Sobolev Independence Criterion

Youssef Mroueh, Tom Sercu, Mattia Rigotti, Inkit Padhi, Cicero Dos Santos *
IBM Research & MIT-IBM Watson AI lab
mroueh,mrigotti@us.ibm.com,inkit.padhi@ibm.com

Abstract

We propose the Sobolev Independence Criterion (SIC), an interpretable dependency measure between a high dimensional random variable X and a response variable Y. SIC decomposes to the sum of feature importance scores and hence can be used for nonlinear feature selection. SIC can be seen as a gradient regularized Integral Probability Metric (IPM) between the joint distribution of the two random variables and the product of their marginals. We use sparsity inducing gradient penalties to promote input sparsity of the critic of the IPM. In the kernel version we show that SIC can be cast as a convex optimization problem by introducing auxiliary variables that play an important role in feature selection as they are normalized feature importance scores. We then present a neural version of SIC where the critic is parameterized as a homogeneous neural network, improving its representation power as well as its interpretability. We conduct experiments validating SIC for feature selection in synthetic and real-world experiments. We show that SIC enables reliable and interpretable discoveries, when used in conjunction with the holdout randomization test and knockoffs to control the False Discovery Rate. Code is available at http://github.com/ibm/sic.

1 Introduction

Feature Selection is an important problem in statistics and machine learning for interpretable predictive modeling and scientific discoveries. Our goal in this paper is to design a dependency measure that is interpretable and can be reliably used to control the False Discovery Rate in feature selection. The mutual information between two random variables X and Y is the most commonly used dependency measure. The mutual information $I(X;Y)$ is defined as the Kullback-Leibler divergence between the joint distribution p_{xy} of X, Y and the product of their marginals $p_x p_y$, $I(X;Y) = \mathrm{KL}(p_{xy}, p_x p_y)$. Mutual information is however challenging to estimate from samples, which motivated the introduction of dependency measures based on other f-divergences or Integral Probability Metrics [1] than the KL divergence. For instance, the Hilbert-Schmidt Independence Criterion (HSIC) [2] uses the Maximum Mean Discrepancy (MMD) [3] to assess the dependency between two variables, i.e. $\mathrm{HSIC}(X,Y) = \mathrm{MMD}(p_{xy}, p_x p_y)$, which can be easily estimated from samples via Kernel mean embeddings in a Reproducing Kernel Hilbert Space (RKHS) [4]. In this paper we introduce the Sobolev Independence Criterion (SIC), a form of gradient regularized Integral Probability Metric (IPM) [5, 6, 7] between the joint distribution and the product of marginals. SIC relies on the statistics of the gradient of a witness function, or critic, for both (1) defining the IPM constraint and (2) finding the features that discriminate between the joint and the marginals. Intuitively, the magnitude of the average gradient with respect to a feature gives an importance score for each feature. Hence, promoting its sparsity is a natural constraint for feature selection.

The paper is organized as follows: we show in Section 2 how sparsity-inducing gradient penalties can be used to define an interpretable dependency measure that we name Sobolev Independence Criterion

*Tom Sercu is now with Facebook AI Research, and Cicero Dos Santos with Amazon AWS AI. The work was done when they were at IBM Research.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

(SIC). We devise an equivalent computational-friendly formulation of SIC in Section 3, that gives rise to additional auxiliary variables η_j. These naturally define normalized feature importance scores that can be used for feature selection. In Section 4 we study the case where the SIC witness function f is restricted to an RKHS and show that it leads to an optimization problem that is jointly convex in f and the importance scores η. We show that in this case SIC decomposes into the sum of feature scores, which is ideal for feature selection. In Section 5 we introduce a Neural version of SIC, which we show preserves the advantages in terms of interpretability when the witness function is parameterized as a homogeneous neural network, and which we show can be optimized using stochastic Block Coordinate Descent. In Section 6 we show how SIC and conditional Generative models can be used to control the False Discovery Rate using the recently introduced Holdout Randomization Test [8] and Knockoffs [9]. We validate SIC and its FDR control on synthetic and real datasets in Section 8.

2 Sobolev Independence Criterion: Interpretable Dependency Measure

Motivation: Feature Selection. We start by motivating gradient-sparsity regularization in SIC as a mean of selecting the features that maintain maximum dependency between two randoms variable X (the input) and Y (the response) defined on two spaces $\mathcal{X} \subset \mathbb{R}^{d_x}$ and $\mathcal{Y} \subset \mathbb{R}^{d_y}$ (in the simplest case $d_y = 1$). Let p_{xy} be the joint distribution of (X, Y) and p_x, p_y be the marginals of X and Y resp. Let D be an Integral Probability Metric associated with a function space $\mathscr{F}$, i.e for two distributions p, q:

$$D(p, q) = \sup_{f \in \mathscr{F}} \mathbb{E}_{x\sim p} f(x) - \mathbb{E}_{x\sim q} f(x).$$

With $p = p_{xy}$ and $q = p_x p_y$ this becomes a generalized definition of Mutual Information. Instead of the usual KL divergence, the metric D with its witness function, or critic, $f(x, y)$ measures the distance between the joint p_{xy} and the product of marginals $p_x p_y$. With this generalized definition of mutual information, the feature selection problem can be formalized as finding a sparse selector or gate $w \in \mathbb{R}^{d_x}$ such that $D(p_{w\odot x,y}, p_{w\odot x} p_y)$ is maximal [10, 11, 12, 13] , i.e. $\sup_{w, \|w\|_{\ell_0} \leq s} D(p_{w\odot x,y}, p_{w\odot x} p_y)$, where $\odot$ is a pointwise multiplication and $\|w\|_{\ell_0} = \#\{j | w_j \neq 0\}$. This problem can be written in the following penalized form:

$$\text{(P)}: \quad \sup_{w} \sup_{f \in \mathscr{F}} \mathbb{E}_{p_{xy}} f(w \odot x, y) - \mathbb{E}_{p_x p_y} f(w \odot x, y) - \lambda ||w||_{\ell_0}.$$

We can relabel $\tilde{f}(x, y) = f(w \odot x, y)$ and write (P) as: $\sup_{\tilde{f} \in \tilde{\mathscr{F}}} \mathbb{E}_{p_{xy}} \tilde{f}(x, y) - \mathbb{E}_{p_x p_y} \tilde{f}(x, y)$, where $\tilde{\mathscr{F}} = \{\tilde{f} | \tilde{f}(x, y) = f(w \odot x, y) | f \in \mathscr{F}, \|w\|_{\ell_0} \leq s\}$. Observe that we have: $\frac{\partial \tilde{f}}{\partial x_j} = w_j \frac{\partial f(w\odot x, y)}{\partial x_j}$. Since w_j is sparse the gradient of $\tilde{f}$ is sparse on the support of p_{xy} and $p_x p_y$. Hence, we can reformulate the problem (P) as follows:

$$\text{(SIC)}: \quad \sup_{f \in \mathscr{F}} \mathbb{E}_{p_{xy}} f(x, y) - \mathbb{E}_{p_x p_y} f(x, y) - \lambda P_S(f),$$

where $P_S(f)$ is a penalty that controls the sparsity of the gradient of the witness function f on the support of the measures. Controlling the nonlinear sparsity of the witness function in (SIC) via its gradients is more general and powerful than the linear sparsity control suggested in the initial form (P), since it takes into account the nonlinear interactions with other variables. In the following Section we formalize this intuition by theoretically examining sparsity-inducing gradient penalties [14].

Sparsity Inducing Gradient Penalties. Gradient penalties have a long history in machine learning and signal processing. In image processing the total variation norm is used for instance as a regularizer to induce smoothness. Splines in Sobolev spaces [15], and manifold learning exploit gradient regularization to promote smoothness and regularity of the estimator. In the context of neural networks, gradient penalties were made possible through double back-propagation introduced in [16] and were shown to promote robustness and better generalization. Such smoothness penalties became popular in deep learning partly following the introduction of WGAN-GP [17], and were used as regularizer for distance measures between distributions in connection to optimal transport theory [5, 6, 7]. Let μ be a dominant measure of p_{xy} and $p_x p_y$ the most commonly used gradient penalties is

$$\Omega_{L^2}(f) = \mathbb{E}_{(x,y)\sim\mu} \|\nabla_x f(x, y)\|^2 .$$

While this penalty promotes smoothness, it does not control the desired sparsity as discussed in the previous section. We therefore elect to instead use the nonlinear sparsity penalty introduced in [14] :

$\Omega_{\ell_0}(f) = \#\{j | \mathbb{E}_{(x,y)\sim\mu} \left|\frac{\partial f(x,y)}{\partial x_j}\right|^2 \neq 0\}$, and its relaxation :

$$\Omega_S(f) = \sum_{j=1}^{d_x} \sqrt{\mathbb{E}_{(x,y)\sim\mu} \left|\frac{\partial f(x,y)}{\partial x_j}\right|^2}.$$

As discussed in [14], $\mathbb{E}_{(x,y)\sim\mu} \left|\frac{\partial f(x,y)}{\partial x_j}\right|^2 = 0$ implies that f is constant with respect to variable x_j, if the function f is continuously differentiable and the support of μ is connected. These considerations motivate the following definition of the *Sobolev Independence Criterion* (SIC):

$$\text{SIC}_{(L_1)^2}(p_{xy}, p_x p_y) = \sup_{f\in\mathscr{F}} \mathbb{E}_{p_{xy}} f(x,y) - \mathbb{E}_{p_x p_y} f(x,y) - \frac{\lambda}{2}\left(\Omega_S(f)\right)^2 - \frac{\rho}{2}\mathbb{E}_\mu f^2(x,y).$$

Note that we add a ℓ_1-like penalty ($\Omega_S(f)$) to ensure sparsity and an ℓ_2-like penalty ($\mathbb{E}_\mu f^2(x,y)$) to ensure stability. This is similar to practices with linear models such as Elastic net.

Here we will consider $\mu = p_x p_y$ (although we could also use $\mu = \frac{1}{2}(p_{xy} + p_x p_y)$). Then, given samples $\{(x_i, y_i), i = 1, \ldots, N\}$ from the joint probability distribution p_{xy} and *iid* samples $\{(x_i, \tilde{y}_i), i = 1, \ldots, N\}$ from $p_x p_y$, SIC can be estimated as follows:

$$\widehat{\text{SIC}}_{(L_1)^2}(p_{xy}, p_x p_y) = \sup_{f\in\mathscr{F}} \frac{1}{N}\sum_{i=1}^N f(x_i, y_i) - \frac{1}{N}\sum_{i=1}^N f(x_i, \tilde{y}_i) - \frac{\lambda}{2}\left(\hat{\Omega}_S(f)\right)^2 - \frac{\rho}{2}\frac{1}{N}\sum_{i=1}^N f^2(x_i, \tilde{y}_i),$$

where $\hat{\Omega}_S(f) = \sum_{j=1}^{d_x} \sqrt{\frac{1}{N}\sum_{i=1}^N \left|\frac{\partial f(x_i,\tilde{y}_i)}{\partial x_j}\right|^2}$.

Remark 1. *Throughout this paper we consider feature selection only on x since y is thought of as the response. Nevertheless, in many other problems one can perform feature selection on x and y jointly, which can be simply achieved by also controlling the sparsity of $\nabla_y f(x,y)$ in a similar way.*

3 Equivalent Forms of SIC with η-trick

As it was just presented, the SIC objective is a difficult function to optimize in practice. First of all, the expectation appears after the square root in the gradient penalties, resulting in a non-smooth term (since the derivative of square root is not continuous at 0). Moreover, the fact that the expectation is inside the nonlinearity introduces a gradient estimation bias when the optimization of the SIC objective is performed using stochastic gradient descent (i.e. using mini-batches). We alleviate these problems (non-smoothness and biased expectation estimation) by making the expectation linear in the objective thanks to the introduction of auxiliary variables η_j that will end up playing an important role in this work. This is achieved thanks to a variational form of the square root that is derived from the following Lemma (which was used for a similar purpose as ours when alleviating the non-smoothness of mixed norms encountered in multiple kernel learning and group sparsity norms):

Lemma 1 ([18],[19])**.** *Let $a_j, j = 1 \ldots d$, $a_j > 0$ we have: $\left(\sum_{j=1}^d \sqrt{a_j}\right)^2 = \inf\{\sum_{j=1}^d \frac{a_j}{\eta_j} : \eta, \eta_j > 0 \sum_{j=1}^d \eta_j = 1\}$, optimum achieved at $\eta_j = \sqrt{a_j}/\sum_j \sqrt{a_j}$.*

We alleviate first the issue of non smoothness of the square root by adding an $\varepsilon \in (0,1)$, and we define: $\Omega_{S,\varepsilon} = \sum_{j=1}^{d_x} \sqrt{\mathbb{E}_{(x,y)\sim\mu} \left|\frac{\partial f(x,y)}{\partial x_j}\right|^2 + \varepsilon}$. Using Lemma 1 the nonlinear sparsity inducing gradient penalty can be written as :

$$(\Omega_{S,\varepsilon}(f))^2 = \inf\{\sum_{j=1}^{d_x} \frac{\mathbb{E}_{p_x p_y} \left|\frac{\partial f(x,y)}{\partial x_j}\right|^2 + \varepsilon}{\eta_j} : \eta, \eta_j > 0, \sum_{j=1}^{d_x} \eta_j = 1\},$$

where the optimum is achieved for : $\eta^*_{j,\varepsilon} = \frac{\beta_j}{\sum_{k=1}^{d_x} \beta_k}$, where $\beta_j^2 = \mathbb{E}_{p_x p_y} \left|\frac{\partial f(x,y)}{\partial x_j}\right|^2 + \varepsilon$. We refer to $\eta^*_{j,\varepsilon}$ as the normalized importance score of feature j. Note that η_j is a distribution over the features and gives a natural ranking between the features. Hence, substituting $\Omega(S)(f)$ with $\Omega_{S,\varepsilon}(f)$ in its equivalent form we obtain the ε perturbed SIC:

$$\text{SIC}_{(L_1)^2,\varepsilon}(p_{xy}, p_x p_y) = -\inf\{L_\varepsilon(f,\eta) : f \in \mathscr{F}, \eta_j, \eta_j > 0, \sum_{j=1}^{d_x} \eta_j = 1\}$$

where $L_\varepsilon(f,\eta) = -\Delta(f, p_{xy}, p_x p_y) + \frac{\lambda}{2}\sum_{j=1}^{d_x} \frac{\mathbb{E}_{p_x p_y}\left|\frac{\partial f(x,y)}{\partial x_j}\right|^2 + \varepsilon}{\eta_j} + \frac{\rho}{2}\mathbb{E}_{p_x p_y} f^2(x,y)$, and $\Delta(f, p_{xy}, p_x p_y) = \mathbb{E}_{p_{xy}} f(x,y) - \mathbb{E}_{p_x p_y} f(x,y)$. Finally, SIC can be empirically estimated as

$$\widehat{\text{SIC}}_{(L_1)^2,\varepsilon}(p_{xy}, p_x p_y) = -\inf\{\hat{L}_\varepsilon(f,\eta) : f \in \mathscr{F}, \eta_j, \eta_j > 0, \sum_{j=1}^{d_x} \eta_j = 1\}$$

where $\hat{L}_\varepsilon(f,\eta) = -\hat{\Delta}(f, p_{xy}, p_x p_y) + \frac{\lambda}{2}\sum_{j=1}^{d_x} \frac{\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\partial f(x_i,\tilde{y}_i)}{\partial x_j}\right|^2 + \varepsilon}{\eta_j} + \frac{\rho}{2}\frac{1}{N}\sum_{i=1}^{N} f^2(x_i,\tilde{y}_i)$, and main the objective $\hat{\Delta}(f, p_{xy}, p_x p_y) = \frac{1}{N}\sum_{i=1}^{N} f(x_i, y_i) - \frac{1}{N}\sum_{i=1}^{N} f(x_i, \tilde{y}_i)$.

Remark 2 (Group Sparsity)**.** *We can define similarly nonlinear group sparsity, if we would like our critic to depends on subsets of coordinates. Let $G_k, k = 1, \ldots, K$ be an overlapping or non overlapping group : $\Omega_{gS}(f) = \sum_{k=1}^{K}\sqrt{\sum_{j\in G_k}\mathbb{E}_{p_x p_y}\left|\frac{\partial f(x,y)}{\partial x_j}\right|^2}$. The η-trick applies naturally.*

4 Convex Sobolev Independence Criterion in Fixed Feature Spaces

We will now specify the function space $\mathscr{F}$ in SIC and consider in this Section critics of the form:

$$\mathscr{F} = \{f | f(x,y) = \langle u, \Phi_\omega(x,y)\rangle, \|u\|_2 \leq \gamma\},$$

where $\Phi_\omega : \mathcal{X}\times\mathcal{Y} \to \mathbb{R}^m$ is a fixed finite dimensional feature map. We define the mean embeddings of the joint distribution p_{xy} and product of marginals $p_x p_y$ as follow: $\mu(p_{xy}) = \mathbb{E}_{p_{xy}}[\Phi_\omega(x,y)]$, $\mu(p_x p_y) = \mathbb{E}_{p_x p_y}[\Phi_\omega(x,y)] \in \mathbb{R}^m$. Define the covariance embedding of $p_x p_y$ as $C(p_x p_y) = \mathbb{E}_{p_x p_y}[\Phi_\omega(x,y)\otimes\Phi_\omega(x,y)] \in \mathbb{R}^{m\times m}$ and finally define the Gramian of derivatives embedding for coordinate j as $D_j(p_x p_y) = \mathbb{E}_{p_x p_y}[\frac{\partial\Phi_\omega(x,y)}{\partial x_j}\otimes\frac{\partial\Phi_\omega(x,y)}{\partial x_j}] \in \mathbb{R}^{m\times m}$. We can write the constraint $\|u\|_2 \leq \gamma$ as the penalty term $-\tau\|u\|^2$. Define $L_\varepsilon(u,\eta) = \langle u, \mu(p_x p_y) - \mu(p_{xy})\rangle + \frac{1}{2}\left\langle u, \left(\lambda\sum_{j=1}^{d_x}\frac{D_j(p_x p_y)+\varepsilon}{\eta_j} + \rho C(p_x p_y) + \tau I_m\right)u\right\rangle$. Observe that :

$$\text{SIC}_{(L^1)^2,\varepsilon}(p_{xy}, p_x p_y) = -\inf\{L_\varepsilon(u,\eta) : u \in \mathbb{R}^m, \eta_j, \eta_j > 0, \sum_{j=1}^{d_x}\eta_j = 1\}.$$

We start by remarking that SIC is a form of gradient regularized maximum mean discrepancy [3]. Previous MMD work comparing joint and product of marginals did not use the concept of nonlinear sparsity. For example the Hilbert-Schmidt Independence Criterion (HSIC) [2] uses $\Phi_\omega(x,y) = \phi(x)\otimes\psi(y)$ with a constraint $||u||_2 \leq 1$. CCA and related kernel measures of dependence [20, 21] use L_2^2 constraints $L_2^2(p_x)$ and $L_2^2(p_y)$ on each function space separately.

Optimization Properties of Convex SIC We analyze in this Section the Optimization properties of SIC. Theorem 1 shows that the $\text{SIC}_{(L^1)^2,\varepsilon}$ loss function is jointly strictly convex in (u,η) and hence admits a unique solution that solves a fixed point problem.

Theorem 1 (Existence of a solution, Uniqueness, Convexity and Continuity)**.** *Note that $L(u,\eta) = L_{\varepsilon=0}(u,\eta)$. The following properties hold for the SIC loss:*

1) $L(u,\eta)$ is differentiable and jointly convex in (u,η). $L(u,\eta)$ is not continuous for η, such that $\eta_j = 0$ for some j.

2) Smoothing, Perturbed SIC: For $\varepsilon \in (0,1)$, $L_\varepsilon(u,\eta) = L(u,\eta) + \frac{\lambda}{2}\sum_{j=1}^{d_x}\frac{\varepsilon}{\eta_j}$ is jointly strictly convex and has compact level sets on the probability simplex, and admits a unique minimizer $(u_\varepsilon^*, \eta_\varepsilon^*)$.

3) The unique minimizer of $L_\varepsilon(u,\eta)$ is a solution of the following fixed point problem: $u_\varepsilon^* = \left(\lambda\sum_{j=1}^{d_x}\frac{D_j(p_x p_y)}{\eta_j^*} + \rho C(p_x p_y) + \tau I_m\right)^{-1}(\mu(p_{xy}) - \mu(p_x p_y))$, and $\eta_{j,\varepsilon}^* = \frac{\sqrt{\langle u_\varepsilon^*, D_j(p_x p_y)u_\varepsilon^*\rangle + \varepsilon}}{\sum_{k=1}^{d_x}\sqrt{\langle u_\varepsilon^*, D_k(p_x p_y)u_\varepsilon^*\rangle + \varepsilon}}$.

The following Theorem shows that a solution of the unperturbed SIC problem can be obtained from the smoothed $\text{SIC}_{(L_1)^2,\varepsilon}$ in the limit $\varepsilon \to 0$:

Theorem 2 (From Perturbed SIC to SIC)**.** *Consider a sequence ε_ℓ, $\varepsilon_\ell \to 0$ as $\ell \to \infty$, and consider a sequence of minimizers $(u^*_{\varepsilon_\ell}, \eta^*_\ell)$ of $L_{\varepsilon_\ell}(u, \eta)$, and let (u^*, η^*) be the limit of this sequence, then (u^*, η^*) is a minimizer of $L(u, \eta)$.*

Interpretability of SIC. The following corollary shows that SIC can be written in terms of the importance scores of the features, since at optimum the main objective is proportional to the constraint term. It is to the best of our knowledge the first dependency criterion that decomposes in the sum of contributions of each coordinate, and hence it is an interpretable dependency measure. Moreover, η^*_j are normalized importance scores of each feature j, and their ranking can be used to assess feature importance.

Corollary 1 (Interpretability of Convex SIC)**.** *Let (u^*, η^*) be the limit defined in Theorem 2. Define $f^*(x, y) = \langle u^*, \Phi_\omega(x, y)\rangle$, and $\|f^*\|_{\mathscr{F}} = \|u^*\|$. We have that*

$$\begin{aligned} SIC_{(L^1)^2}(p_{xy}, p_x p_y) &= \frac{1}{2}\left(\mathbb{E}_{p_{xy}} f^*(x, y) - \mathbb{E}_{p_x p_y} f^*(x, y)\right) \\ &= \frac{\lambda}{2}\left(\sum_{j=1}^{d_x} \sqrt{\mathbb{E}_{p_x p_y} |\frac{\partial f^*(x, y)}{\partial x_j}|^2}\right)^2 + \frac{\rho}{2}\mathbb{E}_{p_x p_y} f^{*,2}(x, y) + \frac{\tau}{2}||f^*||^2_{\mathscr{F}}. \end{aligned}$$

Moreover, $\sqrt{\mathbb{E}_{p_x p_y} |\frac{\partial f^*(x,y)}{\partial x_j}|^2} = \eta^*_j \Omega_{S,L_1}(f^*)$ and $\sum_{j=1}^{d_x} \eta_j = 1$. The terms η^*_j can be seen as quantifying how much dependency as measured by SIC can be explained by a coordinate j. Ranking of η^*_j can be used to rank influence of coordinates.

Thanks to the joint convexity and the smoothness of the perturbed SIC, we can solve convex empirical SIC using alternating minimization on u and η or block coordinate descent using first order methods such as gradient descent on u and mirror descent [22] on η that are known to be globally convergent in this case (see Appendix A for more details).

5 Non Convex Neural SIC with Deep ReLU Networks

While Convex SIC enjoys a lot of theoretical properties, a crucial short-coming is the need to choose a feature map Φ_ω that essentially goes back to the choice of a kernel in classical kernel methods. As an alternative, we propose to learn the feature map as a deep neural network. The architecture of the network can be problem dependent, but we focus here on a particular architecture: Deep ReLU Networks with biases removed. As we show below, using our sparsity inducing gradient penalties with such networks, results in input sparsity at the level of the witness function f of SIC. This is desirable since it allows for an interpretable model, similar to the effect of Lasso with Linear models, our sparsity inducing gradient penalties result in a nonlinear self-explainable witness function f [23], with explicit sparse dependency on the inputs.

Deep ReLU Networks with no biases, homogeneity and Input Sparsity via Gradient Penalties. We start by invoking the Euler Theorem for homogeneous functions:

Theorem 3 (Euler Theorem for Homogeneous Functions)**.** *A continuously differentiable function f is defined as homogeneous of degree k if $f(\lambda x) = \lambda^k f(x), \forall \lambda \in \mathbb{R}$. The Theorem states that f is homogeneous of degree k if and only if $kf(x) = \langle \nabla_x f(x), x\rangle = \sum_{j=1}^{d_x} \frac{\partial f(x)}{\partial x_j} x_j$.*

Now consider *deep ReLU networks with biases removed* for any number of layers L: $\mathscr{F}_{ReLu} = \{f | f(x, y) = \langle u, \Phi_\omega(x)\rangle$, where $\Phi_\omega(x, y) = \sigma(W_L \ldots \sigma(W_2\sigma(W_1[x, y]))), u \in \mathbb{R}^m, \Phi_\omega : \mathbb{R}^{d_x+d_y} \to \mathbb{R}^m\}$, where $\sigma(t) = \max(t, 0), W_j$ are linear weights. Any $f \in \mathscr{F}_{ReLU}$ is clearly homogeneous of degree 1. As an immediate consequence of Euler Theorem we then have: $f(x, y) = \langle \nabla_x f(x, y), x\rangle + \langle \nabla_y f(x, y), y\rangle$. The first term is similar to a linear term in a linear model, the second term can be seen as a bias. Using our sparsity-inducing gradient penalties with such networks guarantees that on average on the support of a dominant measure the gradients with respect to x are sparse. Intuitively, the gradients wrt x act like the weight in linear models, and our sparsity inducing gradient penalty act like the ℓ_1 regularization of Lasso. The main advantage compared to Lasso is that we have a highly nonlinear decision function, that has better capacity of capturing dependencies between X and Y.

Non-convex SIC with Stochastic Block Coordinate Descent (BCD). We define the empirical non convex $SIC_{(L^1)^2}$ using this function space $\mathscr{F}_{\text{ReLu}}$ as follows:

$$\widehat{\text{SIC}}_{(L^1)^2}(p_{xy}, p_x p_y) = -\inf\{\hat{L}(f_\theta, \eta) : f_\theta \in \mathscr{F}_{\boldsymbol{ReLU}}, \eta_j, \eta_j > 0, \sum_{j=1}^{d_x} \eta_j = 1\},$$

where $\theta = (vec(W_1) \dots vec(W_L), u)$ are the network parameters. Algorithm 3 in Appendix B summarizes our stochastic BCD algorithm for training the Neural SIC. The algorithm consists of SGD updates to θ and mirror descent updates to η.

Boosted SIC. When training Neural SIC, we can obtain different critics f_ℓ and importance scores η_ℓ, by varying random seeds or hyper-parameters (architecture, batch size etc). Inspired by importance scores in random forest, we define **Boosted SIC** as the arithmetic mean or the geometric mean of η_ℓ.

6 FDR Control and the Holdout Randomization Test/ Knockoffs.

Controlling the False Discovery Rate (FDR) in Feature Selection is an important problem for reproducible discoveries. In a nutshell, for a feature selection problem given the ground-truth set of features $\mathcal{S}$, and a feature selection method such as SIC that gives a candidate set $\hat{S}$, our goal is to maximize the TPR (True Positive Rate) or the power, and to keep the False Discovery Rate (FDR) under Control. TPR and FDR are defined as follows:

$$\text{TPR} := \mathbb{E}\left[\frac{\#\{i : i \in \hat{S} \cap \mathcal{S}\}}{\#\{i : i \in \mathcal{S}\}}\right] \quad \text{FDR} := \mathbb{E}\left[\frac{\#\{i : i \in \hat{S} \backslash \mathcal{S}\}}{\#\{i : i \in \hat{S}\}}\right]. \tag{1}$$

We explore in this paper two methods that provably control the FDR: 1) The Holdout Randomization Test (HRT) introduced in [8], that we specialize for SIC in Algorithm 4; 2) Knockoffs introduced in [9] that can be used with any basic feature selection method such as Neural SIC, and guarantees provable FDR control.

HRT-SIC. We are interested in measuring the conditional dependency between a feature x_j and the response variable y conditionally on the other features noted x_{-j}. Hence we have the following null hypothesis: $H_0 : x_j \perp\!\!\!\perp y \,|x_{-j} \iff p_{xy} = p_{x_j|x_{-j}} p_{y|x_{-j}} p_{x_{-j}}$. In order to simulate the null hypothesis, we propose to use generative models for sampling from $x_j|x_{-j}$ (See Appendix D). The principle in HRT [8] that we specify here for SIC in Algorithm 4 (given in Appendix B) is the following: instead of refitting SIC under H_0, we evaluate the mean of the witness function of SIC on a holdout set sampled under H_0 (using conditional generators for R rounds). The deviation of the mean of the witness function under H_0 from its mean on a holdout from the real distribution gives us p-values. We use the Benjamini-Hochberg [24] procedure on those p-values to achieve a target FDR. We apply HRT-SIC on a shortlist of pre-selected features per their ranking of η_j.

Knockoffs-SIC. Knockoffs [25] work by finding control variables called knockoffs $\tilde{x}$ that mimic the behavior of the real features x and provably control the FDR [9]. We use here Gaussian knockoffs [9] and train SIC on the concatenation of $[x, \tilde{x}]$, i.e we train $SIC([X; \tilde{X}], Y)$ and obtain η that has now twice the dimension d_x, i.e for each real feature j, there is the real importance score η_j and the knockoff importance score η_{j+d_x}. knockoffs-SIC consists in using the statistics $W_j = \eta_j - \eta_{j+d_x}$ and the knockoff filter [9] to select features based on the sign of W_j (See Alg. 5 in Appendix).

7 Relation to Previous Work

Kernel/Neural Measure of Dependencies. As discussed earlier SIC can be seen as a *sparse* gradient regularized MMD [3, 7] and relates to the Sobolev Discrepancy of [5, 6]. Feature selection with MMD was introduced in [10] and is based on backward elimination of features by recomputing MMD on the ablated vectors. SIC has the advantage of fitting one critic that has interpretable feature scores. Related to the MMD is the Hilbert Schmidt Independence Criterion (HSIC) and other variants of kernel dependency measures introduced in [2, 21]. None of those criteria has a nonparametric sparsity constraint on its witness function that allows for explainability and feature selection. Other Neural measures of dependencies such as MINE [26] estimate the KL divergence using neural networks, or that of [27] that estimates a proxy to the Wasserstein distance using Neural Networks.

Interpretability, Sparsity, Saliency and Sensitivity Analysis. Lasso and elastic net [28] are interpretable linear models that exploit sparsity, but are limited to linear relationships. Random forests

[29] have a heuristic for determining feature importance and are successful in practice as they can capture nonlinear relationships similar to SIC. We believe SIC can potentially leverage the deep learning toolkit for going beyond tabular data where random forests excel, to more structured data such as time series or graph data. Finally, SIC relates to saliency based post-hoc interpretation of deep models such as [30, 31, 32]. While those method use the gradient information for a post-hoc analysis, SIC incorporates this information to guide the learning towards the important features. As discussed in Section 2.1 many recent works introduce deep networks with input sparsity control through a learned gate or a penalty on the weights of the network [11, 12, 13]. SIC exploits a stronger notion of sparsity that leverages the relationship between the different covariates.

8 Experiments

Synthetic Data Validation. We first validate our methods and compare them to baseline models in simulation studies on synthetic datasets where the ground truth is available by construction. For this we generate the data according to a model $y = f(x) + \epsilon$ where the model $f(\cdot)$ and the noise ϵ define the specific synthetic dataset (see Appendix F.1). In particular, the value of y only depends on a subset of features x_i, $i = 1, \ldots, p$ through $f(\cdot)$, and performance is quantified in terms of TPR and FDR in discovering them among the irrelevant features. We experiment with two datasets: **A) Complex multivariate synthetic data (SinExp)**, which is generated from a complex multivariate model proposed in [33] Sec 5.3, where 6 *ground truth* features x_i out of 50 generate the output y through a non-linearity involving the product and composition of the $\cos$, $\sin$ and $\exp$ functions (see Appendix F.1). We therefore dub this dataset SinExp. To increase the difficulty even further, we introduce a pairwise correlation between all features of 0.5. In Fig. 1 we show results for datasets of 125 and 500 samples repeated 100 times comparing performance of our models with the one of two baselines: Elastic Net (EN) and Random Forest (RF). **B) Liang Dataset.** We show results on the benchmark dataset proposed by [34], specifically the *generalized* Liang dataset matching most of the setup from [8] Sec 5.1. We provide dataset details and results in Appendix F.1 (Results in Figure 2).

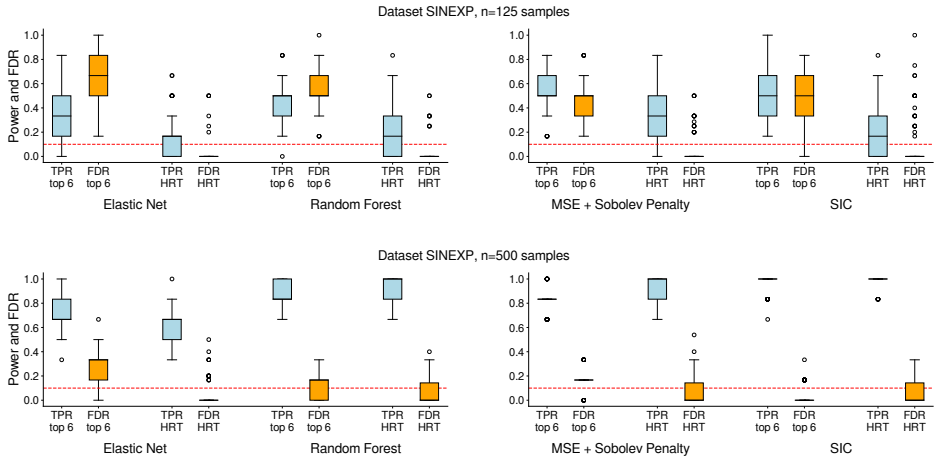

Figure 1: SinExp synthetic dataset. TPR and FDR of Elastic Net (EN) and Random Forest (RF) baseline models (left panels) are compared to our methods: a 2-hidden layer neural network with no biases trained to minimize an objective comprising an MSE cost and a Sobolev Penalty term (MSE + Sobolev Penalty), and the same network trained to optimize SIC criterion (right panels), for datasets of 125 samples (top panels) and 500 samples (bottom panels). For all models TPR and FDR are computed by selecting the top 6 features in order of feature importance (which for EN is defined as the absolute value of the weight of a feature, for RF is the out-of-bag error associated to it (see [35]), and for our method is the value of its η). Selecting the first 6 features is useful to compare models, but assumes *oracle knowledge* of the fact that there are 6 ground truth features. We therefore also compute FDR and TPR after selecting features using the HRT method of [8] among the top 20 features. HRT estimates the importance of a feature quantifying its effect on the distribution of y on a holdout set by replacing its values with samples from a conditional distribution (see Section 6). We use HRT to control FDR rate at 10% (red horizontal dotted line). Standard box plots are generated over 100 repetitions of each simulation.

Feature Selection on Drug Response dataset. We consider as a real-world application the Cancer Cell Line Encyclopedia (CCLE) dataset [36], described in Appendix F.2. We study the result of using the normalized importance scores η_j from SIC for (heuristic) feature selection, against features selected by Elastic Net. Table 1 shows the heldout MSE of a predictor trained on selected features, averaged over 100 runs (each run: new randomized 90%/10% data split, NN initialization). The goal here is to quantify the predictiveness of features selected by SIC on its own, without the full randomized testing machinery. The SIC critic and regressor NN were respectively the big_critic and $regressor\_NN$ described with training details in Appendix F.3, while the random forest is trained with default hyper parameters from scikit-learn [37]. We can see that, with just η_j, informative features are selected for the downstream regression task, with performance comparable to those selected by ElasticNet, which was trained explicitly for this task. The features selected with high η_j values and their overlap with the features selected by ElasticNet are listed in Appendix F.2 Table 3.

	NN	RF
All 7251 features	1.160 ± 3.990	0.783 ± 0.167
Elastic-Net1 [36] top-7	0.864 ± 0.432	0.931 ± 0.215
Elastic-Net2 [8] top-10	$\mathbf{0.663 \pm 0.161}$	0.830 ± 0.190
SIC top-7	0.728 ± 0.166	0.856 ± 0.189
SIC top-10	0.706 ± 0.158	$\mathbf{0.817 \pm 0.173}$
SIC top-15	0.734 ± 0.168	0.859 ± 0.202

Table 1: CCLE results on downstream regression task. Heldout MSE for drug PLX4720 prediction based on selected features. Columns: neural network (NN) and random forest (RF) regressors.

HIV-1 Drug Resistance with Knockoffs-SIC. The second real-world dataset that we analyze is the HIV-1 Drug Resistance[38], which consists in detecting mutations associated with resistance to a drug type. For our experiments we use all the three classes of drugs: Protease Inhibitors (PIs), Nucleoside Reverse Transcriptase Inhibitors (NRTIs), and Non-nucleoside Reverse Transcriptase Inhibitors (NNRTIs). We use the pre-processing of each dataset (<drug-class, drug-type>) of the knockoff tutorial [39] made available by the authors. Concretely, we construct a dataset $(X, \tilde{X})$ of the concatenation of the real data and Gaussian knockoffs [9], and fit $SIC([X, \tilde{X}], Y)$. As explained in Section 6, we use in the knockoff filter the statistics $W_j = \eta_j - \eta_{j+d_x}$, i.e. the difference of SIC importance scores between each feature and its corresponding knockoff. For SIC experiments, we use $small\_critic$ architecture (See Appendix F.3 for training details). We use Boosted SIC, by varying the batch sizes in $N \in \{10, 30, 50\}$, and computing the geometric mean of η produced by those three setups as the feature importance needed for Knockoffs. Results are summarized in Table 2.

Drug Class	Drug Type	Knockoff with GLM			Boosted SIC Knockoff		
		TD	FD	FDP	TD	FD	FDP
PIs	APV	19	3	**0.13**	17	5	0.22
	ATV	22	8	0.26	19	1	**0.05**
	IDV	19	12	0.38	15	3	**0.16**
	LPV	16	1	**0.05**	14	2	0.12
	NFV	24	7	0.22	19	5	**0.21**
	RTV	19	8	0.29	12	2	**0.20**
	SQV	17	4	**0.19**	14	8	0.36
NRTIs	X3TC	0	0	0	7	0	**0**
	ABC	10	1	0.09	11	1	**0.08**
	AZT	16	4	**0.2**	12	5	0.29
	D4T	6	1	0.14	8	0	**0**
	DDI	0	0	0	8	0	**0**
NNRTIs	DLV	10	13	0.56	8	10	**0.55**
	EFV	11	11	0.5	11	10	**0.47**
	NVP	7	10	**0.58**	7	11	0.611

Table 2: Comparison of applying (knockoff filter + GLM) and (Knockoff filter+Boosted SIC). For each <drug-class, drug-type> we compared the True Discoveries (TD), False Discoveries(FD) and False Discovery Proportion (FDP). Knockoff with Boosted SIC keeps FDP under control without compromising power, and succeeds in making true discoveries that GLM with knockoffs doesn't find.

9 Conclusion

We introduced in this paper the Sobolev Independence Criterion (SIC), a dependency measure that gives rise to feature importance which can be used for feature selection and interpretable decision making. We laid down the theoretical foundations of SIC and showed how it can be used in conjunction with the Holdout Randomization Test and Knockoffs to control the FDR, enabling reliable discoveries. We demonstrated the merits of SIC for feature selection in extensive synthetic and real-world experiments with controlled FDR.

References

[1] Bharath K. Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Scholkopf, and Gert R. G. Lanckriet. On integral probability metrics, ϕ-divergences and binary classification. 2009.

[2] A. Gretton, K. Fukumizu, CH. Teo, L. Song, B. Schölkopf, and AJ. Smola. A kernel statistical test of independence. In *Advances in neural information processing systems 20*, 2008.

[3] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *JMLR*, 2012.

[4] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Arxiv*, 2017.

[5] Youssef Mroueh, Chun-Liang Li, Tom Sercu, Anant Raj, and Yu Cheng. Sobolev gan. *ICLR*, 2018.

[6] Youssef Mroueh, Tom Sercu, and Anant Raj. Sobolev descent. In *AISTATS*, 2019.

[7] Michael Arbel, Dougal J. Sutherland, Mikolaj Binkowski, and Arthur Gretton. On gradient regularizers for mmd gans. *NeurIPS*, 2018.

[8] W. Tansey, V. Veitch, H. Zhang, R. Rabadan, and D. M. Blei. The holdout randomization test: Principled and easy black box feature selection. *arXiv preprint arXiv:1811.00645*, 2018.

[9] Emmanuel Candes, Yingying Fan, Lucas Janson, and Jinchi Lv. Panning for gold: model-x knockoffs for high dimensional controlled variable selection. 2018.

[10] Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. Feature selection via dependence maximization. *J. Mach. Learn. Res.*, 2012.

[11] Jean Feng and Noah Simon. Sparse-input neural networks for high-dimensional nonparametric regression and classification. 2017.

[12] Mao Ye and Yan Sun. Variable selection via penalized neural network: a drop-out-one loss approach. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[13] Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. Deep supervised feature selection using stochastic gates. *Arxiv*, 2018.

[14] Lorenzo Rosasco, Silvia Villa, Sofia Mosci, Matteo Santoro, and Alessandro Verri. Nonparametric sparsity and regularization. *J. Mach. Learn. Res.*, 2013.

[15] Grace Wahba. Smoothing noisy data with spline functions. *Numerische mathematik*, 24(4), 1975.

[16] Harris Drucker and Yann LeCun. Improving generalization performance using double back-propagation. *IEEE Transactions on Neural Networks*, 1992.

[17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv:1704.00028*, 2017.

[18] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Mach. Learn.*, 2008.

[19] Francis Bach, Rodolphe Jenatton, and Julien Mairal. *Optimization with Sparsity-Inducing Penalties (Foundations and Trends(R) in Machine Learning)*. Now Publishers Inc., Hanover, MA, USA, 2011.

[20] H.D. Vinod. Canonical ridge and econometrics of joint production. *Journal of Econometrics*, 1976.

[21] Kenji Fukumizu, Arthur Gretton, Xiaohai Sun, and Bernhard Schölkopf. Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems 20*. 2008.

[22] Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Oper. Res. Lett.*, 2003.

[23] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems 31*. 2018.

[24] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A Practical and powerful approach to multiple testing. *J. Roy. Statist. Soc.*, 57:289–300, 1995.

[25] Rina Foygel Barber, Emmanuel J Candès, et al. Controlling the false discovery rate via knockoffs. *The Annals of Statistics*, 43(5):2055–2085, 2015.

[26] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: Mutual information neural estimation, 2018.

[27] Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning, 2019.

[28] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer New York Inc., 2001.

[29] Leo Breiman. Random forests. *Mach. Learn.*, 2001.

[30] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[31] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *International Conference on Learning Representations (Workshop Track).*, 2014.

[32] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 2015.

[33] Jean Feng and Noah Simon. Sparse-input neural networks for high-dimensional nonparametric regression and classification. *arXiv preprint arXiv:1711.07592*, 2017.

[34] Faming Liang, Qizhai Li, and Lei Zhou. Bayesian neural networks for selection of drug sensitive genes. *Journal of the American Statistical Association*, 113(523), 2018.

[35] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[36] Jordi Barretina, Giordano Caponigro, Nicolas Stransky, Kavitha Venkatesan, Adam A Margolin, Sungjoon Kim, Christopher J Wilson, Joseph Lehár, Gregory V Kryukov, Dmitriy Sonkin, et al. The cancer cell line encyclopedia enables predictive modelling of anticancer drug sensitivity. *Nature*, 483(7391):603, 2012.

[37] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[38] Soo-Yon Rhee, Jonathan Taylor, Gauhar Wadhera, Asa Ben-Hur, Douglas L Brutlag, and Robert W Shafer. Genotypic predictors of human immunodeficiency virus type 1 drug resistance. *Proceedings of the National Academy of Sciences*, 103(46):17355–17360, 2006.

[39] Matteo Sesia and Evan Patterson. R tutorial for knockoffs - 4. https://web.stanford.edu/group/candes/knockoffs/software/knockoffs/tutorial-4-r.html, 2017.

[40] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.*, 109, 2001.

[41] Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 2013.

[42] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pages 4790–4798, 2016.

[43] Ethan Perez, Harm de Vries, Florian Strub, Vincent Dumoulin, and Aaron Courville. Learning visual reasoning without strong priors. *arXiv preprint arXiv:1707.03017*, 2017.

[44] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[45] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[46] Yaniv Romano, Matteo Sesia, and Emmanuel Candès. Deep knockoffs. *Journal of the American Statistical Association*, pages 1–27, 2019.

A Algorithms for Convex SIC

Algorithms and Empirical Convex SIC from Samples. Given samples from the joint and the marginals, it is easy to see that the empirical loss $\hat{L}_\varepsilon$ can be written in the same way with empirical feature mean embeddings $\hat{\mu}(p_{xy}) = \frac{1}{N}\sum_{i=1}^N \Phi_\omega(x_i, y_i)$ and $\hat{\mu}(p_x p_y) = \frac{1}{N}\sum_{i=1}^N \Phi_\omega(x_i, \tilde{y}_i)$, covariances $\hat{C}(p_x p_y) = \frac{1}{N}\sum_{i=1}^N \Phi_\omega(x_i, \tilde{y}_i) \otimes \Phi_\omega(x_i, \tilde{y}_i)$ and derivatives grammians $\hat{D}_j(p_x p_y) = \frac{1}{N}\sum_{i=1}^N \frac{\partial \Phi_\omega(x_i, \tilde{y}_i)}{\partial x_j} \otimes \frac{\partial \Phi_\omega(x,y)}{\partial x_j}$. Given the strict convexity of $\hat{L}_\varepsilon$ jointly in u and η, alternating optimization as given in Algorithm 1 in Appendix is known to be convergent to a global optima (Theorem 4.1 in [40]). Similarly Block Coordinate Descent (BCD) using first order methods as given in Algorithms 3 and 2 (in Appendix): gradient descent on u and mirror descent on η (in order to satisfy the simplex constraint [22]) are also known to be globally convergent (Theo 2 in [41].)

Algorithm 1 Alternating Optimization

Inputs: ε,λ, τ, ρ , Φ_ω
Initialize $\hat{\eta}_j = \frac{1}{d_x}, \forall j$, $\hat{\delta} = \hat{\mu}(p_{xy}) - \hat{\mu}(p_x p_y)$
for $i = 1 \dots$ Maxiter **do**
$\hat{u} \leftarrow$
$\left(\lambda \sum_{j=1}^{d_x} \frac{\hat{D}_j(p_x p_y)}{\hat{\eta}_j} + \rho \hat{C}(p_x p_y) + \tau I_m\right)^{-1} \hat{\delta}$
$\hat{\eta}_j \leftarrow \frac{\sqrt{\langle \hat{u}, \hat{D}_j(p_x p_y)\hat{u}\rangle + \varepsilon}}{\sum_{k=1}^{d_x} \sqrt{\langle \hat{u}, \hat{D}_k(p_x p_y)\hat{u}\rangle + \varepsilon}}$
end for
Output: $\hat{u}, \hat{\eta}$

Algorithm 2 Block Coordinate Descent

Inputs: ε,λ, τ, ρ, α, α_η (learning rates),Φ_ω
Initialize $\hat{\eta}_j = \frac{1}{d_x}, \forall j$, Softmax$(z) = e^z / \sum_{j=1}^{d_x} e^{z_j}$
for $i = 1 \dots$ Maxiter **do**
Gradient step u:
$\hat{u} \leftarrow \hat{u} - \alpha \frac{\partial \hat{L}_\varepsilon(\hat{u}, \hat{\eta})}{\partial u}$
Mirror Descent η :
logit $\leftarrow \log(\hat{\eta}) - \alpha_\eta \frac{\partial \hat{L}_\varepsilon(\hat{u}, \hat{\eta})}{\partial \eta}$
$\hat{\eta} \leftarrow$ Softmax(logit) {stable implementation of softmax}
end for
Output: $\hat{u}, \hat{\eta}$

B Algorithms for Neural SIC, HRT-SIC and Model-X Knockoff SIC

Algorithm 3 *(non convex)* Neural SIC(X, Y) (Stochastic BCD)

Inputs: X, Y dataset $X \in \mathbb{R}^{N \times d_x}, Y \in \mathbb{R}^{N \times d_y}$, such that $(x_i = X_{i,.}, y_i = Y_{i,.}) \sim p_{xy}$
Hyperparameters: ε,λ, τ, ρ, α_θ, α_η (learning rates)
Initialize $\eta_j = \frac{1}{d_x}, \forall j$, Softmax$(z) = e^z / \sum_{j=1}^{d_x} e^{z_j}$
for $iter = 1 \dots$ Maxiter **do**
Fetch a minibatch of size N $(x_i, y_i) \sim p_{xy}$
Fetch a minibatch of size N $(x_i, \tilde{y}_i) \sim p_x p_y$ {$\tilde{y}_i$ obtained by permuting rows of Y}
Stochastic Gradient step on θ:
$\theta \leftarrow \theta - \alpha_\theta \frac{\partial \hat{L}(f_\theta, \eta)}{\partial \theta}$ {We use ADAM}
Mirror Descent η :
logit $\leftarrow \log(\eta) - \alpha_\eta \frac{\partial \hat{L}(f_\theta, \eta)}{\partial \eta}$
$\eta \leftarrow$ Softmax(logit) {stable implementation of softmax}
end for
Output: f_θ, η

Algorithm 4 HRT With SIC (X, Y)

Inputs: $D_{train} = (X_{tr}, Y_{tr})$, a Heldout set $D_{\text{Holdout}} = (X, Y)$, features Cutoff K
SIC: $(f_{\theta^*}, \eta_*) = \text{SIC}(D_{train})$ {Alg. 3}
Score of witness on Hold out : $S^* = \text{MEAN}(f_{\theta^*}(X, Y))$
Conditional Generators Pre-trained conditional Generator : $G(x_{-j}, j)$ predicts $X_j | X_{-j}$
Shortlist : $I = \text{INDEXTOPK}(\eta)$
{p-values for $j \in I$; randomizations tests}
for $j \in I$ **do**
for $r = 1 \dots R$ **do**
Construct $\tilde{X}$, $\tilde{X}_{.,k} = X_{.,k} \forall k \neq j$ and $\tilde{X}_{.,j} = G(X_{-j}, j)$ {Simulate Null Hyp.}
$S_{j,r} = \text{MEAN}(f_{\theta^*}(\tilde{X}, Y))$ {Score of witness function on the Null}
end for
$p_j = \frac{1}{R+1}\left(1 + \sum_{r=1}^R 1_{S_j^r \geq S^*}\right)$
end for
discoveries =**BH**(p,targetFDR) {Benjamini-Hochberg Procedure}
Output: discoveries

Algorithm 5 Model-X Knockoffs FDR control with SIC

Inputs: $D_{train} = (X_{tr}, Y_{tr})$, Model-X knockoff features $\tilde{X} \sim \text{ModelX}(X_{tr})$, target FDR q
Train SIC: $(f_{\theta^*}, \eta) = \text{SIC}([X_{tr}, \tilde{X}], Y)$, {Alg. 3} where $[X_{tr}, \tilde{X}]$ is the concatenation of X_{tr} and knockoffs $\tilde{X}$
for $j = 1, \ldots, d_X$ **do**
  Compute importance score of j feature: $W_j = \eta_j - \eta_{j+d_x}$, where η_{j+d_X} is the η of feature knockoff $\tilde{X}_j$
end for
Compute threshold $\tau > 0$ by setting $\tau = \min\left\{t > 0 : \frac{\#\{j:W_j \leq -t\}}{\#\{j:W_j \geq t\}} \leq q\right\}$
Output: discoveries $\{j : W_j > \tau\}$

C Proofs

Proof of Theorem 1. 1) Let $\delta = \mu(p_{xy}) - \mu(p_x p_y)$.

We have

$$L(u,\eta) = -\langle u, \delta\rangle + \frac{1}{2}\langle u, (\rho C(p_x p_y) + \tau I_m)u\rangle + \frac{\lambda}{2}\sum_j \frac{\langle u, D_j(p_x p_y)u\rangle}{\eta_j}, u \in \mathbb{R}^m \text{ and } \eta \in \Delta^{d_x}$$

where Δ^{d_x} is the probability simplex. L is the sum of a linear tem and quadratic terms (convex in u) and a function of the form

$$f(u,\eta) = \frac{1}{2}\sum_{j=1}^{d_x} \frac{u^\top A_j u}{\eta_j}$$

where A_j are PSD matrices, and η is in the probability simplex (convex). Hence it is enough to show that f is jointly convex. Let $g(w,\eta) = \frac{w^\top A w}{\eta}, \eta > 0$. The Hessian of $g(w,\eta)$, Hg has the following form:

$$Hg(w,\eta) = \left[\begin{array}{c|c} \frac{\partial^2 L}{\partial w \otimes \partial w} & \frac{\partial^2 L}{\partial w \partial \eta} \\ \hline \frac{\partial^2 L}{\partial \eta \partial w} & \frac{\partial^2 L}{\partial \eta^2} \end{array}\right] = \left[\begin{array}{c|c} \frac{A}{\eta} & -\frac{Aw}{\eta^2} \\ \hline -\frac{w^\top A}{\eta^2} & \frac{w^\top A w}{\eta^3} \end{array}\right]$$

Let us prove that for all $(w,\eta), \eta_j >, \forall j 0$:

$$(w',\eta')^\top Hg(w,\eta)(w',\eta') \geq 0, \forall (w',\eta'), \eta'_j > 0, \forall j$$

We have :

$$\begin{aligned}(w',\eta')^\top Hg(w,\eta)(w',\eta') &= \frac{\langle w', Aw'\rangle}{\eta} - 2\eta'\frac{\langle w', Aw\rangle}{\eta^2} + \eta'^2\frac{w^\top A w}{\eta^3} \\ &= \frac{1}{\eta}\left(\langle w', Aw'\rangle - \frac{2\eta'}{\eta}\langle w', Aw\rangle + \frac{\eta'^2}{\eta^2} w^\top A w\right) \\ &= \frac{1}{\eta}\left\|A^{\frac{1}{2}}w' - \frac{\eta'}{\eta}A^{\frac{1}{2}}w\right\|_2^2 \geq 0 \text{ for } \eta > 0\end{aligned}$$

Now back to f it is easy to see that :

$$(w',\eta')^\top Hf(w,\eta)(w',\eta') = \sum_{j=1}^{d_x}\frac{1}{\eta_j}\left\|A_j^{\frac{1}{2}}w' - \frac{\eta'_j}{\eta_j}A_j^{\frac{1}{2}}w\right\|_2^2 \geq 0 \text{ for } \eta \in \Delta_j^{d_x}, \eta_j > 0.$$

Hence the loss L is jointly convex in (u,η). Due to discontinuity at $\eta_j = 0$ the loss is not continuous .

2) It is easy to see that the hessian becomes definite:

$$(w',\eta')^\top H L_\varepsilon(w,\eta)(w',\eta') = \sum_{j=1}^{d_x} \frac{1}{\eta_j}\left(\left\| A_j^{\frac{1}{2}} w' - \frac{\eta'_j}{\eta_j} A_j^{\frac{1}{2}} w \right\|_2^2 + \varepsilon(\frac{\eta'_j}{\eta_j})^2\right) > 0 \text{ for } \eta \in \Delta_j^{d_x}, \eta_j, \eta'_j > 0,$$

and $L_\varepsilon(u,\eta)$ is jointly strictly convex, u is unconstrained and η belongs to a convex set (the probability simplex) and hence admits a unique minimizer.

3) The unique minimizer satisfies first order optimality conditions for the following Lagragian:

$$\mathscr{L}(u,\eta,\xi) = L_\varepsilon(u,\eta) + \xi(\sum_j \eta_j - 1)$$

$$\frac{\partial \mathscr{L}(u,\eta,\xi)}{\partial u} = -\delta + \left(\lambda \sum_{j=1}^{d_x} \frac{D_j(p_x p_y)}{\eta_j} + \rho C(p_x p_y) + \tau I_m\right) u = 0$$

and

$$\frac{\partial \mathscr{L}(u,\eta,\xi)}{\partial \eta_j} = -\frac{\lambda}{2}\frac{\langle u, D_j(p_x p_y)u\rangle + \varepsilon}{\eta_j^2} + \xi = 0$$

and

$$\frac{\partial \mathscr{L}(u,\eta,\xi)}{\partial \xi} = \sum_j \eta_j - 1 = 0$$

Hence:

$$u_\varepsilon^* = \left(\lambda \sum_{j=1}^{d_x} \frac{D_j(p_x p_y)}{\eta_j^*} + \rho C(p_x p_y) + \tau I_m\right)^{-1} (\mu(p_{xy}) - \mu(p_x p_y))$$

and :

$$\eta_{j,\varepsilon}^* = \frac{\sqrt{\langle u_\varepsilon^*, D_j(p_x p_y)u_\varepsilon^*\rangle + \varepsilon}}{\sum_{k=1}^{d_x}\sqrt{\langle u_\varepsilon^*, D_k(p_x p_y)u_\varepsilon^*\rangle + \varepsilon}}.$$

□

Proof of Theorem 2. The proof follows similar proof in Argryou 2008.

$$S_\varepsilon(u) = L(u_\varepsilon, \eta(u_\varepsilon)) = -\langle u,\delta\rangle + \frac{1}{2}\langle u, (\rho C(p_x p_y) + \tau I_m)u\rangle + \frac{\lambda}{2}\left(\sum_j \sqrt{\langle u, D_j(p_x p_y)u\rangle + \varepsilon}\right)^2$$

Let $\{(u_{\ell_n}, \eta_{\ell_n}(u_{\ell_n})), n \in \mathbb{N}\}$ be a limiting subsequence of minimizers of $L_{\varepsilon_{\ell_n}}(.,.)$ and let (u^*, η^*) be its limit as $n \to \infty$. From the definition of $S_\varepsilon(u)$, we see that $\min_u S_\varepsilon(u)$ decreases as ε decreases to zero, and admits a limit $\bar{S} = \min_u S_0(u)$. Hence $S_{\varepsilon_{\ell_n}} \to \bar{S}$. Note that $S_\varepsilon(u)$ is continuous in both ε and u and we have finally $S_0(u^*) = \bar{S}$, and u^* is a minimizer of S_0. □

Proof of Corollary 1 . The optimum $(u_\varepsilon^*, \eta_\varepsilon^*)$ satisfies:

$$-\delta + \left(\lambda \sum_{j=1}^{d_x} \frac{D_j(p_x p_y)}{\eta_j} + \rho C(p_x p_y) + \tau I_m\right) u_\varepsilon^* = 0$$

Let $f^*(x) = \langle u, \Phi_\omega(x,y)\rangle$ and define $||f^*_\varepsilon||_{\mathscr{F}} = \|u^*_\varepsilon\|_2$. It follows that $\eta^*_j = \frac{\sqrt{\mathbb{E}_{p_x p_y}\left|\frac{\partial f^*_\varepsilon(x,y)}{\partial x_j}\right|^2+\varepsilon}}{\sum_k \sqrt{\mathbb{E}_{p_x p_y}\left|\frac{\partial f^*_\varepsilon(x,y)}{\partial x_k}\right|^2+\varepsilon}}$

$$
\begin{aligned}
\text{Note that we have} \quad & \mathbb{E}_{p_{xy}} f^*_\varepsilon(x,y) - \mathbb{E}_{p_x p_y} f^*_\varepsilon(x,y) \\
= \quad & \langle \delta, u^*_\varepsilon \rangle \\
= \quad & \left\langle u^*_\varepsilon, \left(\lambda \sum_{j=1}^{d_x} \frac{D_j(p_x p_y)}{\eta^*_{j,\varepsilon}} + \rho C(p_x p_y) + \tau I_m \right) u^*_\varepsilon \right\rangle \\
= \quad & \lambda \left(\sum_{j=1}^{d_x} \sqrt{\mathbb{E}_{p_x p_y} |\frac{\partial f^*_\varepsilon(x,y)}{\partial x_j}|^2 + \varepsilon} \right)^2 + \rho \mathbb{E}_{p_x p_y} f^{*,2}_\varepsilon(x,y) + \tau ||f^*_\varepsilon||^2_{\mathscr{F}}
\end{aligned}
$$

$$
\begin{aligned}
SIC_{(L^1)^2,\varepsilon} \quad = \quad & \mathbb{E}_{p_{xy}} f^*_\varepsilon(x,y) - \mathbb{E}_{p_x p_y} f^*_\varepsilon(x,y) - \frac{1}{2}(\lambda \left(\sum_{j=1}^{d_x} \sqrt{\mathbb{E}_{p_x p_y} |\frac{\partial f^*_\varepsilon(x,y)}{\partial x_j}|^2 + \varepsilon} \right)^2 \\
+ \quad & \rho \mathbb{E}_{p_x p_y} f^{*,2}_\varepsilon(x,y) + \tau ||f^*_\varepsilon||^2_{\mathscr{F}}) \\
= \quad & \frac{\lambda}{2} \left(\sum_{j=1}^{d_x} \sqrt{\mathbb{E}_{p_x p_y} |\frac{\partial f^*_\varepsilon(x,y)}{\partial x_j}|^2 + \varepsilon} \right)^2 + \frac{\rho}{2} \mathbb{E}_{p_x p_y} f^{*,2}_\varepsilon(x,y) + \frac{\tau}{2} ||f^*_\varepsilon||^2_{\mathscr{F}} \\
= \quad & \frac{1}{2} \left(\mathbb{E}_{p_{xy}} f^*_\varepsilon(x,y) - \mathbb{E}_{p_x p_y} f^*_\varepsilon(x,y) \right)
\end{aligned}
$$

We conclude by taking $\varepsilon \to 0$. □

D FDR Control with HRT and Conditional Generative Models

The Holdout Randomization Test (HRT) is a principled method to produce valid p-values for each feature, that enables the control over the false discovery of a predictive model [8]. The p-value associated to each feature x_j essentially quantifies the result of a conditional independence test with the null hypothesis stating that x_j is independent of the output y, conditioned on all the remaining features $\mathbf{x}_{-j} = (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_p)$. This in practice requires the availability of an estimate of the complete conditional of each feature x_j, i.e. of $P(x_j|\mathbf{x}_{-j})$. HRT then samples the values of x_j from this conditional distribution to obtain the p-value associated to it. Taking inspiration from neural network models for conditional generation (see e.g. [42]) we train a neural network to act as a generator of a features x_j given the remaining features $\mathbf{x}_{-j}$ as inputs, as a replacement for the conditional distributions $P(x_j|\mathbf{x}_{-j})$. In all of our tasks, one three-layer neural network with 200 ReLU units and Conditional Batch Normalization (BCN) [43] applied to all hidden layers serves as generator for all features $j = 1, \ldots, p$. A sample from $P(x_j|\mathbf{x}_{-j})$ is generated by giving as input to the network an index j indicating the feature to generate, and a sample $\mathbf{x}_{-j} \sim P(\mathbf{x}_{-j})$, represented as a sample from the full joint distribution $\mathbf{x} \sim P(x_1, \ldots, x_p)$, with feature j being masked out. In practice, the index j and $\mathbf{x} \sim P(x_1, \ldots, x_p)$ are given as inputs to the generator, and the neural network model does the masking, and sends the index j to the CBN modules which normalize their inputs using j-dependent centering and normalization parameters. The output of the generator is a n_{bins}-dimensional softmax over bins tessellating the range of the distribution of x_j, such that the bins are uniform quantiles of the inverse CDF of the distribution of x_j estimated over the training set. In all simulations we used a number of bins $n_{bins} = 100$.

Generators are trained randomly sampling an index $j = 1, \ldots, p$ for each sample $\mathbf{x}$ in the training set, and minimizing the cross-entropy loss between the output of the generator neural network $Gen(j, \mathbf{x})$ and x_j using mini-batch SGD. In particular, we used the Adam optimizer [44] with the default pytorch [45] parameters and learning rate $\lambda = 0.003$ which is halved every 20 epochs, and batch size of 128.

E Discussion of SIC: Consistency, Computational Complexity and FDR Control

SIC consistency. In order to recover the correct conditional independence we elected to use FDR control techniques to perform those dependent hypotheses testing (btw coordinates). By combining SIC with HRT and knockoffs we can guarantee that the correct dependency is recovered while the FDR is under control. For the consistency of SIC in the classical sense, one needs to analyze the solution of SIC, when the critic is not constrained to belonging to an RKHS. This can be done by studying the solution of the equivalent PDE corresponding to this problem (which is challenging, but we think can also be managed through the η- trick). Then one would proceed by finding 1) conditions under which this solution exists in the RKHS, 2) generalization bounds from samples to the population solution in the RKHS. We leave this analysis for future work.

Computational Complexity of Neural SIC. The cost of training SIC with SGD and mirror descent has the same scaling in the size of the problem as training the base regressor neural network via back-propagation. The only additional overhead is the gradient penalty, where the cost is that of double back-propagation. In our experiments, this added computational cost is not an issue when training is performed on GPU.

SIC-HRT versus SIC-Knockoffs. For a comparison between HRT and knockoffs, we refer the reader to [8], which shows similar performance for either method in terms of controlling FDR. Each method has its advantages. In HRT most of the computation is in 1) training the generative models, and 2) performing the randomization test, i.e. forwarding the data through the critic and computing p-values for each coordinate for R runs. On the other hand, if knockoff features can be modelled as a multivariate Gaussian, controlling FDR with knockoffs can be done very cheaply, since it does not require randomization tests. If instead knockoff features have to be generated through nonlinear models, knockoffs can be computationally expensive as well (see for example [46]).

F Experimental details

F.1 Synthetic Datasets

F.1.1 Complex Multivariate Synthetic Dataset (SinExp)

The SinExp dataset is generated from a complex multivariate model proposed in [33] Sec 5.3, where 6 features x_i out of 50 generate the output y through a non-linearity involving the product and composition of the $\cos$, $\sin$ and $\exp$ functions, as follows:

$$y = \sin(x_1(x_1 + x_2)) \cos(x_3 + x_4 x_5) \sin(e^{x_5} + e^{x_6} - x_2).$$

We increase the difficulty even further by introducing a pairwise correlation between all features of 0.5. We perform experiments using datasets of 125 and 500 samples. For each sample size, 100 independent datasets are generated.

F.1.2 Liang Dataset

Liang Dataset is a variant of the synthetic dataset proposed by [34]. The dataset prescribes a regression model with 500-dimensional correlated input features x, where the 1-D regression target y depends on the first 40 features only (the last 460 correlated features are ignored). In the original dataset proposed by [34], y depends on 4 features only, this more complex version of the dataset that uses 40 features was proposed by [8]. The target y is computed as follows:

$$y = \sum_{j=0}^{9} [w_{4j} x_{4j} + w_{4j+1} x_{4j+1} + \tanh(w_{4j+2} x_{4j+2} + w_{4j+3} x_{4j+3})] + \sigma\epsilon\,, \tag{2}$$

with $\sigma = 0.5$ and $\epsilon \sim \mathcal{N}(0, 1)$. As in [8], the 500 features are generated to have 0.5 correlation coefficient with each other,

$$x_j = (\rho + z_j)/2\,, \quad j = 1, \ldots, 500\,, \tag{3}$$

where ρ and z_j are independently generated from $\mathcal{N}(0, 1)$.

Our experimental results are the average over 100 generated datasets, each consisting of 500 train and 500 heldout samples.

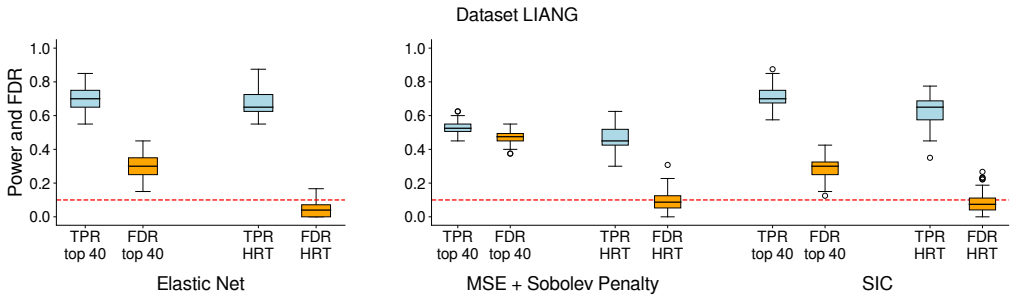

Figure 2: Liang synthetic dataset. TPR and FDR of Elastic Net baseline models (left panels) are compared against our methods, analogously to Fig. 1. Differently from Fig. 1, however, TPR and FDR are computed by selecting the top 40 features in order of importance (since this dataset was generated with 40 ground truth features). Moreover, HRT is used to select features among the top 100 most important features.

F.2 CCLE Dataset

The Cancer Cell Line Encyclopedia (CCLE) dataset [36] provides data about of anti-cancer drug response in cancer cell lines. The dataset contains the phenotypic response measured as the area under the dose-response curve (AUC) for a variety of drugs that were tested against hundreds of cell lines. [36] analyzed each cell to obtain gene mutation and expression features. The total number of data points (cells) is 479. We followed the preprocessing steps by [8] and first screened the genomic features to filter out features with less than 0.1 magnitude Pearson correlation to the AUC. This resulted in a final set of about 7K features. The main goal in this task is to discover the genomic features associated with drug response. Following [8], we perform experiments for the drug PLX4720. Table 3 presents the top-10 genomic features selected by SIC according to η_j values. In Sec. 8, we also present quantitative results that show the effectiveness of these features when used to train regression models.

	Genomic Feature	η_j
0	BRAF.V600E_MUT *	0.011837
1	ACKR3	0.011712
2	RP11-349I1.2	0.010534
3	BRAF_MUT †	0.010449
4	UBE2V1P5	0.010420
5	EPB41L3	0.010163
6	C11orf85 †	0.009622
7	RP11-395F4.1	0.009449
8	SERPINA9	0.009387
9	RN7SKP281	0.009369

Table 3: Top-10 Genomic Features selected by SIC according to η_j values. These are the most important features for high mutual information with PLX4720 response variable, on the CCLE dataset. * indicates feature also discovered by Elastic Net and HRT [8]. † indicates feature also discovered by Elastic Net in original CCLE paper [36].

F.3 SIC Neural Network descriptions and training details

The first critic network used in the experiments (with SinExp and HIV-1 datasets) is a standard three-layer ReLU dropout network with no biases, i.e. small_critic. When using this network, the

inputs X and Y are first concatenated then given as input to the network. The two first layers have size 100, while the last layer has size 1. We train the network using Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$, weight_decay=1e-4 learning rate α_η = 1e-3 and $\alpha_\eta = 0.1$, and perform 4000 training iterations/updates, computed with batches of size 100. All NNs used in our experiments were implemented using PyTorch [45].

```
small_critic(
  (branchxy): Sequential(
    (0): Linear(in_features=51, out_features=100, bias=False)
    (1): ReLU()
    (2): Dropout(p=0.3)
    (3): Linear(in_features=100, out_features=100, bias=False)
    (4): ReLU()
    (5): Dropout(p=0.3)
    (6): Linear(in_features=100, out_features=1, bias=False)
  )
)
```

The critic network used in the experiments with Liang and CCLE datasets contains two different branches that separately process the inputs X (*branchx*) and Y (*branchy*), then the output of these two branches are concatenated and processed by a final branch that contains three-layer LeakyReLU network (*branchxy*). We name this network *big_critic* (see figure bellow for details about layer sizes). This network is trained with the same Adam settings as above for 4000 updates (Liang) and 8000 updates (CCLE).

```
big_critic(
  (branchx): Sequential(
    (0): Linear(in_features=500, out_features=100, bias=True)
    (1): LeakyReLU(negative_slope=0.01)
    (2): Linear(in_features=100, out_features=100, bias=True)
    (3): LeakyReLU(negative_slope=0.01)
  )
  (branchy): Sequential(
    (0): Linear(in_features=1, out_features=100, bias=True)
    (1): LeakyReLU(negative_slope=0.01)
    (2): Linear(in_features=100, out_features=100, bias=True)
    (3): LeakyReLU(negative_slope=0.01)
  )
  (branchxy): Sequential(
    (0): Linear(in_features=200, out_features=100, bias=True)
    (1): LeakyReLU(negative_slope=0.01)
    (2): Linear(in_features=100, out_features=100, bias=True)
    (3): LeakyReLU(negative_slope=0.01)
    (4): Linear(in_features=100, out_features=1, bias=True)
  )
)
```

The regressor NN used for the downstream regression task in Section 8 is a standard three-layer ReLU dropout network. This regressor NN was trained with the same Adam settings as above for 1000 updates with a batchSize of 16. We did not perform any hyperparameter tuning or model selection on heldout MSE performance.

```
regressor_NN(
  (net): Sequential(
    (0): Linear(in_features=7251, out_features=100, bias=True)
    (1): ReLU()
    (2): Dropout(p=0.3)
    (3): Linear(in_features=100, out_features=100, bias=True)
    (4): ReLU()
    (5): Dropout(p=0.3)
```

```
    (6): Linear(in_features=100, out_features=1, bias=True)
  )
)
```