Explainability as statistical inference

Hugo Henri Joseph Senetaire [1] **Damien Garreau** [2] **Jes Frellsen** [*1] **Pierre-Alexandre Mattei** [*2]

Abstract

A wide variety of model explanation approaches have been proposed in recent years, all guided by very different rationales and heuristics. In this paper, we take a new route and cast interpretability as a statistical inference problem. We propose a general deep probabilistic model designed to produce interpretable predictions. The model's parameters can be learned via maximum likelihood, and the method can be adapted to any predictor network architecture, and any type of prediction problem. Our model is akin to amortized interpretability methods, where a neural network is used as a selector to allow for fast interpretation at inference time. Several popular interpretability methods are shown to be particular cases of regularized maximum likelihood for our general model. Using our framework, we identify imputation as a common issue of these models. We propose new datasets with ground truth selection which allow for the evaluation of the features importance map and show experimentally that multiple imputation provides more reasonable interpretations.

1 Introduction

Fueled by the recent advances in deep learning, machine learning models are becoming omnipresent in society. Their widespread use for decision making or predictions in critical fields leads to a growing need for transparency and interpretability of these methods. While Rudin (2019) argues that we should always favor interpretable models for high-stake decisions, in practice, black-box methods are used due to their superior predictive power. Researchers have proposed a variety of model explanation approaches for black-box models, and we refer to Linardatos et al. (2021) for a recent survey. Finding interpretable models is hard. The lack of consensus for evaluating an explanation (Afchar et al., 2021; Jethani et al., 2021a; Liu et al., 2021; Hooker et al., 2019) makes it difficult to assess the qualities of the different methods. Many leads are explored such as interpretability for unsupervised models (Crabbé & van der Schaar, 2022; Moshkovitz et al., 2020), using concept embedding models to obtain high-level explanation (Zarlenga et al., 2022), global feature selection (Tibshirani, 1996; Lemhadri et al., 2021). In this paper, we will focus on methods that offer an understanding of which features are important for the prediction of a given instance in a supervised learning setting. These types of methods are called instance-wise feature selection and quantify how much a prediction changes when only a subset of features is shown to the model.

Related work Many different rationales enable explainability with feature importance maps with different successes.

Gradient-based methods leverage the gradient of the target with respect to the input to get feature importance. For instance, Seo et al. (2018) create saliency maps by back-propagating gradient through the model. However, many works showed that these methods do not provide reliable explanations (Adebayo et al., 2018; Kindermans et al., 2019; Hooker et al., 2019; Slack et al., 2020).

Other methods proposed to approximate locally a complicated black-box model using simpler models. Ribeiro et al. (2016) fit an interpretable linear regression locally around a given instance with LIME while Lundberg & Lee (2017) approximates Shapley values (Shapley, 1953) for every instance with SHAP. Wang et al. (2021) proposed an algorithm that selects the most probable subset of features maximizing the same objective as the predictor called probabilistic sufficient explanation.

In practice, these methods focus on local explanations for a single instance. An evaluation of the selection for a single image requires a large number of passes through the black-box model. To alleviate this issue, (Wang et al., 2021) proposed to combine a beam search with probabilistic circuits (Choi et al., 2020). This allowed for tractable and faster calculation of the probabilistic sufficient explanations but only

[*]Equal contribution [1]Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark [2]Université Côte d'Azur, Inria, Maasai, LJAD, CNRS, Nice, France. Correspondence to: Hugo Henri Joseph Senetaire <hhjs@dtu.dk>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

for specific classifiers thus losing the generality of previous methods.

It is of particular interest to obtain explanations of multiple instances using amortized explanation methods. The idea of such methods is to train a *selector* network that will learn to predict the selection instead of calculating it from the ground up for every instance. While there is a higher cost of entry due to training an extra network, the interpretation at test time is much faster since we do not require multiple passes through the predictor. Such methods benefit from having a separated predictor and selector as they can explain pretrained predictors or train both predictor and selector at the same time with any architecture. Following Lundberg & Lee (2017), Jethani et al. (2021b) proposed to obtain Shapley values efficiently using a selector network. Chen et al. (2018); Yoon et al. (2018) both proposed to train a selector that selects a minimum subset of features while maximizing an information-theoretical threshold. Lei et al. (2016) focused on natural language processing (NLP) tasks and proposed to learn jointly a predictor and a selector. The latter selects a coherent subset of words from a given document that maximize the prediction objective.

Finally, other methods constrained the architecture of the predictor to be self-explainable thus losing the modularity of previous examples. The self-explainable network framework (SENN) (Alvarez Melis & Jaakkola, 2018) requires a specific structure of predictor to allow self-explainability by mimicking linear regression where the parameters of the linear regression can vary depending on the input. Moreover, SENN can be trained directly using the input features but works better for concept-based interpretation where the concept and the linear regression are learned jointly.

In this paper, we propose **LEX (Latent Variable as Explanation)** a modular self-interpretable probabilistic model class that allows for instance-wise feature selection. LEX is composed of four different modules: a predictor, a selector, an imputation scheme and some regularization. We show that up to different optimization procedures, other existing amortized explanation methods (L2X, Chen et al. 2018; Invase, Yoon et al. 2018; REAL-X Jethani et al. 2021a; and rationale selection Lei et al. 2016) optimize a single objective that can be framed as the maximization of the likelihood of a LEX model. LEX can be used either "In-Situ", where the selector and predictor are trained jointly, or "Post-Hoc", to explain an already learned predictor. Through the framework formulation, we carefully evaluate the importance of each module and identify the imputation step as a potential limit. To conduct properly this empirical assessment, we propose two new datasets to evaluate the performance of instance-wise feature selection and experimentally show that using multiple imputation leads to more plausible selections, both on our new datasets and more standard benchmarks. These new datasets could also be used to assess different intepretability methods beyond our framework.

Notations Random variables are capitalized, their realizations are not. Superscripts correspond to the index of realizations and subscripts correspond to the considered feature. For instance, x_j^i corresponds to the ith realization of the random variable X_j, which is the jth feature of the random variable X. Let $j \in [\![1, D]\!]$, x_{-j} is defined as the vector $(x_0, \dots, x_{j-1}, x_{j+1}, \dots, x_D)$, *i.e.*, the vector with i^{th} dimension removed. Let $z \in \{0,1\}^D$, then X_z is defined as the vector $(x_j)_{\{j|z_j=1\}}$, where we only select the dimensions where $z = 1$, and x_{1-z} denotes the vector $(x_j)_{\{j|z_j=0\}}$, where we only select the dimension where $z = 0$. In particular, X_z is $\|z\|_0$-dimensional and x_{1-z} is $(D - \|z\|_0)$-dimensional.

2 Casting interpretability as statistical inference

Let $\mathcal{X} = \prod_{d=1}^{D} \mathcal{X}_i$ be a D-dimensional feature space and $\mathcal{Y}$ be the target space. We consider two random variables $\mathbf{X} = (X_1, \dots, X_D)$ and $Y \in \mathcal{Y}$ following the true data generating distribution $p_{\text{data}}(x, y)$. We have access to N i.i.d. realisations of these two random variables, $x^1, \dots, x^N \in \mathcal{X}$ and labels $y^1, \dots, y^N \in \mathcal{Y}$. We want to approximate the conditional distribution of the labels $p_{\text{data}}(y|x)$ and discover which subset of features are useful for every local prediction.

2.1 Starting with a standard predictive model

To approximate this conditional distribution, a standard approach would be to consider a predictive model $\Phi(y|f_\theta(x))$, where $f_\theta : \mathbb{R}^D \to H$ is a neural network and $(\Phi(\cdot|\eta))_{\eta \in H}$ is a parametric family of densities over the target space, here parameterized by the output of f_θ. Usually, Φ is a categorical distribution for a classification task and a normal distribution for a regression task. The model being posited, various approaches exist to find the optimal parameters such as maximum likelihood or Bayesian inference. This method is just a description of the usual setting in deep learning. This is the starting point for our models, from which we will derive a latent variable as explanation model.

2.2 Latent variable as explanation (LEX)

As discussed in Section 1, the prediction model $\Phi(y|f_\theta(x))$ is not interpretable by itself in general. Our goal is to embed it within a general interpretable probabilistic model. In addition, we want this explanation to be easily understandable by a human, thus we propose to have a score per feature defining the importance of this feature for prediction. We propose to create a latent $Z \in \{0,1\}^D$ that corresponds

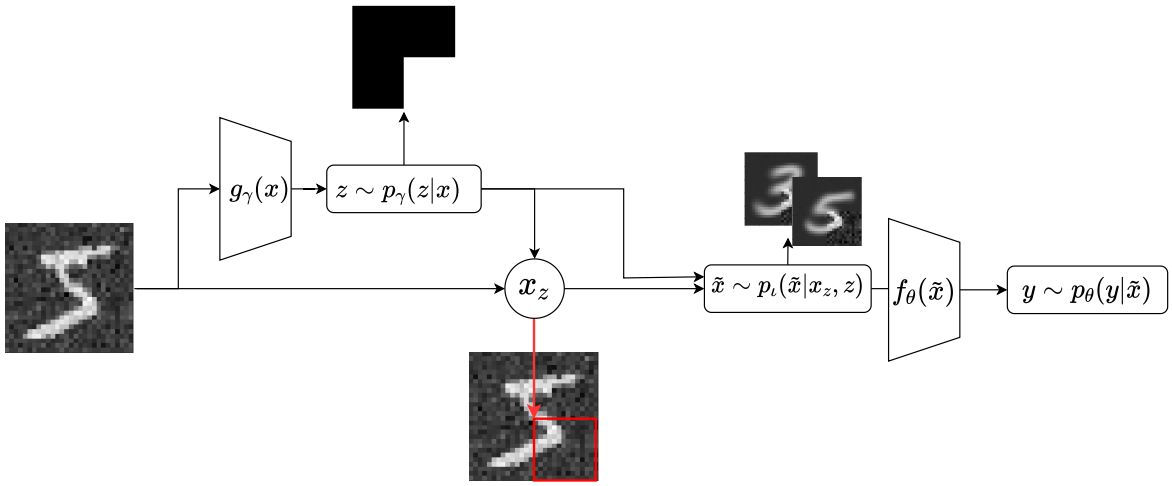

Figure 1: The LEX pipeline allows us to transform any prediction model into an explainable one. In supervised learning, a standard approach uses a function f_θ (usually a neural network) to parameterize a prediction distribution p_θ. In that framework, we would feed the input data directly to the neural network f_θ. Within the LEX pipeline, we obtain a distribution of masks p_γ parameterized by a neural network g_γ from the input data. Samples from this mask distribution are applied to the original image x to produce incomplete samples x_z. We implicitly remove features by sampling imputed samples $\tilde{x}$ given the masked image using a generative model p_ι conditioned on both the mask and the original image. These samples are then fed to a classifier f_θ to obtain a prediction. As opposed to previous methods, multiple imputation allows us to minimise the encoding happening in the mask and to get a more faithful selection.

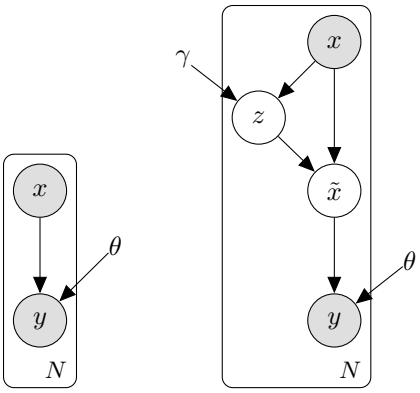

Figure 2: *Left panel:* graphical model of a standard predictive model. We propose to embed this model in a latent explainer model using the construction of the *right panel*.

to a subset of selected features. The idea is that if $Z_d = 1$, then feature d is used by the predictor, and conversely.

We endow this latent variable with a distribution $p_\gamma(z|x)$. This distribution p_γ is parametrized by a neural network $g_\gamma : \mathcal{X} \to [0,1]^D$ called the *selector* with weights $\gamma \in \Gamma$. To obtain the importance feature map of an input x, we look at its average selection $\mathbb{E}_\gamma[Z|x]$ (see Figure 8). A common parametrization is choosing the latent variable Z to be distributed as a product of independent Bernoulli variables $p_\gamma(z|x) = \prod_{d=1}^{D} \mathcal{B}(z_d|g_\gamma(x)_d)$. With that parametrization, the importance feature map of an input x is directly given by the output of the selector g_γ. For instance, Yoon et al. (2018) and Jethani et al. (2021a) use a parametrization with independent Bernoulli variables and obtain the feature importance map directly from g_γ. L2X (Chen et al., 2018) uses a neural network g_γ called the selector but since the parametrization of p_γ is not an independent Bernoulli, they obtain their importance feature map by ranking the features' importance with the weights of the output of g_γ. FastSHAP (Jethani et al., 2021b) also uses a similar network g_γ to predict Shapley values deterministically.

In the next sub-section, we will define how feature turn-off should be incorporated in the model, *i.e.*, we will define $p_\theta(y|x,z)$, the predictive distribution given that some features are ignored. With these model assumptions, the predictive distribution will be the average over likely interpretations,

$$p_{\theta,\gamma}(y|x) = \sum_{z \in \{0,1\}^D} p_\theta(y|x,z) p_\gamma(z|x)\,. \tag{1}$$

2.3 Turning off features

We want our explanation to be model-agnostic, *i.e.*, to embed any kind of predictor into a latent explanation. To that end, we want to make use of f_θ the same way we would in the setting without selection in Section 2.1, with the same input dimension. Hence we implicitly turn off features by considering an imputed vector $\tilde{X}$. Given x and z, $\tilde{X}$ is defined by the following generative process:

- We sample the turned off features according to a con-

ditional distribution called an imputing scheme $\hat{X} \sim p_\iota(\hat{X}_{1-z}|x_z)$.
- Then, we define $\tilde{X}_j = x_j$ if $z_j = 1$ and $\hat{X}_j$ if $z_j = 0$.

For instance, a simple imputing scheme is constant imputation, *i.e.*, $\tilde{X}$ is put to $c \in \mathbb{R}$ whenever $z_j = 0$. One could also use the true conditional imputation $p_\iota(\hat{X}_{1-z}|x_z) = p_{\text{data}}(X_{1-z}|x_z)$. Borrowing terminology from the missing data literature, we call single imputations the ones produced by deterministic schemes like constant imputation and multiple imputations the ones produced by truly stochastic schemes (when $\hat{X}|x_z$ does not follow a Dirac distribution). A key example of multiple imputation scheme is the true conditional $p_\iota(\hat{X}_{1-z}|x_z) = p_{\text{data}}(X_{1-z}|x_z)$. Denoting the density for the generative process above $p(\tilde{x}|x, z)$, we define the predictive distribution given that some features are ignored as

$$p_\theta(y|x, z) = \mathbb{E}_{\tilde{X}\sim p(\tilde{X}|x,z)}\Phi(y|f_\theta(\tilde{X})) \tag{2}$$

$$= \mathbb{E}_{\tilde{X}\sim p(\tilde{X}|x,z)}p_\theta(y|\tilde{X})\,. \tag{3}$$

Figure 1 shows how the model unfolds. This construction allows us to define the following factorisation of the complete model in Figure 2:

$$p_{\gamma,\theta}(y, x, \tilde{x}, z) = p_\theta(y|\tilde{x})p_\iota(\tilde{x}|z, x)p_\gamma(z|x)p(x)\,. \tag{4}$$

2.4 Statistical inference with LEX

Now that LEX is cast as a statistical model, it is natural to infer the parameters using maximum likelihood estimation. The log-likelihood function is

$$\mathcal{L}(\theta, \gamma) = \sum_{n=1}^{N} \log[\mathbb{E}_{Z\sim p_\gamma(\cdot|x^n)}\mathbb{E}_{\tilde{X}\sim p(\cdot|(x^n),Z)}p_\theta(y^n|\tilde{X})]\,. \tag{5}$$

Maximizing the previous display is quite challenging since we have to optimize the parameters of an expectation over a discrete space inside a log. We derive in Appendix E good gradient estimators for $\mathcal{L}(\theta, \gamma)$, and accordingly, the model can be optimized using stochastic gradient descent.

If $p_\gamma(z|x)$ can be any conditional distribution, then the model can just refuse to learn something explainable by setting $p_\gamma(z|x) = \mathbf{1}_{z=\mathbf{1}_d}$. All features are then always turned on. Experimentally, it appears that some implicit regularization or differences in the initialisation of the neural networks may prevent the model from selecting everything. Yet without any regularization, we leave this possibility to chance. Note that this regularization problem appears in other model explanation approaches. For instance, LIME (Ribeiro et al., 2016) fits a linear regression locally around a given instance and proposes to penalize by the number of parameters used by the linear model.

The first type of constraint we can add is an explicit function-based regularization $R : \{0, 1\}^D \to \mathbb{R}$. This function is to be strictly increasing with respect to inclusion of the mask so that the model reaches a trade-off between selection and prediction score. The regularization strength is then controlled by a positive hyperparameter λ. For instance, Yoon et al. (2018) proposed to used an L1-regularization on the average selection map $R(z) = \|z\|$. While this allows for a varying number of feature selected per instance, the optimal λ is difficult to find in practice. Another method considered by Chen et al. (2018) and Xie & Ermon (2019) is to enforce the selection within the parametrization of p_γ. Indeed, they build distributions such that any sampled subset have a fixed number of selected features.

LEX is all around LEX is a modular framework for which we can compare elements for each of the parametrizations:

- the distribution family and parametrization for the predictor p_θ;
- the distribution family and parametrization for the selector p_γ;
- the regularization function $\mathrm{R} : \{0, 1\}^D \to \mathbb{R}$ to ensure some selection happens. This regularization can be implicit in the distribution family of the selector;
- the imputation function p_ι, probabilistic or deterministic, that handles feature turn-off.

By choosing different combinations, we can obtain a model that fits our framework and express interpretability as the following maximum likelihood problem:

$$\max_{\theta,\gamma} \sum_{n=1}^{N} \Big[\log \mathbb{E}_{Z\sim p_\gamma(\cdot|x^n)}\mathbb{E}_{\tilde{X}\sim p(\cdot|x^n,Z)}p_\theta(y^n|\tilde{X}) - \lambda\mathbb{E}_{Z\sim p_\gamma(\cdot|x^n)}[\mathrm{R}(Z)]\Big]\,. \tag{6}$$

Many models, though starting from different formulations of interpretability, minimize a cost function that is a hidden maximum likelihood estimation of a parametrization that fits our framework. Indeed, L2X (Chen et al., 2018) frames their objective from a mutual information point of view, while Invase (Yoon et al., 2018) and REAL-X (Jethani et al., 2021a), whose objective is derived from Invase's, frame their objective from a Kullback-Leibler divergence between $Y|X$ and $Y|X_Z$. We can cast their optimization objective as the maximization of the log-likelihood of a LEX model. Rationale selection in Natural language processing (Lei et al., 2016) consists in learning a masking model to select a specific subset of words from text documents for downstream tasks in NLP. To enforce short and coherent rationales, they use an explicit regularization

which is a combination of L1 regularization (which favours sparsity of the mask) and continuity regularization (which favours selecting words following each other). The imputation scheme corresponds to either removing the missing words if the predictor can handle multiple-sized inputs or constant imputation where missing words are replaced with the padding value. As opposed to the multiple imputation scheme that we use (that can be generalized to any modality), the imputation scheme used in (Lei et al., 2016) is ad-hoc to text. Such an imputation would not make sense in computer vision or tabular data as the usual models in such modalities can only be fed inputs with fixed dimensions and there are no specific values for padding.

We refer to Table 1 for an overview, and to Appendix A for the full derivations of these equivalences. Such a unified framework provides a sensible unified evaluation method for different parametrizations. This allows us to compare different modules fairly. In that regard, we propose to study the imputation module we found most critical, following notably (Jethani et al., 2021a) in Section 4.

The benefits of casting explainability as inference LEX is another tool in the rich toolbox of probabilistic models. This means that, while we chose maximum-likelihood as a natural method of inference, we could train a LEX model using any standard inference technique like Bayesian inference or mixup (Zhang et al., 2018), and LEX will inherit all the compelling properties of the chosen inference technique (*e.g.* the consistency/efficiency of maximum likelihood). Beyond inference, LEX can be used in other contexts that involve a likelihood, *e.g.*, likelihood ratio tests or decision-theoretic problems. A more practical benefit is that we may use LEX as an interpretable building block within a more complex graphical model, without having to change its loss function. One may for instance use LEX to build an explainable deep Cox model (Zhong et al., 2022) trained via maximum partial likelihood.

2.5 LEX unifies Post-Hoc and In-Situ interpretations

While our model can be trained from scratch as a self-interpretable model (we call this setting In-Situ), we can also use it to explain a given black-box model in the same statistical learning framework (this setting is called Post-Hoc). In the In-Situ regime, the selection learns the minimum amount of features to get a prediction closer to the true output of the data, while in the Post-hoc regime, the selection learns the minimum amount of features to recreate the prediction of the fully observed input of a given classifier $p_m(y|x)$ to explain. The distinction between the "In-Situ" regime and the "Post-hoc" regime is mentioned, for instance, in Watson & Floridi (2021) as crucial. We distinguish four types of regimes:

Table 1: Existing models and their parametrization in the LEX framework.

Model	Sampling (p_γ) & Regularization (R)	Imputation (p_ι)	Training regime
L2X	Subset Sampling & Implicit	0 imputation	Surrogate PostHoc
Invase	Bernoulli & L1	0 imputation	Surrogate PostHoc
REAL-X	Bernoulli & L1	Surrogate 0 imputation	Fixed θ In-Situ / Surrogate PostHoc
Rationale Selection	Bernoulli, L1 & continuity regularization	Removed or imputed with padding value	Free In-Situ

Free In-Situ regime training an interpretable model from scratch using the random variable $Y \sim p_{\text{data}}$ as target.

Fix-θ In-Situ regime training only the selection part of the interpretable model using a fixed classifier but still using the random variable $Y \sim p_{\text{data}}(Y|X)$ as a target. In that setting, we do not get an explanation of how p_θ predicts its output but an explanation map for the full LEX model $p_{\theta,\gamma}$.

Self Post-Hoc regime training only the selection part of the model using a fixed classifier p_θ, but the target is given by the output of the same fixed classifier when using the full information $Y \sim \Phi(\cdot|f_\theta(x))$. This can be understood as a Fix-θ In-Situ regime where the dataset is generated from $p_{\text{data}}(x)p_\theta(y|x)$.

Surrogate Post-Hoc regime training both the selection and the classification part but the target Y is following the distribution of a given fixed classifier $p_m(y|x)$. The full model $p_{\theta,\gamma}$ is trained to mimic the behaviour of the model $p_m(y|x)$. This can be understood as a Free In-Situ regime where the dataset is generated from $p_{\text{data}}(x)p_m(y|x)$.

Depending on the situation, there is a more suited training regime. For instance, when one can only access a fixed black box model, one should train PostHoc. Depending on the imputation method, one may choose to train a surrogate to mimic the original predictor or impute directly with the evaluated black box model. If the black-box model can be changed, one may improve and explain predictions using Fix-θ In-Situ regime. Finally, training a model Free InSitu allows one to get a self-interpretable model from scratch.

3 How to turn off features?

We want to create an interpretable model where an unselected variable is not "seen" by the classifier of the model. For instance, for a given selection set $z \in \{0,1\}^D$ and a given predictor p_θ, a variable x_{1-z} is not "seen" by the predictor when averaging all the possible outputs by the

model over the unobserved input given the observed part:

$$p_{\theta,\iota}(y|x_z) = \int p_\theta(y|x_{1-z}, x_z) p_\iota(x_{1-z}|x_z) \mathrm{d}x_{1-z} \,. \quad (7)$$

Following Covert et al. (2021), we want to use a multiple imputation scheme that mimics the behaviour of the true conditional $p_\iota \approx p_{\text{data}}$. Chen et al. (2018) and Yoon et al. (2018) proposed to use 0 imputation to remove some features. Jethani et al. (2021a) showed that using such single imputation can lead to issues. Notably, training the selector and predictor jointly can lead to the selector encoding the output target for the classifier making the selection incorrect. Rong et al. (2022) also raised this issue showing class information leakage through masking in many post-hoc methods. Jethani et al. (2021a) proposed instead to separate the optimization of θ and γ. They first train $p_{\iota,\theta}(y|x, z) = p_\theta(y|x \cdot z + (1 - z)c)$ by sampling randomly z from independent Bernoulli $\mathcal{B}(0.5)$. This training step should ensure that $p_{\theta,\iota}(y|x, z)$ approximates the restricted predictor $\int p_{\text{data}}(y|x_{1-z}, x_z) p_{\text{data}}(x_{1-z}|x_z) \mathrm{d}x_{1-z}$. Note that Olsen et al. (2023) also studies the influence of imputation but only considers Shapley Values. Then, the selector part of the model p_γ is trained optimizing Equation (6) with the fixed θ.

However, training $p_{\theta,\iota}(y|x, z)$ with constant imputation to approximate the restricted predictor is difficult. Indeed, Le Morvan et al. (2021) showed that the optimal $p_{\theta,\iota}(y|x, z)$ suffers from discontinuities which makes it hard to approximate with a neural network. If $p_{\theta,\iota}(y|x, z)$ is not correctly approximating $p_{\text{data}}(y|x_z)$, there is no guarantee that the selection will be meaningful. Jethani et al. (2021a) advocate the use of a constant that is outside the support. Yet, all the experiments are made using 0 imputation which is inside the support of the input distribution. Having a constant imputation inside the domain of the input distribution may lead to further discrepancy between $p_{\theta,\iota}(y|x, z)$ and the restricted predictor. Indeed, Ipsen et al. (2022) showed that using constant imputation inside the domain to learn a discriminative model with missing data leads to some artefacts in the prediction.

On the other hand, we propose to approximate the true conditional distribution by using a multiple imputation scheme. This generative model should allow for fast sampling of the quantity $p_\iota(\tilde{x}|x, z)$. Depending on the complexity of the dataset, obtaining an imputing scheme allowing for fast and efficient masked imputation can be complicated. We show that we can come up with simpler imputing schemes that perform as well or better than the fixed constant ones. For instance, we propose to train a mixture of diagonal Gaussians to sample the missing values or to randomly sample instances from the validation dataset and replace the missing features with values from the validation sample. We describe several different methods for multiple imputations in Appendix B.

4 Experiments

There are many different sets of parameters to choose from in the LEX model. In this section, we want to study the influence of the imputation method in the LEX models as motivated in the previous section and in recent papers (Covert et al., 2021; Jethani et al., 2021a; Rong et al., 2022).

We use an implicit regularization constraining the distribution p_γ to sample a fixed proportion of the features. We call this proportion the selection rate. This choice of regularization allows comparing all sets of parameters on the same "credit" for selection. In Appendix F.2, we provide more results using L1-regularization and study different parametrizations in Appendices F.2.1 and F.2.2.

Evaluation of features' importance map is hard. These evaluation methods rely very often on synthetic datasets where a ground truth is available (Liu et al., 2021; Afchar et al., 2021) or retrain a full model using the selected variables (Hooker et al., 2019). We consider more complicated datasets than these synthetic datasets. We create three datasets using MNIST, FashionMNIST, and CelebA as starting points. These created datasets provide information only on a subset of features that should not be selected since they are not informative for the prediction task. It allows us to consider any selection outside this remaining subset as an error in selection. We call this ground truth: the maximum selection ground truth.

To compare our selections to these ground truths, we look at the true positive rate (TPR), false positive rate (FPR), and false discovery rate (FDR) of the selection map. To that end, we sample at test time a 100 mask samples from p_γ, and we calculate the three measures for each one of the 100 masks. We then show the average of the selection performance over these 100 masks. To compare the prediction performance, we consider the accuracy of the output averaged over the mask samples.

We compare different LEX parameterizations to the self-explainable neural network (SENN Alvarez Melis & Jaakkola (2018)) using all the features for regression instead of learned concepts. Though all our LEX experiments are trained In-Situ, we compare them in the two image datasets to standard Post-Hoc methods: LIME (Ribeiro et al., 2016), SHAP (Lundberg & Lee, 2017), and FASTSHAP (Jethani et al., 2021b). To do so, we train a classifier with the same architecture as before and apply the PostHoc methods on it. Details on the experiments can be found in Appendix D.

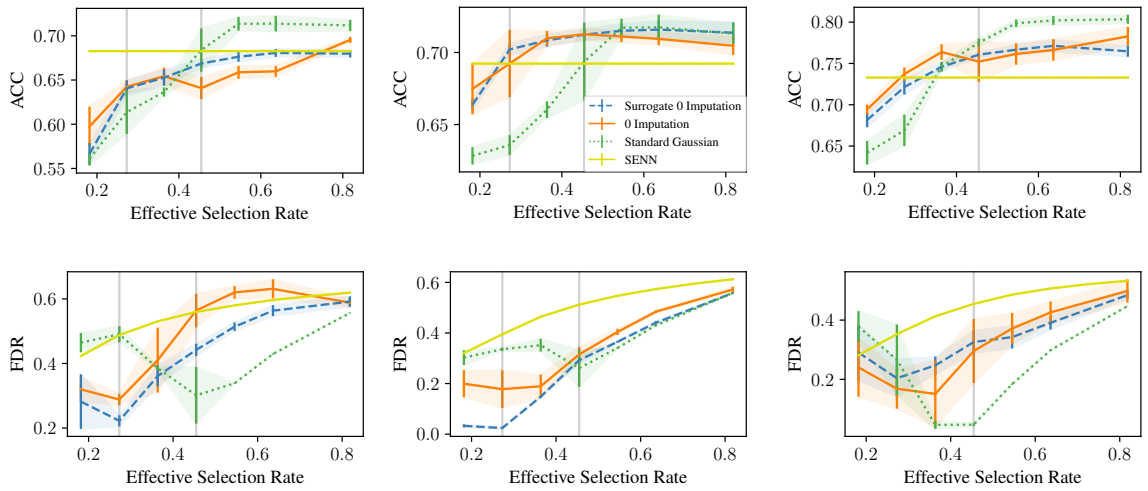

Figure 3: Performances of LEX with different imputations. 0 imputation (solid orange line) corresponds to the imputation method of Invase/L2X, Surrogate 0 imputation (blue dashed line) is the imputation method of REAL-X. The standard Gaussian is the true conditional imputation method from the model (green dotted curve). We also conducted experiments on self-explainable neural networks (SENN) in dark continuous green. The reported accuracy is obtained using all features. Columns correspond to the three synthetic datasets (S1, S2, S3) and lines correspond to the different measure of quality of the model (Accuracy, FDR, TPR). We report the mean and the standard deviation over 5 folds/generated datasets.

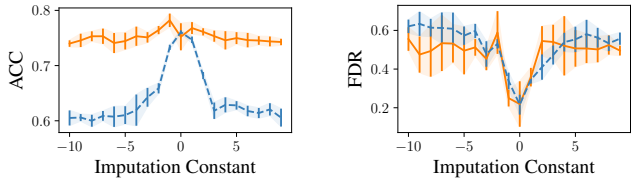

Figure 4: Performance of LEX (mean/std over 5 datasets) with varying constant imputation (orange solid line) and surrogate constant imputation (blue dashed line) on S3 using the true selection rate. Though Invase/L2X (resp. REAL-X) uses constant imputation (resp. surrogate constant), all these methods used only 0 as constant.

4.1 Artificial datasets

Datasets We study 3 synthetic datasets (S1, S2, S3) proposed by (Chen et al., 2018), where the features are generated with standard Gaussian. Depending on the sign of the control flow feature (x_{11}), the target will be generated according to a Bernoulli distribution parameterized by one among the three functions described in Appendix C. These functions use a different number of features to generate the target. Thus, depending on the sign of the control flow feature, S1 and S2 either need 3 or 5 features to fully generate the target while S3 always requires 5.

For each dataset, we generate 5 different datasets containing each 10,000 train samples and 10,000 test samples. For every dataset, we train different types of imputation with a selection rate in $[\frac{2}{11}, \frac{3}{11}, \ldots, \frac{9}{11}]$, *i.e.*, we select $2, \ldots, 9$ features. We then report the results in Figure 3 with their standard deviation across each 5 generated datasets. In the following experiments, we compare a multiple imputation based on a standard Gaussian and constant imputation with and without surrogate using a constant (as used by Yoon et al., 2018; Chen et al., 2018; Jethani et al., 2021a).

Selection evaluation The ground truth selection for S1 and S2 have two different true selection rates depending on the sign of the 11th feature (shown as two grey lines in Figure 3). In Figure 3, for S1, using multiple imputations outperforms other methods as soon as the number of selected variables approaches the larger of the two true selection rates. For S2, LEX performs better most of the time when the selection rate is higher than the larger true selection rate but has a higher variance in performance. LEX outperforms both a 0-imputation and a surrogate with 0-imputation on S3 as soon as the selection rate is close to the true selection rate (shown as the grey line) while still maintaining a better accuracy. All models outperform SENN while allowing for better accuracy.

Dependency on the constant of imputation We now focus on S3 and we consider the same experiment with 5 generated datasets but with a fixed selection rate $\frac{5}{11}$ corresponding to the true number of features, $k = 5$. We train a single model for both the constant imputation with and without a surrogate varying the constant from -10 to 9. In Figure 4, the quality of both the selection and the accuracy depends on the value of the imputation constant. Both the surrogate constant imputation and constant imputation

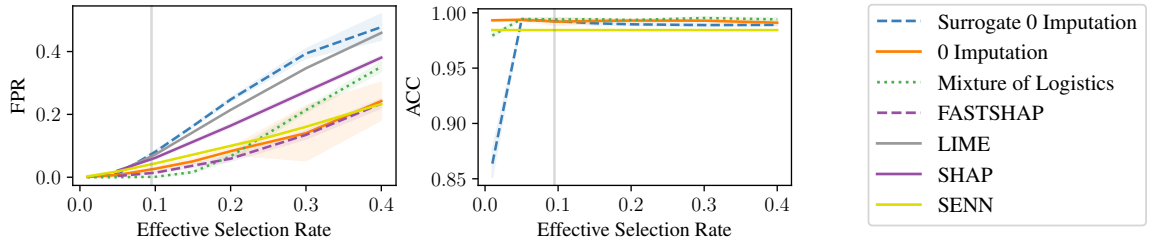

Figure 5: Performances of LEX on the Switching Panel MNIST dataset with different imputations. Surrogate 0 imputation corresponds to the parametrization of REAL-X, 0 imputation corresponds to the parametrization of Invase/L2X. We report the mean and standard deviation over 5 folds/generated datasets. We also report results on Post-Hoc methods (LIME, SHAP, FASTSHAP) and self-explainable neural networks (SENN). For the Post-Hoc methods, the predictor trained without any selection module and with full data has an accuracy of 0.97 on average over the 5 folds/generated datasets which is similar to the result obtained with the method trained In-Situ.

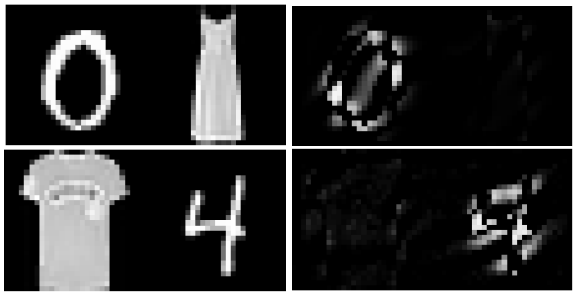

Figure 6: Samples from SP-MNIST and their LEX explanations with a mixture of logistics. The two top images correspond to a single sample of class 0 and the two bottom to a single sample of class 4.

perform better when the constant is within the domain of the input distribution. The performance drops drastically outside the range $[-1, 1]$. Jethani et al. (2021a) suggested using a constant outside the domain of imputation, which is clearly sub-optimal. Further results in Figure 11 explicit this dependency on all three synthetic datasets.

4.2 Switching Panels Datasets

We want to use more complex datasets than the synthetic ones while still keeping some ground truth explanations. To create such a dataset, we randomly sample a single image from both MNIST and FashionMNIST (Xiao et al., 2017) and arrange them in a random order to create a new image (Figure 6). The target output will be given by the label of the MNIST digit for Switching Panels MNIST (SP-MNIST) and by the label of the FashionMNISTimage for Switching Panels FashionMNIST (SP-FashionMNIST).

Given enough information from the panel from which the target is generated, the classifier should not need to see any information from the second panel. If a selection uses the panel from the dataset that is not the target dataset, it means that the predictor is leveraging information from the mask. We consider that the maximum ground truth selection is the panel corresponding to the target image.

We train every model on $45{,}000$ images randomly selected on the train dataset from MNIST (the remaining $15{,}000$ images are used in the validation dataset). For each model, we make 5 runs of the model on 5 generated datasets (*i.e.* for each generated dataset, the panels are redistributed at random) and report the evaluation of each model on the same test dataset with their standard deviation over the 5 folds. All method uses a U-NET (Ronneberger et al., 2015) for the selection and a fully convolutional neural network for the classification. Note that in practice for MNIST, only 19% of pixels are lit on average per image. The true minimum effective rate for our dataset SP-MNIST should therefore be around 10% of the pixels. We evaluate the results of our model around that reasonable estimate of the effective rate. The selected pixels around this selection rate should still be within the ground truth panel.

In Figure 5, we can see that LEX using multiple imputation with a mixture of logistics to approximate the true conditional imputation outperforms both using a constant imputation (L2X/Invase) and a surrogate constant imputation (REAL-X). The selection also outperforms the Post-Hoc methods. Indeed, around the estimated true selection rate (10%), less selection occurs in the incorrect panel while still maintaining high predictive performance (similar to the Post-Hoc methods predictive performance). Figure 6 shows that LEX uses information from the correct panel of SP-MNIST. We highlight further results and drawbacks in Appendix F.

4.3 CelebA smile

The CelebA dataset (Liu et al., 2015) is a large-scale face attribute dataset with more than $200,000$ images which provides 40 attributes and landmarks or positions of important features in the dataset. Using these landmarks, we

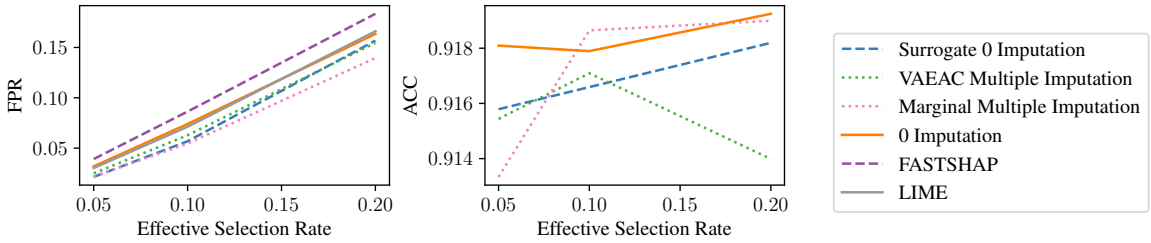

Figure 7: Performances of LEX on the CelebA Smile dataset with different methods of approximation for the true conditional distribution. For the Post-Hoc methods, the predictor trained without any selection module and with full data has an accuracy of 0.92 on average over the 5 folds/generated datasets which is similar to the result obtained with the method trained In-Situ.

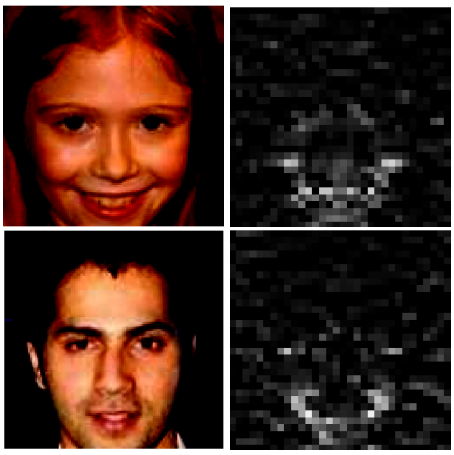

Figure 8: Samples from CelebA smile and their LEX explanations with a 10% rate. Using constant imputation leads to less visually compelling results (Appendix F). The ground truth masks used for evaluation are available in Appendix C.

are able to create a dataset with a minimal ground truth selection. The assignment associated with the CelebA smile dataset is a classification task to predict whether a person is smiling. We leverage the use of the landmark mouth associated with the two extremities of the mouth to create a ground truth selection in a box located around the mouth. We make the hypothesis that the model should look at this region to correctly classify if the face is smiling or not in the picture (see Appendix C and Figure 9 for details and examples).

In Figure 7, we evaluate two methods of multiple imputation: the VAEAC (Ivanov et al., 2019), which is a generative model allowing for the generation of imputed samples given any conditional mask z, and a marginal multiple imputation. Both our multiple imputation methods perform similarly compared to the constant imputation method both in selection and accuracy. In Figure 8, we see that LEX uses the information around the mouth to predict whether the face is smiling or not which is a sensible selection. See Appendix F for further results and samples.

5 Conclusion

We proposed a framework, LEX, casting interpretability as a maximum likelihood problem. We have shown that LEX encompasses several existing models. We provided 2 datasets on complex data with a ground truth to evaluate the feature importance map. Using these, we compared many imputation methods to remove features and showed experimentally the advantages of multiple imputation compared to other constant imputation methods. These new datasets can be used for other explanation methods than the amortized ones we focused on here.

The framing of interpretability as a statistical learning problem allowed us to use maximum likelihood to find optimal parameters. One could explore other methods for optimizing the parameters, such as Bayesian inference. Interpretation maps are more easily readable when they provide smooth segmentations for images. Another avenue for future work could be to study the use of different parametrizations or regularization that favors connected masks (*i.e.* neighbouring pixels have more incentives to share the same selection) to allow for smoother interpretability maps.

Acknowledgements

HHJS and JF were funded by the Independent Research Fund Denmark (9131-00082B). Furthermore, JF was in part funded by the Novo Nordisk Foundation (NNF20OC0062606 and NNF20OC0065611) and the Innovation Fund Denmark (0175-00014B). This work has also been supported by the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002. DG acknowledges the support of ANR through project NIM-ML (ANR-21-CE23-0005-01) and EU Horizon 2020 project AI4Media (contract no. 951911).

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. Sanity checks for saliency maps. *Advances in Neural Information Processing Systems*, 31, 2018.

Afchar, D., Guigue, V., and Hennequin, R. Towards rigorous interpretations: a formalisation of feature attribution. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 76–86, 2021.

Alvarez Melis, D. and Jaakkola, T. Towards robust interpretability with self-explaining neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2016.

Chen, J., Song, L., Wainwright, M., and Jordan, M. Learning to explain: An information-theoretic perspective on model interpretation. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 883–892, 2018.

Choi, Y., Vergari, A., and Van den Broeck, G. Probabilistic circuits: A unifying framework for tractable probabilistic models. *Unpublished manuscript, URL:* `http://starai.cs.ucla.edu/papers/ProbCirc20.pdf`, 2020.

Covert, I. C., Lundberg, S., and Lee, S.-I. Explaining by removing: A unified framework for model explanation. *The Journal of Machine Learning Research*, 22(1): 9477–9566, 2021.

Crabbé, J. and van der Schaar, M. Label-free explainability for unsupervised models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 4391–4420, 2022.

Domke, J. and Sheldon, D. R. Importance weighting and variational inference. *Advances in Neural Information Processing Systems*, 31, 2018.

Hooker, S., Erhan, D., Kindermans, P.-J., and Kim, B. A benchmark for interpretability methods in deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Ipsen, N., Mattei, P.-A., and Frellsen, J. How to deal with missing data in supervised deep learning? In *International Conference on Learning Representations*, 2022.

Ivanov, O., Figurnov, M., and Vetrov, D. Variational autoencoder with arbitrary conditioning. In *International Conference on Learning Representations*, 2019.

Jethani, N., Sudarshan, M., Aphinyanaphongs, Y., and Ranganath, R. Have we learned to explain?: How interpretability methods can learn to encode predictions in their interpretations. In *International Conference on Artificial Intelligence and Statistics*, pp. 1459–1467, 2021a.

Jethani, N., Sudarshan, M., Covert, I. C., Lee, S.-I., and Ranganath, R. FastSHAP: Real-time Shapley value estimation. In *International Conference on Learning Representations*, 2021b.

Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., and Kim, B. The (un) reliability of saliency methods. *Explainable AI: Interpreting, explaining and visualizing deep learning*, pp. 267–280, 2019.

Le Morvan, M., Josse, J., Scornet, E., and Varoquaux, G. What's a good imputation to predict with missing values? *Advances in Neural Information Processing Systems*, 34, 2021.

Lei, T., Barzilay, R., and Jaakkola, T. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 107–117. Association for Computational Linguistics, 2016.

Lemhadri, I., Ruan, F., Abraham, L., and Tibshirani, R. Lassonet: A neural network with feature sparsity. *The Journal of Machine Learning Research*, 22(1):5633–5661, 2021.

Linardatos, P., Papastefanopoulos, V., and Kotsiantis, S. Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2021.

Liu, Y., Khandagale, S., White, C., and Neiswanger, W. Synthetic benchmarks for scientific research in explainable machine learning. *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 2017.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. Monte Carlo gradient estimation in machine learning. *The Journal of Machine Learning Research*, 21(132):1–62, 2020.

Moshkovitz, M., Dasgupta, S., Rashtchian, C., and Frost, N. Explainable k-means and k-medians clustering. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 7055–7065, 2020.

Olsen, L. H. B., Glad, I. K., Jullum, M., and Aas, K. A comparative study of methods for estimating conditional Shapley values and when to use them. *arXiv preprint arXiv:2305.09536*, 2023.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

Ribeiro, M. T., Singh, S., and Guestrin, C. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Rong, Y., Leemann, T., Borisov, V., Kasneci, G., and Kasneci, E. A consistent and efficient evaluation strategy for attribution methods. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 18770–18795, 2022.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. PixelCNN++: Improving the pixelCNN with discretized logistic mixture likelihood and other modifications. In *International Conference on Learning Representations*, 2017.

Seo, J., Choe, J., Koo, J., Jeon, S., Kim, B., and Jeon, T. Noise-adding methods of saliency map as series of higher order partial derivative. *arXiv preprint arXiv:1806.03000*, 2018.

Shapley, L. S. A value for n-person games. *Contributions to the Theory of Games, number 28 in Annals of Mathematics Studies, pages 307–317*, II, 1953.

Slack, D., Hilgard, S., Jia, E., Singh, S., and Lakkaraju, H. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pp. 180–186. Association for Computing Machinery, 2020.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 12, 1999.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Tucker, G., Lawson, D., Gu, S., and Maddison, C. J. Doubly reparameterized gradient estimators for Monte Carlo objectives. In *International Conference on Learning Representations*, 2019.

Wang, E., Khosravi, P., and Broeck, G. V. d. Probabilistic sufficient explanations. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pp. 3082–3088, 2021.

Watson, D. S. and Floridi, L. The explanation game: a formal framework for interpretable machine learning. In *Ethics, Governance, and Policies in Artificial Intelligence*, pp. 185–219. Springer, 2021.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xie, S. M. and Ermon, S. Reparameterizable subset sampling via continuous relaxations. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pp. 3919–3925. AAAI Press, 2019.

Yoon, J., Jordon, J., and van der Schaar, M. INVASE: Instance-wise variable selection using neural networks. In *International Conference on Learning Representations*, 2018.

Zarlenga, M. E., Barbiero, P., Ciravegna, G., Marra, G., Giannini, F., Diligenti, M., Shams, Z., Precioso, F., Melacci, S., Weller, A., Lio, P., and Jamnik, M. Concept embedding models: Beyond the accuracy-explainability

trade-off. *Advances in Neural Information Processing Systems*, 35, 2022.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Zhong, Q., Mueller, J., and Wang, J.-L. Deep learning for the partially linear cox model. *The Annals of Statistics*, 50(3):1348–1375, 2022.

A Other methods falling into the LEX framework

In this section, we show how to cast three existing models in our statistical learning setting. Though the original papers framed their wanted interpretabiliy through different points of view, we will show that what they actually optimize can be seen as *likelihood lower bounds of specific LEX models*.

A.1 Learning to explain (L2X, Chen et al., 2018)

Let us consider a fixed classifier p_m. For a fixed number of features k, we want to train a selector p_γ (this corresponds to $V(\theta, \cdot)$ in the original paper) that will allow selection for this exact number of features (using subset sampling) only and a predictor p_θ (this corresponds to g_α in the original paper) such that the full model $p_{\theta,\gamma}$ will mimic the behaviour of p_m. (Chen et al., 2018) uses 0-imputation to parameterize the imputation distribution. We can rewrite Eq. (6) in (Chen et al., 2018) using our notation as

$$\begin{aligned}
&\mathbb{E}_{X\sim p_{\text{data}}, Z\sim p_\gamma(\cdot|X)}\left[\sum_y p_m(y|X)\log p_\theta(y|X\cdot Z)\right] \\
&= \mathbb{E}_{X\sim p_{\text{data}}, Y\sim p_m, Z\sim p_\gamma(\cdot|X), \tilde{X}\sim p_\iota(\cdot|X,Z)}[\log p_\theta(Y|\tilde{X})] \\
&\leq \mathbb{E}_{X\sim p_{\text{data}}, Y\sim p_m}[\log \mathbb{E}_{Z\sim p_\gamma(\cdot|X), \tilde{X}\sim p_\iota(\cdot|X,Z)} \log p_\theta(Y|\tilde{X})]\,,
\end{aligned} \tag{8}$$

where we used Jensen inequality to obtain this last inequality. Thus the L2X objective is a lower bound of the likelihood in the post-hoc setting as the target distribution comes from the fixed classifier p_m we want to explain. The objective described is shed under the light of the Surrogate Post-Hoc regime. By considering p_m as the true data distribution p_{data}, this can be extended to the Free In-Situ regime.

A.2 Invase (Yoon et al., 2018)

We consider a selector p_γ (this corresponds to $\pi_\theta(x, s)$ in the original paper) and a classifier p_θ (this corresponds f^ϕ in the original paper) using a 0-imputation to parameterise the imputation module. We consider a regularization function controlled by parameter $\lambda \in \mathcal{R}$ such that for any $z \in [0,1]^D$, $R(z) = \|z\|_0$. They also train a baseline classifier, since this classifier is simply used to reduce variance in the estimated gradient and does not change the objective for the optimization in θ and γ. In this paper, they consider only 0 imputation so for any given $(x, z) \in \mathcal{X} \times [0,1]^D$, $\tilde{x} = x \cdot z$.

We can rewrite Eq. (5) in (Yoon et al., 2018) using our notation and not considering the baseline as :

$$\hat{l}(z, x, y) = -\sum_{i=1}^{c} y_i \log p_\theta(y_i|x, z)\,. \tag{9}$$

The loss for the selector network ($l_2(\theta)$ in the paper) is :

$$\begin{aligned}
&\int_{x,y} p_{\text{data}}(x,y) \sum_{z\in\{0,1\}^D} [p_\gamma(z|x)\hat{l}(z,x,y) + \lambda\|z\|_0]\mathrm{d}x\mathrm{d}y \\
&= \mathbb{E}_{X,Y\sim p_{\text{data}}}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[\hat{l}(z,x,y) + \lambda\|z\|_0] \\
&= \mathbb{E}_{X,Y\sim p_{\text{data}}}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[-\log p_\theta(Y|X,Z) + \lambda\|z\|_0] \\
&= \mathbb{E}_{X,Y\sim p_{\text{data}}}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[-\log p_\theta(Y|X\cdot Z) + \lambda\|z\|_0] \\
&= \mathbb{E}_{X,Y\sim p_{\text{data}}}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[-\log \mathbb{E}_{\tilde{X}\sim p_\iota(\cdot|x,z)} p_\theta(Y|\tilde{X}) + \lambda\|z\|_0] \\
&= \mathbb{E}_{X,Y\sim p_{\text{data}}}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[-\log \mathbb{E}_{\tilde{X}\sim p_\iota(\cdot|x,z)} p_\theta(Y|\tilde{X})] + \lambda\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[\|z\|_0] \\
&= \mathbb{E}_{X,Y\sim p_{\text{data}}}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[-\log \mathbb{E}_{\tilde{X}\sim p_\iota(\cdot|x,z)} p_\theta(Y|\tilde{X})] + \lambda\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[R(Z)] \\
&\geq \mathbb{E}_{X,Y\sim p_{\text{data}}}\left[[-\log \mathbb{E}_{Z\sim p_\gamma(\cdot|X)}\mathbb{E}_{\tilde{X}\sim p_\iota(\cdot|x,z)} p_\theta(Y|\tilde{X})] + \lambda\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[R(Z)]\right]\,.
\end{aligned}$$

Though the loss for the optimisation of the selector with γ and the loss for the optimisation in θ are separated in the original article, they only differ by the regularization $\mathbb{E}_{Z\sim p_\gamma}[\lambda\|z\|_0]$ which is independent of θ. Thus, minimizing both θ and γ using the loss for the selector network is equivalent to the complete minimization set up in (Yoon et al., 2018).

The last equation is exactly the negative log-likelihood of the LEX model with a parametrization described above. Hence the minimisation target of INVASE is a minimisation of an upper bound of the negative log-likelihood.

A.3 REAL-X (Jethani et al., 2021a)

Let us consider a fixed classifier p_m. Similarly to INVASE, we consider a selector p_γ (this corresponds to q_{sel}) and a classifier p_θ (which correspond to q_{pred}) and a regularizer R controlled by $\lambda \in \mathcal{R}$.

We can rewrite Eq. (3) from (Jethani et al., 2021a) in our framework as

$$\begin{aligned}&\mathbb{E}_{X\sim p_{\text{data}}}\mathbb{E}_{Y\sim p_m}\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[\log p_\theta(Y|X\cdot Z)] - \lambda\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[R(Z)]\\ &\leq \mathbb{E}_{X\sim p_{\text{data}}}\mathbb{E}_{Y\sim p_m}[\log \mathbb{E}_{Z\sim p_\gamma(\cdot|X)}p_\theta(Y|X\cdot Z)] - \lambda\mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[R(Z)]\,.\end{aligned}$$

The last equation is a lower bound on the log-likelihood of a LEX model parametrized as described above.

As opposed to the previous method, the predictor is trained on a different loss than the selector:

$$\mathbb{E}_{X\sim p_{\text{data}}}\mathbb{E}_{Y\sim p_m}\mathbb{E}_{Z\sim \mathcal{B}(0.5)}[\log p_\theta(Y|X\cdot Z)]\,,$$

where $\mathcal{B}(0.5)$ is a distribution of independent Bernoulli for the mask.

This loss is completely independent of the selector's parameters. Thus, training first the predictor network p_θ until convergence and then training the selector p_γ is equivalent to the alternating training in Algorithm 1 in (Jethani et al., 2021a).

We can consider that the associated LEX model is always trained with a fixed θ and REAL-X is maximizing the lower bound of the log-likelihood of a LEX parametrization in a fixed θ setting.

A.4 Rationale selection (Lei et al., 2016)

We consider a free classifier p_θ (which corresponds to the encoder) and a selector p_γ (which corresponds to the generator) and two regularization functions such that for any $z \in [0,1]^D$ $R_1(z) = ||z||_1$ controlling the sparsity and $R_2(z) = \sum_{d=1}^{D}|z_t - z_{t-1}|$ enforcing continuity of the selection. Their relative strengths are regulated by two parameters λ_1 and λ_2. Given a data point $(x,y) \in \mathcal{X}\times\mathcal{Y}$ and $z \in [0,1]^D$ sampled from $p_\gamma(x,y)$, the classifier is fed $\tilde{x} = z\cdot x + (1-z)\cdot pad$ where pad is the padding value of the embedding space. If the classifier p_θ can handle multiple dimensions input, then $\tilde{x} = x_z$.

The objective from (Lei et al., 2016) can be expressed with our notation :

$$\mathbb{E}_{(X,Y)\sim p_{data}, Z\sim p_\gamma(\cdot|x)}[\log p_\theta(Y|\tilde{X}) + \lambda_1 R_1(Z) + \lambda_2 R_2(Z)]. \tag{10}$$

Using Jensen equation, we get :

$$\begin{aligned}&\mathbb{E}_{(X,Y)\sim p_{data}, Z\sim p_\gamma(\cdot|X)}[\log p_\theta(Y|\tilde{X}) + \lambda_1 R_1(Z) + \lambda_2 R_2(Z)]\\ &\leq \mathbb{E}_{(X,Y)\sim p_{data}}\left[\log \mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[p_\theta(Y|\tilde{X})] + \mathbb{E}_{Z\sim p_\gamma(\cdot|X)}[\lambda_1 R_1(Z) + \lambda_2 R_2(Z)]\right].\end{aligned} \tag{11}$$

Thus, the rationale selection objective is a lower bound of the log-likelihood of a LEX parameterization in a free In-Situ setting with two regularization function.

B Multiple imputation scheme

B.1 VAEAC

VAEAC (Ivanov et al., 2019) is a arbitrary conditioned variational autoencoder that allows us to sample from an approximation of the true conditional distribution. To impute a single example, we first sample a latent h according to p_{prior} using a prior network, then sample the unobserved features p_{gen} using the generative network

$$p_\iota(\tilde{x}|x,z) = \int_h p_{prior}(h|x_z,z)p_{gen}(\tilde{x}_{1-z}|x_z,z,h)\mathbf{1}_{\tilde{x}_z = x_z}\,. \tag{12}$$

We only use the VAEAC for the CelebA Smile experiment and leverage the architecture and weights provided in the paper.

B.2 Gaussian Mixture Model (GMM)

The Gaussian mixture model allows for fast imputation of masked samples for low to medium size dataset when using spherical Gaussians. Before training, we fit the mixture model to the fully observed dataset by maximizing the log likelihood of the model Equation (13) using the expectation maximization algorithm. In practice, we use the Gaussian Mixture Model library from (Pedregosa et al., 2011) to obtain the parameters $(\pi^k, \mu^k, \Sigma^k)_K$. In the case of spherical Gaussians, Σ_k is a diagonal matrix. thus $\forall k, \mu_k, \Sigma_k$ are the size of the input dimension D.

$$\mathcal{L}_{\text{GMM}}(x) = \sum_k \pi^k \mathcal{N}(x|\mu^k, \Sigma^k)\,. \tag{13}$$

To obtain a sample imputing the missing variable, we start by calculating $p(k|x, z)$. In the particular case of spherical Gaussian, this allows us to consider only the unmasked features when calculating this quantity.

$$p(k|x, z) = \frac{\mathcal{N}(x_z|(\mu^k)_z, (\Sigma^k)_z)}{\sum_{k'} \mathcal{N}(x_z|(\mu^{k'})_z, (\Sigma^{k'})_z)}\,. \tag{14}$$

Finally, we sample a center k from the previous conditional distribution and a sample from the associated Gaussian:

$$p_\iota(\tilde{x}|x, z) = \sum_k p(k|x, z)\mathcal{N}(\tilde{X}|\mu^k, \Sigma^k)\,. \tag{15}$$

In practice, to train the Gaussian mixture model on discrete image, we add some uniform noise to the input data to help the learning of the input data.

B.3 Means of Gaussian mixture model (Means of GMM)

We propose an extension of the previous GMM model to get more in distribution samples from the dataset. After sampling a center from the conditional distribution Equation (14), instead of resampling the imputed values from the Gaussian distribution, we can use directly the means of the centers of the sampled center as imputed data.

$$p_\iota(\tilde{x}|x, z) = \sum_k p(k|x, z)\mathbf{1}_{\tilde{x}=\mu^k}\,. \tag{16}$$

B.4 Dataset Gaussian Mixture Model (GMM Dataset)

In a second extension of the previous GMM imputation we make use of the validation dataset to create imputations. To that end, we store the quantity $p_{\text{resampling}}(x^i_{val}|k)$ for every center k and every data point x^i_{val} in the validation set, where

$$p_{\text{resampling}}(x^i_{val}|k) = \frac{\mathcal{N}(x^i_{val}|\mu_k, \Sigma_k)}{\sum_j \mathcal{N}(x^j_{val}|\mu_k, \Sigma_k)}\,. \tag{17}$$

After sampling a center from the conditional distribution, we sample one example according to the distribution in Equation (17). The imputation distribution is :

$$p_\iota(\tilde{x}|x, z) = \sum_k p(k|x, z) \sum_j p_{\text{resampling}}(x^j_{\text{val}}|k)\mathbf{1}_{\tilde{x}=x^j_{\text{val}}}\,. \tag{18}$$

B.5 KMeans and Validation Dataset (KMeans Dataset)

By using the KMeans instead of the GMM for the GMM Dataset, we can obtain a simpler multiple imputation methods. To that end, we calculate the minimum distance between a masked input x_z and the masked centers of the clusters μ_k. We select cluster $k*$ closest to the input data and we can sample uniformly any image from the validation dataset belonging to this cluster.

B.6 Mixture of logistics

We propose to use a mixture of discretized logistic as an approximation for the true conditional distribution. For each center k of the mixture among K centers, we consider a set of D center μ_d^k and scale parameters s_d^k which allows the creation of a discretized logistic distribution for each pixel similar to the construction in (Salimans et al., 2017). We obtain the parameters by maximizing the log-likelihood of the model Equation (19):

$$p(x) = \sum_k \pi_k \sum_{d\in[0,D]} \text{logistic}(x_d|\mu_d^k, s_d^k)\,, \tag{19}$$

where $\text{logistic}(x_d|\mu_d^k, s_d^k) = \sigma((x_d + 0.5 - \mu_d^k)/s_d^k)) - \sigma((x_d - 0.5 - \mu_d^k)/s_d^k))]$ and σ is the logistic sigmoid function. In the edge cases of a pixel value equal to 0, we replace $x - 0.5$ by $-\infty$ and for 255 we replace $x + 0.5$ by $+\infty$.

We initialise the model means and weights by using the K-Means algorithm from Sklearn (Pedregosa et al., 2011). We then learn the model either by stochastic gradient ascent on the likelihood or by stochastic EM (both methods leads to similar choice of parameters).

Similarly to the GMM, we sample an imputation by first sampling a center from the mixture using the distribution $p(k|x, z)$, where

$$p(k|x, z) = \frac{\sum_{d\in[0,D]} \text{logistic}(x|\mu_d^k, s_d^k)}{\sum_{k'} \pi_{k'} \sum_{d\in[0,D]} \text{logistic}(x|\mu_d^{k'}, s_d^{k'})}\,. \tag{20}$$

We can then sample the imputed data using the parameters obtained from the subset restricted sample, that is :

$$p_\iota(\tilde{x}|x, z) = \sum_k p(k|x, z) \sum_{d|z_d=1} \text{logistic}(x_d|\mu_d^k, s_d^k)\,. \tag{21}$$

B.7 Means of Mixture of logistics

Instead of sampling from the logistic distribution, after sampling a center k from the conditional distribution Equation (14), we can use directly the means of the sampled centers as imputed data, that is,

$$p_\iota(\tilde{x}|x, z) = \sum_k p(k|x, z) \sum_d \mathbf{1}_{\tilde{x}_d=\mu_d^k}\,. \tag{22}$$

B.8 Validation Dataset (Marginal)

We can sample randomly from the validation dataset to replace the missing value from the unobserved dataset. This corresponds to approximating the conditional true imputation $p_{data}(x_{1-z}|x_z)$ by the unconditional marginal distribution $p_{data}(x_{1-z})$. The imputation distribution is the following :

$$p_\iota(\tilde{x}|x, z) = p_{\text{data}}(\tilde{x}_{1-z})\,. \tag{23}$$

C Dataset details

C.1 Synthetic Dataset generation

The input features for the synthetic datasets follow the generation procedure :

$$\{x_i\}_{i=1}^{11} \sim \mathcal{N}(0, 1) \quad y \sim \mathcal{B}\left(\frac{1}{1 + f(x)}\right)\,. \tag{24}$$

We consider different functions f for different situations:

- $f_A(x) = \exp(x_1 x_2)$,
- $f_B(x) = \exp(\sum_{i=3}^{6} x_i^2 - 4)$,

- $f_C(x) = \exp(-10\sin(0.2x_7) + |x_8| + x_9 + e^{x_{-10}} - 2.4)$.

This leads to the following datasets:

- **S1**:

$$f(x) = \begin{cases} f_A(x) & \text{if } x_{11} < 0 \\ f_B(x) & \text{if } x_{11} \geq 0. \end{cases} \tag{25}$$

- **S2**:

$$f(x) = \begin{cases} f_A(x) & \text{if } x_{11} < 0 \\ f_C(x) & \text{if } x_{11} \geq 0. \end{cases} \tag{26}$$

- **S3**:

$$f(x) = \begin{cases} f_B(x) & \text{if } x_{11} < 0 \\ f_C(x) & \text{if } x_{11} \geq 0. \end{cases} \tag{27}$$

C.2 Panel MNIST and FashionMNIST

Both dataset MMNIST and FashionMNIST are transformed in the same fashion as (Jethani et al., 2021a) by dividing the input features by 255. Note that this transformation will affect the choice of the optimal constant of imputation for LEX. We create the train set and validation set of the panel dataset by using only images from the train datasets of MNIST and FashionMNIST and the test set by using only images from the test datasets. The split between train and validation is split randomly with proportion $80\%, 20\%$. Hence, the train dataset of the switching panels input contain 48,000 images, the validation dataset contains 12,000 images and the test dataset 10,000 images.

C.3 CelebA Smile

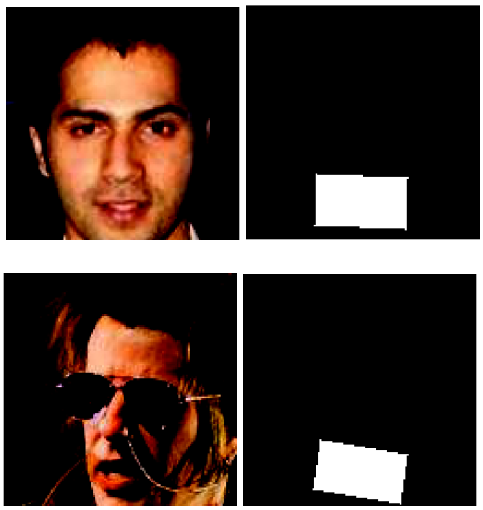

Figure 9: Two examples from the CelebA smile dataset associated with their ground truth selection.

The CelebA dataset consists of $162,770$ train, 19867 validation and 19962 test color images of faces of celebrities of size 178×218. Before training any model, we crop the image to keep a 128×128 pixels size square in the center of the image (using Torchvision's *CenterCrop* function and we normalize the channels in datasets. Note that we use this center crop to be able to use directly the weights and parametrization provided by the author of (Ivanov et al., 2019) with CelebA for the VAEAC. We define the target of this dataset using the attribute given for smiling in the dataset.

We can create the maximum ground truth boxes by using the landmarks position of the mouth. Given the position of both extremities of the mouth, we can obtain both the direction and the lenght of the mouth. Supposing not only the mouth but also the region around is useful for the classifier, we create the maximum ground truth boxes using a rectangle oriented following the direction of the mouth, centered at the center of both mouth extremities and with height corresponding to two times the lenght of the mouth and width corresponding to the length of the mouth.

D Experiment Details

For every experiment, we use an importance weighted lower bound with 10 importance samples for the mask and a single important sample for the imputed values. We estimate the monte carlo gradient using REBAR with the weighted reservoir sampling and the relaxed subset sampling distribution as control variate. We evaluate the selection by averaging the FPR, TPR and FDR of a 100 mask samples. The accuracy is calculated using a 100 importance mask sample.

LIME (Ribeiro et al., 2016) is a PostHoc method fitting a linear model in the neighborhood of the target image we want to explain. This does not allow us to compare directly with the LEX method as the coefficients obtained correspond to the

weights attributed to each superpixel in the linear regression. In order to compare to LEX's map of importance feature, we create a map associating to each feature the absolute value of the corresponding superpixel weight, this value corresponds to the importance of each pixel. Then we select the features with the highest coefficient until the given selection rate is reached. If some features have the same importance, we select a random subset to fill in the selection rate. We then used this selection map to compute the FPR, TPR and FDR. We use the implementation associated with the paper associated and the quickshift algorithm from Pedregosa et al. (2011) for the segmentation algorithm.

Self-explainable neural networks (SENN) (Alvarez Melis & Jaakkola, 2018) extend regular linear regression $\theta^T x$ (where $\theta \in \mathcal{R}^d$) by learning a function giving parameter per instance $\theta(x)^T x$. The magnitude of the parameters $\theta(x)(y)$ for a label $y \in \mathcal{Y}$ allows us to quantify the importance of each feature for this specific instance. To enforce regularization and prevent overfitting, a sparsity regularizer (i.e. L1-loss) and a robustness regularizer (i.e. weight decay) are used during training. In order to compare to LEX, we trained SENN with the this implementation using a large grid of parameters for both regularizers and chose the hyperparameters maximizing the validation accuracy as proposed in Alvarez Melis & Jaakkola (2018). We calculate the FPR, FDR and TPR of the selection by choosing, for each rate, the features with maximum magnitude $|\theta(x)_j|$. As opposed to the LEX framework, the number of selected features is not part of the model but only defined for the evaluation.

SHAP (Lundberg & Lee, 2017) and FASTSHAP (Jethani et al., 2021b) are PostHoc methods that allow to calculate the Shapley-Values (Shapley, 1953) of a target image. Similarly to LIME, we create the importance feature map by selecting the features with the highest absolute Shapley-Values until we reach the selection rate. We use the function DeepExplainer from the implementation of SHAP associated with the original paper. We use the implementation of FASTSHAP associated with the paper.

Synthetic Dataset The predictor p_θ is parameterized with a fully connected neural network f_θ with 3 hidden layers while the selector p_γ is parameterized with a fully connected neural network g_γ with 2 hidden layers. The hidden layers are dimension 200 and use ReLU activations. The predictor has a softmax activation for classification while the selector uses a sigmoid activation. We trained all the methods for 1000 epochs using Adam for optimisation with a learning rate 10^{-4} and weight decay 10^{-3} with a batch size of 1000. Both selector and predictor are free during training for experiments with constant imputation and multiple imputation. When using a surrogate imputation, the surrogate is trained at the same time as the selector according to the algorithm in (Jethani et al., 2021a).

We evaluate SENN (Alvarez Melis & Jaakkola, 2018) using a neural network with 2 hidden layers of dimension 200 and ReLU activations. We use a large grid for the sparsity regularizer ($[2 \times 10^{-5}, 2 \times 10^{-4}, 2 \times 10^{-3}]$) and the robustness regularizer ($[0, 1 \times 10^{-3}, 1 \times 10^{-2}, 1 \times 10^{-1}, 1]$) and reported the results with maximum validation accuracy.

Switching Panels The predictor is composed by 2 sequential convolution block that outputs respectively 32, 64 filters. Each block is composed with 2 convolutional layers and an average pooling layer. We fed the output of the last convolutional block to a fully connected layer with a softmax activation. The selector is a U-Net (Ronneberger et al., 2015) with 3 down sampling and up sampling blocks and a sigmoid activation. The U-Net outputs a 28x56x1 image mask corresponding to the parameters for each pixel in the image. We trained all the methods for 100 epochs using Adam for optimisation with a learning rate 10^{-4} and weight decay 10^{-3} with a batch size of 64. Both selector and predictor are free during training for experiment with constant imputation and multiple imputation. When using a surrogate imputation, the surrogate is trained at the same time as the selector according to the algorithm in Jethani et al. (2021a).

We evaluate SENN (Alvarez Melis & Jaakkola, 2018) using a U-Net with 3 down sampling and up sampling blocks and a sigmoid activation. The U-Net outputs a 28x56x1 image mask corresponding to the parameters for each pixel in the image. We use the same large grid for the sparsity regularizer ($[2 \times 10^{-5}, 2 \times 10^{-4}, 2 \times 10^{-3}]$) and the robustness regularizer ($[0, 1 \times 10^{-3}, 1 \times 10^{-2}, 1 \times 10^{-1}, 1]$) and reported the results with maximum validation accuracy.

For LIME (Ribeiro et al., 2016), we train 5 different networks f_θ on the 5 generated datasets with the same architecture described above for 1000 epochs with a learning rate 10^-4 and a batch size of 64. We parameterized Quickshift with a ratio of 0.1, a kernel size of 1.0 and a max dist of 10. For SHAP, we use the same networks as LIME and use as background the validation dataset. This parametrization allow an average of 25 superpixels on the train dataset.

The surrogate and selection networks of FASTSHAP have the same architecture as the network trained for LEX. We train each network separately for 1000 epochs with a batch size of 64.

CelebA the predictor neural network f_θ is composed by 4 sequential convolution block that outputs respectively 32, 64, 128 and 256 filters. Each block is composed with 2 convolutional layers and an average pooling layer. We fed the output of the last convolutional block to a fully connected layer with a softmax activation. The selector is a U-Net (Ronneberger et al., 2015) with 5 down sampling and up sampling blocks and a softmax activation. We lower the number of possible mask by creating a 32×32 grid of 4×4 pixels over the whole image. The U-Net outputs a $32 \times 32 \times 1$ image where each feature corresponds to the parametrization of a 4×4 pixel square in the original image.

For experiments with a constant imputation or a multiple imputation, we train the predictor for 10 epochs. We then train the selector using the pre-trained fixed classifier for 10 epochs. For experiments with a surrogate constant imputation, we train the surrogate for 10 epochs using an independent Bernoulli distribution to mask every pixel (this corresponds to the objective of EVAL-X in Eq (5) in (Jethani et al., 2021a)). We then train the selector using the pretrained surrogate for 10 epochs. We optimize every neural netowrk using Adam with a learning rate 10^{-4} and weight decay 10^{-3} with a batch size of 32

For LIME (Ribeiro et al., 2016), we train f_θ with the same architecture described above for 10 epochs with a learning rate 10^-4 and a batch size of 32. We parameterized Quickshift with a ratio of 0.5, a kernel size of 2.0 and a max dist of 100. For SHAP, we use the same networks as LIME and use as background the validation dataset.

The surrogate and selection network of FASTSHAP have the same architecture as the network trained for LEX. We train each network separately for 1000 epochs with a batch size of 32 and a learning rate 10^{-4}.

E A difficult optimization problem

E.1 An importance weighted lower bound

To maximise Equation (6), one difficulty resides in the two expectations inside the log. Using Jensen inequality, we can get a lower bound on this log-likelihood.

$$\begin{aligned}
\mathcal{L}(\theta, \gamma) &= \sum_{n=1}^{N} \log[\mathbb{E}_{Z \sim p_\gamma(.|x^n)} \mathbb{E}_{\tilde{X} \sim p_\iota(.|x^n, Z)} p_\theta(y^n|\tilde{X})] \\
&\geq \sum_{n=1}^{N} \mathbb{E}_{Z \sim p_\gamma(.|x^n)} \log[\mathbb{E}_{\tilde{X} \sim p_\iota(.|x^n, Z)} p_\theta(y^n|\tilde{X})] \\
&\geq \sum_{n=1}^{N} \mathbb{E}_{Z \sim p_\gamma(.|x^n)} \mathbb{E}_{\tilde{X} \sim p_\iota(.|x^n, Z)} [\log p_\theta(y^n|\tilde{X})] \,.
\end{aligned}$$

This bound may be too loose and give a poor estimate of the likelihood of the model. Instead, we propose to use importance weighted variational inference (IWAE) (Burda et al., 2016) so we can have a tighter lower bound than with Jensen inequality. Note that we have to apply two times the IWAE, for the expectation on masks and on the imputed features. For each data point x_n, we sample L mask importance samples and LK imputations importance samples.

The IWAE lower of the log-likelihood on the mask with L mask samples

$$\mathcal{L}(\theta, \gamma) \geq \mathcal{L}_L(\theta, \gamma) \,, \tag{28}$$

where

$$\mathcal{L}_L(\theta, \gamma) = \sum_{n=1}^{N} \mathbb{E}_{Z^{n,1}, \ldots, Z^{n,L} \sim p_\gamma(.|x^n)} \mathbb{E}_{\tilde{X}^{n,1}, \ldots, \tilde{X}^{n,L}, \sim p_\iota(.|Z^{n,l}, x^n)} [\log \frac{1}{L} \sum_{l=1}^{L} p_\theta(y^n|\tilde{X}^{n,l})] \,.$$

For a fixed L mask imputations, the IWAE lower bound with K imputation samples is

$$\mathcal{L}_L(\theta, \gamma) \geq \mathcal{L}_{L,K}(\theta, \gamma) \,, \tag{29}$$

where

$$\mathcal{L}_{L,K}(\theta, \gamma) = \sum_{n=1}^{N} \mathbb{E}_{Z^{n,1}, \ldots, Z^{n,L} \sim p_\gamma(\cdot|x^n)} \mathbb{E}_{\tilde{X}^{n,1,1}, \ldots, \tilde{X}^{n,L,K} \sim p_\iota(\cdot|Z^{n,l}, x^n)} \left[\log \frac{1}{L} \frac{1}{K} \sum_{l=1}^{L} \sum_{k=1}^{K} p_\theta(y^n|\tilde{X}^{n,l,k}) \right] \,.$$

Theorem 3 of (Domke & Sheldon, 2018) ensures that when $L \to +\infty$,

$$\mathcal{L}_L(\theta, \gamma) = \mathcal{L}(\theta, \gamma) + O\left(\frac{1}{L}\right) ,$$

and, for a fixed L,

$$\mathcal{L}_{L,K}(\theta, \gamma) = \mathcal{L}_L(\theta, \gamma) + O\left(\frac{1}{K}\right) .$$

Thus, using a large number of importance samples for both the mask and the imputation ensures that we are maximizing a tight lower bound of the log-likelihood of the model.

Note that other models falling into the LEX framework are optimizing a lower bound of the log-likelihood (see Appendix A for details) with the Jensen inequality. However, casting their method into the statistical learning framework motivates to choose a tighter lower bound of this log-likelihood which improves results in classification and selection.

E.2 Gradient Monte Carlo Estimator

We want to train the maximum likelihood model with stochastic gradient descent. Using Monte Carlo gradient estimator for θ is straightforward. Finding Monte Carlo Gradient estimator for γ is more complicated because the expectation on masks depends on γ and we sample from a discrete space $\{0, 1\}^D$. A simple way of getting an estimator for this gradient is by using a policy gradient estimator (Sutton et al., 1999) or Score Function Gradient estimator (Mohamed et al., 2020).

On the other hand, by relaxing the discreteness of the distribution, it is possible to reparametrize the expectation in γ and use a pathwise monte carlo gradient estimator(Mohamed et al., 2020). These estimators introduce some bias but lower the variance of the estimation. For instance, (Yoon et al., 2018) proposed to use the concrete distribution (Maddison et al., 2017) which is a continuous relaxation of the discrete Bernoulli distribution. Similarly, (Xie & Ermon, 2019), (Chen et al., 2018) used some forms of continuous relaxations of subset sampling distribution.

RealX (Jethani et al., 2021a) proposed to use REBAR (Tucker et al., 2019) to further reduce the variance of these gradient estimators while still keeping an unbiased estimator thereof by using the relaxation of the discrete distribution as a control variate for a score function gradient estimator.

When using multiple imputation, there is no possibility to use a continuous mask as the imputation distribution. To that end, we leverage the allowed reparametrization of the continuous relaxation of the discrete distribution but still apply a straight-through estimator (Bengio et al., 2013). When using independent Bernoulli for p_γ, we consider a thresholded straight through estimator with threshold $t = 0.5$. For relaxed subset sampling, we use a top K function for the straight-through estimator with k being the number of features to be selected. Using this new straight-through estimator for the different continuously relaxed distribution, we can either use a pathwise Monte Carlo gradient estimator or a variation of REBAR where the relaxation is modified by the straight-through function.

F Extended results

F.1 Extended results with a fixed sampling rate

Synthethic dataset In Figure 10, we compare different parameterizations of LEX (0 imputation, surrogate 0 imputation and multiple imputations with standard Gaussian) and SENN using all available metrics (FDR, FPR and TPR). Provided the chosen selection rate is higher than the actual selection rate of the dataset, multiple imputation outperforms any other parametrization or SENN on all metrics.

Furthermore, SENN suffers from the same issue as constant imputation raised in (Jethani et al., 2021a). If there is a Control flow feature (x_{11} in the tabular datasets we use) there is no guarantee that it will be selected. Indeed, the "linear regression parameters" $\theta(x)$ will handle the control flow but will not show its importance. On the other hand, multiple imputation with standard gaussian selects the control flow as soon as the selection rate reaches the true value.

Finally, Figure 11 extends Figure 4 and shows that the performances of the surrogate constant imputation depend on the choice of the constant. While (Jethani et al., 2021a) recommends in the second footnote to choose a constant outside the input domain, we obtained the best results using 0 imputation which is exactly the mean imputation of all the synthetic datasets.

SP-MNIST In Figure 12, we observe results for different constants using a surrogate imputation on SP-MNIST. We see that changing the constant of imputation may drastically change the performance of the selection. As opposed to synthetic dataset, the best results are obtained with a 3 imputation. In that experiment, 0 performs poorly even though 0 is a decent estimation of the mean imputation for MNIST.

On Figure 21, we can observe the average of a 100 samples from a model trained with different constant imputation and a surrogate function and an average rate of selection of 5%. We can see that the constant imputation drastically changes the shape of the imputation even though we are using a surrogate function. When using 0, the selection model seems to try to recreate the full image instead of selecting the correct panel. Using constants 1 and 3 seems to imitate the negative of the shape of the two on the correct panel. Finally, when using constant -1, the selection recreates the target number in both panels to facilitate classification. These samples using a surrogate function can be considered cheating as the selection is used to improve the classification results. As opposed, samples using multiple imputation on Figure 22 are less dependent on the type of multiple imputation used.

SP-FashionMNIST Following the same procedure as switching panels MNIST, we conduct experiments on the dataset switching panels FashionMNIST. Since on average 50% of the pixels are lit in FashionMNIST, we expect the true selection rate to be around 25 % of the total number of pixels in Switching Panels FashionMNIST.

In Figure 14, we compare LEX with two methods of multiple imputation, the Mixture of Logistics and GMM Dataset. These two multiple imputations outperforms their constant imputation counterparts with and without surrogate near the expected true rate of selection. Using the mixture of logistics allows to maintain a strong accuracy, similar to the accuracy of both constant imputation methods. In Figure 15, we can observe that the selections obtained with the surrogate constant imputation is more robust to the change in constant imputation compared to the switching panels MNIST dataset but the variations are still higher that the variations obtain with many different multiple imputation in Figure 16.

CelebA Smile On Figure 17, as opposed to the results on SP-MNIST and the synthetic datasets, we see that the performance of LEX using a surrogate constant imputation does not depend on the choice of a constant.

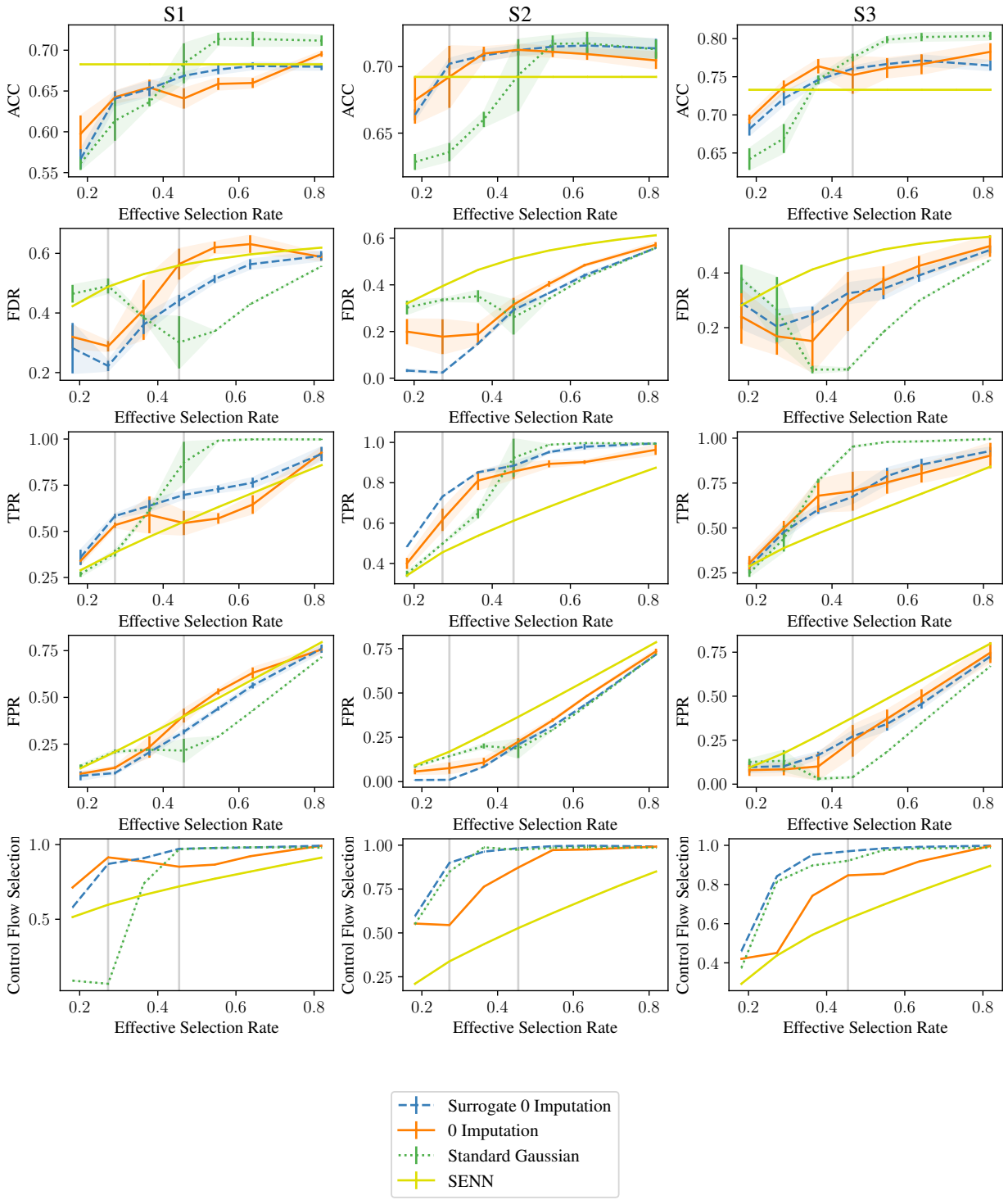

Figure 10: Performances of LEX with different imputations. 0 imputation (solid orange line) corresponds to the imputation method of Invase/L2X, Surrogate 0 imputation (blue dashed line) is the imputation method of REAL-X. The standard Gaussian is the true conditional imputation method from the model (green dotted curve). We also conducted experiments on self-explainable neural networks (SENN) in dark continuous green. The columns correspond to the three synthetic datasets (S1, S2, S3) and the lines correspond to the different measure of quality of the model (Accuracy, FDR, TPR and control flow selection). We report the mean and the standard deviation over 5 folds/generated datasets.

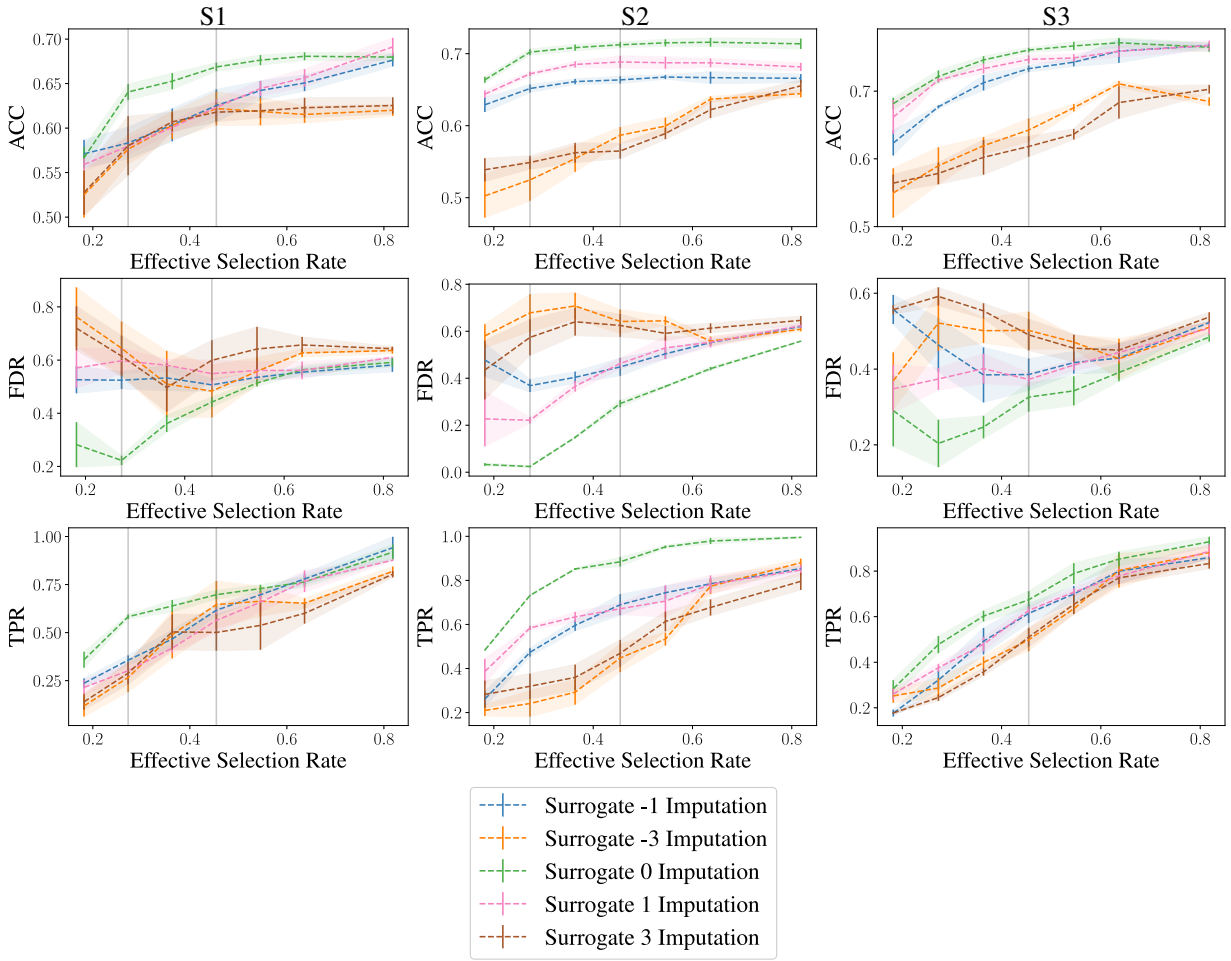

Figure 11: Performances of LEX with different imputation constants using a surrogate constant imputation on the synthetic datasets. This corresponds to the REAL-X parametrization with different constant imputation. Columns corresponds to the three synthetic datasets (S1, S2, S3) and lines corresponds to the different measure of quality of the model (Accuracy, FDR, TPR). [mean $\pm$ std over 5 folds/generated dataset]

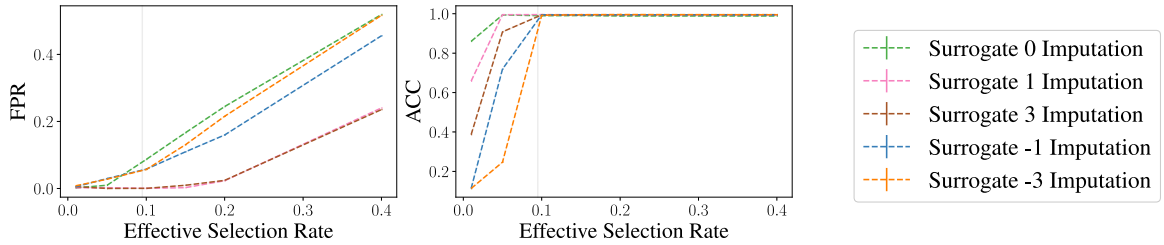

Figure 12: Performance of LEX using a surrogate constant imputation for different imputation constants on the SP-MNIST dataset.

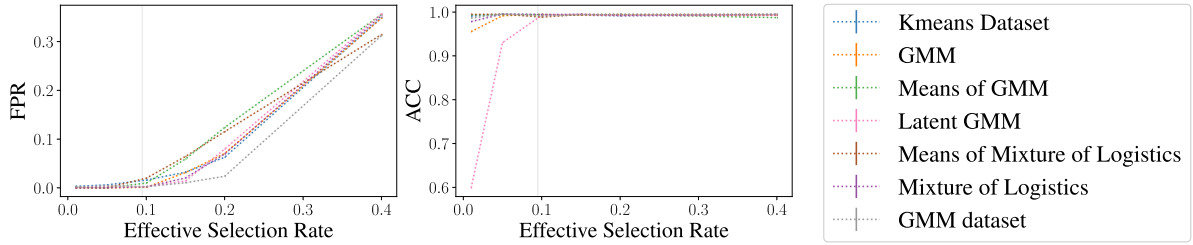

Figure 13: Performances of the LEX method with different methods of multiple imputation on the SP-MNIST dataset.

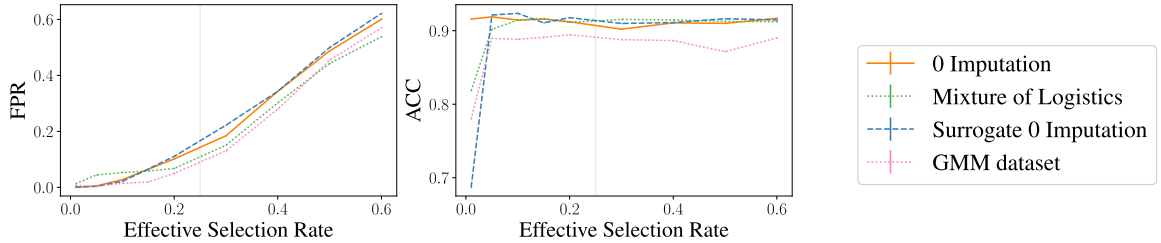

Figure 14: Performances of LEX on the SP-Fashion dataset with different methods of approximation for the true conditional imputation.

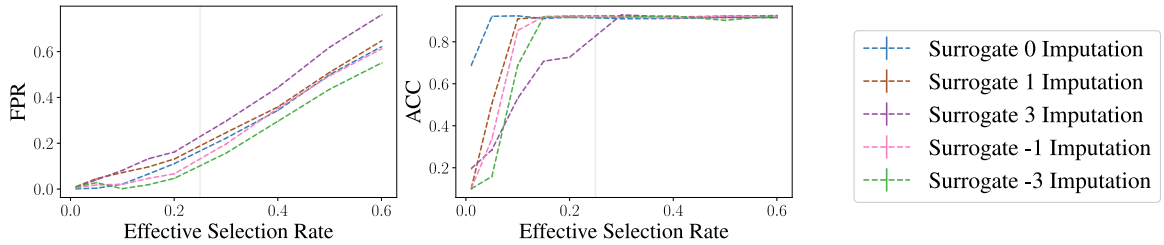

Figure 15: Performance of LEX using a surrogate constant imputation for different imputation constants on the SP-MNIST dataset.

F.2 Extended results using L1-regularization

To provide a better analysis of the influence of the imputation method on the performance of LEX, we fixed all the other parameters hence slightly changing the original methods (Invase and REAL-X). We differed from the original implementations in two ways, the choice of the regularization method and sampling distribution p_γ as well as the choice of the Monte-Carlo gradient estimator. Finding an optimal λ with L1-regularization is difficult as different ranges of λ lead to

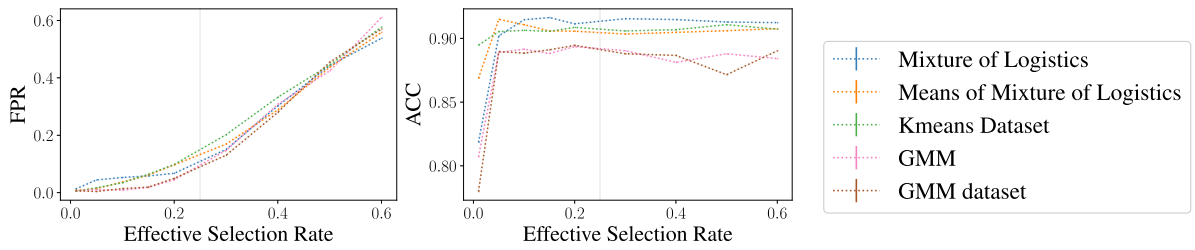

Figure 16: Performances of the LEX method with different methods of multiple imputation on the SP-FashionMNIST dataset.

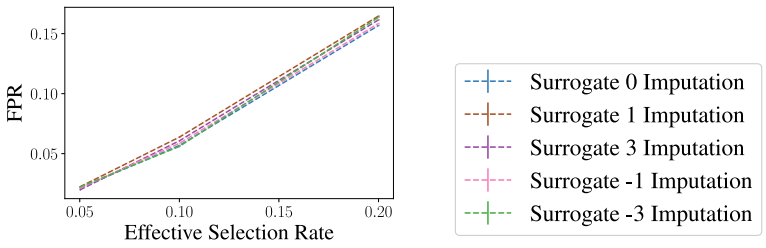

Figure 17: Performances on the CelebA smile dataset with different constant for the surrogate constant imputation.

different performances and different rates of selection depending on the datasets, the imputations (see Appendix F.2.1). While we can estimate an adequate selection rate with intuition from the dataset, no such intuition is available for λ which requires a long exploration. In that section, we will study how these choices might affect the performances of LEX and compare to the original implementations using the SP-MNIST dataset.

F.2.1 On the difficulty of tuning λ

We proposed in Section 4 to study LEX with a fixed selection rate because finding an optimal λ requires an extensive search and makes it difficult to compare the results between sets of parameters. In this section, we study how different λ lead to very different rates of selection depending on the sets of parameters considered.

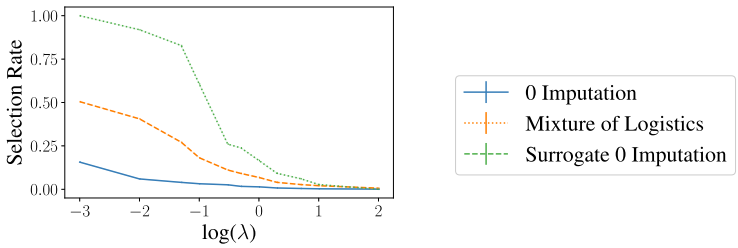

Figure 18: Effective selection rate depending on the value of λ of the same LEX model for different types of imputation trained on the SP-MNIST dataset. Depending on the choice of imputation, different ranges of λ induce different ranges of selection rate which thwart the comparison between different parameterizations.

We fix the regularization to L1-Regularization and the distribution p_γ to be an independent Bernoulli (this is the same setting for distribution and regularization as Invase and REAL-X) and fix the Monte-Carlo gradient estimator to REBAR. We study the evolution of the rate of selection with parameter λ varying in $[0.01, 0.05, 0.1, 0.3, 0.5, 1.0, 2.0, 5.0, 10.0, 100.0]$. We observe in Figure 18 that depending on the imputation, the evolution of the rate of selection is very different. Since we want to compare different methods on the same "credit" of selection (ie the same average rate of selection), we have to search on very large range of λ in practice.

F.2.2 Influence of the Monte-Carlo gradient estimation

In their original implementation, L2X, Invase and REAL-X use different Monte-Carlo gradient estimators for the optimization. L2X uses Gumbel-Softmax (Maddison et al., 2017) a continuous relaxation of the discrete Bernoulli distribution which is a biased but low variance estimator. Invase uses the REINFORCE (Sutton et al., 1999) estimator, an unbiased but high variance estimator. They proposed to control the variance using a baseline network, trained without selection, as a control variate. REAL-X uses the REBAR (Tucker et al., 2019) estimator, a REINFORCE estimator using Gumbel-Softmax as a control variate. We chose to focus on REBAR as it can be considered an improvement over REINFORCE and Gumbel-Softmax.

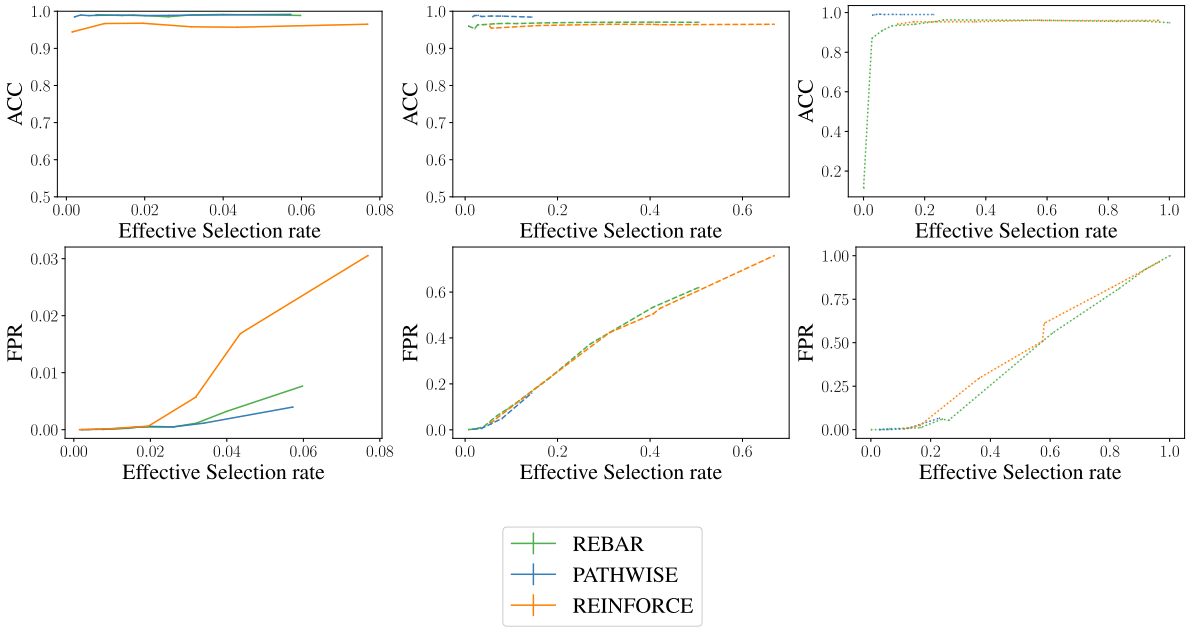

Figure 19: Comparison of the performance of LEX models with L1-regularization with 3 different Monte-Carlo gradient estimator on the SP-MNIST dataset.. λ varies in $[0.1, 0.3, 0.5, 1.0, 3.0, 5.0, 10.0,]$. First column corresponds to 0-imputation mimicking the behaviour of INVASE for different MC gradient estimator, second column corresponds to surrogate 0-imputation mimicking the behaviour of REAL-X and third column corresponds to multiple imputation with a Mixture of Logistics.

In this section, we fix the regularization to L1-Regularization and the distribution p_γ to be an independent Bernoulli and we compare the difference in performance depending on the Monte-Carlo gradient estimator (REINFORCE, Gumbel-Softmax, REBAR) for different methods of imputation. Note that plain REINFORCE slightly differs from Invase (because of the baseline control variate) but they admitted in the reviews for their paper (Yoon et al., 2018) that using the control variate did not improve the results. In Figure 19, we observe that changing the Monte-Carlo gradient estimator leads to similar results in prediction and selection. However, changing the Monte-Carlo gradient estimator changes how the choice of λ impact the selection rate.

F.2.3 Comparisons with the set-ups of L2X, Invase and REAL-X

Here we propose a comparison of the original set-ups of L2X, Invase and REAL-X to two parameterisations of LEX models with multiple imputation. The first one is the same set-up as in 4, using REBAR and an implicit regularization with a fixed rate of selection. For the second set-up, we train using REBAR but consider an Independent Bernoulli for the selection distribution regularized by an explicit L1-Regularisation (note that this is the same regularization and distribution of Invase and REAL-X). λ varies in $[0.1, 0.3, 0.5, 1.0, 2.0, 5.0, 10.0]$ for the method with L1-Regularization.

On Figure 20, we see that our method (i.e. using multiple imputation with a Mixture of Logistics) provides the best performances in the vicinity of the expected selection rate. On the other hand, even with λ very close to 0, Invase still

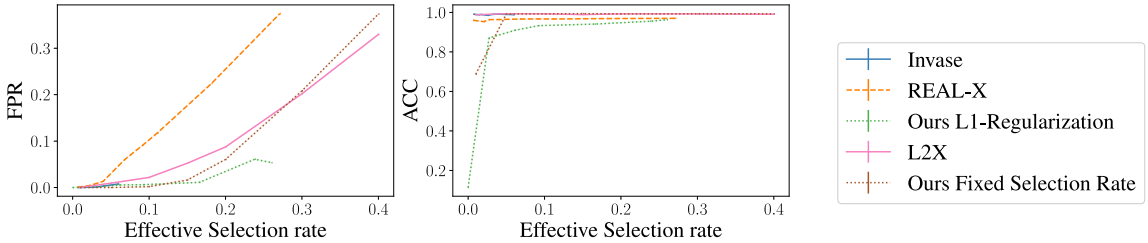

Figure 20: Performances of LEX trained on the SP-MNIST dataset using the same sets of parameters as the original implementation of L2X, Invase, and REAL-X. We compare them to two sets of parameters with Mixture Of Logistics imputation (denoted as Ours) using two types of regularization: L1-Regularization and implicit regularization with a fixed selection rate. The models with L1-regularization were trained on the same grid $\lambda \in [0.1, 0.3, 0.5, 1.0, 2.0, 5.0, 10.0,]$.

selects a very small subset of the features making it difficult to compare to the other methods. Moreover, the accuracy of Invase, REAL-X, and L2X do not decrease with the effective selection rate which suggests that these methods encode the target output in the mask selection.

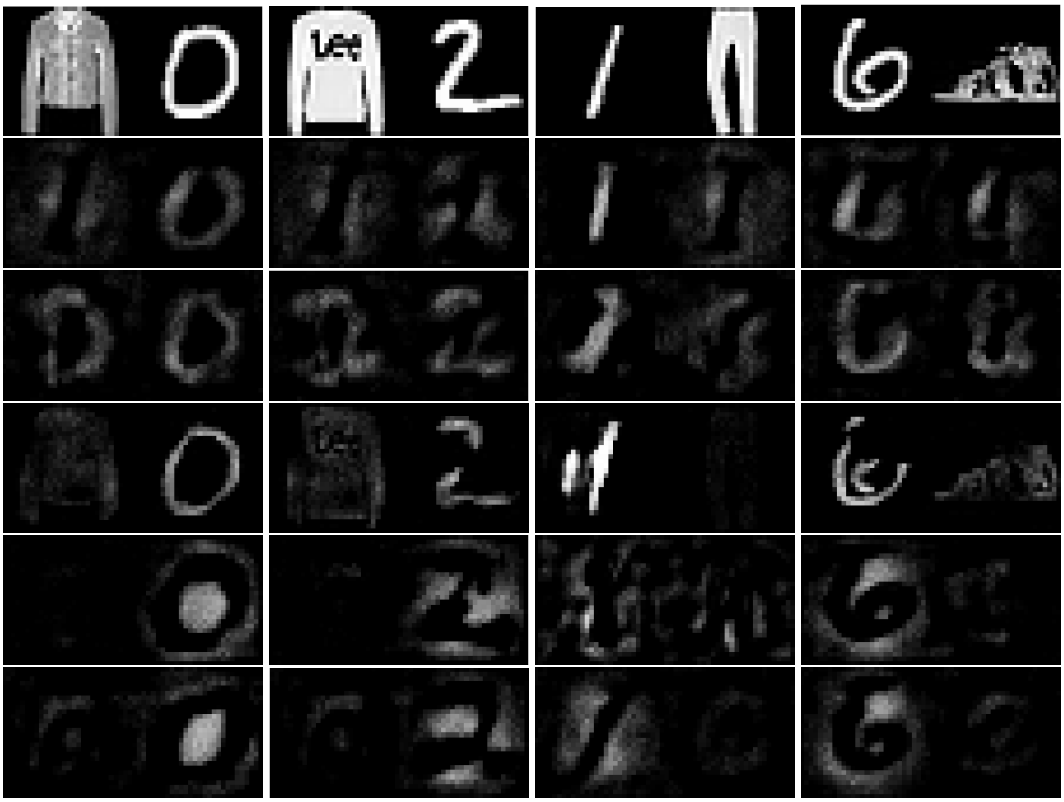

Figure 21: Each figure corresponds to the average of 100 mask samples from the selector trained using a surrogate constant of imputation for different constant. From top to bottom, we have the input data, and constants $-3, -1, 0, 1, 3$. The selector was parametrized by a subset sampling with a rate of selection of 5%. Not only the selection happen in the wrong panel for most of the constant, the selected features change drastically with the choice of the constant.

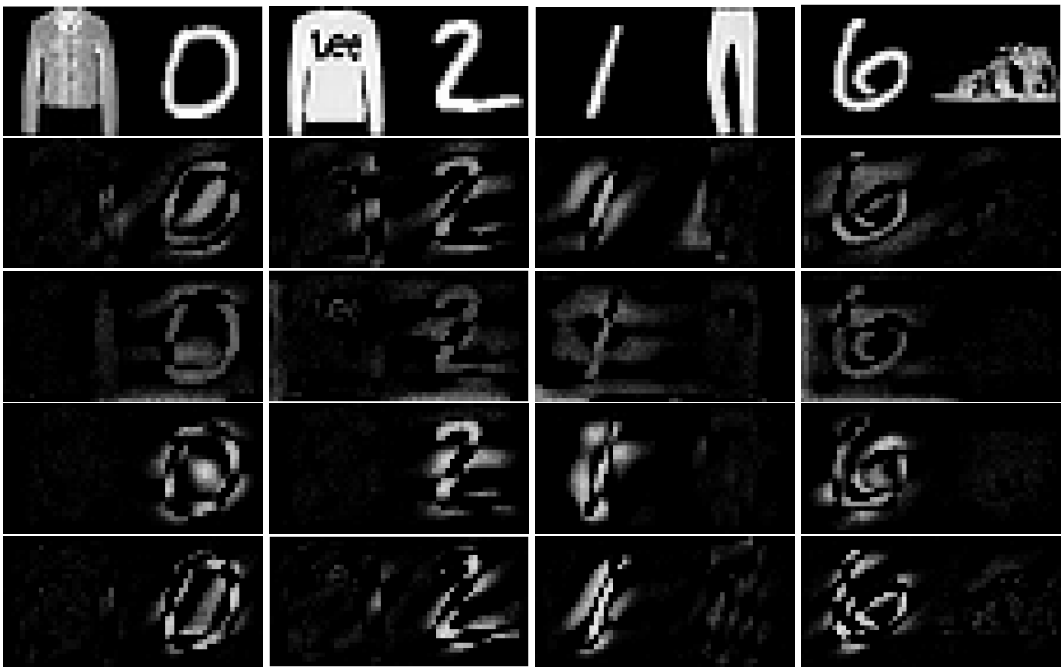

Figure 22: Each figure corresponds to the average of 100 mask samples from the selector trained using different multiple imputation. From top to bottom, the different multiple imputation are : KMeans Dataset, Means of GMM, Mixture of Logitics, means of mixture of logistics. The selector was parametrized by a subset sampling with a rate of selection of 5%.

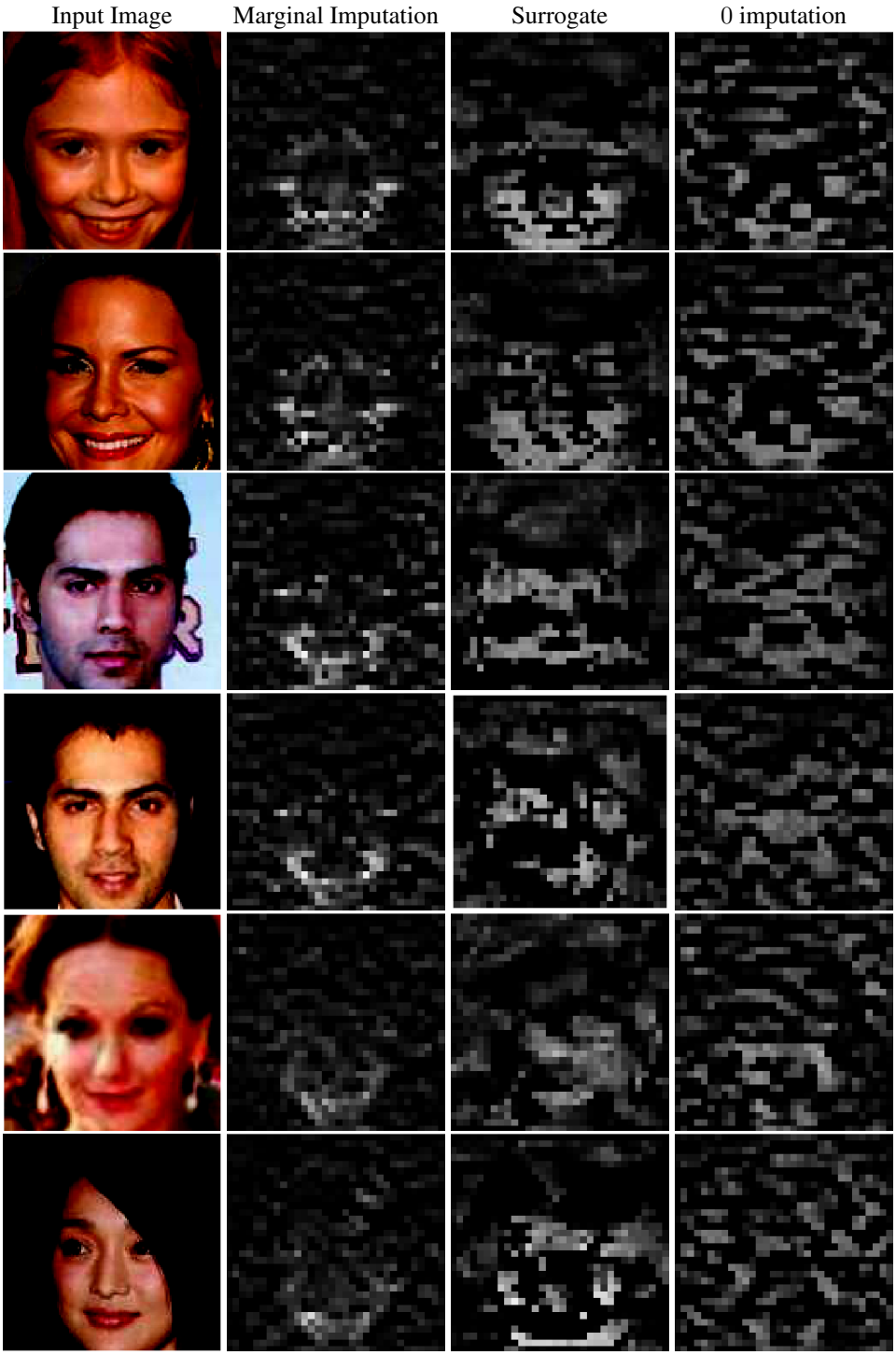

Figure 23: Selection obtained averaging over 100 samples from the distribution p_γ parametrized by a subset sampling distribution selecting 10% of the pixels. Marginal imputation consists in replacing the missing pixels with samples from the validation dataset.