SCALABLE FEATURE SELECTION VIA SPARSE LEARNABLE MASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a canonical approach for feature selection, sparse learnable masks (SLM). SLM integrates learnable sparse masks into end-to-end training. For the fundamental non-differentiability challenge of selecting a desired number of features, we propose duo mechanisms for automatic mask scaling to achieve the desired feature sparsity, and gradually tempering this sparsity for effective learning. In addition, SLM employs a novel objective that maximizes the mutual information (MI) between the selected features and the labels, in an efficient and scalable way. Empirically, SLM achieves state-of-the-art results on several benchmark datasets, often by a significant margin, especially on real-world challenging datasets.

1 INTRODUCTION

In many machine learning scenarios, a significant portion of the input features may be irrelevant to the output, especially with modern data management tools allowing easy construction of large-scale datasets by combining many different data sources. 'Feature selection', or filtering the most relevant features for the downstream task, is an everlasting problem in machine learning, with many methods proposed to date and used in numerous applications (Guyon & Elisseeff, 2003; Li et al., 2017; Dash & Liu, 1997).

Feature selection can bring a multitude of benefits in machine learning. Smaller number of features can yield superior generalization and hence better test accuracy, by minimizing information extraction from spurious patterns that do not hold consistently (Sagawa et al., 2020), and by utilizing the model capacity in an optimal way on the most relevant features. In addition, reducing the number of input features can decrease the computational complexity and cost for deployed models, as the models need to learn the mapping from smaller dimensional input data, and the infrastructure need to support only the selected features likely decreases. Lastly, it helps with interpretability and controllability, as the human users can focus their efforts to understand the model to a smaller subset of input features.

Feature selection has been studied with numerous approaches, as summarized in §2. For the optimal test accuracy, a feature selection method should consider the predictive model itself, as the optimal set of features would depend on the how the mapping is done between the inputs and outputs. Such 'Embedded' methods have been approached in different ways, such as via regularization and its extensions (Lemhadri et al., 2019). Especially when deep learning is considered, the fundamental challenge is that the selection operation (given the target count for selected features), being non-differentiable. This necessitates design of soft approximations for feature selection operator, united with the end-to-end task learning.

In this paper, we propose Sparse Learnable Masks (SLM), a novel approach for scalable feature selection. SLM can be integrated into any deep learning architecture, given the optimization is gradient-descent based. While the vast majority of the feature selection methods cannot take advantage of end-to-end learning, SLM is jointly trained with the predictive model. SLM provides benefits for scaling – yielding efficient feature selection even when the number of features or samples are very large. SLM proposes an effective way of modifying the learnable masks to select the exact number desired features, addressing the differentiability challenges. In addition, SLM uses a novel mutual information (MI) regularizer, based on a quadratic relaxation of the MI between the labels and the selected features, conditioned on the probability that a feature is selected. We demonstrate state-of-the-art feature selection results with SLM in different scenarios across a wide range of datasets.

2 RELATED WORK

Feature selection methods: Numerous methods have been studied for feature selection, broadly categorized under 3 groups (Guyon & Elisseeff, 2003):

- *Wrappers* utilize a given predictive model as black box to score subsets of variables according to their predictive power. As exhaustive search is NP-hard and computationally intractable, efficient search strategies become important, including Greedy methods such as forward selection (selected features are progressively incorporated into larger subsets) and backward elimination (starting with the set of all features and progressively eliminating the least promising ones). Among other advanced methods, HSIC-Lasso (Yamada et al., 2014) is a notable one, proposing a feature-wise kernelized Lasso for capturing non-linear dependencies. *Wrappers* are especially difficult to integrate with modern deep learning, as the training complexity would get prohibitively large.
- *Filters* select subsets of variables as a pre-processing step, independent of the predictive model. One popular method is using Fisher score (Gu et al., 2012) which selects features to maximize/minimize the distances between data points in different/same classes in the space spanned by the selected features. Another commonly-used method is principal feature analysis (PFA) (Lu et al., 2007b), based on principal component analysis. (Pan et al., 2020) uses adversarial validation to select the features, based on how much their characteristics differ between training and test splits, as a way to improve robustness. There are also various methods based on MI maximization (Ding & Peng, 2005), selecting features independent of the predictive model (unlike SLM). CMIM (Fleuret, 2004) maximizes the *conditional* MI between selected features and the class labels to account for feature inter-dependence. JMIM (Bennasar et al., 2015) maximizes the *joint* MI between class labels and the selected features, while addressing overconfidence in features that correlate with already-selected features, with greedy search that selects features one at a time. (Zadeh et al., 2017) formulates feature selection as a diversity maximization problem using a MI-based metric amongst features. The fundamental disadvantage of filter-based methods, of not being optimized with the predictive models, results in them often yielding suboptimal performance.
- *Embedded* methods perform selection in the process of training and are usually specific to given predictive models. Their value proposition for efficient feature selection comes from combining feature selection and learning. A well-known example is the Lasso regularization (Tibshirani, 1996), which employs feature selection by varying the strength of the L1 regularization. (Feng & Simon, 2017) extends this idea by proposing an input-sparse neural network, where the input weights are penalized using the group Lasso penalty. (Lemhadri et al., 2019) is a recently proposed version that only a subset of the features are selected using input-to-output residual connections, allowing features to have non-zero weights only if their skip-layer connections are active.

Masking in deep neural networks: Masking the inputs to control information propagation is a commonly-used approach in deep learning. Attention-based architectures, such as the Transformer (Vaswani et al., 2017) and the Perceiver (Jaegle et al., 2021), show strong results across many domains, with learnable key and query representations, whose alignment yield the masks that control the contribution of corresponding value representations. While these effectively reweight the inputs, they typically do not completely mask out (i.e. yielding zero attention weight) the inputs. Towards this end, various works have focused on bringing sparsity into masking, such as based on thresholding (Zhao et al., 2019) or sparse normalization (Correia et al., 2019). TabNet (Arik & Pfister, 2019) is one notable architecture, based on directly generating sparse attention masks and applying them sequentially to input data, which can perform sample-dependent feature selection. All these perform sample-wise masking or input selection, not global, i.e. the masks depend on the input features. On the other hand, our goal in this paper is to explore global feature selection. We note that sample-wise masking can be indeed more beneficial for accuracy, as the model would have higher capacity to focus on the important features that may depend on the actual input itself. However, it does not bring the computational complexity saving and overfitting robustness benefits of global feature selection.

3 METHODS

SLM is based on the canonical idea of integrating a feature selection layer into the model architecture, as part of gradient-descent based learning. Alg. 1 overviews the training procedure of SLM. The following sections present the key components of it in detail.

Let $\mathbf{x} \in R^{F_0}$ denote the input data (covariate) and y is the target, such as class labels. We assume that the input data are numerical – categorical features can be mapped to embeddings. SLM is integrated with the the predictor model f_θ with learnable parameters θ, that is applied to the selected features $\mathbf{x_{sp}} \in \mathbb{R}^{F_t}$, where F_t denotes the number of selected features at step t. The predictor model can be any architecture such as a multi-layer perceptron (MLP) or TabNet (Arik & Pfister, 2019), as long as it uses gradient-descent based training. We use the notation of multiplication of a binary mask $\mathcal{M}_{sp}$ for feature selection operation.

ALGORITHM 1 Training for SLM-based feature selection.

Input: Input data $\mathbf{x}$ with target labels y
Input: Total training steps N
Initialize: Learnable mask argument $\mathcal{M} \leftarrow$ all ones vector
for $t = 1$ **to** N **do**
  Obtain the number of selected features F_t using Eq 3 for step t.
  Generate **sparse mask** $\mathcal{M}_{\text{sp}} = \text{sparsemax}(\mathcal{M})$.
  Select and **weight** input features: $\mathbf{x_{sp}} = \mathbf{x} \cdot \mathcal{M}_{\text{sp}}$. Non-selected features are zeroed out.
  Input the selected features into the predictor $f_\theta(\mathbf{x_{sp}})$ for the downstream task.
  Compute **training task loss** $l(\mathbf{x}_{sp}, y)$ and **MI loss** $E(\mathbf{x}_{sp}, y)$ in Eq 9.
  Update the parameters θ and $\mathcal{M}$, using task loss l and MI loss E.

3.1 SPARSE ATTENTION

We propose to combine sparse non-linear normalization with learnable feature selection vectors in SLM. Motivated its success in encouraging most small values to be exactly mapped to zero, we employ sparsemax normalization (Martins & Astudillo, 2016) to achieve feature sparsity, rather than approaches like thresholding of other normalization approaches (Zhao et al., 2019). Sparsemax achieves highly-sparse selection by returning the Euclidean projection of the input vector onto the probability simplex:

$$\text{sparsemax}(\boldsymbol{v}) := \text{argmin}_{\boldsymbol{p} \in \Delta^{K-1}} \|\boldsymbol{p} - \boldsymbol{v}\|^2 \tag{1}$$

In sparsemax, the probability simplex projection in $\text{sparsemax}(\boldsymbol{v})$ scales the top values in $\boldsymbol{v}$ so they are more equidistributed over $[0, 1]$ than softmax with thresholding would yield. This equidistribution leads to greater feature weight separation, encouraging to discriminate amongst the features. Sparsemax is applied to the normalized mask argument to obtain sparse feature mask:

$$\mathcal{M}_{\text{sp}} := \text{sparsemax}(\mathcal{M}). \tag{2}$$

3.2 TEMPERING FEATURE SPARSITY

Starting training on only a randomly selected subset of features likely leads to suboptimal learning in the initial steps, and if feature selection converges before the predictor converges, the predictor would be trained with suboptimal features. To alleviate these and improve training stability, we propose gradually decreasing the number of features selected until reaching the target F_N:

$$F_t = \begin{cases} F_0 - t/N_{tmp}(F_0 - F_N) & \text{if } t < N_{tmp} \\ F_N & \text{if } t \geq N_{tmp}, \end{cases} \tag{3}$$

where F_t is the number of selected features at step t, N_{tmp} is the tempering threshold. In our experiments, we simply set $N_{tmp} = N/2$ as it was observed to be a reasonable value across a wide range of datasets. To further stabilize training, instead of continuously decreasing the number of features, we decrease the number of features at five evenly spaced steps. This tempering allows the model to learn from more than the final target number of features during training – an advantage not shared by baseline methods. Furthermore, learning from all features initially likely provides a more robust initialization compared to starting learning with the target number of features, as the randomness in the initial selection is seldom optimal.

3.3 MASK SCALING TO YIELD DESIRED NUMBER OF SELECTED FEATURES

Following its formulation, sparsemax does not yield a predetermined number of non-zero elements, as the sparsity depends on the location on the probability simplex Δ^{K-1} that $\boldsymbol{v}$ projects onto. For a non-uniform vector $\boldsymbol{v} \in \mathbb{R}^K$, we can adjust its projection onto Δ^{K-1} by multiplying $\boldsymbol{v}$ by a positive scalar. In particular, a sufficiently large scalar increases the sparsity, while a sufficiently small scalar decreases the sparsity. To illustrate this, we give a simple example in two dimensions, as illustrated in Fig 1.

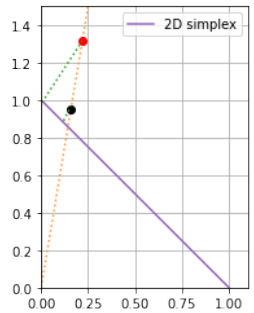

Figure 1: Scaling $\boldsymbol{v}$ from the black to the red point moves its projection (green dotted line) onto Δ^1 closer to the simplex boundary.

Example 3.1 (Adjusting sparsemax($\boldsymbol{v}$) sparsity by scaling). The probability simplex Δ^1 in $\mathbb{R}^2$ is the line connecting $(0, 1)$ and $(1, 0)$, with these two points as the simplex boundary. Let $\boldsymbol{v} = (x, y)$ be a point in $\mathbb{R}^2$, and (z, w) its projection onto Δ^1. We show that by varying multiplier m, sparsemax($m\boldsymbol{v}$) would have a varying degree of sparsity. The projection $(z, w) = \text{sparsemax}((x, y))$ is the unique point that satisfies $(z, w) = \text{argmin}_{(z,w)}(\|y - w\|^2 + \|x - z\|^2)$, (z, w) elementwise positive, and $z + w = 1$. As we scale (x, y) with m, $\text{sparsemax}(m(x, y)) = argmin_{(z,w)}(\|my - w\|^2 + \|mx - z\|^2)$. This projection distance expands to

$$\begin{aligned} d(z, w) &\coloneqq \|my - w\|^2 + \|mx - z\|^2 \\ &= m^2y^2 - 2myw + w^2 + m^2x^2 - 2mxz + z^2 \end{aligned}$$

Hence, $d(0, 1) - d(0.5, 0.5) = mx - my + 0.5$, which means that for any (x, y) and m with $y > x$, sparsemax($m(x, y)$) is closer to $(0, 1) \in \Delta^1$ whenever $m > 1/(2(y - x))$, and closer to $(0.5, 0.5)$ otherwise. Since projection is linear, this means varying the multiplier m varies the sparsity of sparsemax((x, y)).

This example conveys the intuition that larger multipliers lead to sparser outputs. More generally, one can show:

Lemma 3.2. *Given a vector $\boldsymbol{v} \in \mathbb{R}^K$, to obtain F nonzero elements in sparsemax($\boldsymbol{v}$), $\boldsymbol{v}$ should be multiplied with the scalar*

$$\boldsymbol{m} = \begin{cases} \left(\sum_{i=1}^{F+1} v_{(i)} - (F+1) * v_{(F+1)}\right)^{-1} & \text{if } |\textit{sparsemax}(\boldsymbol{v}) > 0| > F \\ \left(\sum_{i=1}^{F} v_{(i)} - F * v_F\right)^{-1} & \text{if } |\textit{sparsemax}(\boldsymbol{v}) > 0| < F, \end{cases} \tag{4}$$

where $v_{(1)} \geq v_{(2)} \ldots \geq v_{(K)}$ denote sorted elements of $\boldsymbol{v}$ in descending order.

The proof can be found in §A.3. Lemma 3.2 allows us to scale the mask to achieve the desired number of non-zero features.

3.4 Mutual Information Maximization

As an inductive bias to the model that accounts for sample labels during feature selection, we propose to maximize the mutual information (MI) between the distribution of the selected features and the distribution of the labels. Specifically, we condition the MI on the probability that a feature is selected, as given by the mask $\mathcal{M}$. This stands in contrast to prior MI-based feature selection works such as (Fleuret, 2004; Bennasar et al., 2015), which yield binary decisions on whether to select a feature.

Let X denote the random variable representing the features, and Y the random variable representing the labels, with value spaces $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$. Methods based on maximizing either the conditional or the joint MI between selected features and labels require the computation of an exponential number of probabilities, the optimization of which is intractable (Fleuret, 2004). Therefore, we propose a quadratic relaxation of MI, which is end-to-end differentiable. When we model X and Y as random variables, their MI $I(X, Y)$ can be defined and reformulated as:

$$\begin{aligned} I(X, Y) &\coloneqq \sum\nolimits_{x \in \mathcal{X}} \sum\nolimits_{y \in \mathcal{Y}} P_{X,Y}(x, y) \log \frac{P_{x,y}(x, y)}{P_X(x) P_Y(y)} \\ &= \left(\sum\nolimits_{x \in \mathcal{X}} \sum\nolimits_{y \in \mathcal{Y}} P_{X,Y}(x, y) \log \frac{P_{X,Y}(x, y)}{P_X(x)}\right) - \sum\nolimits_{y \in \mathcal{Y}} P_Y(y) \log P_Y(y), \end{aligned} \tag{5}$$

where the second step derives from marginalizing over $\mathcal{X}$. Since the second term above does not depend on features X, it can be ignored during optimization.

Quadratic relaxation. We propose a quadratic relaxation $I_q(X,Y)$ of Eq 5 to simplify $I(X,Y)$ and its optimization, while retaining much of its properties:

$$I_q(X,Y) \coloneqq \left(\sum\nolimits_{x\in\mathcal{X}} \sum\nolimits_{y\in\mathcal{Y}} P_{X,Y}(x,y)^2/P_X(x)\right) - \sum\nolimits_{y\in\mathcal{Y}} P_Y(y)^2. \tag{6}$$

Here, terms of the form $p\log q$ are relaxed to pq. Note that both $p\log q$ and pq are convex with respect to p and q, and hence have the same correlation behavior with respect to p and q. From an optimization perspective, $I_q(X,Y)$ is a good approximation of $I(X,Y)$ where $P_{X,Y}(X,Y)/P_X(x)$ and $P_Y(y)$ in Eq 6 lie in the neighborhood $(1-\delta, 1+\delta)$. In this neighborhood, using Taylor expansion: $\log(q)=\log(q_0)+(q-q_0)/q_0-(q-q_0)^2/2q_0^2+\cdots$ When $q_0=1$, this becomes $\log(q)\approx(q-1)-(q-1)^2/2=-3/2+2q-q^2/2$, hence, $p\log(q)$ has the second order approximation $-3p/2+2pq$ (or $-3p/2+2p^2$ when $p=q$). Applying this to Eq 5, p is $P_{X,Y}(x,y)$ in the first term and $P_Y(y)$ in the second. Since both $P_{X,Y}(x,y)$ and $P_Y(y)$ are probabilities, and hence must sum to 1 across the label space for any given sample, the linear term $-3p/2$ does not affect gradient descent optimization. Normalization is a hard constraint enforced during training that supersedes this linear term in the objective. Therefore, during optimization, $P_{X,Y}(x,y)\log(P_{X,Y}(x,y)/P_X(x))$ and $P_{X,Y}(x,y)^2/P_X(x)$, and thus $I_q(X,Y)$ and $I(X,Y)$, agree on their second order approximation. Note that the proposed relaxation is a variant of the commonly-used quadratic approximation based on Taylor's theorem (Shafer, 1974; Hsieh et al., 2011).

Relating MI $I_q(X,Y)$ to model error $E(X,Y)$. Next we connect $I_q(X,Y)$ with the model's predictions using Lagrange multipliers. Let $R(x,y):\mathcal{X}\times\mathcal{Y}\to[0,1]$ denote the model's probability output for sample x and outcome y. Below we model the discrete label case, e.g. for classification; the case where labels are continuous can be reduced to the discrete case by quantization (Fleuret, 2004). First, we define the quadratic error term $E(X,Y)$ in terms of $R(x,y)$, and expand:

$$\begin{aligned}
E(X,Y) &\coloneqq \sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} P_{X,Y}(x,y)\left((1-R(x,y))^2+\sum\nolimits_{y'\in\mathcal{Y}\setminus y} R(x,y')^2\right)\\
&= \sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} P_{X,Y}(x,y)\left(1-2R(x,y)+R(x,y)^2+\sum\nolimits_{y'\in\mathcal{Y}\setminus y} R(x,y')^2\right)\\
&= \sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} P_{X,Y}(x,y) - 2\sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} P_{X,Y}(x,y)R(x,y)\\
&\qquad + \sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y},y'\in\mathcal{Y}} P_{X,Y}(x,y)R(x,y')^2 \quad \triangleleft \text{Combine last two terms and expand.}\\
&= 1-2\sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} P_{X,Y}(x,y)R(x,y) + \sum\nolimits_{x\in\mathcal{X},y'\in\mathcal{Y}} P_X(x)R(x,y')^2 \quad \triangleleft \text{Marginalize.}
\end{aligned} \tag{7}$$

Theorem 3.3. *Let X and Y denote the random variables representing the features and labels, respectively, and $\mathcal{Y}$ the value space for Y, then maximizing the quadratic relaxation of mutual information $I_q(X,Y)$ is equivalent to minimizing the error $E(X,Y)$. More specifically,*

$$E(X,Y) = 1 - \sum\nolimits_{y\in\mathcal{Y}} P_Y(y)^2 - I_q(X,Y)$$

The proof uses Lagrange multipliers to solve for the optimal model predictions in terms of $P_{X,Y}(x,y)$ and $P_X(x)$, this can then be used to express the objective $E(X,Y)$ as a function of $I_q(X,Y)$. The full proof can be found in §A.4.

Application to feature selection. Now, we apply this finding concretely to feature selection, by selecting a given number of features that minimize $E(X,Y)$. Given a dataset, let $\mathcal{I}$ denote the index set of the dataset samples, $\mathcal{J}$ the index set of the features, and $\mathcal{L}$ the set of possible labels. Let $\mathcal{S}\subset\mathcal{J}$ denote the index set of features selected, $X_i^{\mathcal{S}}$ the random variable representing a selected subset of features for the ith sample, and Y_i the random variable representing the label for the ith sample. Then, the joint probability can be written as $P_{X,Y}(x,y) = |\{i\in\mathcal{I}|X_i^{\mathcal{S}}=x, Y_i=y\}|/|I|$. Plugging this into the definition of $E(X,Y)$ we obtain:

$$
\begin{aligned}
E(X,Y) &\coloneqq \sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} P_{X,Y}(x,y)\left((1-R(x,y))^2+\sum\nolimits_{y\neq Y_i} R(x,y')^2\right)\\
&= \sum\nolimits_{x\in\mathcal{X},y\in\mathcal{Y}} \frac{|\{i\in\mathcal{I}\mid X_i^{\mathcal{S}}=x, Y_i=y\}|}{|\mathcal{I}|}\left((1-R(X_i^{\mathcal{S}},Y_i))^2+\sum\nolimits_{y\neq Y_i} R(X_i^{\mathcal{S}},y)^2\right)\\
&= \sum\nolimits_{i\in\mathcal{I}}\left((1-R(X_i^{\mathcal{S}},Y_i))^2+\sum\nolimits_{y\neq Y_i} R(X_i^{\mathcal{S}},y)^2/|\mathcal{I}|\right) \qquad (8)
\end{aligned}
$$

During training, Eq. 8 is minimized under the following consistency constraint: for two samples i_1 and i_2 that have the same values in the selected features, i.e. $X_{i_1}^{\mathcal{S}} = X_{i_2}^{\mathcal{S}}$, their model predictions must be the same, i.e. $R(X_{i_1}^{\mathcal{S}}, Y_{i_1}) = R(X_{i_2}^{\mathcal{S}}, Y_{i_2})$. To encourage the model to satisfy this constraint, we turn it into a soft consistency regularization term r_{cs}, converting constrained optimization to unconstrained optimization with regularization:

$$
r_{\text{cs}} \coloneqq \sum\nolimits_{\{i_1,i_2\}\in\mathcal{I}^2, i_1<i_2} P(X_{i_1}^{\mathcal{S}} = X_{i_2}^{\mathcal{S}})\left(R(X_{i_1}^{\mathcal{S}},Y_{i_1}) - R(X_{i_2}^{\mathcal{S}},Y_{i_2})\right)^2,
$$

where $P(X_{i_1}^{\mathcal{S}} = X_{i_2}^{\mathcal{S}})$ is the probability that the samples X_{i_1} and X_{i_2} take the same values in the selected feature set $\mathcal{S}$.

Let the learned mask consists of probabilities $\mathcal{M} = \{p_j\}_{j\in\mathcal{J}}$, i.e. p_j is the probability that feature j is selected, then $P(X_{i_1}^{\mathcal{S}} = X_{i_2}^{\mathcal{S}}) = \prod_{X_{i_1}^{(j)}\neq X_{i_2}^{(j)}}(1-p_j)$, i.e. $P(X_{i_1}^{\mathcal{S}} = X_{i_2}^{\mathcal{S}})$ is the product over probabilities that feature j is not selected, if X_{i_1} and X_{i_2} differ at feature j. (The difference in a feature that is not selected does not contribute to $P(X_{i_1}^{\mathcal{S}} = X_{i_2}^{\mathcal{S}})$). In this probabilistic form, the consistency regularizer also encourages the selection of features with diverse ranges, since it encourages high p_j for the features with many $X_{i_1}^{(j)} \neq X_{i_2}^{(j)}$ pairs. Therefore, the regularized objective to maximize the MI $I(X,Y)$ between the selected features and the labels becomes:

$$
E(X,Y) = \sum\nolimits_{i\in\mathcal{I}}\left((1-R(X_i^{\mathcal{S}},Y_i))^2+\sum\nolimits_{y\neq Y_i} R(X_i^{\mathcal{S}},y)^2\right)/|\mathcal{I}| + r_{cs}, \qquad (9)
$$

where

$$
r_{cs} = \sum\nolimits_{\{i_1,i_2\}\in\mathcal{I}^2, i_1<i_2}\left(\prod\nolimits_{X_{i_1}^{(j)}\neq X_{i_2}^{(j)}}(1-p_j)\left(R(X_{i_1}^{\mathcal{S}},Y_{i_1}) - R(X_{i_2}^{\mathcal{S}},Y_{i_2})\right)^2\right) \qquad (10)
$$

In practice, r_{cs} can be enforced batch-wise, and can be efficiently vectorized for the parallel computation of all $X_{i_1}^{(j)} \neq X_{i_2}^{(j)}$ pairs per batch. When the labels are in the continuous space, the minimization objective with the consistency regularizer is derived the exact the same way to yield:

$$
E(X,Y) = \sum\nolimits_{i\in\mathcal{I}}\left(Y_i - R(X_i^{\mathcal{S}})\right)^2/|\mathcal{I}| + r_{cs}.
$$

3.5 Computational complexity

The sparsemax operation is dominated by sorting, and hence has complexity $O(F_0 \log F_0)$ per sample, with an overall complexity of $O(nF_0 \log F_0)$. The consistency regularizer r_{cs} in the MI-maximizing objective $E(X,Y)$ has complexity $O(nbF_N)$, as the calculation $\prod_{X_{i_1}^{(j)}\neq X_{i_2}^{(j)}}(1-p_j)\left(R(X_{i_1}^{\mathcal{S}},Y_{i_1}) - R(X_{i_2}^{\mathcal{S}},Y_{i_2})\right)^2$ in Eq 10 occurs over the selected feature index set $j\in\mathcal{S}$, and is done between each sample and others in its batch. The non-regularizer component in $E(X,Y)$ has complexity nc, where c is the constant for the number of discrete or binned labels. Assuming an MLP classifier with h hidden units, which has complexity $O(nh^2)$, the overall algorithm has complexity $O(nF_0 \log F_0 + nbF_N + nc + nh^2)$, i.e. the dependence on the total number of features is $\widetilde{O}(F_0)$, making SLM is amenable to scaling to a large number of features.

4 Experiments

4.1 Datasets and Settings

We present the efficacy of SLM in feature selection on wide range of datasets from numerous domains. For all experiments, we ensure fair comparison by employing similar hyperparameter search space

and budget – to search for hyperparameters such as batch size and learning rate for each baseline method and dataset, we conduct an extensive random search within the search grid, by randomly generating a value within a conceivable range. We run a total of 300 trials for each method-dataset combination to ensure good coverage. We tune all hyperparameters based on the validation accuracy.

We first consider the real-world benchmark datasets in (Lemhadri et al., 2019). **Mice** consists of protein expression levels measured in the cortex of normal and trisomic mice who had been exposed to different experimental conditions. Each feature is the expression level of one protein. **MNIST** and **Fashion-MNIST** consist of 28-by-28 grayscale images of hand-written digits and clothing items, respectively. The images are converted to tabular data by treating each pixel as a separate feature. **Isolet** consists of preprocessed speech data of people speaking the names of the letters in the English alphabet with each feature being one of the preprocessed quantities, including spectral coefficients and sonorant features. **Coil-20** consists of centered gray-scale images of 20 objects taken at certain pose intervals, hence the features are image pixels. **Activity** consists of sensor data collected from a smartphone mounted on subjects while they performed several activities such as walking or standing. For these datasets, we use the exact same data splits and preprocessing approaches with (Lemhadri et al., 2019) for fair comparison, as well as the same model hyperparameter search space.[1] In addition, we consider the **Ames** housing dataset (Cock, 2011), with the goal of predicting residential housing prices based on each home's features; as well as the IEEE-CIS **Fraud** Detection dataset (Kaggle, 2022), with the goal of identifying fraudulent transactions from numerous transaction and identity dependent features. Additional details on the datasets can be found in §A.1.

We benchmark SLM against a variety of competitive methods. The mutual information (**MI**) based feature selection baseline uses entropy estimation from k-nearest neighbors distances as described in (Kraskov et al., 2004; Ross, 2014) to estimate MI. Tree-based methods yield Gini importance scores, which can be used for feature selection. For this we benchmark two commonly used methods: random forest (**RF**) (Breiman, 2001), an ensemble of independent trees, and **XGBoost** (Chen & Guestrin, 2016), a scalable end-to-end tree boosting system. We furthermore benchmark against methods as discussed in §2: **LassoNet** (Lemhadri et al., 2019), which uses residual connections to allow the network to learn whether to use any given feature in a particular layer; feature importance ranking based on the **Fischer** score (Gu et al., 2012); principal feature analysis (**PFA**) (Lu et al., 2007a), a PCA-based feature selection method; and **HSIC-Lasso** (Yamada et al., 2014), which uses kernel learning to find non-linear feature interactions. Lastly, we benchmark against **linear** regression, where feature importance is determined by the learned feature coefficients. When available, we use results from (Lemhadri et al., 2019).

4.2 Task performance with feature selection

In practice, the ground truth feature importance is unknown, and accurate predictor performance is typically the end goal, therefore we focus on the task predictive accuracy given a limited number of features as the metric.

First, we consider selecting 50 features across wide range of high dimensional datasets (most with >400 features). Table 1 shows that the SLM consistently yields competitive performance, outperforming all methods in all cases except on Mice and Ames, for which the performance is saturated due to small numbers of original features, making feature selection less relevant. Most feature selection methods are not consistent in their performance, we observe cases that one method might be competitive at one dataset while giving poor results on another. On the other hand, SLM's strong performance is consistent. Interestingly, we observe that there are cases where SLM even beats the baseline of using all features, which can likely be attributed to superior generalization when the limited model capacity is focused on the most salient features.

Next, we focus on the Fraud dataset specifically, a large-scale dataset for the complex task from many heterogeneous features. The adversarial nature of the task, the nature of fraudsters adapt themselves and yielding different fraud patterns, cause the data to be highly non i.i.d. (Grover et al., 2022), thus making feature selection important given that high capacity models can be prone to overfitting and poor generalization. Table 2 shows that SLM outperforms other methods consistently for different number of selected features, and its performance degradation with much less number of features

[1] We use a single layer multi-layer perceptron (MLP) as the predictor, where the number of units is chosen from $[M/3, 2M/3, M, 4M/3]$.

Method	Mice↑	MNIST↑	Fashion↑	Isolet↑	Coil-20↑	Activity↑	Ames↓	Fraud↑
All-features	0.990	0.928	0.833	0.953	0.996	0.956	0.283	0.782
SLM (ours)	0.981	**0.953**	**0.835**	**0.919**	**0.996**	**0.947**	**0.274**	**0.911**
Fisher	0.944	0.813	0.671	0.793	0.986	0.769	0.286	0.743
HSIC-Lasso	0.958	0.870	0.785	0.877	0.972	0.829	**0.273**	0.846
PFA	0.939	0.873	0.793	0.863	0.975	0.779	0.356	0.852
XGBoost	0.968	0.913	0.832	0.879	0.986	0.926	0.403	0.872
MI	0.949	0.882	0.645	0.751	0.976	0.883	0.335	0.711
Linear	0.982	0.452	0.787	0.760	0.983	0.914	0.318	0.871
Anova	**0.995**	0.113	0.719	0.811	0.986	0.901	0.358	0.744
RF	0.967	0.928	0.829	0.892	0.993	0.893	0.303	0.773
Lassonet	0.958	0.873	0.800	0.885	0.991	0.849	0.342	0.842

Table 1: Test performance on real-world benchmarks with 50 selected features. SLM outperforms competitive baselines. The metrics reported are AUC for Fraud, since there is a high class imbalance; hence AUC is reported; the median RMSE on normalized labels is reported for Ames; and accuracy is reported on all other datasets. The arrow next to each dataset indicates whether a higher or lower value is more optimal. These test results are selected based on the best validation set performance during 300 hyperparameter grid search trials.

is smaller. Indeed, the AUC with 20 features out of 432, is >10% better than using all features, indicating improved generalization.

Method	Test AUC ↑		
	20 features	50 features	100 features
SLM (ours)	**89.32**	**91.06**	**91.75**
Anova	71.81	74.41	82.91
RF	72.16	77.29	78.72
Linear	84.32	87.11	87.46
MI	65.37	71.05	74.91
XGBoost	85.45	87.24	81.67

Table 2: Test AUC on Fraud dataset with different feature selection methods, choosing 20, 50 or 100 features. The superiority of SLM persists across different numbers of selected features.

4.3 Ablation Studies

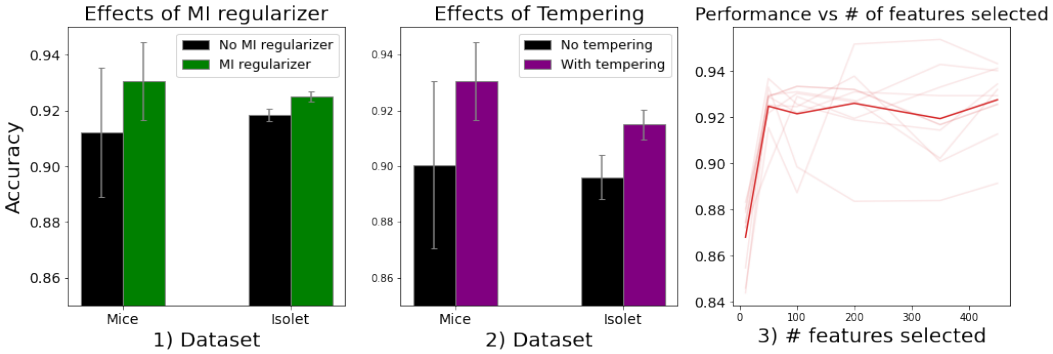

Figure 2: (1) and (2) show ablation studies on the effect of mutual information regularization and tempering the number of features. (3) shows the task accuracy as a function of the number of features selected on the activity dataset. The dark line shows the average of ten random hyperparameter trials, shown with light hue, demonstrating that task performance can be near-optimal even with a small subset of features.

We study the utility of SLM components in this section, in particular the effects of the MI regularizer and tempering the number of features, which gradually decreases the number of selected features

from the full feature set to the target number. The effects are measured by randomly selecting ten hyperparameter settings and a seed, and recording the average performance with or without either MI regularizer or tempering. Fig 2 shows that both MI regularization and tempering positively affect task performance. This is consistent with the theory developed in §3: the MI regularizer encourages maximal mutual information sharing between the labels and the selected features; and tempering allows the model to initialize learning based on all features, rather than a randomly selected subset.

5 Discussion

Feature importance intepretability. SLM learns a sparse mask $\mathcal{M}$ that contains the feature selection coefficients. We show that this approach yields superior results with end-to-end learning by allowing a smooth transition between selecting and un-selecting features. In addition, SLM can also be used for interpretation of global feature importance during inference, yielding the importance ranking of selected features, similar to other commonly-used methods like SHAP (Lundberg & Lee, 2017). This can be highly desired in high-stakes applications such as healthcare or finance, where an importance score can be more useful than simply whether a feature is selected or not.

Feature interdependence during selection. Compared to prior MI-based feature selectors (Ding & Peng, 2005; Fleuret, 2004; Bennasar et al., 2015), SLM accounts for feature inter-dependence by learning inter-dependent probabilities $\{p_j\}_j$ for the selected feature, where $\{p_j\}_j$ jointly maximize the MI between features and labels. Furthermore, SLM learns feature selection and the task objective in an end-to-end way, which alleviates the selection of repetitive features that may individually be predictive, as gradient descent favors increasing the probability for a non-redundant and loss decreasing but less predictive feature over an individually predictive but redundant feature.

Improved model generalization via feature selection. Feature selection can help improve generalization beyond the training set, especially for high capacity models like deep neural networks, which can easily overfit patterns from spurious features that do not hold across training and test data splits (Arjovsky et al., 2019). For instance, Table 1 shows that on some datasets, especially with SLM, prediction on a subset of features can outperform that on all features. Furthermore, Fig 2 shows that task performance can reach near-optimum with even a small subset of all features.

Relation to other MI estimations in deep learning. MI-based objectives have been used in other deep learning methods, such as InfoNCE (Oord et al., 2018), InfoGAN (Chen et al., 2016), and Deep Graph Infomax (Velickovic et al., 2019). To estimate MI, these typically train classifiers on samples drawn from the joint distribution and the product of the marginals. In contrast, SLM proposes a quadratic relaxation of MI optimization, applied to feature selection by converting MI maximization to minimizing a loss function. SLM does not need to sample from the joint or marginal distributions, a potentially computationally intensive process. Furthermore, prior works (Chen et al., 2016; Velickovic et al., 2019) often require a contrastive term in estimation of MI, requiring negative sampling, a process that is not needed in SLM.

Future work. SLM can be integrated into unsupervised or semi-supervised learning, with modified objectives. In addition, our results indicate more significant outperformance for datasets with non i.i.d. characteristics as feature selection can effectively reduce the feature dimensionality and reduce the risk of overfitting to the spurious correlations of irrelevant features. We believe more explorations along this direction can bring even bigger improvements, for example self-supervised constraints can be proposed to learn features with more stationarity characteristics. Lastly, feature selection for data with structure (e.g. temporal or graph) is an interesting extension, which might be based on modifying SLM to apply attention to entire time-series or graph data.

6 Conclusion

In this work, we introduced SLM, a sparse learnable mask based feature selection framework that maximizes the MI between features and labels, while optimizing the training objective end-to-end. The feature mask learned not only allows a smooth, probabilistic selection of features, but also gives way to post-hoc interpretation of feature importance. SLM demonstrates competitive performance against SOTA baselines, and opens door to future applications in domains such as graph or time series representation learning.

REPRODUCIBILITY STATEMENT

§3 gives detailed description of the methodology used. All results derive from repeated cross validation. In particular, the main results reported in 1 derive from extensive grid search during 300 trials for each method-dataset combination, where the test result based on the best validation performance is reported. Code will be open-sourced after the review process.

REFERENCES

Sercan O. Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning, 2019.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2019. URL https://arxiv.org/abs/1907.02893.

Mohamed Bennasar, Yulia Hicks, and Rossitza Setchi. Feature selection using joint mutual information maximisation. *Expert Systems with Applications*, 42(22):8520–8532, 2015. URL https://www.sciencedirect.com/science/article/pii/S0957417415004674.

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

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *KDD*, 2016. doi: 10.1145/2939672.2939785.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016.

De Cock. Ames, iowa: Alternative to the boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19, 2011.

Gonçalo M. Correia, Vlad Niculae, and André F. T. Martins. Adaptively sparse transformers. *CoRR*, abs/1909.00015, 2019. URL http://arxiv.org/abs/1909.00015.

M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(1):131–156, 1997. ISSN 1088-467X. doi: https://doi.org/10.1016/S1088-467X(97)00008-5. URL https://www.sciencedirect.com/science/article/pii/S1088467X97000085.

Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.

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

François Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine learning research*, 5(9), 2004.

Prince Grover, Zheng Li, Jianbo Liu, Jakub Zablocki, Hao Zhou, Julia Xu, and Anqi Cheng. Fdb: Fraud dataset benchmark, 2022. URL https://arxiv.org/abs/2208.14417.

Quanquan Gu, Zhenhui Li, and Jiawei Han. Generalized fisher score for feature selection, 2012. URL https://arxiv.org/abs/1202.3725.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, mar 2003.

Cho-Jui Hsieh, Inderjit Dhillon, Pradeep Ravikumar, and Mátyás Sustik. Sparse inverse covariance matrix estimation using quadratic approximation. *Advances in neural information processing systems*, 24, 2011.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention, 2021.

Kaggle. IEEE-CIS fraud detection, 2022. URL https://www.kaggle.com/c/ieee-fraud-detection.

A. Kraskov, H. Stogbauer, and P. Grassberger. Estimating mutual information. *Phys. Rev. E*, 69, 2004.

Ismael Lemhadri, Feng Ruan, Louis Abraham, and Robert Tibshirani. Lassonet: A neural network with feature sparsity. *arXiv:1907.12207*, 2019.

Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Comput. Surv.*, 50(6), dec 2017. ISSN 0360-0300. doi: 10.1145/3136625. URL https://doi.org/10.1145/3136625.

Yijuan Lu, Ira Cohen, Xiang Sean Zhou, and Qi Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th ACM international conference on Multimedia*, pp. 301–304, 2007a.

Yijuan Lu, Ira Cohen, Xiang Sean Zhou, and Qi Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th ACM International Conference on Multimedia*, MM '07, pp. 301–304, New York, NY, USA, 2007b. Association for Computing Machinery. ISBN 9781595937025. doi: 10.1145/1291233.1291297. URL https://doi.org/10.1145/1291233.1291297.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4765–4774. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pp. 1614–1623. PMLR, 2016.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018. URL https://arxiv.org/abs/1807.03748.

Jing Pan, Vincent Pham, Mohan Dorairaj, Huigang Chen, and Jeong-Yoon Lee. Adversarial validation approach to concept drift problem in automated machine learning systems. *CoRR*, abs/2004.03045, 2020. URL https://arxiv.org/abs/2004.03045.

B. C. Ross. Mutual information between discrete and continuous data sets. *PLoS ONE*, 9, 2014.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8346–8356. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sagawa20a.html.

Robert E Shafer. On quadratic approximation. *SIAM Journal on Numerical Analysis*, 11(2):447–460, 1974.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *ICLR*, 2(3):4, 2019.

Makoto Yamada, Wittawat Jitkrittum, Leonid Sigal, Eric P. Xing, and Masashi Sugiyama. High-dimensional feature selection by feature-wise kernelized lasso. *Neural Computation*, 26(1): 185–207, jan 2014. doi: 10.1162/neco_a_00537.

Sepehr Abbasi Zadeh, Mehrdad Ghadiri, Vahab Mirrokni, and Morteza Zadimoghaddam. Scalable feature selection via distributed diversity maximization. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Guangxiang Zhao, Junyang Lin, Zhiyuan Zhang, Xuancheng Ren, Qi Su, and Xu Sun. Explicit sparse transformer: Concentrated attention through explicit selection, 2019.

A APPENDIX

A.1 DATASET CHARACTERISTICS

Table 3 summarizes the characteristics of the datasets used in the experiments.

Dataset	Number of samples	Number of features	Number of classes
Mice	1080	77	8
MNIST	10000	784	10
Fashion	10000	784	10
Isolet	7797	617	26
Coil-20	1440	400	20
Activity	5744	561	6
Ames	1460	81	N/A
IEEE Fraud	590540	432	2

Table 3: Attributes of datasets used in experiments.

A.2 EXPERIMENTAL DETAILS

As described, we use hyperparameter tuning based on the validation accuracy for all cases. We use the Adam optimizer for training, with exponential decay. For benchmarks from (Lemhadri et al., 2019), for a fair comparison, our hyperparameter search space is same as the original paper. For Fraud, which is larger and more complex, we extend the search space as in Table 4.

Hyperparameter	Search space
Batch size	[512, 1024, 2048, 4096]
Learning rate	[0.001, 0.003, 0.01]
Decay steps	[1000, 10000]
Decay rate	[0.7, 0.9, 0.95, 0.99]
Number of epochs	[30, 100, 200]
Number of hidden units	[50, 100, 200]
Number of layers	[1, 2, 3, 4]
Mask temperature	[0.8, 0.9, 0.95, 0.99]

Table 4: Hyperparameter tuning search space for experiments on the Fraud dataset.

For baselines such as LassoNet, we tune additional method-specific hyperparameters. For instance, for LassoNet, in addition to the hyperparameters, we also tune the ℓ_2 penalization on the skip connection, the hierarchy parameter, and the dropout rate. For XGBoost, we also tune the number of estimators and the maximum tree depth.

A.3 PROOF OF LEMMA 3.2

Lemma 3.2. Given a vector $\boldsymbol{v} \in \mathbb{R}^K$, to obtain F nonzero elements in sparsemax($\boldsymbol{v}$), $\boldsymbol{v}$ should be multiplied with the scalar

$$\boldsymbol{m} = \begin{cases} \left(\sum_{i=1}^{F+1} v_{(i)} - (F+1) * v_{(F+1)}\right)^{-1} & \text{if } |\text{sparsemax}(\boldsymbol{v}) > 0| > F \\ \left(\sum_{i=1}^{F} v_{(i)} - F * v_F\right)^{-1} & \text{if } |\text{sparsemax}(\boldsymbol{v}) > 0| < F, \end{cases} \tag{11}$$

where $v_{(1)} \geq v_{(2)} \ldots \geq v_{(K)}$ denote sorted elements of $\boldsymbol{v}$ in descending order.

Proof. We first show the case when $|\text{sparsemax}(\boldsymbol{v}) > 0| > F$, i.e. the sparsity needs to be increased (the case where spasity needs to be decreased works analogously). By (Martins & Astudillo, 2016), the projection of $\boldsymbol{v}$ onto Δ^{K-1} in Eq 1 takes the form $\text{sparsemax}(\boldsymbol{v}) = [\boldsymbol{v} - \tau(\boldsymbol{v})]_+$, where $[x]_+ =$

$\max\{0, x\}$, and τ takes the form $\tau = \frac{\left(\sum_{i \leq k(\boldsymbol{v})} v_{(i)}\right) - 1}{k(\boldsymbol{v})}$ with $k(\boldsymbol{v})$ defined as the index

$$k(\boldsymbol{v}) \coloneqq \max\left\{k \in \{1, \ldots, K\} \mid 1 + kv_{(k)} > \sum\nolimits_{i \leq k} v_{(i)}\right\}. \tag{12}$$

Hence, increasing the sparsity such that sparsemax outputs only F nonzero elements, i.e. decreasing the index $k(\boldsymbol{v})$ to F, requires finding the smallest $\boldsymbol{m}$ such that $1 + (F+1)\boldsymbol{m}v_{(F+1)} > \sum_{i \leq (F+1)} \boldsymbol{m}v_{(i)}$ does not hold, i.e. $F+1$ must be the first k to fail the condition $1 + kv_{(k)} > \sum_{i \leq k} v_{(i)}$. Rewriting this condition in terms of F we obtain:

$$\begin{aligned} 1 + (F+1)\boldsymbol{m}v_{(F+1)} &> \sum\nolimits_{i \leq (F+1)} \boldsymbol{m}v_{(i)} \\ \text{implies } \; 1 &> \boldsymbol{m}\left(\sum\nolimits_{i \leq (F+1)} v_{(i)} - (F+1)v_{(F+1)}\right) \end{aligned} \tag{13}$$

The smallest $\boldsymbol{m}$ such that condition Eq. 13 does not hold is $\boldsymbol{m} = \left(\left(\sum_{i=1}^{F+1} v_{(i)}\right) - (F+1) * v_{(F+1)}\right)^{-1}$, which given Eq 12 implies $\boldsymbol{m}\boldsymbol{v}$ has F nonzero elements. Analogously, to derive the multiplier for $\boldsymbol{v}$ to decrease sparsemax$(\boldsymbol{v})$ sparsity, we need to increase the index $k(\boldsymbol{v})$ to F. This requires finding the largest $\boldsymbol{m}$ such that $1 + F(\boldsymbol{m}v_F) > \sum_{i \leq F} \boldsymbol{m}v_{(i)}$ holds, which implies: $\boldsymbol{m} = \left(\sum_{i=1}^{F} v_{(i)} - F * v_{(F)}\right)^{-1}$. □

A.4 Proof of Theorem 3.3

Theorem 3.3. With notation as above, maximizing the quadratic relaxation of mutual information $I_q(X, Y)$ is equivalent to minimizing the error $E(X, Y)$. More specifically,

$$E(X, Y) = 1 - \sum\nolimits_{y \in \mathcal{Y}} P_Y(y)^2 - I_q(X, Y)$$

Proof. During training, the model seeks to produce the optimal predictions $R(x, y)$ that *minimize* $E(X, Y)$, while satisfying the constraint $\sum_{y \in \mathcal{Y}} R(x, y) = 1$. Hence we can apply Lagrange multipliers to solve for the optimal $R(x, y)$. Taking the derivatives of $E(X, Y)$ and the constraint $g(X, Y) = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} R(x, y) - |\mathcal{X}|$ with respect to $R(x, y)$:

$$\begin{aligned} E'(X, Y) &= \sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} -2P_{X,Y}(x, y) + 2P_X(x)R(x, y) \\ g'(X, Y) &= \sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} 1 \end{aligned} \tag{14}$$

By Lagrange multiplier theory, for an optimum set of model predictions $R^*(X, Y)$, there exists some λ such that $E'(X, Y) = \lambda g'(X, Y)$. Marginalizing $E'(X, Y)$ over Y yields:

$$E'(X, Y) = \sum\nolimits_{x \in \mathcal{X}} -2P_X(x) + 2P_X(x) = 0 \quad \triangleleft \text{Since } \sum\nolimits_{y \in \mathcal{Y}} R(x, y) = 1$$

Therefore, by Eq 14, $R(x, y) = P_{X,Y}(x, y)/P_X(x)$. Plugging this into Eq 7, we obtain an expression relating the mutual information $I_q(X, Y)$ and the error $E(X, Y)$:

$$\begin{aligned} E(X, Y) &= 1 - 2\sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} P_{X,Y}(x, y)R(x, y) + \sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} P_X(x)R(x, y)^2 \\ &= 1 - 2\sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} P_{X,Y}(x, y)\frac{P_{X,Y}(x, y)}{P_X(x)} + \sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} \frac{P_{X,Y}(x, y)^2}{P_X(x)} \\ &= 1 - \sum\nolimits_{x \in \mathcal{X}, y \in \mathcal{Y}} \frac{P_{X,Y}(x, y)^2}{P_X(x)} \\ &= 1 - \sum\nolimits_{y \in \mathcal{Y}} P_Y(y)^2 - I_q(X, Y) \quad \triangleleft \text{By Eq 6} \end{aligned}$$

Since $P_Y(y)$ is fixed for a given dataset, maximizing $I_q(X, Y)$ is equivalent to minimizing $E(X, Y)$. □