Discovering relevant input features for predicting a target variable is a key scientific question. However, in many domains, such as medicine and biology, feature selection is confounded by a scarcity of labeled samples coupled with significant correlations among features. In this paper, we propose a novel deep learning approach to feature selection that addresses both challenges simultaneously. First, we pre-train the network using unlabeled samples within a self-supervised learning framework by solving pretext tasks that require the network to learn informative representations from partial feature sets. Then, we fine-tune the pre-trained network to discover relevant features using labeled samples. During both training phases, we explicitly account for the correlation structure of the input features by generating correlated gate vectors from a multivariate Bernoulli distribution. Experiments on multiple real-world datasets including clinical and omics demonstrate that our model discovers relevant features that provide superior prediction performance compared to the state-of-the-art benchmarks in practical scenarios where there is often limited labeled data and high correlations among features.

1 INTRODUCTION

High-dimensional datasets are increasingly prevalent in a variety of fields, including critical domains such as medicine and biology. Discovering features responsible for the target variable is an essential but challenging step in the analysis of such data. For example, while next generation sequencing can detect the expression of tens of thousands of genes per sample (e.g., RNA-seq), many genetic disorders stem from the variation in only a few groups of related genes (Jackson et al., 2018). Identification of such disease-related factors is crucial for the design of therapeutic treatments. Furthermore, feature selection has additional benefits, including reduced experimental costs, greater interpretability, and improved model generalization (Min et al., 2014; Ribeiro et al., 2016). However, the high-dimensional and often noisy nature of such data prevents relevant features from being readily discovered. Moreover, in many domains there is a scarcity of label information due to cost, privacy reasons, or experimental study design. While deep learning approaches have resulted in improvements in feature selection (e.g., Yamada et al. [2020], Lemhdri et al. [2021], see Related Work), such methods typically assume access to many labeled samples. With low sample size, feature selection methods, in particular those employing deep learning, have shown a propensity to overfit high-dimensional data (Kuncheva et al., 2020).

This issue is further exacerbated by the inherent structure of such data. Many (high-dimensional) datasets exhibit substantial inter-correlations or multicollinearity – i.e., there exist features that are (highly) correlated among themselves – which impacts the performance of feature selection methods (Chong & Jun, 2005; Katrutsa & Strijov, 2017; Belsley et al., 2005). In particular, this structure can cause parameter estimates to be unstable (Dobson & Barnett, 2018) and, in the extreme, can prevent variable effects from being separated, confounding the problem of feature selection (Meloun et al., 2002). Existing deep learning-based methods for feature selection do not (explicitly) consider the correlations between features which will often result in the selection of redundant features.

*Equal contribution
When labeled samples are scarce, large numbers of unlabeled samples are often available (e.g., Perez-Riverol et al. [2019]). However, the importance of unlabeled data has been overlooked in feature selection despite its potential to prevent the model selection process from overfitting by allowing informative feature representations to be learned. Therefore, the development of methods for feature selection that can exploit both labeled and unlabeled data is of great practical importance.

Contributions. We propose a novel method for feature selection that addresses two key challenges in real-world scenarios: limited labeled data and correlated features. We use a self-supervised approach to train an encoder using unlabeled data via two pretext tasks: feature vector reconstruction and gate vector estimation. This pre-conditions the encoder to learn informative representations from partial feature sets, aligning the self-supervision with the model selection process of the downstream feature selection task. In addition, we introduce a novel gating procedure that accounts for the correlation structure of the input features. This ensures the pretext tasks remain challenging by preventing the model from memorizing trivial relations between features. Moreover, unlike previous deep learning-based feature selection methods, the correlated gate vectors encourage our method to select the most relevant features by making multiple correlated features compete against each other.

We validate our approach through experiments on synthetic and multiple real-world datasets, including clinical and omics, where only a small number of labeled samples are available. Our model discovers relevant features that provide superior prediction performance compared to state-of-the-art benchmarks, and we corroborate these features with supporting medical and scientific literature.

2 RELATED WORK

Feature Selection Methods. Feature selection is a well-studied problem, with a number of proposed solutions including wrapper (Kohavi & John, 1997) and filter (Liu & Setiono, 1996; Kira & Rendell, 1992) methods. Recent advances in deep learning provide an elegant way of training embedded feature selection methods by jointly learning to perform feature selection while training a prediction network (Huang et al., 2020). These methods learn to perform the non-differentiable process of selecting feature subsets by approximating it either with Lasso or elastic net penalization (Li et al., 2016), using an MCMC sampling approach (Liang et al., 2018), or more recently with continuous relaxation using independent Gaussian random variables (Yamada et al., 2020). However, supervised feature selection methods can fail to identify relevant features when limited labeled samples are available due to overfitting (Kuncheva et al., 2020), impacting their suitability in many real-world scenarios. Moreover, these methods do not consider the underlying correlation structure of the input features which can be problematic when selecting relevant features among (highly) correlated ones.

Relatively few feature selection methods use unlabeled samples. Abid et al. (2019) used auto-encoders to identify a pre-specified number of features that are sufficient for reconstructing the data. Lindenbaum et al. (2020) improved the well-known Laplacian score (He et al., 2005) by selecting feature subsets that better capture the “local structure” of the data. However, both approaches are fully unsupervised and, without the guidance of label information, can fail to identify features relevant to the target outcome. While a number of semi-supervised feature selection methods have been proposed (Sheikhpour et al., 2017), they have typically been extensions of traditional methods, such as a Laplacian score with modified affinity scores using labels (Zhao et al., 2008) or manifold regularization based on linear SVMs (Dai et al., 2013). To the best of our knowledge, this is the first deep learning framework that fully utilizes both labeled and unlabeled samples for feature selection.

Self-Supervised Learning. Self-supervised learning methods create (weak) supervised signals from unlabeled data, employing contrastive learning or pretext task(s) to provide surrogate labels. While self-supervised learning has found success in computer vision (Chen et al., 2020) and natural language processing (Devlin et al., 2018), the tabular domain, which is most relevant in feature selection, has been largely neglected. Methods for tabular data seek to reconstruct data either based on a corrupted sample alone (Vincent et al., 2008; Arik & Pfister, 2019; Yin et al., 2020) or with knowledge of which entries have been corrupted (Pathak et al., 2016). Recently, Yoon et al. (2020) jointly learn to recover the original sample and predict the mask vector used to corrupt the data.

While our pretext tasks are also reconstruction-based, we propose a novel method for generating the gate vector used to produce the input feature vector. This is particularly important when there exists substantial correlation between features (see Section 5). All existing methods have used indepen-
dicting the target as part of the model selection process. Denote ∗
Embedded feature selection aims to select a subset, \( \mathcal{S} \),
case letters to denote realizations of random variables.
\( x \)
features (e.g., gene expressions) and the target outcome (e.g., disease traits) whose realizations are
one-hot encoding of
\( Y \)
classification tasks (i.e.,
where
\( \delta \)
following optimization problem:
\[ \minimize_{f \in \mathcal{F}, \mathcal{S} \subseteq [p]} \mathbb{E}_{x \sim p_X, y \sim p_{XY}} \left[ \ell_Y (y, f(x_S)) \right] \text{ subject to } |\mathcal{S}| \leq \delta \] (1)
where \( \delta \) constrains the number of selected features and \( \ell_Y (y, y') = -\sum_{c=1}^{C} y_c \log y_{c}' \) for \( C \)-way classification tasks (i.e., \( \mathcal{Y} = \{1, \cdots, C\} \) and \( \ell_Y (y, y') = \|y - y'\|_2^2 \) for regression tasks (i.e., \( \mathcal{Y} = \mathbb{R}^m \)). Unfortunately, the combinatorial problem in (1) becomes intractable for high-dimensional data as the search space increases exponentially with \( p \). Hence, we instead focus on a relaxation by converting the combinatorial search in (1) into a search over the space of possibly correlated binary random variables. It is worth highlighting that unlike existing work (e.g., Yamada et al. (2020); Yoon et al. (2019)), we do not assume independence among these random variables.

Let \( M = (M_1, \cdots, M_p) \in \{0, 1\}^p \) be binary random variables governed by distribution \( p_M \), whose realization \( m \) is referred to as the gate vector, for indicating selection of the corresponding features. Then, the selected features given gate vector \( m \) can be written as
\[ \tilde{x} \triangleq m \odot x + (1 - m) \odot \bar{x} \] (2)
where \( \bar{x} = \mathbb{E}_{x \sim p_X} [x] \) and \( \odot \) indicates element-wise multiplication. Here, we replace not-selected features with their means to resolve any issue when a feature having zero value (e.g., turned-off genes) has specific meaning. Finally, we can approximately achieve (1) by jointly learning the model selection \( f \) and the gate vector distribution \( p_M \) based on the following optimization problem:
\[ \minimize_{f, p_M} \mathbb{E}_{x \sim p_X, y \sim p_{XY}} \mathbb{E}_{m \sim p_M} \left[ \ell_Y (y, f(\tilde{x})) + \beta \|m\|_0 \right] \] (3)
where \( f \) is implemented as a neural network and \( \beta \) is a balancing coefficient that controls the number of features to be selected.

**Challenges.** In practice (especially in the healthcare domain), there are two main challenges that confound selecting relevant features for predicting the target: (i) inter-correlation or multicollinearity, which is the existence of features that are (highly) correlated among themselves, and (ii) the absence of sufficient labeled samples. These challenges make embedded feature selection vulnerable to overfitting. More specifically, model selection (i.e., learning \( f \)) and feature selection (i.e., learning \( p_M \)) are conducted jointly. Either one being overfit to correlated or noisy irrelevant features will end up discovering spurious relations and poor prediction. For instance, if the network is biased toward irrelevant features, the selection probability (i.e., importance) of those features will be increased as if those features were "discriminative" or "predictive", and vice versa.

\(^1\)To enable neural networks to be trained with varying feature subsets, we set \( * = \bar{x} \).

\(^2\)For \( \mathcal{Y} = \{1, \cdots, C\} \), we will occasionally abuse notation and write \( y_c \) to denote the \( c \)-th element of the one-hot encoding of \( y \) when clear in the context.
cumulative distribution function (CDF) of random variables to construct a valid multivariate Bernoulli distribution. The Gaussian copula is a multivariate distribution where dependencies among gates are transferred from the correlation structure of the input features. To this goal, we use the Gaussian copula instead of assuming independence in the gating mechanism. We model the gate vector distribution by the correlation structure of the input features. To this goal, we use the Gaussian copula to model the gate vector distribution.

Instead of assuming independence in the gating mechanism, we model the gate vector distribution as a multivariate Bernoulli distribution where dependencies among gates are determined by the correlation structure of the input features. This is achieved by using a Gaussian copula to construct a valid multivariate Bernoulli distribution. The Gaussian copula is a multivariate cumulative distribution function (CDF) of random variables \((U_1, \ldots, U_p)\) over the unit cube \([0, 1]^p\) with uniform marginals, where \(U_k \sim \text{Uniform}(0, 1)\) for \(k \in [p]\). Formally, given the correlation matrix \(R \in [-1, 1]^p\) that captures the correlation structure of \(X\), the Gaussian copula is defined as:

\[
C_R(U_1, \ldots, U_p) = \Phi_R(\Phi^{-1}(U_1), \ldots, \Phi^{-1}(U_p))
\]

where \(\Phi_R\) stands for the joint CDF of a multivariate Gaussian distribution with mean zero vector and correlation matrix \(R\), and \(\Phi^{-1}\) is the inverse CDF of the standard univariate Gaussian distribution.

Define \(\pi = (\pi_1, \ldots, \pi_p) \in [0, 1]^p\) to be the selection probability that governs the multivariate Bernoulli distribution used to generate the gate vector \(m\). Then, given \(C_R\), we can generate the gate vector from a multivariate Bernoulli distribution, which is denoted as \(m \sim \text{MultiBern}(\pi; R)\), based on the correlated random variables \((U_1, \ldots, U_p)\) that preserve the correlation structure of the input features; formally, \(m_k = 1\) if \(U_k \leq \pi_k\) and \(m_k = 0\) if \(U_k > \pi_k\) for \(k \in [p]\).

Using the correlation structure of the input features when generating the gate vector has the following two advantages: (i) during the Self-Supervision Phase, correlated gate vectors encourage the network not to rely on trivial signals from highly correlated features when solving the pretext tasks, which may lead to inefficient pre-training and sub-optimal downstream performance; (ii) during the Supervision Phase, correlated gate vectors encourage the network to select only the most rele-
vant features by increasing the chance that correlated features are selected jointly and thus compete against each other.

4.2 Self-Supervision Phase

Self-supervised learning uses unlabeled samples to automatically generate a (weak) supervisory signal by solving relevant pretext tasks in the absence of labeled data. The learned representations can then be used to solve downstream tasks. To this goal, we begin by defining two pretext tasks motivated by recent success in self-supervised learning for tabular data (Yoon et al., 2020). The pretext tasks to be jointly solved are: (i) reconstructing the original input $x$ based on a randomly selected subset of input features $\tilde{x}$, and (ii) simultaneously estimating the gate vector $m$ that defined which features were selected and which features were not (i.e., replaced by their mean value).

To solve these pretext tasks, we formally define the encoder $f$ which features were selected and which features were not (i.e., replaced by their mean value).

To account for the correlations between features, we generate gate vectors using a multivariate Bernoulli distribution. This is important since the correlated gate vectors encourage the network to select only the most relevant features by making multiple correlated features compete against each other.

4.3 Supervision Phase

During the Supervision Phase, the encoder, predictor, and the selection probability, which we refer to as the feature selection network, are jointly updated to solve the feature selection objective \[ (5) \] . The predictor is formally defined as follows:

- **Predictor**, $f_\theta : \mathcal{Z} \rightarrow \mathcal{Y}$, that takes as input the latent representations of the selected subset of features, i.e., $z = f_\theta(\tilde{x})$, and outputs predictions on the target outcome. Together with the encoder, the model selection function is given by $f = f_\theta \circ f_\theta$. To account for the correlations between features, we generate gate vectors using a multivariate Bernoulli distribution. This is important since the correlated gate vectors encourage the network to select only the most relevant features by making multiple correlated features compete against
each other. However, solving (3) directly using such gate vectors is intractable since $p_M(m)$ has no differentiable closed-form expression. Instead, we adopt a continuous relaxation of the multivariate Bernoulli distribution (Wang & Yin, 2020) by applying the reparameterization trick to the correlated uniform random variables $(U_1, \ldots, U_p)$ that preserve the correlation structure $R$.

Formally, given selection probability $\pi = (\pi_1, \ldots, \pi_p)$ and the multivariate uniform random variables $(U_1, \ldots, U_p)$ from Gaussian copula $C_R$, we can generate relaxed gate vector $\hat{m} = (\hat{m}_1, \ldots, \hat{m}_p) \in (0, 1)^p$ based on the following reparameterization trick (Wang & Yin, 2020):

$$\hat{m}_k = \sigma \left( \frac{1}{\tau} \left( \log \pi_k - \log(1 - \pi_k) + \log U_k - \log(1 - U_k) \right) \right)$$

where $\sigma(x) = (1 + \exp(-x))^{-1}$ is the sigmoid function. Such a relaxation is parameterized by $\pi$, a pre-specified covariance matrix $R$, and a temperature hyper-parameter $\tau \in (0, \infty)$. Similar to the Relaxed Bernoulli distribution (Maddison et al., 2017), (6) is differentiable with respect to $\pi$.

In practice, the gate vector generation process in (6) can be handled as a deterministic transformation of samples from a standard multivariate Gaussian random variable. More specifically, we first draw samples from the multivariate Gaussian distribution $v \sim \mathcal{N}(0, R)$ by $v = (v_1, \ldots, v_p) = L \epsilon$ where $\epsilon \sim \mathcal{N}(0, 1)$ and $L$ is a lower triangular matrix with positive diagonal entries, which is derived by the Cholesky factorization of the covariance matrix, i.e., $R = LL^T$. Next, we generate correlated uniform random variables $u_k = \Phi(v_k)$ for $k \in [p]$. Finally, we can generate a relaxed multivariate Bernoulli variable $\hat{m}_k \in (0, 1)$ by plugging in $u_k$ and $\pi_k$ into (6).

Under the continuous relaxation, the regularization term in (3) that induces sparsity of selected features can be simply derived as $\mathbb{E}_{m \sim p_M} \|m\|_0 = \sum_{k=1}^p P(U_k \leq \pi_k) = \sum_{k=1}^p \pi_k$. Overall, we can rewrite our objective as

$$\min_{\theta, \phi, \pi} \quad \mathbb{E}_{x, y \sim p_{XY}, \epsilon \sim \mathcal{N}(0, 1)} \left[ \ell_Y(y, f_{\phi} \circ f_{\theta} (\hat{m} \odot x + (1 - \hat{m}) \odot x)) + \beta \sum_{k=1}^p \pi_k \right].$$

The model selection function $f = f_{\phi} \circ f_{\theta}$ and the selection probability $\pi$ are updated jointly via gradient descent. Pseudo-code of SEFS can be found in Appendix A.

## Experiments

In this section, we evaluate the performance of SEFS and multiple feature selection methods using a synthetic and several real-world datasets. Further experiments together with detailed information regarding all experiments, benchmarks, and datasets can be found in the Appendix.

**Benchmarks.** We compare SEFS with 7 feature selection methods including 3 conventional methods and 4 state-of-the-art methods. The conventional methods include LASSO regularized linear/logistic model (Lasso, Tibshirani, 1996), extremely randomized tree ensembles (Tree, Geurts et al., 2006), and Laplacian Score that preserves locality structure (L-Score, He et al., 2005)). State-of-the-art deep learning methods include Bayesian neural networks for feature selection (BNNsel, Liang et al., 2018)), feature selection based on continuous relaxation (STG, Yamada et al., 2020), unsupervised feature selection based on gated Laplacian (DUFs, Lindenbaum et al., 2020). We also include an extension of STG to the semi-supervised setting by augmenting the loss with a reconstruction task that estimates the original features from the gated inputs (STG (SS)). Among the benchmarks, three are able to use both labeled and unlabeled samples: DUFs is fully unsupervised, while both L-Score and STG (SS) are semi-supervised approaches that explicitly use the label.

To highlight our novel two-step training process, we also include a variant of our method that is trained without employing the Self-Supervision Phase, which is denoted as SEFS (no SS).

**Performance Metrics.** For the synthetic experiments, we use the true positive rate (TPR) to assess whether discovered features are truly relevant compared to the ground-truth. With real-world data, however, it is not possible to calculate the TPR since the ground truth relevance is often not known. Hence, we instead evaluate the prediction performance of the discovered features obtained by different feature selection methods. For classification tasks, we use the area under the receiver operating characteristic curve (AUROC) and area under the precision recall curve (AUPRC), and for regression tasks, mean squared error (MSE). This is an indirect way of assessing the relevance of the
discovered features on the target tasks: we first conduct feature selection based on each method and then train a multi-layer perceptron (MLP) to perform predictions on top of the feature-selected data. Training a separate MLP isolates the effect of having different model selections – such as linear models for Lasso, ensemble trees for Tree, and deep neural networks for BNNsel, STG and SEFS – and thereby provides a fair comparison of the discovered features.

**Experimental Setup.** For each dataset, we divide the available samples into labeled and unlabeled. Labeled samples are randomly split into training and testing sets. Within the training set, \( n_l \) labeled samples from the training set are used to train the feature selection methods and the entire labeled samples are used to evaluate the discovered features. As a result, while features are selected based on limited sample sizes, a significantly larger number of labeled samples are used to validate the discovered features. Details of the data construction process is given in Appendix [C].

**Implementation.** Implementation details, including hyper-parameters and the construction of covariance matrix \( R \), together with details for all benchmarks can be found in Appendix [B].

### 5.1 Synthetic: Block-Structured Noisy Two-Moons Dataset

To motivate and confirm our intuition of applying self-supervision and correlated gate vectors in feature selection, we begin by evaluating SEFS via a set of synthetic experiments that addresses two key challenges – i.e., limited labeled data and correlated features – in real-world scenarios.

**Dataset Description.** We consider the well-known 2-dimensional “Two-Moons” dataset with additive noise with variance \( \sigma^2 = 0.1 \). The outcome label (i.e., \( Y = \{0, 1\} \)) is generated solely based on these 2 features (i.e., \( x_1 \) and \( x_2 \)). We augment 8 auxiliary feature dimensions that are irrelevant to target outcomes by sampling from the standard Normal distribution. Finally, we introduce a correlated block-structure by augmenting each feature with 9 highly-correlated features (correlation: 0.94) by injecting small white noise (i.e., \( \mathcal{N}(0, 0.3^2) \)), yielding 100 features in total.

**Quantitative Analysis.** In Figure 3(a), we compare the average TPR with varying numbers of labeled samples \( n_l = \{10, 20, 40, 60, 80\} \) while fixing the number of unlabeled samples \( n_u = 1000 \). We define \( x_1 \) (or \( x_2 \)) as correctly discovered if and only if \( x_1 \) (or \( x_2 \)) has the first or second highest feature importance. SEFS benefits from learning the data structure using unlabeled samples and provides significant gain over all baselines (see Appendix [D] for a full comparison). The majority of benchmarks struggle to identify \( x_1 \) in particular due to \( x_2 \) being more discriminative than \( x_1 \), and therefore noisy features that are correlated with \( x_2 \) are often selected (Table S.4). We highlight the benefit of the correlated gating procedure by comparing SEFS (no SS) and SEFS (no SS, indep), a variant that uses the independent Bernoulli distribution to generate gate vectors. Our novel gating procedure significantly improves the TPR even in the fully supervised setting (Figure 3(a)).

We further investigate the utility of the Self-Supervision Phase in SEFS. To this goal, we introduce two counterparts that replace our self-supervised training with (i) a conventional auto-encoder (SEFS (AE)) and (ii) a variant of our model with an uncorrelated gating procedure using independent Bernoulli random variables (SEFS (indep)). In all cases, we employ the same feature selection process as SEFS to isolate the impact of self-supervision. In Figure 3(b), we evaluate these variants in terms of the TPR with a varying number of unlabeled samples \( n_u = 200, 500, 1000, 2000, \) and 4000 while fixing the number of labeled samples \( n_l = 10 \). (i.e., 5 for each label). Our method significantly outperforms the other variants, improving the TPR over SEFS (no SS) even with a relative

![Figure 3: Results on the Block-Structured Noisy Two-Moons dataset.](image)
small number of unlabeled samples. Moreover, we further evaluate the quality of the learned representation by assessing the discriminative performance of the learned representations given only the ground-truth features (i.e., setting $m_1 = m_2 = 1$ and $m_k = 0$ for $k \notin \{1, 2\}$). As shown in Table 1, the proposed gating procedure in SEFS helps the encoder to learn meaningful structure and results in the most performant representations, further validating the self-supervised training procedure.

5.2 CLINICAL: UKCF DATASET

**Dataset Description.** The UK Cystic Fibrosis registry (UKCF) records annual follow-ups for 6,754 adults over the period 2008–2015. We include patients with observations in at least three consecutive years. Each patient is associated with $p=245$ clinical variables (11 static and $3 \times 78$ time-varying features), including demographics, genetic mutations, clinical tests, and therapeutic management. We set respiratory failure (death or lung transplant) in the next 5 years as the label.

**Quantitative Results.** In Table 2, we compare the prediction performance of feature selection methods with $n_l = 32$, $n_u = 4754$ and $|S| = 10$. As expected, SEFS benefits from learning the data structure utilizing the unlabeled samples and discovers features that provide significantly greater discriminative performance over all other methods (Table 2). Fully supervised feature selection methods struggle to discover relevant features with only a few labeled samples. The benefit of self-supervision can be clearly seen by comparing SEFS with SEFS (no SS). Our results also highlight that fully unsupervised feature selection methods (i.e., L-Score and DUFS) struggle to discover relevant features as these methods cannot select features that effectively distinguish the target outcomes without the guidance of label information.

**Qualitative Results.** Two interesting features identified by SEFS but that were not discovered by other methods were PI Allele 1 and 2 that are related to pancreatic functions. More specifically, pancreatic function studies have demonstrated that (types of) mutations in the CFTR gene, which cause CF, are highly associated with dysfunction in organ systems and the pathology of CF patients (Sosnay et al., 2013; Gibson-Corley et al., 2016). Further details of the most frequently discovered features and supporting evidence of their clinical relevance can be found in Appendix D.2.

5.3 PROTEOMICS: CCLE DATASET

**Dataset Description.** We study the response of heterogeneous cancer cell lines to 11 different drugs where the goal is to identify proteins associated with the cell line response based on proteomic measurements from the Cancer Cell Line Encyclopedia (CCLE, Barretina et al., 2012). CCLE is a small dataset containing 899 cancer cell lines (i.e., samples) described by $p=196$ protein expressions. The real-valued drug response (i.e., $Y \in \mathbb{R}$) is available for 458 samples and is missing for the remaining 441 samples (thus unlabeled). To benefit from self-supervised learning, we integrate the RPPA measurements on 7,329 samples from The Cancer Genome Atlas (TCGA), creating overall 7,770 unlabeled samples (i.e., $n_u = 7770$). While the two datasets are not generated from the same distribution, we expect some benefit from learning basic correlations among features, even if not all correlations from CCLE are preserved.

**Quantitative Results.** In Figure 4, we compare the ranking of SEFS and the benchmarks across 11 drugs ($n_l = 20$, $n_u = 7770$, $|S| = 10$). SEFS is the best performing method for 9 drugs and is always in the top 3 (median rank: 1, average rank: 1.36). Despite the majority of unlabeled data originating from a different source, SEFS outperforms SEFS (no SS) in every experiment. While

---

Table 2: AUROC and AUPRC (mean ± 95%-CI) for $|S| = 10$ discovered features for the UKCF dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>AUROC</th>
<th>AUPRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>0.767±0.054</td>
<td>0.409±0.083</td>
</tr>
<tr>
<td>Tree</td>
<td>0.807±0.036</td>
<td>0.463±0.069</td>
</tr>
<tr>
<td>BNNSel</td>
<td>0.650±0.051</td>
<td>0.269±0.069</td>
</tr>
<tr>
<td>STG</td>
<td>0.781±0.048</td>
<td>0.440±0.082</td>
</tr>
<tr>
<td>DUFS</td>
<td>0.799±0.039</td>
<td>0.398±0.062</td>
</tr>
<tr>
<td>L-Score</td>
<td>0.668±0.010</td>
<td>0.213±0.017</td>
</tr>
<tr>
<td>STG (SS)</td>
<td>0.810±0.036</td>
<td>0.477±0.043</td>
</tr>
<tr>
<td>SEFS (no SS)</td>
<td>0.785±0.044</td>
<td>0.412±0.083</td>
</tr>
<tr>
<td>SEFS</td>
<td>0.846±0.013</td>
<td>0.532±0.027</td>
</tr>
</tbody>
</table>

Table 3: Frequently discovered features by SEFS for Panobinosstat (CCLE). Features in blue are not chosen by SEFS (no SS). Supporting references provided in Appendix D.3.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Proteins</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Caveolin-1</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>YAP-pS127</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>PRAS40-pT246</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>VHL</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>Sre-pY416</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>TAZ</td>
<td>✓</td>
</tr>
<tr>
<td>7</td>
<td>14-3-3-β</td>
<td>✓</td>
</tr>
<tr>
<td>8</td>
<td>Fibronectin</td>
<td>✓</td>
</tr>
<tr>
<td>9</td>
<td>GSK3-pS9</td>
<td>✓</td>
</tr>
<tr>
<td>10</td>
<td>MSH2</td>
<td>✓</td>
</tr>
</tbody>
</table>
we would expect further gain from more similar unlabeled data, our results highlight that there is
potential benefit even when unlabeled data is only partially related to the labeled samples.

Qualitative Results. Now we have demonstrated the superior predictive performance of the features
discovered using SEFS, we seek to validate these features in the scientific literature. Here, we
focus our analysis on panobinostat, a histone deacetylase (HDAC) inhibitor used in the treatment of
several cancers. We found strong supporting evidence for 9 of the top 10 ranked features selected by
SEFS (Table 3, see Appendix D.3 for full details). In addition, the remaining feature (fibronectin)
is strongly implicated in cancer, but we did not find literature specifically relating it to panobinostat or
HDAC-mediated pathways. We provide supporting evidence for other drugs in Appendix D.3.

5.4 TRANSCRIPTOMICS: PBMC DATASET

Dataset Description. We focus on distinguishing sub-populations of T-cells (i.e., \( \mathcal{Y} = \{0, 1\} \)),
namely naive and regulatory T-cells, from purified populations of peripheral blood monocytes
(PBMCs)\(^5\) based on transcriptomic measurements (i.e., mRNA sequence). The PBMC dataset con-
sists of 20,742 samples described by \( p = 19256 \) protein-coding genes. We primarily used
\( n_l = 20 \), \( n_u = 15557 \), with the remaining 5,165 labeled samples used to evaluate the discovered features.

Quantitative and Qualitative Results. Similarly to the performance on other datasets, SEFS sig-
nificantly outperforms all benchmarks (Table 4). Figure 5 displays only features selected with a
frequency \( \geq 0.4 \). It is immediately apparent that only SEFS consistently identifies the same fea-
tures, while Lasso and Tree typically select different features, demonstrating the robustness of our
approach. We provide supporting evidence for the importance of these features in Appendix D.4.

6 CONCLUSION

In this paper, we proposed SEFS, a self-supervised feature selection framework that is able to lever-
age the abundant quantities of unlabeled data that are often available, achieving state-of-the-art
performance when limited labeled data is available. Using synthetic data, we motivate and confirm
our intuition as to why self-supervision and our novel gating procedure can benefit feature selec-
tion methods. Through experiments on multiple real-world datasets, we validate that our model
disCOVERS features that provide superior prediction performance and corroborate the vast majority of
the discovered features with supporting medical and scientific literature. As with all feature selec-
tion methods, the discovered features based on SEFS should be further experimentally verified or
evaluated by a domain expert prior to deployment in practice.

\(^5\)https://support.10xgenomics.com/single-cell-gene-expression/datasets
ACKNOWLEDGMENTS

We thank anonymous reviewers as well as members of the vanderschaar-lab for many insightful comments and suggestions. CL was supported through the IITP grant funded by the Korea government (MSIT) (No. 2021-0-01341, AI Graduate School Program, CAU). FI and MvdS are supported by the National Science Foundation (NSF), grant number 1722516. MvdS is additionally supported by the Office of Naval Research (ONR).

ETHICS STATEMENT

Like all methods for selecting features or providing the importance of features from observational data, SEFS relies on an assumption (i.e., the important features subset should be enough to provide good prediction power). Consequently, all identified features should undergo additional evaluation or verification by domain experts prior to deployment in practice. In this work, we highlight direct/indirect impact of our method through experiments on real-world clinical and omics datasets including the UKCF, CCLE, and PBMC datasets. The use of these datasets was in accordance with the guidance of the respective data providers and domain experts.

REPRODUCIBILITY STATEMENT

The source code of SEFS and the synthetic dataset are provided in the Supplementary Material and is also available at [https://github.com/chl8856/SEFS](https://github.com/chl8856/SEFS). All details of our method and the experimental setup are included in either the main manuscript or the Appendix: a description of the method can be found in Section 4, pseudo-code for SEFS is described in Appendix A, Appendix B provides implementation details for both SEFS and the benchmarks, Appendix C provides details about the real-world datasets.

REFERENCES


A PSEUDO-CODE OF SEFS

SEFS is trained via a two-step training procedure. We provide pseudo-codes for the Self-Supervision Phase in Algorithm 1 and for the Supervision Phase in Algorithm 2.

The source code for SEFS is available in [https://github.com/chl8856/SEFS](https://github.com/chl8856/SEFS).

Algorithm 1 Pseudo-code for the Self-Supervision Phase of SEFS

**Input**: Dataset $D_u = \{x^i\}_{i=1}^{n_u}$, coefficient $\alpha$, selection probability $\pi$, mini-batch size $n_{mb}$, learning rate $\eta$

**Output**: SEFS parameter $\theta$

Initialize parameters $(\theta, \psi_1, \psi_2)$

Compute the mean value: $\bar{x} = \frac{1}{n_u} \sum_{i=1}^{n_u} x^i$

Compute the correlation matrix: $R$ where $R_{kj} = \frac{|C_{kj}|}{\sqrt{C_{kk} \cdot C_{jj}}}$ and $C_{kj} = \sum_{i=1}^{n_u} (x^i_k - \bar{x}_k)(x^i_j - \bar{x}_j)$

repeat

Sample a mini-batch of $n_{mb}$ unlabeled data samples: $\{x^i\}_{i=1}^{n_{mb}} \sim D_u$

for $i = 1, \ldots, n_{mb}$ do

Sample gate vector: $m^i \sim \text{MultiBern}(\pi; R)$

Generate feature subset: $\tilde{x}^i = m \odot x^i + (1 - m) \odot \bar{x}$

Estimate feature vector: $\hat{x}^i = h_{\psi_1} \circ f_\theta(\tilde{x}^i)$

Estimate gate vector: $\hat{m}^i = h_{\psi_2} \circ f_\theta(\hat{x}^i)$

end for

Update the encoder parameter $\theta$:

$$\theta \leftarrow \theta - \eta \nabla_{\theta} \left( \frac{1}{n_{mb}} \sum_{i=1}^{n_{mb}} \ell_X(x^i, \hat{x}^i) + \alpha \cdot \ell_M(m^i, \hat{m}^i) \right)$$

Update the feature vector estimator parameter $\psi_1$:

$$\psi_1 \leftarrow \psi_1 - \eta \nabla_{\psi_1} \left( \frac{1}{n_{mb}} \sum_{i=1}^{n_{mb}} \ell_X(x^i, \hat{x}^i) \right)$$

Update the gate vector estimator parameter $\psi_2$:

$$\psi_2 \leftarrow \psi_2 - \eta \nabla_{\psi_2} \left( \frac{1}{n_{mb}} \sum_{i=1}^{n_{mb}} \alpha \cdot \ell_M(m^i, \hat{m}^i) \right)$$

until convergence

B IMPLEMENTATION DETAILS

The hyper-parameters of SEFS and those of the benchmarks are chosen via a grid search. For all methods, we use 20% of the overall training set as the validation set, which will then be unseen for training feature selection methods with chosen hyper-parameters.

Once feature selection methods are trained utilizing a small subset of the overall training set as illustrated in Figure S.1, we train MLPs (with 3 layers and 100 nodes) with feature-selected data. As described in Section 5, training a separate MLP isolates the effect of having different model selections and thereby provides a fair comparison among the discovered features.

It is worth highlighting that generating correlated gate vectors can be computationally burdensome for high-dimensional data due to the matrix multiplication in Algorithm 3 and 4. To avoid such an issue, we first apply thresholding (e.g., 0.7 for the PBMC dataset) to the correlation matrix $R$, group...
Algorithm 2 Pseudo-code for the Supervision Phase of SEFS

**Input:** Dataset $D_l = \{x^i, y^i\}_{i=1}^{n_l}$, pre-trained encoder $\theta$, coefficient $\beta$, mini-batch size $n_{mb}$, learning rate $\eta$

**Output:** SEFS parameters $(\theta, \phi, \pi)$

Initialize parameters $(\phi, \pi)$

Compute the mean value: $\bar{x} = \frac{1}{n_u} \sum_{i=1}^{n_u} x^i$

Compute the correlation matrix: $R_kj = \frac{|C_kj|}{\sqrt{C_{kk} \cdot C_{jj}}} \quad \text{and} \quad C_kj = \sum_{i=1}^{n_u} (x^i_k - \bar{x}_k)(x^i_j - \bar{x}_j)$

repeat

Sample a mini-batch of $n_{mb}$ labeled data samples: $\{x^i, y^i\}_{i=1}^{n_{mb}} \sim D_l$

for $i = 1, \cdots, n_{mb}$ do

Sample gate vector: $\tilde{m}^i \sim \text{Relaxed-MultiBern}(\pi; R)$

Generate feature subset: $\tilde{x}^i \leftarrow \tilde{m} \odot x^i + (1 - \tilde{m}) \odot \bar{x}$

Predict outcome given the feature subset: $\hat{y}^i \leftarrow f_\phi \circ f_\theta(\tilde{x}^i)$

end for

Fine-tune the encoder parameter $\theta$:

$$\theta \leftarrow \theta - \eta \nabla_\theta \left( \frac{1}{n_{mb}} \sum_{i=1}^{n_{mb}} \ell_Y(y^i, \hat{y}^i) \right)$$

Update the predictor parameter $\phi$:

$$\phi \leftarrow \phi - \eta \nabla_\phi \left( \frac{1}{n_{mb}} \sum_{i=1}^{n_{mb}} \ell_Y(y^i, \hat{y}^i) \right)$$

Update the selection probability parameter $\pi$:

$$\pi \leftarrow \pi - \eta \nabla_\pi \left( \frac{1}{n_{mb}} \sum_{i=1}^{n_{mb}} \ell_Y(y^i, \hat{y}^i) + \beta \sum_{k=1}^{p} \pi_k \right)$$

until convergence

Algorithm 3 Pseudo-code for MultiBern($\pi; R$)

**Input:** selection probability $\pi$, correlation matrix $R$

**Output:** correlated gate vector $m$

Draw a standard normal sample: $\epsilon \sim \mathcal{N}(0, I)$

Compute $L = \text{Cholesky-Decomposition}(R)$

Generate a multivariate Gaussian vector $v = L\epsilon$

for $k = 1, \cdots, p$ do

Apply element-wise Gaussian CDF: $u_k = \Phi(v_k)$

Generate correlated gate: $m_k = \mathbb{1}(u_k \leq \pi_k)$

end for

features based on the agglomerative clustering\footnote{Implemented using Python package scikit-learn} using the correlation matrix as the similarity measure, and then conduct block-wise matrix multiplication. Overall, the generated gates for features within the same group will maintain the correlation structure. This significantly reduces the computational complexity in Algorithm 3 and 4 as it scales quadratically with respect to the largest block size (i.e., the number of features grouped in the largest block). Thus, if the largest block size remains the same, the complexity of generating the correlated gate vectors will only increase linearly with the feature dimension (since the number of blocks would increase linearly).
Algorithm 4 Pseudo-code for Relaxed-MultiBern($\pi; R$) (Wang & Yin, 2020)

**Input:** selection probability $\pi$, correlation matrix $R$, temperature $\tau$

**Output:** correlated gate vector $\tilde{m}$

Draw a standard normal sample: $\epsilon \sim N(0, I)$

Compute $L = \text{Cholesky-Decomposition}(R)$

Generate a multivariate Gaussian vector $v = L\epsilon$

for $k = 1, \ldots, p$

Apply element-wise Gaussian CDF: $u_k = \Phi(v_k)$

Apply reparameterization trick (Maddison et al., 2017):

$$\tilde{m}_k = \sigma \left( \frac{1}{\tau} \left( \log \pi_k - \log (1 - \pi_k) + \log u_k - \log (1 - u_k) \right) \right)$$

where $\sigma(x) = \frac{1}{1 + \exp(-x)}$.

end for

Throughout the experiments, training SEFS takes approximately 30 minutes to 1 hour for each phase (similar to that of STG) on a single GPU machine.

B.1 SEFS

The overall training process of SEFS consists of 4 different network components that are implemented as neural networks: (i) encoder $f_{\theta}$, (ii) predictor $f_{\phi}$, (iii) feature vector estimator $h_{\phi_1}$, and (iv) gate vector estimator $h_{\phi_2}$. We use a fully-connected network as the baseline architecture for all the network components.

For the Semi-Supervision Phase, coefficients ($\alpha$, $\pi$) and the hyper-parameters – including the number of hidden units, the number of nodes, and the number of layers – of the encoder, the feature vector estimator, and the gate vector estimator are chosen utilizing a grid search. We choose hyper-parameter values that achieve the lowest validation loss.

For the Supervision Phase, coefficient $\beta$ and the hyper-parameters – including the number of nodes and the number of layers – of the predictor are chosen utilizing a grid search (note that the hyper-parameters of the encoder is fixed after the Semi-Supervision Phase). We choose hyper-parameter values that achieve the best validation performance by using the outputs of the predictor ($i.e., \hat{y} = f_{\phi} \circ f_{\theta}(x)$).

The permitted values of coefficients ($\alpha$, $\beta$, $\pi$) and the hyper-parameters are listed in Table S.1.

B.2 Benchmarks

Throughout the experiments, we compared SEFS with feature selection methods ranging from conventional approaches – such as Lasso (Tibshirani, 1996) and Tree (Geurts et al., 2006) – to the state-of-the-art approaches – such as STG (Yamada et al., 2020) and DUFS (Lindenbaum et al., 2020). Further descriptions of the benchmarks and implementation details are as follows:

- **Lasso**
  Lasso is a well-known embedded method whose objective is to minimize the prediction loss (i.e., cross-entropy loss for classification tasks and mean squared error for regression tasks) while enforcing the $l_1$-penalty to achieve sparsity among input features (Tibshirani, 1996). We implement Lasso regression for regression tasks and Logistic regression with $l_1$-penalty for classification tasks. The hyper-parameter to control the input sparsity is chosen from $\{0.001, 0.01, 0.1, 1, 10, 100\}$.

- **Tree**
  ExtraTree (Geurts et al., 2006) is an ensemble of decision trees that has become a popular choice for classification and regression tasks for tabular data. We use the feature importance

---

7 The specification of the machine is: CPU – Intel Core i7-8700K, GPU – NVIDIA GeForce GTX 1080Ti, and RAM – 64GB DDR4

8 Implemented using Python package scikit-learn

9 Implemented using Python package scikit-learn
Table S.1: Hyper-parameters of SEFS.

<table>
<thead>
<tr>
<th>Block</th>
<th>Set of Hyper-Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization</td>
<td>Xavier (Glorot &amp; Bengio, 2010)</td>
</tr>
<tr>
<td>Optimization</td>
<td>Adam (Kingma &amp; Ba, 2015)</td>
</tr>
<tr>
<td>Mini-batch size*</td>
<td>32</td>
</tr>
<tr>
<td>Non-linearity</td>
<td>ReLu</td>
</tr>
</tbody>
</table>

**Semi-Supervision Phase**

- Learning rate ($\eta$) \{0.0001, 0.001, 0.01\}
- No. of hidden units \{10, 30, 50, 100\}
- No. of layers \{1, 2, 3\}
- No. of nodes \{10, 30, 50, 100, 300, 500\}
- Coeff. $\alpha$ \{0.01, 0.1, 1.0, 10, 100\}
- Coeff. $\pi$ \{0.2, 0.4, 0.6, 0.8\}

**Supervision Phase**

- Temperature $\tau$: 1.0
- Dropout: 0.3
- Learning rate ($\eta_1$, $\eta_2$) \{0.0001, 0.001, 0.01\}
- No. of layers \{1, 2, 3\}
- No. of nodes \{10, 30, 50, 100, 300, 500\}
- Coeff. $\beta$ \{0.01, 0.1, 1.0, 10, 100\}

*We used $n_l$ as the mini-batch size for the cases with the number of labeled samples smaller than 32, i.e., $\min(n_l, 32)$.

obtained by ExtraTree to conduct feature selection. The number of trees and the maximum depth are chosen from \{10, 50, 100, 300, 500\} and \{1, 2, 3, 4\}, respectively.

- L-Score\(^{10}\): The Laplacian Score (L-Score) \cite{He_2005} is an unsupervised (filter) feature selection method that quantifies the importance of input features by the ability to preserve local structure of the data, which is captured by the Laplacian matrix. We further modify the affinity score between a pair of labeled samples following the description in \cite{Sheikhpour_2017}, i.e., the affinity score becomes 1 if the two samples have the same label and 0 otherwise. We construct the Laplacian matrix based on the Gaussian kernel using both labeled and unlabeled samples with 10-nearest neighbors and setting the kernel bandwidth as the median Euclidean distance.

- BNNsel\(^{11}\): BNNsel is an embedded feature selection method based on Bayesian neural networks that overcomes the non-differentiability in the selection process by utilizing MCMC sampling \cite{Liang_2018}. The number of hidden units is chosen among \{3 (default value), 5, 10, 30, 50\} with a fixed prior probability 0.025 (default value).

- STG\(^{12}\): STG is an embedded feature selection method using deep neural networks that overcomes the non-differentiable process of selecting feature subsets via continuous relaxation using Gaussian random variables \cite{Yamada_2020}. We use a fully-connected network as the baseline architecture where the number of nodes and the number of layers are chosen among \{10, 30, 50, 100, 300, 500\} and \{1, 2, 3, 4, 5\}, respectively. The hyper-parameter to control the sparsity of input features is chosen among \{0.001, 0.01, 0.1, 1, 10, 100\}.

- STG (SS): We propose an extension of STG to the semi-supervised setting. To this goal, we introduce a reconstruction task in addition to the prediction task of the original STG. More specifically, we utilize an auxiliary network, i.e., decoder, which reconstructs the original input features given the gated features by minimizing the reconstruction loss. The overall network is trained based on the combination of the prediction loss and the reconstruction loss. (Here, we adopt a hyper-parameter $\alpha \in \{1.0, 0.1, 0.01, 0.001\}$ to balance the two losses). We use a fully-connected network as the baseline architecture where the number of nodes and the number of layers are chosen among \{10, 30, 50, 100, 300, 500\} and \{1, 2, 3, 4, 5\}, respectively. The hyper-

\(^{10}\)We explicitly implement L-Score for semi-supervised setting based on the description in \cite{Sheikhpour_2017}.

\(^{11}\)https://rdrr.io/cran/BNN/man/BNNsel.html

\(^{12}\)https://github.com/runopti/stg
Table S.2: Summary statistics of the dataset.

<table>
<thead>
<tr>
<th>Data Source</th>
<th>UKCF</th>
<th>PBMC</th>
<th>CCLE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Type</strong></td>
<td>Clinical</td>
<td>Transcriptomics</td>
<td>Proteomics</td>
</tr>
<tr>
<td><strong>Target Outcome</strong></td>
<td>Resp. failure</td>
<td>Cell-type class</td>
<td>Drug response</td>
</tr>
<tr>
<td><strong>Feature Dimension</strong></td>
<td>285</td>
<td>19256</td>
<td>196</td>
</tr>
<tr>
<td><strong>No. Labeled Samples (FS–Train)</strong></td>
<td>32</td>
<td>1296</td>
<td>15</td>
</tr>
<tr>
<td><strong>No. Labeled Samples (Eval.–Train)</strong></td>
<td>1000</td>
<td>3889</td>
<td>146</td>
</tr>
<tr>
<td><strong>No. Labeled Samples (Eval.–Test)</strong></td>
<td>4754</td>
<td>15557</td>
<td>7770</td>
</tr>
<tr>
<td><strong>No. Unlabeled Samples</strong></td>
<td>32</td>
<td>20</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>1296</td>
<td>25</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>146</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>223</td>
<td>27</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>225</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>227</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>228</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>229</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>229</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>7770</td>
<td>7770</td>
<td>7770</td>
</tr>
<tr>
<td></td>
<td>7770</td>
<td>7770</td>
<td>7770</td>
</tr>
<tr>
<td></td>
<td>7770</td>
<td>7770</td>
<td>7770</td>
</tr>
</tbody>
</table>

Lapa: Lapatinib, PD: PD-0325901, PF: PF2341066, Topo: Topotecan

Figure S.1: Illustration of dataset splits for (a) UKCF, (b) PBMC, and (c) CCLE – Lapatinib datasets.

Parameter to control the sparsity of input features is chosen among \{0.001, 0.01, 0.1, 1, 10, 100\}.

- **DUFS** \footnote{https://github.com/Ofirlin/DUFS} is an unsupervised feature selection method that improves the Laplacian score \cite{he2005laplacian} by replacing it with a gated variant computed on a subset of features \cite{lindenbaum2020dufs}. We use the parameter-free version of DUFS which constructs the Laplacian matrix with 2-nearest neighbors and setting the kernel bandwidth as the median Euclidean distance. Both labeled and unlabeled samples are used for training.

C Detailed Data Descriptions

In Section 5 of the manuscript, we evaluate SEFS and the benchmarks with multiple healthcare datasets from three different sources, i.e., UKCF (clinical), CCLE (proteomics), and PBMC (transcriptomics). The summary statistics of these datasets are provided in Table S.2.
Throughout the experiments on the real-world healthcare and omics datasets, we consider a practical scenario for feature selection under low labeled data regime where only a small number of labeled samples are available while having a sufficient amount of unlabeled sample. To this goal, we construct datasets based on the following steps: First, we obtain enough unlabeled samples by utilizing samples without labels that are readily present in the original data (e.g., UKCF dataset has censored patients and CCLE dataset has samples without drug response), treating a portion of labeled samples as unlabeled (e.g., UKCF and PBMC datasets), and by integrating samples from different sources (e.g., CCLE dataset). Second, we randomly split the labeled samples into training and testing sets. Here, both the training and the testing sets are used for evaluating discovered features from different feature selection methods. This assures that the discovered features generalize well to unseen samples. Third, we randomly select a small subset of training samples for training different feature selection methods. The overall data construction steps are illustrated in Figure S.1.

**Correlation Structure.** The correlation structure of the datasets are illustrated in Figure S.2. We construct the correlation matrix $R$ as defined in Algorithm 1 and 2: formally, each element of $R$ is given as $R_{kj} = \frac{C_{kj}}{\sqrt{C_{kk}C_{jj}}} = \frac{\sum_{i=1}^{n-1}(x_{ik} - \bar{x}_k)(x_{ij} - \bar{x}_j)}{n-1}$. It is worth highlighting that the PBMC dataset contains many features that have low correlation with other features; thus, after thresholding at 0.7, only 426 features with high correlation remain.

To quantify the amount of collinearity, we calculated the variance inflation factor (VIF), which is an index that measures how much the variance (the square of the estimate’s standard deviation) of an estimated regression coefficient is increased because of collinearity. While there is not a strict cut-off indicating collinearity, a general rule of thumb is that VIFs exceeding 10 are signs of serious multicollinearity (Menard, 2001; Vittinghoff et al., 2011; James et al., 2013). We note this is a conservative cut-off, with guidelines suggesting that VIFs greater than 5 (Menard, 2001; Vittinghoff et al., 2011) indicate considerable collinearity. In Table S.3, we provide the VIFs of features in each dataset; here the VIFs for the PBMC dataset are not available due to the computational complexity. As seen in the table, there are strong signs of multicollinearity in both datasets with the VIFs for 23.5% of the features in the UKCF dataset and all the features in the CCLE dataset exceeding 10.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Mean</th>
<th>Min</th>
<th>Max</th>
<th>VIF &gt; 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>UKCF</td>
<td>11.33</td>
<td>1.04</td>
<td>391.84</td>
<td>23.50%</td>
</tr>
<tr>
<td>CCLE</td>
<td>80.81</td>
<td>15.00</td>
<td>462.97</td>
<td>100%</td>
</tr>
</tbody>
</table>

**D ADDITIONAL EXPERIMENTS**

**D.1 SYNTHETIC: BLOCK-STRUCTURED NOISY TWO-MOONS DATASET**

**Quantitative Results.** Table S.4 shows the average TPRs for all benchmarks. As can be seen in the table, SEFS outperforms all benchmarks, with the majority of methods struggling to identify Feature 1 in particular. This is due to Feature 2 being more discriminative than Feature 1 and therefore the noisy features that are correlated with Feature 2 are often selected. It is worth highlighting that L-Score, DUFS, and STG (SS) fail to discover discriminative features in almost all cases due to the correlation structure of the data that cannot be addressed by the similarity metrics or reconstruction employed by these methods.

**D.2 CLINICAL: UKCF DATASET**

**Quantitative Results.** Figure S.3 displays features discovered with a frequency $\geq 0.4$, where we consider a feature is discovered (for all three feature selection methods) if that feature is within the top 20 highest feature importance. It is immediately apparent that SEFS consistently identifies the same features, while Lasso and Tree typically select different features, demonstrating the robustness of our approach.
Table S.4: The average TPRs on the Two-Moons dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Feature 1</th>
<th>Feature 2</th>
<th>Average</th>
<th>Feature 1</th>
<th>Feature 2</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>0.45</td>
<td>0.98</td>
<td>0.72</td>
<td>0.77</td>
<td>1.00</td>
<td>0.89</td>
</tr>
<tr>
<td>Tree</td>
<td>0.02</td>
<td>0.46</td>
<td>0.24</td>
<td>0.05</td>
<td>0.88</td>
<td>0.47</td>
</tr>
<tr>
<td>STG</td>
<td>0.14</td>
<td>0.96</td>
<td>0.55</td>
<td>0.21</td>
<td>1.00</td>
<td>0.61</td>
</tr>
<tr>
<td>DUFS</td>
<td>0.02</td>
<td>0.03</td>
<td>0.03</td>
<td>0.01</td>
<td>0.02</td>
<td>0.02</td>
</tr>
<tr>
<td>L-Score</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>STG (SS)</td>
<td>0.09</td>
<td>1.00</td>
<td>0.55</td>
<td>0.09</td>
<td>0.94</td>
<td>0.52</td>
</tr>
<tr>
<td>SEFS (no SS)</td>
<td>0.68</td>
<td>0.99</td>
<td>0.84</td>
<td>0.98</td>
<td>1.00</td>
<td>0.99</td>
</tr>
<tr>
<td>SEFS</td>
<td>0.92</td>
<td>1.00</td>
<td>0.96</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Figure S.3: Comparison of discovered features based on Lasso, Tree, and SEFS for the UKCF dataset with \( n_l = 32 \).

**Qualitative Results.** Other than PI Allele 1 and 2 that are related to pancreatic functions as described in Section 5.2, SEFS consistently discovered features related to the lung function scores (i.e., \( \text{FEV}_1 \% \text{ pred} \text{ and } \text{FEV}_1 \)). It is worth highlighting that, features related to the lung function scores are frequently selected despite their relatively high correlation. We conjecture that the trajectory of the lung functions scores measured at different times (with the highest frequency of the latest measurement) plays an important role for predicting the respiratory failure as supported in Taylor-Robinson et al. (2012); Adler & Liou (2016).

D.3 PROTEOMICS: CCLE DATASET

**Quantitative Results.** In Table S.5, we compare the prediction performance of 5 and 10 discovered features (i.e., \( |S| = 5 \text{ and } 10 \)) for the 11 drugs – AZD6244, Erlotinib, Irinotecan, L-685458, Lapatinib, PD-0325901, PF2341066, PLX4720, Paclitaxel, Panobinostat, and Topotecan – reported in the manuscript. Despite the majority of unlabeled data originating from a different source, our method benefits from learning the underlying data structure from unlabeled samples. SEFS consistently displays improvements (except for Erlotinib with \( |S| = 5 \text{ and } 10 \), L-685458 with \( |S| = 5 \), and PF2341066 with \( |S| = 5 \text{ and } 10 \)) in performance for different drugs and varying numbers of discovered features, outperforming SEFS (no SS) in every experiment. While we would expect further gain from more similar unlabeled data, our results highlight the effectiveness of self-supervision even when the unlabeled data is only partially related to the labeled samples.

**Qualitative Results.** In the main manuscript, we validated the features identified by SEFS for panobinostat in the scientific literature. In Table S.6, we provide details of the references for panobinostat and also provide supporting evidence for the selected features for lapatinib and irinotecan.

Lapatinib is a drug used primarily to treat breast cancer, as well as other solid tumours. It is a dual tyrosine kinase inhibitor which inhibits the epidermal growth factor receptor (EGFR) and human epidermal growth factor receptor 2 (HER2) receptors. In total, we found supporting literature for 8
Table S.5: Comparison of the MSE (mean ± 95%-CI) given discovered features for the CCLE dataset. (Lower the better.)

<table>
<thead>
<tr>
<th>Methods</th>
<th>AZD6244</th>
<th>Erlotinib</th>
<th>Irinotecan</th>
<th>L-685458</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>1.44±0.03</td>
<td>1.49±0.03</td>
<td>1.42±0.02</td>
<td>1.43±0.03</td>
</tr>
<tr>
<td>Tree</td>
<td>1.39±0.04</td>
<td>1.42±0.04</td>
<td>1.43±0.02</td>
<td>1.38±0.03</td>
</tr>
<tr>
<td>BNNuel</td>
<td>1.39±0.04</td>
<td>1.41±0.04</td>
<td>0.40±0.01</td>
<td>0.41±0.01</td>
</tr>
<tr>
<td>STG</td>
<td>1.38±0.03</td>
<td>1.40±0.04</td>
<td>1.46±0.02</td>
<td>1.39±0.03</td>
</tr>
<tr>
<td>DUF5</td>
<td>1.42±0.04</td>
<td>1.43±0.04</td>
<td>0.42±0.01</td>
<td>0.44±0.01</td>
</tr>
<tr>
<td>L-Score</td>
<td>1.39±0.03</td>
<td>1.40±0.04</td>
<td>0.43±0.02</td>
<td>0.44±0.01</td>
</tr>
<tr>
<td>STG (SS)</td>
<td>1.39±0.04</td>
<td>1.41±0.04</td>
<td>0.41±0.01</td>
<td>0.44±0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>Lapatinib</th>
<th>PD-0325901</th>
<th>FF2341066</th>
<th>PLX4720</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.45±0.01</td>
<td>0.45±0.02</td>
<td>2.16±0.05</td>
<td>2.10±0.05</td>
</tr>
<tr>
<td>Tree</td>
<td>0.41±0.02</td>
<td>0.44±0.02</td>
<td>2.16±0.04</td>
<td>2.17±0.05</td>
</tr>
<tr>
<td>BNNuel</td>
<td>0.45±0.02</td>
<td>0.47±0.02</td>
<td>2.17±0.05</td>
<td>2.18±0.05</td>
</tr>
<tr>
<td>STG</td>
<td>0.43±0.02</td>
<td>0.46±0.02</td>
<td>2.21±0.06</td>
<td>2.17±0.06</td>
</tr>
<tr>
<td>DUF5</td>
<td>0.43±0.01</td>
<td>0.46±0.01</td>
<td>2.18±0.05</td>
<td>2.23±0.04</td>
</tr>
<tr>
<td>L-Score</td>
<td>0.43±0.01</td>
<td>0.45±0.01</td>
<td>2.43±0.04</td>
<td>2.26±0.05</td>
</tr>
<tr>
<td>STG (SS)</td>
<td>0.43±0.01</td>
<td>0.45±0.01</td>
<td>2.18±0.05</td>
<td>2.18±0.05</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>Paclitaxel</th>
<th>Panatinostat</th>
<th>Topotecan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>1.69±0.03</td>
<td>1.60±0.03</td>
<td>0.65±0.01</td>
</tr>
<tr>
<td>Tree</td>
<td>1.63±0.03</td>
<td>1.62±0.03</td>
<td>0.54±0.02</td>
</tr>
<tr>
<td>BNNuel</td>
<td>1.63±0.03</td>
<td>1.62±0.03</td>
<td>0.57±0.02</td>
</tr>
<tr>
<td>STG</td>
<td>1.65±0.03</td>
<td>1.65±0.04</td>
<td>0.57±0.02</td>
</tr>
<tr>
<td>DUF5</td>
<td>1.67±0.04</td>
<td>1.62±0.04</td>
<td>0.60±0.02</td>
</tr>
<tr>
<td>L-Score</td>
<td>1.67±0.03</td>
<td>1.63±0.03</td>
<td>0.64±0.01</td>
</tr>
<tr>
<td>STG (SS)</td>
<td>1.61±0.03</td>
<td>1.62±0.03</td>
<td>0.54±0.02</td>
</tr>
</tbody>
</table>

| SEFS (no SS) | 1.37±0.04 | 1.40±0.04 | 0.40±0.01 | 0.44±0.01 | 1.41±0.03 | 1.35±0.04 | 0.28±0.01 | 0.31±0.02 |
| SEFS     | 1.36±0.03 | 1.40±0.03 | 0.40±0.02 | 0.42±0.02 | 1.35±0.04 | 1.29±0.04 | 0.28±0.01 | 0.30±0.03 |

| Methods | |        |            |          |
|---------| |        |            |          |
| Lasso  | 0.39±0.02 | 0.42±0.02 | 2.17±0.06 | 2.19±0.06 | 0.34±0.01 | 0.37±0.01 | 0.39±0.01 | 0.40±0.02 |
| Tree   | 0.39±0.02 | 0.40±0.02 | 2.16±0.05 | 2.16±0.06 | 0.34±0.01 | 0.36±0.01 | 0.39±0.01 | 0.39±0.02 |

of the top 10 ranked features selected by SEFS (Table S.6b), with five such features not proposed by SEFS (no SS). The other two features selected by SEFS are both established cancer biomarkers with evidence in breast cancer, but we did not find literature specifically relating them to Lapatinib.

Irinotecan is a chemotherapy agent used primarily to treat colon cancer in addition to small cell lung cancer. All of the top 10 features identified by SEFS had supporting scientific literature for their impact on the mechanism and effectiveness of irinotecan or colon cancer (Table S.6c), of which six were not proposed by SEFS (no SS).

### D.4 Transcriptomics: PBMC Dataset

#### Qualitative Results.

In Figure S.4, we compare the frequency of the discovered features based on SEFS with that of the discovered features based on Lasso and Tree. The figure displays only the features that were selected with a frequency equal to or greater than 0.4 by one of the methods. It is worth highlighting that SEFS (no SS) does not discover any feature with frequency equal to or greater than 0.4, which shows the importance of the Self-Supervision Phase. The lack of consistency of discovering features suggests that Lasso and Tree could be overfit to spurious relations in different splits of the data. Only 2 features are discovered at least 40% of the time by either Lasso or Tree (one feature each), compared to 9 for SEFS.

SEFS almost always discovers **S100A4** and **IL32** as relevant features. Both of these genes have been shown to be only expressed, or preferentially expressed, in certain types of T-cells – **S100A4**: Weatherly et al. (2015) and **IL32**: Goda et al. (2006) – which strongly validates the selection of these features. These features are also the most frequent features discovered by Tree, but with much lower frequency, while Lasso does not discover either feature, despite their high relevance. Similarly, while **DUSP1** (also known as **MKP-1** is selected by Lasso, Tree, and SEFS, only our
Table S.6: Frequently discovered proteins (features) by SEFS for the CCLE dataset. Features in blue were not proposed by SEFS (no SS).

(a) Panobinostat

<table>
<thead>
<tr>
<th>Rank</th>
<th>Proteins</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Caveolin-1</td>
<td>(Deb et al., 2016)</td>
</tr>
<tr>
<td>2</td>
<td>YAP-pS127</td>
<td>(Heinemann et al., 2015)</td>
</tr>
<tr>
<td>3</td>
<td>PRAS40-pT246</td>
<td>(Gagner et al., 2016)</td>
</tr>
<tr>
<td>4</td>
<td>VHL</td>
<td>(Kalac et al., 2011)</td>
</tr>
<tr>
<td>5</td>
<td>Src-pY416</td>
<td>(Kostyniuk et al., 2002)</td>
</tr>
<tr>
<td>6</td>
<td>TAZ</td>
<td>(Lee et al., 2017)</td>
</tr>
<tr>
<td>7</td>
<td>14-3-3-β</td>
<td>(Wang et al., 2009)</td>
</tr>
<tr>
<td>8</td>
<td>Fibronectin</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>GSK3-pS9</td>
<td>(Rahmani et al., 2014)</td>
</tr>
<tr>
<td>10</td>
<td>MSH2</td>
<td>(Li et al., 2020)</td>
</tr>
</tbody>
</table>

(b) Lapatinib

<table>
<thead>
<tr>
<th>Rank</th>
<th>Proteins</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Caveolin-1</td>
<td>(Qian et al., 2019)</td>
</tr>
<tr>
<td>2</td>
<td>HER2-pY1248</td>
<td>(Medina &amp; Goodin, 2008)</td>
</tr>
<tr>
<td>3</td>
<td>VHL</td>
<td>(~)</td>
</tr>
<tr>
<td>4</td>
<td>mTOR</td>
<td>(Gayle et al., 2012)</td>
</tr>
<tr>
<td>5</td>
<td>N-Ras</td>
<td>(Gali 'e, 2007)</td>
</tr>
<tr>
<td>6</td>
<td>Claudin-7</td>
<td>(Constantinou et al., 2018)</td>
</tr>
<tr>
<td>7</td>
<td>Rab25</td>
<td>(Cheng et al., 2013)</td>
</tr>
<tr>
<td>8</td>
<td>GSK3,pS9</td>
<td>(Trudi et al., 2020)</td>
</tr>
</tbody>
</table>

(c) Irinotecan

<table>
<thead>
<tr>
<th>Rank</th>
<th>Proteins</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lck</td>
<td>(Harashima et al., 2003)</td>
</tr>
<tr>
<td>2</td>
<td>Statmin</td>
<td>(Peng et al., 2010)</td>
</tr>
<tr>
<td>3</td>
<td>β-Catenin</td>
<td>(Sato et al., 2010)</td>
</tr>
<tr>
<td>4</td>
<td>Bcl-2</td>
<td>(Gupta et al., 2007)</td>
</tr>
<tr>
<td>5</td>
<td>Src-pY416</td>
<td>(Peledpaz &amp; van Herpen, 2013)</td>
</tr>
<tr>
<td>6</td>
<td>Smad4</td>
<td>(Wong et al., 2009)</td>
</tr>
<tr>
<td>7</td>
<td>YAP-pS127</td>
<td>(Noguchi et al., 2018)</td>
</tr>
<tr>
<td>8</td>
<td>Bcl-xL</td>
<td>(Lee et al., 2019)</td>
</tr>
<tr>
<td>9</td>
<td>P3K-p85</td>
<td>(Koizumi et al., 2005)</td>
</tr>
<tr>
<td>10</td>
<td>E-Cadherin</td>
<td>(Bendardaf et al., 2019)</td>
</tr>
</tbody>
</table>

method discovers this feature in the majority of the experiments, despite its importance for T-cell activation and function (Zhang et al., 2009).

In addition, we also find evidence of the importance of JUN (Riera-Sans & Behrens, 2007), SH3BGRL3 (Deng et al., 2006), CCR7 (Schneider et al., 2007), and ribosomal genes RPS13, RPL27, and RPS3A (Procaccini et al., 2016).

Figure S.4: Comparison of discovered features using Lasso, Tree, and SEFS for the PBMC dataset with n_l=20 (i.e., 10 labeled samples for each label).

D.5 Correlated vs Independent Gate Vector Generation in Feature Selection

In this subsection, we investigate how the correlated gate vectors using a multivariate Bernoulli distribution in SEFS improves the feature selection process over independently generated gate vectors. To this goal, in Figure S.5, we provide trajectories of selection probabilities for the ground-truth feature x_2 and its block-correlated noisy features with respect to the number of training iterations. Note that the results are generated only utilizing the Supervision Phase. As claimed in the manuscript, the correlated gate vectors encourage the network to select only the most relevant feature by making
Figure S.5: The trajectory of $\pi_k$’s with respect to the number of iterations for the ground-truth relevant feature $x_2$ (in the blue line) and its block-correlated noisy features (in other colors) using (a) correlated gate vectors using a multivariate Bernoulli distribution and (b) independent gate vectors using an independent Bernoulli distribution.

Figure S.6: Comparison of (a) the TPR performance and (b) the number of discovered features with $\pi_k > 0.5$, with respect to different values of $\beta$, respectively.

D.6 Sensitivity Analysis on Coefficient $\beta$

In this subsection, we provide sensitivity analysis using the Block-Structured Noisy Two-Moons dataset to see the effect of coefficient $\beta$ that controls the sparsity of the selected features. Figure S.6 shows (a) the true discovery rate as defined in Section 5.1 – where we define $x_1$ (or $x_2$) as correctly discovered if and only if $x_1$ (or $x_2$) has the first or the second highest feature importance – and (b) the number of features discovered whose selection probability is higher than 0.5 (i.e., $\pi_k > 0.5$). Here, multiple instances of SEFS with different $\beta$’s are trained utilizing $n_l = 20$ labeled samples (i.e., 10 labeled samples for each label) and pre-trained with $n_u = 1000$. The results are averaged over 100 iterations as described in Section 5.1 of the manuscript.

Figure S.6(a) shows that SEFS trained with setting $\beta = 5, 10$ provides the highest TPR. However, as can be seen in Figure S.6(b), setting the coefficient too small (here, $\beta \leq 1$) will end up discovering too many irrelevant features as if they were relevant – i.e., having many features whose selection probability is above a certain level – and will mislead the discovery of relevant features. Contrarily, setting the coefficient too high (here, $\beta = 10$) can restrict the selection process and may fail to identify relevant features.
APPENDIX REFERENCES


Mohamed Rahmani, Mandy Mayo Aust, Elisa C. Benson, LaShanale Wallace, Jonathan Friedberg, and Steven Grant. PI3k/mtor inhibition markedly potentiates hdac inhibitor activity in nhl cells through bim- and mcl-1-dependent mechanisms in vitro and in vivo. Clinical Cancer Research, 20(18):4849–4860, 2014. doi: 10.1158/1078-0432.CCR-14-0034. URL [https://clincancerres.aacrjournals.org/content/20/18/4849](https://clincancerres.aacrjournals.org/content/20/18/4849).


26