLR. For this experiment, we used one partition of the BRCA dataset, and the models were executed on a server that contains 2 Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz processors with 10 cores each and HyperThreading (40 "virtual" cores), and 128 GBytes of RAM and 1 GPU card (Nvidia GeForce GTX) with 10GB of RAM. The DBLR was trained 250 epochs, which means that for this data set it took approximately 0.47 s to run an epoch.

Model	**Training time (s)**
PCA + LR	0.96
KPCA + LR	2.20
CCA + LR	0.92
PLS + LR	0.35
UMAP + LR	15.25
HSIC + LR	53.17
DBLR (*250 epochs*)	117.71

Table 11: Time consumption of the different methods.