Parallel Feature Selection inspired by Group Testing

Yingbo Zhou* **Utkarsh Porwal***
CSE Department
SUNY at Buffalo
{yingbozh, utkarshp}@buffalo.edu

Ce Zhang
CS Department
University of Wisconsin-Madison
czhang@cs.wisc.edu

Hung Ngo
CSE Department
SUNY at Buffalo
hungngo@buffalo.edu

XuanLong Nguyen
EECS Department
University of Michigan
xuanlong@umich.edu

Christopher Ré
CS Department
Stanford University
chrismre@cs.stanford.edu

Venu Govindaraju
CSE Department
SUNY at Buffalo
govind@buffalo.edu

Abstract

This paper presents a parallel feature selection method for classification that scales up to very high dimensions and large data sizes. Our original method is inspired by group testing theory, under which the feature selection procedure consists of a collection of randomized tests to be performed in parallel. Each test corresponds to a subset of features, for which a scoring function may be applied to measure the relevance of the features in a classification task. We develop a general theory providing sufficient conditions under which true features are guaranteed to be correctly identified. Superior performance of our method is demonstrated on a challenging relation extraction task from a very large data set that have both redundant features and sample size in the order of millions. We present comprehensive comparisons with state-of-the-art feature selection methods on a range of data sets, for which our method exhibits competitive performance in terms of running time and accuracy. Moreover, it also yields substantial speedup when used as a pre-processing step for most other existing methods.

1 Introduction

Feature selection (FS) is a fundamental and classic problem in machine learning [10, 4, 12]. In classification, FS is the following problem: Given a universe U of possible features, identify a subset of features $F \subseteq U$ such that using the features in F one can build a model to best predict the target class. The set F not only influences the model's accuracy, its computational cost, but also the ability of an analyst to understand the resulting model. In applications, such as gene selection from micro-array data [10, 4], text categorization [3], and finance [22], U may contain hundreds of thousands of features from which one wants to select only a small handful for F.

While the overall goal is to have an FS method that is both computationally efficient and statistically sound, natural formulations of the FS problem are known to be NP-hard [2]. For large scale data, scalability is a crucial criterion, because FS often serves not as an end but a means to other sophisticated subsequent learning. In reality, practitioners often resort to heuristic methods, which can broadly be categorized into three types: *wrapper*, *embedded*, and *filter* [10, 4, 12]. In the wrapper method, a classifier is used as a black-box to test on any subset of features. In filter methods no classifier is used; instead, features are selected based on generic statistical properties of the (labeled)

* * denotes equal contribution

data such as mutual information and entropy. Embedded methods have built in mechanisms for FS as an integral part of the classifier training. Devising a mathematically rigorous framework to explain and justify FS heuristics is an emerging research area. Recently Brown et al. [4] considered common FS heuristics using a formulation based on conditional likelihood maximization.

The primary contribution of this paper is a new framework for *parallelizable* feature selection, which is inspired by the theory of *group testing*. By exploiting parallelism in our test design we obtain a FS method that is easily scalable to millions of features and samples or more, while preserving useful statistical properties in terms of classification accuracy, stability and robustness. Recall that group testing is a combinatorial search paradigm [7] in which one wants to identify a small subset of "positive items" from a large universe of possible items. In the original application, items are blood samples of WWII draftees and an item is *positive* if it is infected with syphilis. Testing individual blood sample is very expensive; the group testing approach is to distribute samples into pools in a smart way. If a pool is tested negative, then all samples in the pool are negative. On the other hand, if a pool is tested positive then at least one sample in the pool is positive. We can think of the FS problem in the group testing framework: there is a presumably small, *unknown* subset F of *relevant features* in a large universe of N features. Both FS and group testing algorithms perform the same basic operation: apply a "test" to a subset T of the underlying universe; this test produces a *score*, $s(T)$, that is designed to measure the quality of the features T (or return positive/negative in the group testing case). From the collection of test scores the relevant features are supposed to be identified. Most existing FS algorithms can be thought of as *sequential* instantiations in this framework[1]: we select the set T to test based on the scores of previous tests. For example, let $\mathbf{X} = (X_1, \ldots, X_N)$ be a collection of features (variables) and Y be the class label. In the *joint mutual information* (JMI) method [25], the feature set T is grown sequentially by adding one feature at each iteration. The next feature's score, $s(X_k)$, is defined relative to the set of features already selected in T: $s(X_k) = \sum_{X_j \in T} I(X_k, X_j; Y)$. As each such scoring operation takes a non-negligible amount of time, a sequential method may take a long time to complete.

A key insight is that group testing needs not be done sequentially. With a good pooling design, all the tests can be performed *in parallel* in which we determine the pooling design *without* knowing any pool's test outcome. From the vector of test outcomes, one can identify exactly the collection of positive blood samples. Parallel group testing, commonly called *non-adaptive group testing* (NAGT) is a natural paradigm and has found numerous applications in many areas of mathematics, computer Science, and biology [18]. It is natural to wonder whether a "parallel" FS scheme can be designed for machine learning in the same way NAGT was possible: all feature sets T are specified *in advance*, without knowing the scores of any other tests, and from the final collection of scores the features are identified. This paper initiates a mathematical investigation of this possibility.

At a high level, our *parallel feature selection* (PFS) scheme has three inter-related components: (1) the *test design* indicates the collection of subsets of features to be tested, (2) the *scoring function* $s : 2^{[N]} \to \mathbb{R}$ that assigns a score to each test, and (3) the *feature identification algorithm* that identifies the final selected feature set from the test scores. The design space is thus very large. Every combination of the three components leads to a new PFS scheme.[2] We argue that PFS schemes are preferred over sequential FS for two reasons:

1. *scalability*, the tests in a PFS scheme can be performed in parallel, and thus the scheme can be scaled to large datasets using standard parallel computing techniques, and
2. *stability*, errors in individual trials do not affect PFS methods as dramatically as sequential methods. In fact, we will show in this paper that increasing the number of tests improves the accuracy of our PFS scheme.

We propose and study one such PFS approach. We show that our approach has comparable (and sometimes better) empirical quality compared to previous heuristic approaches while providing sound statistical guarantees and substantially improved scalability.

Our technical contributions We propose a simple approach for the first and the third components of a PFS scheme. For the second component, we prove a sufficient condition on the scoring function under which the feature identification algorithm we propose is guaranteed to identify *exactly* the set

[1] A notable exception is the MIM method, which is easily parallelizable and can be regarded as a special implementation of our framework

[2] It is important to emphasize that this PFS framework is applicable to both filter and wrapper approaches. In the wrapper approach, the score $s(T)$ might be the training error of some classifier, for instance.

of original (true) features. In particular, we introduce a notion called C-*separability*, which roughly indicates the strength of the scoring function in separating a relevant feature from an irrelevant feature. We show that when s is C-separable and we can estimate s, we are able to guarantee exact recovery of the right set of features with high probability. Moreover, when $C > 0$, the number of tests can be asymptotically logarithmic in the number of features in U.

In theory, we provide sufficient conditions (a Naïve Bayes assumption) according to which one can obtain separable scoring functions, including the KL divergence and mutual information (MI). In practice, we demonstrate that MI is separable even when the sufficient condition does not hold, and moreover, on generated synthetic data sets, our method is shown recover *exactly* the relevant features. We proceed to provide a comprehensive evaluation of our method on a range of real-world data sets of both large and small sizes. It is the large scale data sets where our method exhibits superior performance. In particular, for a huge relation extraction data set (TAC-KBP) that has millions redundant features and samples, we outperform all existing methods in accuracy and time, in addition to generating plausible features (in fact, many competing methods could not finish the execution). For the more familiar NIPS 2013 FS Challenge data, our method is also competitive (best or second-best) on the two largest data sets. Since our method hinges on the accuracy of score functions, which is difficult achieve for small data, our performance is more modest in this regime (staying in the middle of the pack in terms of classification accuracy). Nonetheless, we show that our method can be used as a preprocessing step for other FS methods to eliminate a large portion of the feature space, thereby providing substantial computational speedups while retaining the accuracy of those methods.

2 Parallel Feature Selection

The general setting Let N be the total number of input features. For each subset $T \subseteq [N] := \{1, \ldots, N\}$, there is a *score* $s(T)$ normalized to be in $[0, 1]$ that assesses the "quality" of features in T. We select a collection of t *tests*, each of which is a subset $T \subseteq [N]$ such that from the scores of all tests we can identify the *unknown* subset F of d relevant variables that are most important to the classification task. We encode the collection of t tests with a binary matrix $\mathbf{A} = (a_{ij})$ of dimension $t \times N$, where $a_{ij} = 1$ iff feature j belongs to test i. Corresponding to each row i of $\mathbf{A}$ is a "test score" $s_i = s(\{j \mid a_{ij} = 1\}) \in [0, 1]$. Specifying $\mathbf{A}$ is called *test design*, identifying F from the score vector $(s_i)_{i \in [t]}$ is the job of the *feature identification algorithm*. The scheme is inherently parallel because *all* the tests must be specified in advance and executed in parallel; then the features are selected from all the test outcomes.

Test design and feature identification Our test design and feature identification algorithms are *extremely simple*. We construct the test matrix $\mathbf{A}$ randomly by putting a feature in the test with probability p (to be chosen later). Then, from the test scores we *rank* the features and select d top-ranked features. The ranking function is defined as follows. Given a $t \times N$ test matrix $\mathbf{A}$, let $\mathbf{a}^j$ denote its jth column. The dot-product $\langle \mathbf{a}^j, \mathbf{s} \rangle$ is the total score of all the tests that feature j participates in. We define $\rho(j) = \langle \mathbf{a}^j, \mathbf{s} \rangle$ to be the *rank* of feature j with respect to the test matrix $\mathbf{A}$ and the score function s.

The scoring function The crucial piece stiching together the entire scheme is the scoring function. The following theorem explains why the above test design and feature identification strategy make sense, as long as one can choose a scoring function s that satisfies a natural *separability* property. Intuitively, separable scoring functions require that adding more hidden features into a test set increase its score.

Definition 2.1 (Separable scoring function)**.** Let $C \geq 0$ be a real number. The score function $s : 2^{[N]} \to [0, 1]$ is said to be C-*separable* if the following property holds: for every $f \in F$ and $\tilde{f} \notin F$, and for every $T \subseteq [N] - \{f, \tilde{f}\}$, we have $s(T \cup \{f\}) - s(T \cup \{\tilde{f}\}) \geq C$.

In words, with a separable scoring function adding a relevant feature should be better than adding an irrelevant feature to a given subset T of features. Due to space limination, the proofs of the following theorem, propositions, and corollaries can be found in the supplementary materials. The essence of the idea is that, when s can separate relevant features from irrelevant features, with high probability a relevant feature will be ranked higher than an irrelevant feature. Hoeffding's inequality is then used to bound the number of tests.

Theorem 2.2. *Let* $\mathbf{A}$ *be the random* $t \times N$ *test matrix obtained by setting each entry to be* 1 *with probability* $p \in [0,1]$ *and* 0 *with probability* $1-p$*. If the scoring function* s *is* C*-separable, then the expected rank of a feature in* F *is at least the expected rank of a feature not in* F*.*

Furthermore, if $C > 0$*, then for any* $\delta \in (0,1)$*, with probability at least* $1-\delta$ every *feature in* F *has rank higher than* every *feature not in* F*, provided that the number of tests* t *satisfies*

$$t \geq \frac{2}{C^2 p^2 (1-p)^2} \log\left(\frac{d(N-d)}{\delta}\right). \tag{1}$$

By setting $p = 1/2$ in the above theorem, we obtain the following. It is quite remarkable that, assuming we can estimate the scores accurately, we only need about $O(\log N)$ tests to identify F.

Corollary 2.3. *Let* $C > 0$ *be a constant such that there is a* C*-separable scoring function* s*. Let* $d = |F|$*, where* F *is the set of hidden features. Let* $\delta \in (0,1)$ *be an arbitrary constant. Then, there is a distribution of* $t \times N$ *test matrices* $\mathbf{A}$ *with* $t = O(\log(d(N-d)/\delta))$ *such that, by selecting a test matrix randomly from the distribution, the* d *top-ranked features are* exactly *the hidden features with probability at least* $1-\delta$*.*

Of course, in reality estimating the scores accurately is a very difficult problem, both statistically and computationally, depending on what the scoring function is. We elaborate more on this point below. But first, we show that separable scoring functions exist, under certain assumption about the underlying distribution.

Sufficient conditions for separable scoring functions We demonstrate the existence of separable scoring functions given some sufficient conditions on the data. In practice, loss functions such as classification error and other surrogate losses may be used as scoring functions. For binary classification, information-theoretic quantities such as Kullback-Leibler divergence, Hellinger distance and the total variation — all of which special cases of f-divergences [5, 1] — may also be considered. For multi-class classification, mutual information (**MI**) is a popular choice.

The data pairs $(\boldsymbol{X}, Y)$ are assumed to be iid samples from a joint distribution $P(\boldsymbol{X}, Y)$. The following result shows that under the so-called "naive Bayes" condition, i.e., all components of random vector $\boldsymbol{X}$ are conditionally independent given label variable Y, the Kullback-Leibler distance is a separable scoring function in a binary classification setting:

Proposition 2.4. *Consider the binary classification setting, i.e.,* $Y \in \{0,1\}$ *and assume that the naive Bayes condition holds. Define score function to be the Kullback-Leibler divergence:*

$$s(T) := KL(P(\boldsymbol{X}_T|Y=0)||P(\boldsymbol{X}_T|Y=1)).$$

Then s *is a separable scoring function. Moreover,* s *is* C*-separable, where* $C := \min_{f \in F} s(f)$*.*

Proposition 2.5. *Consider the multi-class classification setting, and assume that the naive Bayes condition holds. Moreover, for any pair* $f \in F$ *and* $\tilde{f} \notin F$*, the following holds for any* $T \subseteq [N] - \{f, \tilde{f}\}$

$$I(X_f; Y) - I(X_f; \boldsymbol{X}_T) \geq I(X_{\tilde{f}}; Y) - I(X_{\tilde{f}}; \boldsymbol{X}_T).$$

Then, the MI function $s(T) := I(\mathbf{X}_T; Y)$ *is a separable scoring function.*

We note the naturalness of the condition so required, as quantity $I(X_f; Y) - I(X_f; X_T)$ may be viewed as the relevance of feature f with respect to the label Y, subtracted by the redundancy with other existing features T. If we assume further that $\boldsymbol{X}_{\tilde{f}}$ is independent of both $\boldsymbol{X}_T$ and the label Y, and there is a positive constant C such that $I(X_f; Y) - I(X_f; X_T) \geq C$ for any $f \in F$, then $s(T)$ is obviously a C-separable scoring function. It should be noted that *the naive Bayes conditions are sufficient, but not necessary* for a scoring function to be C-separable.

Separable scoring functions for filters and wrappers. In practice, information-based scoring functions need to be estimated from the data. Consistent estimators of scoring functions such as KL divergence (more generally f-divergences) and MI are available (e.g., [20]). This provides the theoretical support for applying our test technique to filter methods: when the number of training data is sufficiently large, a consistent estimate of a separable scoring function must also be a separable scoring function. On the other hand, a wrapper method uses a classification algorithm's performance as a scoring function for testing. Therefore, the choice of the underlying (surrogate) loss function plays a critical role. The following result provides the existence of loss functions which induce separable scoring functions for the wrapper method:

Proposition 2.6. *Consider the binary classification setting, and let* $P_0^T := P(\boldsymbol{X}_T|Y=0)$, $P_1^T := P(\boldsymbol{X}_T|Y=1)$. *Assume that an* f*-divergence of the form:* $s(T) = \int \phi(dP_0^T/dP_1^T)dP_1^T$ *is a separable scoring function for some convex function* $\phi : \mathbb{R}_+ \to \mathbb{R}$. *Then there exists a surrogate loss function* $l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ *under which the minimum* l*-risk:* $R_l(T) := \inf_g \mathbb{E}\left[l(Y, g(\boldsymbol{X}_T))\right]$ *is also a separable scoring function. Here the infimum is taken over all measurable classifier functions* g *acting on feature input* $\boldsymbol{X}_T$, $\mathbb{E}$ *denotes expectation with respect to the joint distribution of* $\boldsymbol{X}_T$ *and* Y.

This result follows from Theorem 1 of [19], who established a precise correspondence between f-divergences defined by convex ϕ and equivalent classes of surrogate losses l. As a consequence, if the Hellinger distance between P_0^T and P_1^T is separable, then the wrapper method using the Adaboost classifier corresponds to a separable scoring function. Similarly, a separable Kullback-Leibler divergence implies that of a logistic regression based wrapper; while a separable variational distance implies that of a SVM based wrapper.

3 Experimental results

3.1 Synthetic experiments

In this section, we synthetically illustrate that separable scoring functions exist and our PFS framework is sound beyond the Naïve Bayes assumption (**NBA**). We first show that MI is C-separable for large C even when the NBA is violated. The NBA was only needed in Propositions 2.4 and 2.5 in order for the proofs to go through. Then, we show that our framework recovers *exactly* the relevant features for two common classes of input distributions.

We generate $1,000$ data points from two separated 2-D Gaussians with the same covariance matrix but different means, one centered at $(-2,-2)$ and the other at $(2,2)$. We start with the identity covariance matrix, and gradually change the off diagonal element to -0.999, representing highly correlated features. Then, we add 1,000 dimensional zero mean Gaussian noise with the same covariance matrix, where the diagonal is 1 and the off-diagonal elements increases from 0 gradually to 0.999. We then calculate the MI between two features and the class label, and the two features are selected in three settings: 1) the two genuine dimensions; 2) one of the genuine feature and one from the noisy dimensions; 3) two random pair from the noisy dimensions. The MI that we get from these three conditions is shown in Figure 1. It is clear from this figure MI is a separable scoring function, despite the fact that the NBA is violated.

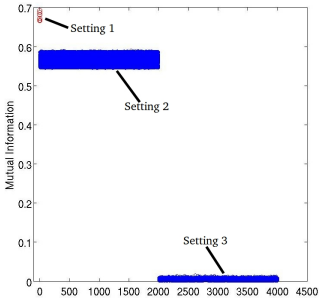


Figure 1: Illustration of MI as a separable scoring function for the case of statistically dependent features. The top left point shows the scores for the 1st setting; the middle points shows the scores for the 2nd setting; and the bottom points shows the scores for the 3rd setting.

We also synthetically evaluated our entire PFS idea, using two multinomials and two Gaussians to generate two binary classification task data. Our PFS scheme is able to capture *exactly* the relevant features in most cases. Details are in the supplementary material section due to lack of space.

3.2 Real-world data experiment results

This section evaluates our approach in terms of accuracy, scalability, and robustness accross a range of real-world data sets: small, medium, and large. We will show that our PFS scheme works very well on medium and large data sets; because, as was shown in Section 3.1, with sufficient data to estimate test scores, we expect our method to work well in terms of accuracy. On the small datasets, our approach is only competitive and does not dominate existing approaches, due to the lack of data to estimate scores well. However, we show that we can still use our PFS scheme as a pre-processing step to filter down the number of dimensions; this step reduces the dimensionality, helps speed up existing FS methods from 3-5 times while keeps their accuracies.

3.2.1 The data sets and competing methods

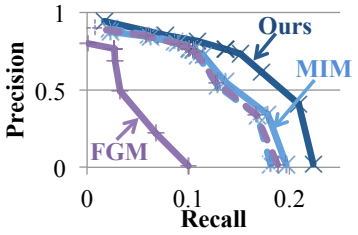
Large: TAC-KBP is a large data set with the number of samples and dimensions in the millions[3]; its domain is on relation extraction from natural language text. **Medium:** GISETTE and MADE-

[3] http://nlp.cs.qc.cuny.edu/kbp/2010/

LON are two largest data sets from the NIPS 2003 feature selection challenge[4], with the number of dimensions in the thousands. **Small:** Colon, Leukemia, Lymph, NCI9, and Lung are chosen from the small Micro-array datasets [6], along with the UCI datasets[5]. These sets typically have a few hundreds to a few thousands variables, with only tens of data samples.

We compared our method with various baseline methods including mutual information maximization[14] (MIM), maximum relevancy minimum redundancy[21] (MRMR), conditional mutual information maximization[9] (CMIM), joint mutual information[25] (JMI), double input symmetrical relevance[16] (DISR), conditional infomax feature extraction[15] (CIFE), interaction capping[11] (ICAP), fast correlation based filter[26] (FCBF), local learning based feature selection [23] (LOGO), and feature generating machine [24] (FGM).

3.2.2 Accuracy



(a) P/R-Curve

Spouse	MemberOf	TopMember
wife	leader	head
pictures	member	executive
husband	rebels	chairman
married	commander	general
widower	iraq leader	president

(b) Keywords in top features

Figure 2: Result from different methods on TAC-KBP dataset. (a) Precision/Recall of different methods; (b) Top-5 keywords appearing in the Top-20 features selected by our method. Dotted lines in (a) are FGM (or MIM) with our approach as pre-processing step.

Accuracy results on large data set. As shown in Figure 2(a), our method dominates both MIM and FGM. Given the same precision, our method achieves 2-14$\times$ higher recall than FGM, and 1.2-2.4$\times$ higher recall than MIM. Other competitors do not finish execution in 12 hours. We compare the top-features produced by our method and MIM, and find that our method is able to extract features that are strong indicators only when they are combined with other features, while MIM, which tests features individually, ignores this type of combination. We then validate that the features selected by our method makes intuitive sense. For each relation, we select the top-20 features and report the keyword in these features.[6] As shown in Figure 2(b), these top-features selected by our method are good indicators of each relation. We also observe that using our approach as the pre-processing step improves the quality of FGM significantly. In Figure 2(a) (the broken lines), we run FGM (MIM) on the top-10K features produced by our approach. We see that running FGM with pre-processing achieves up to 10$\times$ higher recall given the same precision than running FGM on all 1M features.

Accuracy results on medium data sets Since the focus of the evaluation is to analyze the efficacy of feature selection approaches, we employed the same strategy as Brown et al.[4] *i.e.* the final classification is done using k-nearest neighbor classifier with k fixed to three, and applied Euclidean distance[7].

We denote our method by F_k (and W_k), where F denotes filter (and W denotes wrapper method). k denotes the number of tests (*i.e.* let N be the dimension of data, then the total number of tests is kN). We bin each dimension of the data into five equal distanced bins when the data is real valued, otherwise the data is not processed[8]. MI is used as the scoring function for filter method, and log-likelihood is used for scoring the wrapper method. The wrapper we used is logistic regression[9].

For GISETTE we select up to 500 features and for MADELON we select up to 100 features. To get the test results, we use the features according to the smallest validation error for each method, and the results on test set are illustrated in table 4.

[4] http://www.nipsfsc.ecs.soton.ac.uk/datasets/

[5] http://archive.ics.uci.edu/ml/

[6] Following the syntax used by Mintz et al. [17], if a feature has the form $[\Uparrow_{poss}\ wife\ \Downarrow_{prop\_of}]$, we report the keyword as *wife* in Figure 2(b).

[7] The classifier for FGM is linear support vector machine (SVM), since it optimized for the SVM criteria.

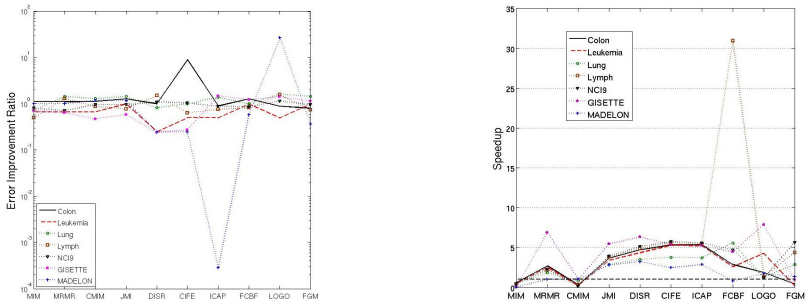
[8] For SVM based method, the real valued data is not processed, and all data is normalized to have unit length.

[9] The logistic regressor used in wrapper is only to get the testing scores, the final classification scheme is still k-NN.

Table 1: Test set balanced error rate (%) from different methods on NIPS datasets

Datasets	Best Perf.	2nd Best Perf.	3rd Best Perf.	Median Perf.	Ours (F_3)	Ours (W_3)	Ours (F_{10})	Ours (W_{10})
GISETTE	**2.15**	3.06	3.09	3.86	4.85	2.72	4.69	2.89
MADELON	10.61	11.28	12.33	25.92	22.61	**10.17**	18.39	**10.50**

Accuracy results on the small data sets. As expected, due to the lack of data to estimate scores, our accuracy performance is average for this data set. Numbers can be found in the supplementary materials. However, as suggested by theorem A.3 (in supplementary materials), our method can also be used as a preprocessing step for other feature selection method to eliminate a large portion of the features. In this case, we use the filter methods to filter out $e + 0.1$ of the input features, where e is the desired proportion of the features that one wants to reserve.



(a) (b)

Figure 3: Result from real world datasets: a) curve showing the ratio between the errors of various methods applied on original data and on filtered data, where a large portion of the dimension is filtered out (value larger than one indicates performance improvement); b) the speed up we get by applying our method as a pre-processing method on various methods across different datasets, the flat dashed line indicates the location where the speed up is one.

Using our method as preprocessing step achieves 3-5 times speedup as compare to the time spend by original methods that take multiple passes through the datasets, and keeps or improves the performance in most of the cases (see figure 3 a and b). The actual running time can be found in supplementary materials.

3.2.3 Scalability

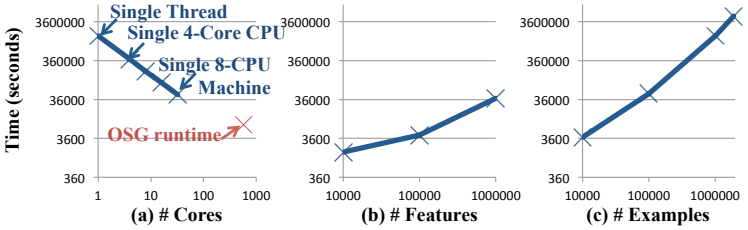


Figure 4: Scalability Experiment of Our Approach

We validate that our method is able to run on large-scale data set efficiently, and the ability to take advantage of parallelism is the key to its scalability.

Experiment Setup Given the TAC-KBP data set, we report the execution time by varying the degree of parallelism, number of features, and number of examples. We first produce a series of data sets by sub-sampling the original data set with different number examples ($\{10^4, 10^5, 10^6\}$) and number of features ($\{10^4, 10^5, 10^6\}$). We also try different degree of parallelism by running our approach using a single thread, 4-threads on a 4-core CPU, 32 threads on a single 8-CPU (4-core/CPU) machine, and multiple machines available in the national Open Science Grid (OSG). For each combination of number of features, number of examples, and degree of parallelism, we estimate the throughput as the number of tests that we can run in 1 second, and estimate the total running time accordingly. We also ran our largest data set (10^6 rows and 10^6 columns) on OSG and report the actual run time.

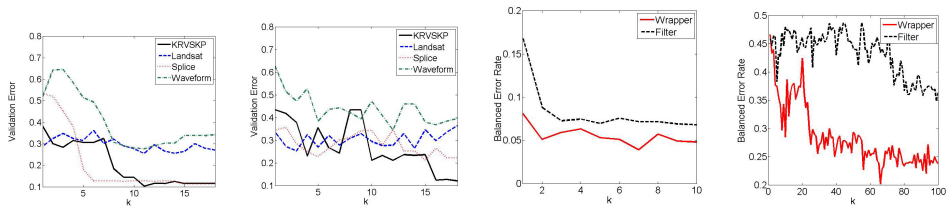
Degree of Parallelism Figure 4(a) reports the (estimated) run time on the largest data set (10^6 rows and 10^6 columns) with different degree of parallelism. We first observe that running our

approach requires non-trivial amount of computational resources–if we only use a single thread, we need about 400 hours to finish our approach. However, the running time of our approach decreases linearly with the number of cores that we used. If we run our approach on a single machine with 32 cores, it finishes in just 11 hours. This linear speed-up behavior allows our approach to scale to very large data set–when we run our approach on the national Open Science Grid, we observed that our approach is able to finish in 2.2 hours (0.7 hours for actual execution, and 1.5 hours for scheduling overhead).

The Impact of Number of Features and Number of Examples Figure 4(b,c) report the run time with different number of features and number of examples, respectively. In Figure 4(b), we fix the number of examples to be 10^5, and vary the number of features, and in Figure 4(c), we fix the number of features to be 10^6 and vary the number of examples. We see that as the number of features or the number of examples increase, our approach uses more time; however, the running time never grows super-linearly. This behavior implies the potential of our approach to scale to even larger data sets.

3.2.4 Stability and robustness

Our method exhibits several robustness properties. In particular, the proof of Theorem 2.2 suggests that as the number of tests are increased the performance also improves. Therefore, in this section we empirically evaluate this observation. We picked four datasets: KRVSKP, Landset, Splice and Waveform from the UCI datasets and both NIPS datasets.



(a) (b) (c) (d)

Figure 5: Change of performance with respect of number of tests on several UCI datasets with (a) filter and (b) wrapper methods; and (c) GISETTE and (d) MADELON datasets.

The trend is pretty clear as can be observed from figure 5. The performance of both wrapper and filter methods improves as we increase the number of tests, which can be attributed to the increase of robustness against inferior estimates for the test scores as the number of tests increases. In addition, apart from MADELON dataset, the performance converges fast, normally around $k = 10 \sim 15$.

Additional stability experiments can be found in the supplementary materials, where we evaluate ours and other methods in terms of consistency index.

Acknowledgements

CR acknowledges the support of DARPA XDATA Program under No. FA8750-12-2-0335 and DEFT Program under No. FA8750-13-2-0039, DARPAs MEMEX program, the NSF CAREER Award under No. IIS-1353606 and EarthCube Award under No. ACI-1343760, the Sloan Research Fellowship, the ONR under awards No. N000141210041 and No. N000141310129, the Moore Foundation, American Family Insurance, Google, and Toshiba. HN is supported by NSF grants CNF-1409551, CCF-1319402, and CNF-1409551. XN is supported in part by NSF grants CCF-1115769, ACI 1342076 , NSF CAREER award under DMS-1351362, and CNS-1409303. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, AFRL, NSF, ONR, or the U.S. government.

References

[1] S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *J. Royal Stat. Soc. Series B*, 28:131–142, 1966.

[2] Edoardo Amaldi and Viggo Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems, 1997.

[3] Ron Bekkerman, Ran El-Yaniv, Naftali Tishby, and Yoad Winter. Distributional word clusters vs. words for text categorization. *J. Mach. Learn. Res.*, 3:1183–1208, March 2003.

[4] Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *JMLR*, 13:27–66, 2012.

[5] I. Csiszár. Information-type measures of difference of probability distributions and indirect observation. *Studia Sci. Math. Hungar*, 2:299–318, 1967.

[6] C. H. Q. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *J. Bioinformatics and Computational Biology*, pages 185–206, 2005.

[7] Ding-Zhu Du and Frank K. Hwang. *Combinatorial group testing and its applications*, volume 12 of *Series on Applied Mathematics*. World Scientific Publishing Co. Inc., River Edge, NJ, second edition, 2000.

[8] Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, Cambridge, 2009.

[9] Francois Fleuret and Isabelle Guyon. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

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

[11] A. Jakulin and I. Bratko. *Machine learning based on attribute interactions: Ph.D. dissertation*. 2005.

[12] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, December 1997.

[13] Ludmila I. Kuncheva. A stability index for feature selection. In *Artificial Intelligence and Applications*, pages 421–427, 2007.

[14] David D. Lewis. Feature selection and feature extraction for text categorization. In *In Proceedings of Speech and Natural Language Workshop*, pages 212–217. Morgan Kaufmann, 1992.

[15] Dahua Lin and Xiaoou Tang. Conditional infomax learning: An integrated framework for feature extraction and fusion. In *ECCV (1)*, pages 68–82, 2006.

[16] P. E. Meyer and G. Bontempi. On the use of variable complementarity for feature selection in cancer classification. In *Proceedings of EvoWorkshop*, pages 91–102. Springer-Verlag, 2006.

[17] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL/IJCNLP*, pages 1003–1011, 2009.

[18] Hung Q. Ngo, Ely Porat, and Atri Rudra. Efficiently decodable compressed sensing by list-recoverable codes and recursion. In *Proceedings of STACS*, volume 14, pages 230–241, 2012.

[19] X. Nguyen, M. J. Wainwright, and M. I. Jordan. On surrogate losses and f-divergences. *Annals of Statistics*, 37(2):876–904, 2009.

[20] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating divergence functionals and the likelihood ratio by by convex risk minimization. *IEEE Trans. on Information Theory*, 56(11):5847–5861, 2010.

[21] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on PAMI*, 27:1226–1238, 2005.

[22] Hervé Stoppiglia, Gérard Dreyfus, Rémi Dubois, and Yacine Oussar. Ranking a random feature for variable and feature selection. *J. Mach. Learn. Res.*, 3:1399–1414, March 2003.

[23] Y. Sun, S. Todorovic, and S. Goodison. Local-learning-based feature selection for high-dimensional data analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1610–1626, Sept 2010.

[24] Mingkui Tan, Li Wang, and Ivor W. Tsang. Learning sparse svm for feature selection on very high dimensional datasets. In *ICML*, pages 1047–1054, 2010.

[25] Howard Hua Yang and John E. Moody. Data visualization and feature selection: New algorithms for nongaussian data. In *NIPS*, pages 687–702, 1999.

[26] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

[27] Ce Zhang, Feng Niu, Christopher Ré, and Jude W. Shavlik. Big data versus the crowd: Looking for relationships in all the right places. In *ACL (1)*, pages 825–834, 2012.

Supplementary Material

A Missing proofs from Section 2

A.1 Proof of Theorem 2.2

Proof. Fix a feature $f \in F$ and a feature $\tilde{f} \notin F$. Recall we used $\mathbf{a}^j$ to denote the jth column of the test matrix $\mathbf{A}$. For each row $i \in [t]$, define the random variable $X_i := a_{if}s_i - a_{i\tilde{f}}s_i$, which is the contribution of the ith test to the difference $\rho(f) - \rho(\tilde{f})$. In particular,

$$\rho(f) - \rho(\tilde{f}) = \langle \mathbf{a}^f, \mathbf{s} \rangle - \langle \mathbf{a}^{\tilde{f}}, \mathbf{s} \rangle = \sum_{i=1}^{t} X_i.$$

The variables X_i are identically and independently distributed. We first estimate $\mathbb{E}[X_i]$. Let T_i denote the ith test, i.e. $T_i = \{j \mid a_{ij} = 1\}$. Then, it is easy to see that $\mathbb{E}[X_i \mid \text{both } f, \tilde{f} \in T_i] = 0$, and $\mathbb{E}[X_i \mid \text{both } f, \tilde{f} \notin T_i] = 0$. Thus, letting $q = 1 - p$ to shorten the notations, we have

$$\begin{aligned}
& \mathbb{E}[X_i] \\
= \;& \mathbb{E}[X_i \mid f \in T_i, \tilde{f} \notin T_i] \cdot \mathbb{P}[f \in T_i, \tilde{f} \notin T_i] + \\
& \mathbb{E}[X_i \mid f \notin T_i, \tilde{f} \in T_i] \cdot \mathbb{P}[f \notin T_i, \tilde{f} \in T_i] \\
= \;& pq\mathbb{E}[X_i \mid f \in T_i, \tilde{f} \notin T_i] + pq\mathbb{E}[X_i \mid f \notin T_i, \tilde{f} \in T_i] \\
= \;& pq \left(\sum_{T \subseteq [N] - \{f, \tilde{f}\}} s(T_i) \cdot \mathbb{P}[T_i = T \cup \{f\} \mid f \in T_i, \tilde{f} \notin T_i] \right) \\
& -pq \left(\sum_{T \subseteq [N] - \{f, \tilde{f}\}} s(T_i) \cdot \mathbb{P}[T_i = T \cup \{\tilde{f}\} \mid f \notin T_i, \tilde{f} \in T_i] \right) \\
= \;& pq \left(\sum_{T \subseteq [N] - \{f, \tilde{f}\}} s(T \cup \{f\}) \cdot p^{|T|} q^{N-2-|T|} \right) \\
& -pq \left(\sum_{T \subseteq [N] - \{f, \tilde{f}\}} s(T \cup \{\tilde{f}\}) \cdot p^{|T|} q^{N-2-|T|} \right) \\
= \;& pq \left(\sum_{T \subseteq [N] - \{f, \tilde{f}\}} \big(s(T \cup \{f\}) - s(T \cup \{\tilde{f}\})\big) \cdot p^{|T|} q^{N-2-|T|} \right) \\
\geq \;& Cpq \left(\sum_{T \subseteq F - \{f, \tilde{f}\}} p^{|T|} q^{N-2-|T|} \right) = Cpq.
\end{aligned}$$

Consequently, when $C \geq 0$ every term in the summation above is non-negative, implying that $\mathbb{E}[X_i] \geq 0$, which in turn implies $\mathbb{E}[\rho(f)] \geq \mathbb{E}[\rho(\tilde{f})]$. Since the X_i are i.i.d. in $[-1, 1]$, by Hoeffding's inequality [8], when $C > 0$ we have

$$\mathbb{P}[\rho(f) - \rho(\tilde{f}) \leq 0] \leq \exp\left\{ \frac{-2t^2C^2p^2q^2}{4t} \right\}.$$

The probability that there is *some* pair $f \in F, \tilde{f} \notin F$ for which $\rho(f) - \rho(\tilde{f}) \leq 0$ is thus at most $d(N-d)\exp\left\{ \frac{-tC^2p^2q^2}{2} \right\}$. The last expression is at most δ when t satisfies (1). □

A.2 Proof of Proposition 2.4 and Proposition 2.5

A special case of separable scoring function is a scoring function satisfying a monotonicity condition: if a subset T_2 of features has more relevant features than another subset T_1, then $s(T_2)$ has to be better than $s(T_1)$.

Definition A.1 (Monotone scoring function)**.** Let $C \geq 0$ be a real number. The score function $s : 2^{[N]} \to [0, 1]$ is said to be *C-monotone* if the following property holds: for any two subsets $T_1, T_2 \subseteq [N]$ such that $T_1 \cap F$ is a proper subset of $T_2 \cap F$, we have $s(T_2) - s(T_1) \geq C$. The 0-monotone scoring functions are called *monotone* for short.

Proposition A.2. *If s is a C-monotone scoring function, then it is a C-separable scoring function.*

Proof. Fix $f \in F$, $\tilde{f} \notin F$, and $T \subset [N] - \{f, \tilde{f}\}$. Let $T_2 = T \cup \{f\}$ and $T_1 = T \cup \{\tilde{f}\}$. Then, $T_1 \cap F \subset T_2 \cap F$. Hence, $s(T_2) - s(T_1) \geq C$, as desired. □

From the above proposition, to show that a function is separable it is sufficient to show that it is monotone.

Proof of Proposition 2.4. Due to conditional independence, it can be checked that $s(T) = \sum_{f \in T} s(f)$. From this the claim can be easily verified. □

Proof of Proposition 2.5. Due to a basic property of mutual information, $s(T \cup \{f\}) = I(\boldsymbol{X}_{T\cup\{f\}}; Y) = H(\boldsymbol{X}_{T\cup\{f\}}) - H(\boldsymbol{X}_{T\cup\{f\}}|Y) = H(\boldsymbol{X}_{T\cup\{f\}}) - H(\boldsymbol{X}_T|Y) - H(X_f|Y)$, where the last identity is due to the conditional independence assumption. Fix $f \in F$ and $\tilde{f} \notin F$. Since $H(\boldsymbol{X}_{T\cup\{f\}}) = H(\boldsymbol{X}_T) + H(X_f) - I(\boldsymbol{X}_T; X_f)$, we have $s(T \cup \{f\}) = I(\boldsymbol{X}_T; Y) + I(X_f; Y) - I(\boldsymbol{X}_T; X_f)$. Combine with a similar formulae for $s(T \cup \{\tilde{f}\})$, we obtain:

$$s(T \cup \{f\}) - s(T \cup \{\tilde{f}\}) = (I(X_f; Y) - I(\boldsymbol{X}_T; X_f)) - (I(X_{\tilde{f}}; Y) - I(\boldsymbol{X}_T; X_{\tilde{f}})) \geq 0,$$

which concludes the proof. □

A.3 On eliminating irrelevant features

The rank $\rho(f)$ of a feature is proportional to the average score of all tests that the feature f participates in. If f is "lucky" enough to participate in tests that contain relevant features, its rank might be inflated. This observation leads to our second idea: we need a way to quickly eliminate features that are likely to be irrelevant.

Theorem A.3. *Let F be the set of hidden relevant features. Let $d = |F|$. Let $\mathbf{A}$ be the random $t \times N$ test matrix obtained by setting each entry to be 1 with probability $p \in [0, 1]$ and 0 with probability $1 - p$. For an irrelevant feature $\tilde{f} \notin F$, let $U_{\tilde{f}}$ denote the total number of tests that $\tilde{f}$ belongs, and $V_{\tilde{f}}$ the total number of tests that $\tilde{f}$ belongs but none of the relevant features belong.*

For any $\delta \in (0, 1)$, and any β such that $0 < \beta < (1 - p)^d$, the following holds:

$$\mathbb{P}[V_{\tilde{f}} \geq \beta U_{\tilde{f}} \text{ for all } \tilde{f} \notin F] \geq 1 - \delta,$$

provided that the total number of tests is at least

$$t \geq \frac{1}{2} \cdot \frac{(1 + \beta)^2}{p^2((1 - p)^d - \beta)^2} \log((N - d)/\delta). \tag{2}$$

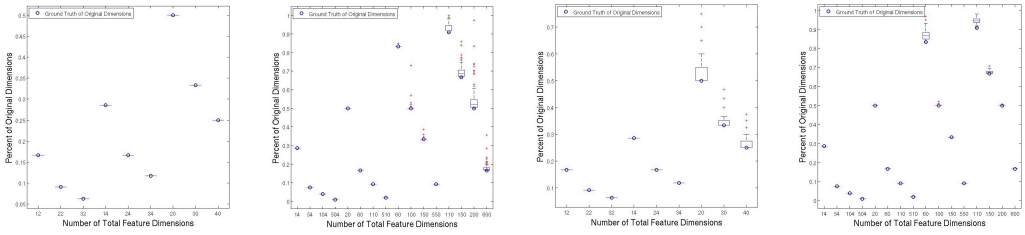
Proof. Let $\tilde{f}$ be an arbitrary irrelevant feature. For each $j \in [t]$, let X_j be the indicator variable for the event that $\tilde{f}$ is in test j, and Y_j be the indicator variable for the event that $\tilde{f}$ belongs to the jth test but none of the relevant features are in test j. Then, $U_{\tilde{f}} = \sum_{j\in[t]} X_j$ and $V_{\tilde{f}} = \sum_{j\in[t]} Y_j$. It follows that $\mathbb{E}[Y_j - \beta X_j] = p(1 - p)^d - \beta p$. Furthermore, we have $Y_j - \beta X_j \in [-\beta, 1]$, and for $j \in [t]$ the variables $Y_j - \beta X_j$ are independent. Hence, by Hoeffding bound we have

$$\begin{aligned}
\mathbb{P}[V_{\tilde{f}} < \beta U_{\tilde{f}}] &= \mathbb{P}[\sum_{j\in[t]} (Y_j - \beta X_j) < 0] \\
&\leq \exp\left\{\frac{-2t^2p^2((1 - p)^d - \beta)^2}{t(1 + \beta)^2}\right\} \\
&= \exp\left\{\frac{-2tp^2((1 - p)^d - \beta)^2}{(1 + \beta)^2}\right\}.
\end{aligned}$$

Hence, due to condition 2,

$$\mathbb{P}[V_{\tilde{f}} < \beta U_{\tilde{f}} \text{ for some } \tilde{f} \notin F] \leq (N - d) \exp\left\{\frac{-2tp^2((1 - p)^d - \beta)^2}{(1 + \beta)^2}\right\} \leq \delta.$$

□



(a) (b) (c) (d)

Figure 6: Box plot from synthetic data on a) the identifiability of original features with abundant data from multinomial distribution, b) the identifiability of original features with reasonable-sized data from multinomial distribution, c) the identifiability of original features with abundant data from discretized Gaussian distribution, d) the identifiability of original features with reasonable-sized data from discretized Gaussian distribution.

The above theorem is useful when we can find a score function such that the tests that contain **no** relevant have low scores, say less than some threshold θ. In that case, the natural algorithm is to first eliminate all features such that at least a β fraction of its tests score lower than θ.

To make use of the above algorithm, we need to set the parameters. For example, suppose we set $p = 1/d$. Then $(1-p)^d = (1-1/d)^d$ is an increasing function in d that tends to $1/e \approx 0.37$ fairly quickly. Hence, $d \geq 4$ we can pick $\beta = 0.25$ (or more). But there is a tradeoff between β and the number of tests t, hence we do not want to pick β to be too close to $(1-p)^d$.

As a second example, suppose we set $p = 1/(2d)$. Then, $(1-p)^d = (1-1/\sqrt{d})^d \to 1/\sqrt{e} \approx 0.61$. In this case we can even pick $\beta = 1/2$.

B Additional details on synthetic experiment results

We evaluate the entire PFS idea synthetically. We generate a simple categorical binary class dataset using two multinomial distributions. Let N_o be the number of original data dimensions (*i.e.* where the data is actually dependent on). The N_n noisy dimensions are generated with uniform probability for all N_n dimensions, so the synthetic data generated is of dimension $N = N_o + N_n$. The number of trials were restricted to five in our data generation. As theorem 2.2 suggests, by setting $p = 0.5$ we only need logarithmic number of tests with respect to the number of feature dimensions, but it will lead to inaccurate score estimation when N is large and the number of data samples are small. Therefore, we first simulate a case where we have abundant samples by sampling $10,000$ samples for each class with $N_o \in \{2, 4, 10\}$ and $N_n \in \{10, 20, 30\}$. We set $p = 0.5$ and $t = \lceil \frac{2}{p^2(1-p)^2} \log(N_o N_n/\delta) \rceil$ where $\delta = 0.01$. In addition, to attain a more realistic setting, we generate $1,000$ samples for each class, with $N_o \in \{4, 10, 50, 100\}$ and $N_n \in \{10, 50, 100, 500\}$. We set $p = \frac{3}{N}$ so that we can get reasonable score estimate and $t = 10N$. To account for the randomness of the test, we ran every experiment 100 times; the result is shown in Figure 6(a) and (b) respectively. It is clear that most of the time all the original dimensions are contained in the top D_o ranked features, in particular, when the score can be estimated reasonably well, the top D_o features contains exactly all the original features (see figure 6 (a) and (c)).

Since not all real world data are categorical valued, we simulated another real-valued binary dataset. The N_o original data dimensions are generated from two Gaussian with mean 3 and -3. They share the same variance, and it is uniformly sampled from the interval $(0, 1]$. The N_n noisy dimensions are generated from Gaussian with mean sample uniformly from interval $[-1, 1]$ and variance uniformly sampled from interval $(0, 1]$. We then quantize each dimension into five equal distanced bins. We use the exactly same settings as the previous experiments, and the result is illustrated in figure 6 (c) and (d), it can be observed that the performance are consistent with the last set of experiments.

C Additional results on the small and medium data sets

C.1 Accuracy and runtime results on Micro-array dataset

The Colon and Leukemia dataset are both binary class dataset that contains 62 samples with 2,000 dimensions and 72 data points with 7,070 dimensions respectively; the Lymph and NCI9 dataset both have 9 classes and respectively contain 96 samples with 4,026 dimensions and 60 samples with 9,712 dimensions; The Lung dataset contains 73 data samples of 325 dimensions and is a 7-class dataset.

We set the maximum number of selected features to be 50. d Models were trained for each dataset with the top d features where d varies from 1 to 50, and we report the best overall *leave-one-out* classification error among all 50 combinations of features. For the wrapper method we set $p = 10/N$ and for filter method we set $p = 4/N$, where N is the dimension of data.

Table 2: Leave one out error on micro-array datasets from various methods

Method/Dataset	Colon	Leukemia	Lung	Lymph	NCI9
MIM	10	2	14	13	25
MIM (Filtered)	10	2	13	13	25
MRMR	9	2	13	7	23
MRMR (Filtered)	9	2	13	7	23
CMIM	9	1	9	9	26
CMIM (Filtered)	9	1	9	9	26
JMI	9	1	11	9	24
JMI (Filtered)	9	1	11	9	24
DISR	8	1	13	11	24
DISR (Filtered)	8	1	13	11	24
CIFE	9	3	19	26	31
CIFE (Filtered)	9	3	9	10	35
ICAP	8	3	10	8	24
ICAP (Filtered)	7	2	9	9	24
FCBF	9	4	11	**6**	24
FCBF (Filtered)	**1**	2	9	14	25
LOGO	10	2	**8**	12	27
LOGO (Filtered)	7	1	11	11	28
FGM	9	2	**8**	7	**21**
FGM (Filtered)	10	1	9	15	25
ours (F_3)	6	1	11	13	31
ours (W_3)	8	**0**	9	11	28
ours (F_{10})	6	1	11	12	28
ours (W_{10})	9	1	15	13	29

C.2 Accuracy and runtime results on the NIPS Datasets

The results on NIPS dataset from different methods are shown in table 4 below.

Note that the numbers we reported are runtimes *without* running tests in parallel. Since our tests are totally independent, the parallel speed up factor will be essentially linear in the partition size.

The NIPS and micro-array datasets experiments were all completed on a machine with I7-3930K 3.20GHZ 6-core CPU and 32GB RAM with 12 threads. The running time of different methods are listed in the following table 5. and 3.

D Additional results on the large dataset

D.1 Top-features on all relations

As we shown inf Figure 2(b), the top-features extracted by our method makes intuitive sense for relations **Spouse**, **MemberOf**, and **TopMember**. Figure 8 shows the result for other relations using the same protocol we described in the body of this paper.

We see that for most relations, the top features selected by our method makes intuitive sense, which implies the effectiveness of our approach. For relations like **per:stateorprovinces_of_residence**, the top keywords are not direct indicator of the relation (although they strongly imply the relation), this is a known problem of how the training set is generated [27, 17], and is orthogonal to the feature selection process.

D.2 On varying the number selected features

Figure 2(a) shows the Precision/Recall on TAC-KBP data set with the number of selected features $K = 1000$. Figure 7 shows the result for $K = 10$ and $K = 100$. We can see that different approaches perform similarly as $K = 1000$ case.

Table 3: Micro-array dataset runtime performance (in seconds)

Methods/Dataset	Colon	Leukemia	Lung	Lymph	NCI9
MIM	1.77	7.27	0.36	6.17	9.08
MIM (Filtered)	0.23	0.78	0.09	0.70	0.96
MRMR	10.96	50.36	2.02	41.72	57.27
MRMR (Filtered)	1.26	4.99	0.39	4.41	5.78
JMI	17.46	76.29	4.45	90.43	127.78
JMI (Filtered)	1.99	7.75	0.83	9.15	12.93
ICAP	37.80	167.25	8.73	180.59	248.40
ICAP (Filtered)	4.28	17.24	1.65	18.38	25.01
DISR	28.13	118.22	7.20	144.71	206.25
DISR (Filtered)	3.17	12.07	1.36	15.22	20.75
CMIM	1.19	3.47	1.12	5.49	2.78
CMIM (Filtered)	0.77	1.42	0.75	2.64	1.16
CIFE	37.53	166.53	8.85	185.69	259.42
CIFE (Filtered)	4.25	16.76	1.64	18.25	25.73
FCBF	14.01	87.17	34.70	3991.4	838.07
FCBF (Filtered)	2.10	18.76	5.50	114.24	158.39
LOGO	32.51	180.58	66.99	156.66	86.84
LOGO (Filtered)	14.87	27.53	52.37	102.49	53.34
FGM	1.73	3.30	4.54	86.71	142.44
FGM (Filtered)	1.15	1.15	0.83	5.33	5.61
ours (F_3)	1.01	5.35	0.25	5.01	6.61
ours (W_3)	19.72	82.87	46.21	1112.22	1599.15
ours (F_{10})	2.80	14.79	0.74	14.50	19.68
ours (W_{10})	66.71	274.12	153.23	3699.2	5233.84

Table 4: Accuracy results from different methods on NIPS datasets

	Datasets			
	GISETTE		MADELON	
Methods	BER (%)	Features (%)	BER (%)	Features (%)
MIM	3.15	9.40	12.33	2.80
MIM (Filtered)	3.08	6.42	12.33	2.80
MRMR	3.69	8.04	47.83	9.40
MRMR (Filtered)	4.58	4.62	46.17	9.20
JMI	4.02	1.94	11.28	2.00
JMI (Filtered)	4.63	5.62	11.28	2.00
ICAP	4.58	6.24	12.33	2.80
ICAP (Filtered)	4.17	4.62	12.33	2.80
DISR	3.06	7.32	10.61	1.80
DISR (Filtered)	2.92	7.02	14.22	2.60
CMIM	4.46	3.16	12.33	2.80
CMIM (Filtered)	4.82	2.22	12.33	2.80
CIFE	7.82	9.74	39.83	10.20
CIFE (Filtered)	7.80	9.62	39.33	3.60
FCBF	16.86	0.02	45.50	0.20
FCBF (Filtered)	16.86	0.02	45.50	0.20
LOGO	3.09	4.00	43.94	17.80
LOGO (Filtered)	3.40	1.38	21.11	11.60
FGM	**2.15**	**0.70**	39.50	9.00
FGM (Filtered)	2.54	1.00	39.11	1.40
ours (F_3)	4.85	9.34	22.61	4.40
ours (W_3)	2.72	6.30	**10.17**	**2.40**
ours (F_{10})	4.69	9.94	18.39	1.40
ours (W_{10})	2.89	9.18	10.50	2.40

Table 5: NIPS dataset runtime performance (in seconds)

Methods/Dataset	GISETTE	MADELON
MIM	0.79	0.04
MIM (Filtered)	0.22	0.01
MRMR	23807.11	5.39
MRMR (Filtered)	3439.96	4.68
JMI	5303.93	7.86
JMI (Filtered)	963.97	2.07
ICAP	30901.43	59.68
ICAP (Filtered)	5866.00	20.08
DISR	5533.05	12.49
DISR (Filtered)	864.76	3.16
CMIM	3162.91	12.88
CMIM (Filtered)	3030.55	11.61
CIFE	30658.80	45.52
CIFE (Filtered)	5715.51	17.72
FCBF	107.57	0.94
FCBF (Filtered)	13.65	0.44
LOGO	19303.79	18.17
LOGO (Filtered)	2441.69	9.73
FGM	36.11	6.48
FGM (Filtered)	27.87	4.09
ours (F_3)	3.32	0.23
ours (W_3)	11.03	0.32
ours (F_{10})	10.48	0.72
ours (W_{10})	36.12	1.05

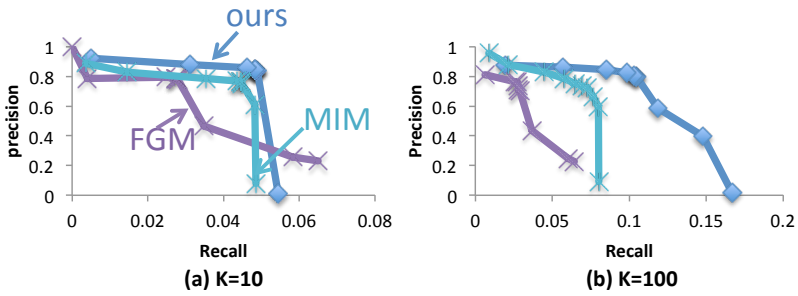


Figure 7: Precision/Recall on TAC-KBP with Number of Features $K = 10$ and $K = 100$.

Relation	Keywords				
org:city_of_headquarters	based	headquarters	COXnet directed seized control of		rulers
org:founded_by	founder	leader	chairman	co-founder	executive
org:parents	employees	owned	unit	divisions	subsidiary
org:subsidaries	employees articles	owned	unit	divisions	subsidiary
org:top_members_employees	head	executive	chairman	general	president
per:children	son	father	daughter	mother	said
per:cities_of_residence	executive	president	chairman	executive	born in
per:city_of_birth	born	ARodriguez	Peavy	told in	native
per:city_of_death	died	died home	died hospital	killed attack city	assassination
per:countries_of_residence	mayor	said in	born in	Democrat	told in
per:employee_of	executive	chairman	president	professor	director
per:member_of	leader	member	rebels	commander	iraq leader
per:parents	son	father	daughter	mother	sons
per:schools_attended	holds degree from	standout	graduate	student	attend
per:siblings	brother	sister	half-brother	found along-with	pregnancy give shops
per:spouse	wife	pictures	husband	married	widower
per:stateorprovinces_of_residence	governor of	senator from	Republican	Democrat	Republican of

Figure 8: Top Keywords for All 17 Relations We Considered in TAC-KBP

E Stability Experiments

Given different data samples from the same distribution, the feature selection algorithm should ideally identify the same set of features assuming there is a unique set of "true" features[10]. However, due to biases incurred during data sampling and the redundancy present in the data, the algorithm may end up identifying different sets of features leading to inconsistency. Kuncheva [13] presented a *consistency index* which measures the consistency between two sets, with a positive value indicating similar sets, negative value for anti-correlation and zero for random relations.

For measuring the consistency index, we take 50 bootstraps from a dataset and select feature on the bootstraps. The consistency index of the dataset from a particular method is taken as the median value from the 50 bootstraps. The box plot of consistency index from different methods on the 15 UCI datasets are shown in figure 9. In general, filter method has relatively higher stability as compared to wrapper method. The stability measure

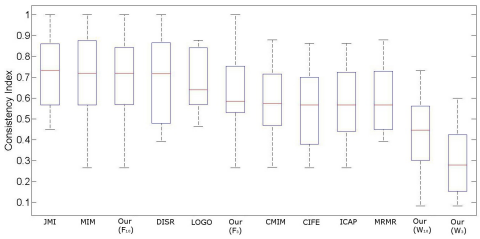


Figure 9: Consistency index across 15 UCI datasets.

of the filter method is very similar to JMI and MIM method, which is attributed to the similarity in obtaining the scores. From the stability measure of our method in figure 9, we can also observe that as we increase the number of tests, the algorithm gets more stable, which confirms the experiments we did in previous section.

[10]This will not hold in case there are multiple subsets of features that are equally good.