GENVISAGE: Rapid Identification of Discriminative and Explainable Feature Pairs for Genomic Analysis

Silu Huang[1‡], Charles Blatti[2‡], Saurabh Sinha[1,2†], Aditya Parameswaran[1,2,3†],

1 Department of Computer Science, University of Illinois Urbana-Champaign, Urbana, IL, 61801, USA
2 Institute of Genomic Biology, University of Illinois Urbana-Champaign, Urbana, IL, 61801, USA
3 School of Information and Department of Electrical Engineering and Computer Sciences, UC Berkeley, Berkeley CA, 94704, USA

‡These authors contributed equally to this work.
†To whom correspondence should be addressed.
* sinhas@illinois.edu, adityagp@berkeley.edu

Abstract

Motivation. A common but critical task in genomic data analysis is finding features that *separate* and thereby help explain differences between two classes of biological objects, e.g., genes that explain the differences between healthy and diseased patients. As lower-cost, high-throughput experimental methods greatly increase the number of samples that are assayed as objects for analysis, computational methods are needed to quickly provide insights into high-dimensional datasets with tens of thousands of objects and features.
Results. We develop an interactive exploration tool called GENVISAGE that rapidly discovers the most discriminative feature pairs that best separate two classes in a dataset, and displays the corresponding visualizations. Since quickly finding top feature pairs is computationally challenging, especially when the numbers of objects and features are large, we propose a suite of optimizations to make GENVISAGE more responsive and demonstrate that our optimizations lead to a *400X* speedup over competitive baselines for multiple biological data sets. With this speedup, GENVISAGE enables the exploration of more large-scale datasets and alternate hypotheses in an interactive and interpretable fashion. We apply GENVISAGE to uncover pairs of genes whose transcriptomic responses significantly discriminate treatments of several chemotherapy drugs.
Availability. Free webserver at `http://genvisage.knoweng.org:443/` with source code at `https://github.com/KnowEnG/Genvisage`

1 Introduction

A common approach to discovery in biology is to construct experiments or analyses that directly contrast two specific classes of biological objects. Examples of this include examining patient samples contrasting tumor versus normal tissue [1], studying the differences in molecular effects of two competing drug treatments [2], or characterizing differentially expressed genes versus genes with unaltered gene expression in a carefully designed experiment [3]. To understand the mechanisms that determine these object classes, researchers often employ statistical and machine learning tools to identify a manageable subset of features, e.g. genes, that accentuate, discriminate, or help explain

the differences between classes, i.e., *separate* the two classes. We therefore refer to this problem as the *separability problem*.

Challenge 1: Explainable Separation with Guarantees. Many tools have been developed in several different biological settings [4–9] that attempt to solve the separability problem by focusing on discovering pairs of features that taken together strongly discriminate the classes. Feature pair methods can provide a better characterization of what distinguishes two object classes by offering insights into the interplay of important features that would not be found using single feature statistical tests [10] or univariate classifiers [11]. Specifically, predictors built with gene feature pairs are more robust to normalization and can achieve better model performance than predictors using single genes as features [5, 6]. On the other hand, methods focused on feature pairs offer the advantage of providing more interpretable or explainable results over more complicated machine learning approaches that return a complex combination of several features to discriminate the classes, such as multivariate regression with LASSO regularization [12] or pattern mining from random forest models [13]. Some existing papers [7, 14–16] employ these more complex machine learning approaches to heuristically return more interpretable feature pairs. However, these heuristic methods do not fully explore the search space nor do they offer a guarantee on the quality of the returned feature pairs.

Challenge 2: Scalability in Data Size. A major problem with current methods that address the separability problem with either feature pairs or more complex machine learning models is that they do not scale to the growing size of genomic data sets. As is often the case with genomics, the biological objects being analyzed (e.g., tissue samples or drug experiments) are frequently represented by high dimensional numeric feature vectors (e.g., transcript abundance measurements). Additionally, with the rise of low-cost sequencing, the possible number of biological objects in a dataset is also increasing and likely to grow in orders of magnitude over the next decade [17]. Applying the standard methods to datasets with tens of thousands of objects and features results in massive running times that preclude interactive exploration of the data. For example, exhaustively searching for the optimal feature pairs from the full space of possibilities in a typical genomic analysis resulted in running times over an hour on a 200 node compute cluster in Watkinson et al. [9].

One reason that more complex machine learning and feature pair based methods do not scale well with the number of features and objects is the the selection of the metric for scoring separability. In Watkinson et al. [9], a metric called synergy is proposed for evaluating the utility of feature pairs, aiming to capture both linear and non-linear aspects of the separability of the two class. Consequently, the intrinsic complexity of these metrics makes them difficult to benefit from optimization techniques. Metrics based only on quantifying *linear* separability, on the other hand, may return a more limited subset of interesting features, but they also may be more intuitive for users to understand and simultaneously enable more performance optimizations and speedups. The linear separability metric has been used in previous studies to identify pairs of genes with expression differences between two cancer types [18] or pairs of motifs that discriminate between different types of genomic sequences [19].

Our Proposal GENVISAGE**: A Scalable and Explainable Tool for Addressing Separability.** Motivated by these observations, we present GENVISAGE, an interactive data exploration tool designed to address the separability problem and scale to the size of large genomic analysis datasets. With GENVISAGE, we not only achieve high separability quality with our carefully formulated problem; but also enable explanations on separation via intuitive visualization; meanwhile, we are capable to handle large scale datasets efficiently — the best of all three world. Specifically, to

enable this scalability, GENVISAGE focuses on returning the top ranking feature pairs that discriminate the objects of separate classes, rather than returning larger subsets of features using more complex and longer to train machine learning approaches. GENVISAGE is also based around a linear separability metric that provides an intuitive interpretation to feature pairs while enabling and simplifying the design of several important performance optimizations. These optimizations include *(a)* elimination of repeated computation for different features pairs; *(b)* pruning poor ranking pairs during early execution; *(c)* sampling with a quality guarantee to further reduce running time; and *(d)* cleverly traversing the search space of feature pairs for improved efficiency.

We applied GENVISAGE to two large genomic datasets with tens of thousands of objects and high-dimensional feature vectors where it is computationally expensive to score the separability for all possible feature pairs. In one, called LINCS, we find pairs of genes whose expression discriminates between perturbagen experiments involving different drug treatments, and in the other, called MSIGDB, we find pairs of annotations (such as pathway membership) that separate differentially expressed cancer genes from other genes. With the carefully designed separability metric of GENVISAGE and its suite of sophisticated optimizations that accelerates evaluation, we are able to *accurately return the highest ranking separating feature pairs for both datasets within two minutes on a single machine*. This reflects a *180X* and *400X* speedup over a competitive baseline for the MSIGDB and LINCS data sets (respectively). We also show that the feature pairs identified by GENVISAGE often more significantly discriminate between the object classes than the corresponding best ranking individual features, even after accounting for the larger search space. Finally, we performed an in-depth analysis for nine distinct drug treatments in the LINCS dataset and found 1070 feature (gene) pairs that had significant separability scores. These gene pairs were enriched in literature support for known relationships between the genes and the drug, as well as known interactions between the genes themselves.

Summarized Benefits of Using GENVISAGE. By focusing on separating feature pairs, GENVISAGE offers researchers the ability to gain additional insight into their object classes beyond singular features, without the prolonged duration needed to train a complex machine learning model. By implementing optimizations that take advantage of a linear separability metric, GENVISAGE enables researchers to quickly explore their data, identify the strongest, most compelling features, and from simple visualizations form hypotheses about the interplay between features and with the object classes. The performance of our tool also allows researchers to investigate multiple definitions of the object classes and investigate alternative hypotheses interactively on the fly, as well as build a feature set to later pass to more in-depth, longer running machine learning-based analysis.

2 Methods

We begin by formally defining the *separability* problem, introducing our separability metric, and finally detailing optimizations that enable the rapid identification of the best separating feature pairs.

2.1 Problem Definition

Let $\mathcal{M}$ be a feature-object matrix of size $m \times N$, where each row is a feature and each column is an object as shown in Figure 1. One example feature-object matrix is one where each object corresponds to a tissue sample from a cancer patient and each feature corresponds to a gene, where the $(i,j)^{th}$ entry represents the expression level of the i^{th} gene in the j^{th} tissue sample. We denote the m features as $\mathcal{F} = \{f_1, f_2, \cdots, f_m\}$ and N objects as $\mathcal{O} = \{o_1, o_2, \cdots, o_N\}$. Each entry $\mathcal{M}_{i,j}$ in $\mathcal{M}$ corresponds to the value of feature f_i for object o_j as illustrated in Figure 1.

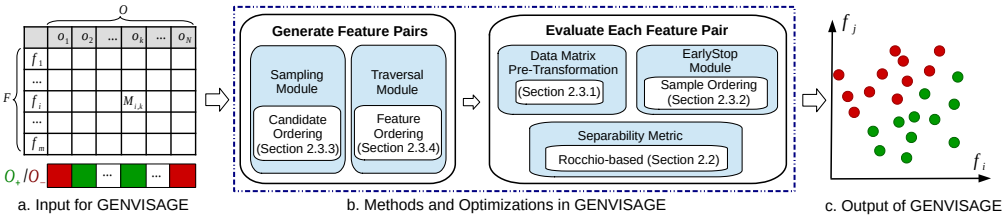


Fig 1. GENVISAGE Workflow. Given (left) a feature-object matrix and green positive and red negative class labels on the objects, GENVISAGE (center) evaluates all pairs of features using several optimizations to identify (right) the top feature pair and its corresponding visualization that best separates the object classes.

We are also given two non-overlapping sets of objects, one with a positive label, $\mathcal{O}_+$ and the other with a negative label, $\mathcal{O}_-$. In our example, tumor samples, $\mathcal{O}_+$, may be assigned the positive label, and the healthy tissue samples, $\mathcal{O}_-$, the negative label. The number of labeled objects, n, is equal to $|\widehat{\mathcal{O}}|$ where $\widehat{\mathcal{O}} = \mathcal{O}_+ \cup \mathcal{O}_-$. Also, let l_k be the label of object $o_k \in \widehat{\mathcal{O}}$, i.e., $l_k = 1$ if o_k is positive and $l_k = -1$ if o_k is negative.

GENVISAGE aims to find feature pairs that best separate the objects in $\mathcal{O}_+$ from those in $\mathcal{O}_-$ using only those features, and then output a visualization that demonstrates the separability (see Figure 1). (We will define the metric for separability subsequently.) A feature pair that leads to a good "visual" separation between the positive and the negative sets may be able to explain or characterize their differences via a interesting, non-trivial relationship among the features. The overall workflow is depicted in Figure 1. We now formally define the separability problem.

Problem 1 (Separability)**.** *Given a feature-object matrix $\mathcal{M}$ and two labeled object sets $(\mathcal{O}_+, \mathcal{O}_-)$, identify the top-k feature pairs (f_i, f_j) that separate $\mathcal{O}_+$ from $\mathcal{O}_-$ based on a given separability metric.*

We will describe our separability metric in Section 2.2, and then discuss optimization techniques in Section 2.3. The notation used in the description of the method is summarized in Supplementary Table B.1.

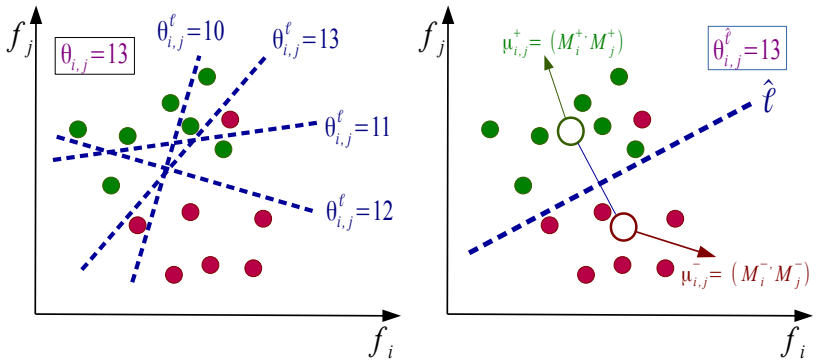
2.2 Separability Metric

Given a feature pair (f_i, f_j) as axes, we can visualize the object sets $\mathcal{O}_+$ and $\mathcal{O}_-$ in a 2-D space, where each object corresponds to a point with x-value and y-value as the object's value on feature f_i and f_j respectively. A desirable (i.e., both interesting and interpretable) visualization would be one in which the objects are *linearly separated*, defined as follows. Two sets of objects, i.e., $\mathcal{O}_+$ and $\mathcal{O}_-$, are said to be *linearly separable* [20] if there exists at least one straight line such that $\mathcal{O}_+$ and $\mathcal{O}_-$ are on opposite side of it. We focus on metrics that capture this linear separation, since it corresponds to an intuitive 2-D visualization. Given a feature pair (f_i, f_j) and a line ℓ, we can predict the label of an object o_k, denoted as $\eta_{i,j}^{\ell,k}$, using Equation 1 below, where w_0, w_i and w_j are coefficients of ℓ and $w_j > 0$:

$$\text{Predicted Label}: \quad \eta_{i,j}^{\ell,k} = sign(w_i \cdot \mathcal{M}_{i,k} + w_j \cdot \mathcal{M}_{j,k} + w_0) \tag{1}$$

If o_k lies above the line ℓ, i.e., o_k has higher value on y-axis than the point on line ℓ with the same value on x-axis as o_k, then $\eta_{i,j}^{\ell,k} = 1$; otherwise, $\eta_{i,j}^{\ell,k} = -1$. Let $\theta_{i,j}^{\ell,k}$ be the indicator variable denoting whether the sign of the predicted label matches the real label l_k: if $\eta_{i,j}^{\ell,k} \cdot l_k = 1$, then $\theta_{i,j}^{\ell,k} = 1$; otherwise, $\theta_{i,j}^{\ell,k} = 0$.

GENVISAGE's separability metric captures *how well the objects in the feature pair's 2-D visualization can be linearly separated*, formally defined next. Given a feature pair



(a) Brute Force (b) Rocchio-Based

Fig 2. Calculating Separability Score $\theta_{i,j}$. The scored separating line can be defined using (a) brute force (few sample lines are shown) or (b) the representative line from a Rocchio-based measure based on the object class centroids (white circles).

(f_i, f_j) and a line ℓ, the separability score of the line (denoted $\theta^{\ell}_{i,j}$) is defined as the sum of the indicators ($\theta^{\ell,k}_{i,j}$) for all objects: $\theta^{\ell}_{i,j} = \sum_k \theta^{\ell,k}_{i,j}$. Figure 2(a) shows separability scores $\theta^{\ell}_{i,j}$ for different separating lines. For example, the separating line with $\theta^{\ell}_{i,j} = 12$ correctly separates six green points and six red points. The final separability score for a feature pair (f_i, f_j) (denoted as $\theta_{i,j}$) is defined as the best separability score $\theta^{\ell}_{i,j}$ among all possible lines ℓ. Accordingly, we define the overall separability error of the feature pair as $err_{i,j} = n - \theta_{i,j}$.

Brute Force Calculation of $\theta_{i,j}$. As suggested in Figure 2(a), the simplest way to calculate $\theta_{i,j}$ is to first enumerate all possible separating lines ℓ and calculate $\theta^{\ell}_{i,j}$ for each of them. We can easily trim down the search space to $O(n^2)$ lines by linking the points corresponding to every two objects in the 2-D plane. This is because the results of all other possible lines can be covered by these $O(n^2)$ lines [21]. Nevertheless, it is still very time-consuming to consider $O(n^2)$ lines for each feature pair (f_i, f_j).

Rocchio-based Measure. We can speed up the process by selecting a single *representative line* L providing us with an estimate of the true separability score $\theta_{i,j}$. In order to achieve a fast and reliable estimate, we select our representative line based on Rocchio's algorithm [22]. Let us denote the centroids of positive objects $\mathcal{O}_+$ and negative objects $\mathcal{O}_-$ for a given (f_i, f_j) as $\mu^{+}_{i,j} = (\mathcal{M}^{+}_{i}, \mathcal{M}^{+}_{j})$ and $\mu^{-}_{i,j} = (\mathcal{M}^{-}_{i}, \mathcal{M}^{-}_{j})$ respectively, where $\mathcal{M}^{+}_{i}$ and $\mathcal{M}^{+}_{j}$ are the values of the centroids of the positive objects on feature f_i and f_j, and $\mathcal{M}^{-}_{i}$ and $\mathcal{M}^{-}_{j}$ are the values of the centroids of the negative objects on feature f_i and f_j. The perpendicular bisector of the line joining the two centroids is selected as the representative separating line L (see Figure 2(b)), with its coefficients corresponding to Equation 1 defined as $w_i = \mathcal{M}^{+}_{i} - \mathcal{M}^{-}_{i}$, $w_j = \mathcal{M}^{+}_{j} - \mathcal{M}^{-}_{j}$, and $w_0 = -(\frac{(\mathcal{M}^{+}_{i})^2 - (\mathcal{M}^{-}_{i})^2}{2} + \frac{(\mathcal{M}^{+}_{j})^2 - (\mathcal{M}^{-}_{j})^2}{2})$.

Brute-force vs. Rocchio-based. Compared to the brute force calculation, the Rocchio-based measure is much more light-weight, but at the cost of accuracy in calculating $\theta_{i,j}$. Intuitively, the representative line is a reasonable proxy to the best separating line since the Rocchio-based measure assigns each object to its nearest centroid. We further empirically demonstrate that $\theta^{L}_{i,j}$ is a good proxy for $\theta_{i,j}$ in Section 3.2. Thus, we will focus on the Rocchio-based measure subsequently, removing L (or ℓ) from the superscripts where it appears, and using $\theta_{i,j}$ and $\theta^{L}_{i,j}$ interchangeably.

2.3 Proposed Suite of Optimizations

In this section, we first analyze the time complexity of identifying the top-k feature pairs using the Rocchio-based measure, and then propose several optimization techniques to reduce the complexity.

Time Complexity Analysis. For a given feature pair (f_i, f_j), if we have already calculated the class centroids for each feature, the separating line L can be calculated in $O(1)$. We can then calculate the number of correctly separated objects $\theta_{i,j}$ via $O(n)$ evaluations. Since there are $O(m^2)$ feature pair candidates, the total time complexity is $O(m^2 n)$, which can be very large, since m and n are typically large.

Optimizations: Overview. To reduce the time complexity, we introduce two categories of optimizations: those that reduce the amount of time for fully evaluating a given feature pair (Section 2.3.1, 2.3.2) and those that reduce the number of feature pairs that require full evaluation (Section 2.3.3, 2.3.4). In the following, we refer to these optimizations as *modules* to indicate that they can be used in any combination—however, in reality, careful engineering is necessary to "stitch" these modules together to multiply the effects of the optimizations.

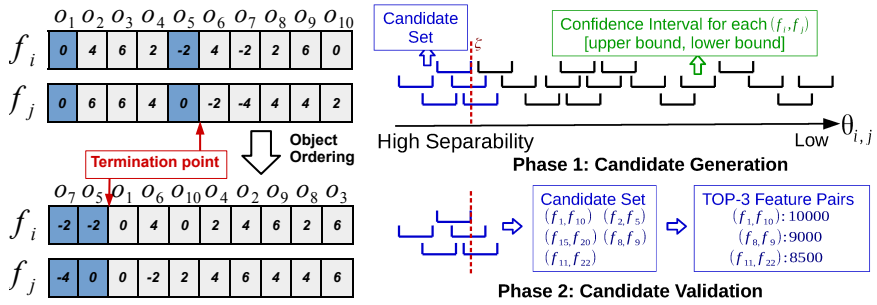
TRANSFORMATION module (Section 2.3.1) reduces redundant calculations across feature pairs by mapping the feature-object matrix $\mathcal{M}$ into a new space that enables faster evaluation of object labeling. EARLYSTOP module (Section 2.3.2) takes advantage of the fact that evaluation of a poorly separating feature pair can be terminated early without having to evaluate the separability of all n objects.

SAMPLING module (Section 2.3.3) first identifies likely top-k feature pair candidates by evaluating their separability on a sampled subset of all objects, and then conducts full evaluations only on these feature pair candidates. Finally, TRAVERSAL module (Section 2.3.4) reduces the number of feature pairs checked by greedily choosing feature pairs based on the separability of the corresponding single features. These optimization modules can be used on their own or combined with each other. In Section 3, we will show how these optimizations modules greatly reduce the running time of finding the top-k separating feature pairs without significantly affecting the accuracy.

2.3.1 Pre-Transformation for Faster Feature Pair Evaluation We observe that there is massive redundancy across $\theta_{i,j}$'s computation of different feature pairs. Motivated by this, we propose the TRANSFORMATION optimization module which will pre-calculate some common computational components once across different features and reuse these components in evaluating the separability for each different feature pair. This TRANSFORMATION module transforms the original $\mathcal{M}_{i,k}$ matrix into another space $\widehat{\mathcal{M}}_{i,k}$ using the identified common feature pair components and updates the separability score equation accordingly. Specifically, with this transformation of the feature-object matrix $\widehat{\mathcal{M}}_{i,k}$, evaluating whether an object was correctly separated is simplified as: if $sign(\widehat{\mathcal{M}}_{i,k} + \widehat{\mathcal{M}}_{j,k}) = 1$, then $\theta^k_{i,j} = 1$; otherwise, $\theta^k_{i,j} = 0$. Details and an example can be found in Supplementary Note A.1 and Supplementary Figure C.1.

2.3.2 Early Termination Given a feature pair (f_i, f_j), we need to scan all the objects to compute the separability score $\theta_{i,j}$. However, since we only need to identify feature pairs in the top-k, we can stop for each feature pair as soon as we can make that determination, without scanning all objects; we call this the EARLYSTOP module.

High Level Idea. We maintain a upper bound τ for the separability error $err_{i,j}$ of the top-k feature pairs. Then, the lower bound of the separability score can be denoted as $(n - \tau)$. Given a feature pair (f_i, f_j), we start to scan the object list until the number of incorrectly classified objects exceeds τ. If so, we can terminate early and prune this feature pair since it cannot be among the top-k. Otherwise, (f_i, f_j) is added to the top-k feature pair set and we update τ accordingly.



(a) EARLYSTOP Example (b) SAMPLING Example

Fig 3. Optimization Module Examples. (a) When evaluating a feature pair with EARLYSTOP module, the transformed $\widehat{\mathcal{M}}$ scores are scanned left to right and each incorrectly classified object is marked in blue. Without object ordering (above), evaluation terminates after five checked objects. When objects are reordered by the most "problematic" (below), the feature pair is rejected after checking only the first two objects. (b) To calculate the top-3 feature pairs with SAMPLING, the confidence interval of $\theta_{i,j}$ is calculated for every feature pair evaluated on the sample set $\mathcal{S}$ (above). The 3^{rd} interval lower bound ζ is obtained (red dotted line), and all feature pairs with a larger upper bound are designated as candidates for validation (blue intervals). The selected candidates (center box) are evaluated on the whole object set $\hat{\mathcal{O}}$ to compute the exact $\theta_{i,j}$ and pick the top-3 (right box).

Enhancement by Object Ordering. Although EARLYSTOP has the potential to always reduce the running time, its benefits are sensitive to the ordering of the objects for evaluation. Since we terminate as soon as we find τ incorrectly classified objects, we can improve our running time if we examine "problematic" objects that are unlikely to be correctly classified relatively early. For this, we order the objects in descending order of the number of single features f_i that incorrectly classify the object o_k, i.e., $\widehat{\mathcal{M}}_{i,k} \leq 0$. Thus, the first object evaluated is the one that is incorrectly classified by the most single features. The benefit of this strategy is illustrated with an example in Figure 3(a).

2.3.3 Sampling-based Estimation

One downside of the EARLYSTOP module is that the improvement in the running time is highly data-dependent. Here, we propose a stochastic method, called SAMPLING, that reduces the number of examined objects. Instead of calculating $\theta_{i,j}$ over the whole object set $\hat{\mathcal{O}}$, SAMPLING works on a sample set drawn from $\hat{\mathcal{O}}$.

High Level Idea. SAMPLING primarily consists of two phases: *candidate generation* and *validation* (Figure 3(b)). In phase one, we estimate the confidence interval of $\theta_{i,j}$ for each feature pair using a sampled set of objects and generate the candidate feature pairs for full evaluation based on where their confidence intervals lie. If the confidence interval overlaps with the score range of the current top-k, then it is selected for evaluation. In phase two (lower half of Figure 3(b)), we evaluate only the feature pairs in the candidate set, calculating $\theta_{i,j}$ over the whole object set, $\hat{\mathcal{O}}$, to obtain the final top-k feature pairs. Unlike our previous optimizations, SAMPLING returns an approximation of the top-k ranking feature pairs.

Candidate Generation. Let $\mathcal{S}$ be a sample set drawn uniformly from $\hat{\mathcal{O}}$. Given a feature pair (f_i, f_j), let $\theta_{i,j}(\mathcal{S})$ be the number of correctly separated objects in $\mathcal{S}$. We can estimate $\tilde{\theta}_{i,j}$ from $\theta_{i,j}(\mathcal{S})$ using $\tilde{\theta}_{i,j} = \frac{\theta_{i,j}(\mathcal{S})}{|\mathcal{S}|} \cdot n$ by assuming the ratio of correctly separated samples in $\mathcal{S}$ is the same as that in $\hat{\mathcal{O}}$. Using Hoeffding's inequality [23], we have that by selecting $\Omega(\frac{1}{\epsilon^2})$ samples, that $\theta_{i,j}$ is in the confidence interval $[\tilde{\theta}_{i,j} - \epsilon n, \tilde{\theta}_{i,j} + \epsilon n]$ with high probability (details in Supplementary Note A.2). Since

the sample size $|\mathcal{S}|$ is independent of the number of objects, this module helps GENVISAGE scale to datasets with large n.

Following the top half of Figure 3(b), we can first calculate the confidence interval of $\theta_{i,j}$ for each feature pair (f_i, f_j). Next, we compute the lower bound of $\theta_{i,j}$ for the top-k feature pairs, denoted as ζ as shown by the red dotted line. Finally, we can prune feature pairs away whose upper bound is smaller than ζ, keeping the candidate set $\mathcal{C}$ of feature pairs depicted by blue intervals. These feature pairs $\mathcal{C}$ will be further validated in phase two, i.e., candidate validation. Typically, $|\mathcal{C}|$ will be orders of magnitude smaller than m^2, the original search space for all feature pairs.

Candidate Validation. We re-evaluate all of the candidates generated from phase one to produce our final feature pair ranking. This evaluation is performed using the whole object set $\hat{\mathcal{O}}$ and the top-k feature pairs are reported (lower half of Figure 3(b)).

Enhancement by Candidate Ordering. In Section 2.3.2 we proposed an enhancement that allows us to terminate computation early by manipulating the order of the objects; here we similarly found a way to reduce the running time by changing the order in which feature pair candidates are validated in phase two. Instead of directly validating each feature pair candidate, we first order the candidates in descending order according to the upper bound of each candidate's confidence interval. Then, we sequentially calculate the full separability score $\theta_{i,j}$ for each feature pair, and update ζ correspondingly. Recall that ζ is the current estimate of the lower bound of $\theta_{i,j}$ for the top-k feature pairs. Finally, we terminate our feature pair validation when the next feature pair's upper bound smaller than the current value of ζ (Figure 4).

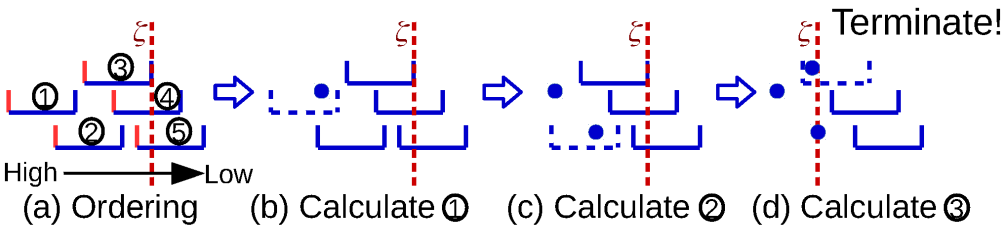


Fig 4. Candidate Ordering Enhancement. (a) Feature pair candidates are sorted by the upper bounds of their confidence intervals (solid red boundary), and the lower bound of the top-3 feature pairs, i.e., ζ, is set (red dotted line). (b,c,d) For each feature pair, we calculate $\theta_{i,j}$ (filled blue circle) using all objects and update ζ if necessary. Note that ζ is increased in (d) after evaluating the third feature pair and since ζ is larger than the upper bound of the fourth feature pair, candidate validation can terminate and return the top ranking pairs.

2.3.4 Search Space Traversal The optimizations discussed so far check fewer than n objects for each feature pair and reduce the number of feature pairs for full evaluation. Our TRAVERSAL module aims to reduce the number of feature pairs considered from m^2 to a smaller number. Instead of examining each feature pair, we only examine a limited number of feature pairs, but in an optimized traversal order. The number of examined feature pairs, χ, determines a trade-off between efficiency and accuracy. Fewer feature pairs checked will result in faster running times, though at the cost of accuracy to the top-k. The order of the feature pairs must be determined carefully and we propose two alternative orderings based on the ranking of single features by their separability scores $\theta_{i,i}$. The first traversal order, called *horizontal traversal*, prioritizes feature pairs that have at least one high ranking single feature in the considered feature pair. The second order, called *vertical traversal*, prioritizes feature pairs where both features have high single feature scores rankings. See Supplementary Figure C.2 for more details and an example.

3 Results

In this section, we illustrate that GENVISAGE rapidly identifies meaningful, significant, and interesting separating feature pairs in real biological datasets. First, we describe the datasets and the algorithms used in our evaluation. Each algorithm that we evaluate represents a combination of optimization modules for ranking top-k feature pairs using our Rocchio-based measure—we report the running time and accuracy of the algorithms. Second, we compare the top-k feature pairs returned by GENVISAGE with the corresponding top-k single features, and examine their significance and support in existing publications. Last, we present some sample visualizations to illustrate the separability of the object classes.

3.1 Evaluation Setup

	$\|\mathcal{F}\|=m$	$\|\mathcal{O}\|=N$	$\|\mathcal{S}\|$	χ	# of $\widehat{\mathcal{O}}$	avg($\|\mathcal{O}_+\|$)	avg($\|\mathcal{O}_-\|$)
MSIGDB	19,912	22,209	400	10^7	10	295	21,914
LINCS	22,268	98,061	400	10^7	40	165	97,897

Table 1. Dataset Statistics. For each dataset, the number of features m, objects N, sample size $|\mathcal{S}|$ used by SAMPLING module, feature pairs χ examined by TRAVERSAL module, number of object sets: # of $\widehat{\mathcal{O}}$, average positive set size: avg($|\mathcal{O}_+|$), and average negative set size: avg($|\mathcal{O}_-|$).

Datasets. We consider datasets from two biological applications (see Table 1): (a) in MSIGDB, we find gene annotations such as pathways and biological processes that separate the differentially expressed genes from the undisturbed genes in specific cancer studies; (b) in LINCS, we find genes whose expression levels can distinguish experiments in which specific drug treatments were administered from others.

In MSIGDB, we are given a feature-object matrix with genes as the objects and gene properties as the features. Rather than being a 0/1 membership indicator matrix, the values of this feature-object matrix indicate the strength of the relationship between the gene and the set of genes that have been annotated with the gene property. Matrix values are calculated using random walks [24] on a heterogeneous network built from prior knowledge found in gene annotation and protein homology databases (see Supplementary Note A.3 for more details). The positive genes for each dataset in MSIGDB are the set of differentially expressed genes (DEGs) in a specific cancer study downloaded from the Molecular Signatures Database (MSigDB) [25]. Each of our tests is an application of GENVISAGE to such a dataset, reporting pairs of properties that separate DEGs of that cancer study (the "positive" set) from all other genes (the "negative" set).

In LINCS, the feature-object matrix contains expression values for different genes (features) across many drug treatment experiments (objects) conducted on the MCF7 cell line by the LINCS L1000 project [26]. The values of the matrix are gene expression values as reported by the "level-4' imputed z-scores measured in the L1000 project. In each dataset, the positive object set includes multiple experiments that used the same drug, at varying dosages and for varying durations. We applied GENVISAGE on each dataset so as to find the top pairs of genes (feature pairs) whose expression values separate the LINCS experiments relating to a single drug from all other LINCS experiments.

Note that the average number of positive objects in any dataset is far fewer than the average number of negative objects. To address this imbalance, we adjust $\theta^{\ell}_{i,j}$ to a weighted sum form: $\theta^{\ell}_{i,j} = \sum_{o_k \in \mathcal{O}_-} \theta^{\ell,k}_{i,j} + \frac{|\mathcal{O}_-|}{|\mathcal{O}_+|} \cdot \sum_{o_k \in \mathcal{O}_+} \theta^{\ell,k}_{i,j}$.

Algorithms. We evaluated six combinations of our optimization modules from Section 2.3, listed in Table 2. For our baseline, we use the algorithm with only the matrix pre-transformation optimization module (TRANSFORMATION). The rightmost

	EARLY-STOP	SAMP-LING	Candidate Ordering	TRAVER-SAL	Approx-mation	Complex-tity
BASELINE	no	no	no	Any	no	$O(m^2n)$
EARLYORDERING	yes	no	no	Any	no	$O(m^2n)$
SAMPONLY	no	yes	no	Any	yes (guaran-tee)	$O(mn + m^2\|\mathcal{S}\| + \|\mathcal{C}\|n)$
SAMPOPT	no	yes	yes	Any	yes (guaran-tee)	$O(mn + m^2\|\mathcal{S}\| + \|\mathcal{C}\|n)$
HORIZSAMPOPT	no	yes	yes	Horizontal	yes (heuris-tic)	$O(mn + \chi\|\mathcal{S}\| + \|\mathcal{C}\|n)$
VERTSAMPOPT	no	yes	yes	Vertical	yes (heuris-tic)	$O(mn + \chi\|\mathcal{S}\| + \|\mathcal{C}\|n)$

Table 2. Selected Algorithms Using Different Optimization Modules. All algorithms, including the BASELINE, are using TRANSFORMATION. In addition, EARLYSTOP and TRAVERSAL are coupled with object ordering and feature ordering by default, respectively. For each algorithm (row), shows which optimization modules are employed, whether the algorithm is returning the exact answer or an approximation answer, and the running time complexity for that combination. The term "guarantee" ("heuristic", resp.) indicates that the returned answer is with (without, resp.) stochastic guarantee. In addition, m and n are the number of features and objects, $\mathcal{S}$ is the sampled set size, χ is the limit on the number of feature pairs considered, and $\mathcal{C}$ is the number of generated feature pair candidates.

column of Table 2 shows the varying time complexity of the algorithms. Consider the HORIZSAMPOPT as an example. First, TRANSFORMATION takes $O(mn)$ time. Then, TRAVERSAL requires a sorting over the feature set, taking $O(m \log m)$ time. Finally, with SAMPLING over χ feature pairs, the running time is reduced from $O(m^2n)$ time to $O(\chi|\mathcal{S}| + |\mathcal{C}|n)$ time, where the first and second term represent the time for candidate generation and candidate validation respectively. Note that $|\mathcal{C}|$ is typically orders of magnitude smaller than χ in HORIZSAMPOPT, as discussed in Section 2.3.3. Combinations of modules beyond the six reported were always inferior to one of the ones shown in the sense that they returned the same top-k feature pairs and had a longer running time. We implemented the algorithms in C++, and conducted the evaluations on a machine with 16 CPUs and 61.9 GB of RAM.

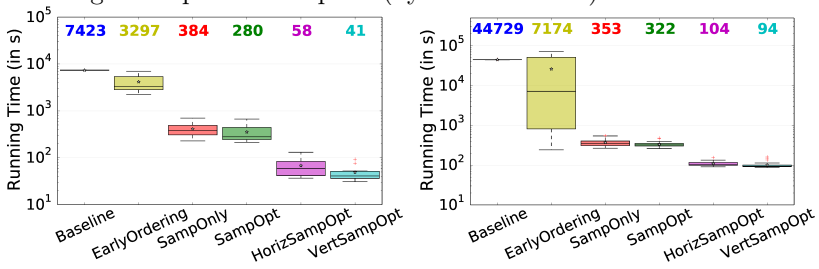
3.2 Comparison of Different Algorithms

In this section, we first justify that Rocchio-based measure is a good proxy for the best possible separating score computed by a brute force method. Then we compare the performance of the algorithms in terms of the running time and the separability of their top-1000 feature pairs.

Accuracy of Rocchio-based approximation. As discussed in Section 2.2, when using brute force, we need to consider $O(n^2)$ lines in order to find the best separating line $\ell^* \leftarrow \arg_\ell \max\{\theta^\ell_{i,j}\}$, with a time complexity of $O(n^2m^2)$ when considering all feature pairs. An alternative is to use Rocchio-based representative separating line L, dramatically reducing $O(n^2)$ lines considered to $O(1)$. Since the brute force method becomes computationally infeasible for datasets with large n, we compared the Rocchio-based measure to the brute force-based measure using specially defined small object sets, $\widehat{\mathcal{O}}$, for the 10 datasets in MSIGDB. For this comparison, the up-regulated genes in each MSIGDB test was defined as the set of positive objects and the down-regulated genes as the set of negative objects, resulting in an average number of

295 objects for each comparison. We call the brute force-based separability score the *true* separability score, since it examines all possible separating lines. We first find the best feature pair using Rocchio-based measure and the brute force based measure separately (potentially different feature pairs) and then calculate the ratio of the true separability scores of the Rocchio versus the brute force best feature pairs. We observe that the Rocchio-based method picks a best feature pair that has true separability score similar to the best pair picked by brute force, with the ratio of the two scores being better than 0.94 in all ten datasets (Supplementary Figure C.3 (a)). Second, for the best feature pairs identified by Rocchio-based method for the ten datasets, we calculate the ratio of the Rocchio-based separability score and the brute force-based separability score, and find the difference to be greater than 0.96 on average (Supplementary Figure C.3 (b)).

Running Time. Figure 5 depicts the running times of our different selected algorithms. Each plotted box corresponds to one algorithm, representing the distribution of running times for finding the top-k feature pairs (by Rocchio score) for all datasets.



(a) MSigDB (b) LINCS

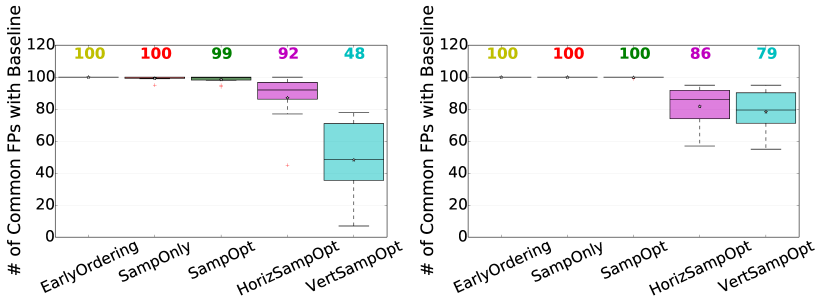
Fig 5. Running Time Comparison. A boxplot for each algorithm is shown with the median value appearing in matching color above. For each boxplot, whiskers are set to be 1.5× the interquartile range, the outliers are shows as red dots, and the average is marked with as a black star. The number on the top shows the median running time for each algorithm.

First, let us compare the median running times among different algorithms. For MSigDB, the Baseline takes more than 2 hours, EarlyOrdering takes less than 1 hour, SampOnly and SampOpt take around 6 and 5 minutes respectively, while HorizSampOpt and VertSampOpt both take only 1 minute on average. Overall, the optimizations result in a reduction of the running time by over 180×. We next examine the effect of different modules on the running time. *(a)* EarlyStop*:* we observe that the EarlyStop module helps achieve a 2× speed up, with the average number of checked objects (genes) reduced from $20K$ to $5K$ (Supplementary Table B.2); *(b)* Sampling*:* the Sampling module helps reduce the running time dramatically, with 20× reduction from Baseline to SampOpt, since on average only $2M$ candidates are generated from all possible $200M$ feature pairs (Supplementary Table B.2); *(c)* Traversal*:* the modules HorizSampOpt and VertSampOpt achieve an additional 6× speed-up compared to SampOpt by terminating after only considering $\chi = 10^7$ feature pairs, approximately $\frac{1}{20}$ of all possible feature pairs. This speedup of HorizSampOpt and VertSampOpt is approaching the limit set by the feature ordering overhead (around $6s$) and the time for the Transformation module (around $8s$) (Supplementary Table B.2). The improvement over SampOpt is not stronger since the candidate generation phase of SampOpt is able to remove a vast amount of the feature pairs from full evaluation that would also be ignored by HorizSampOpt and VertSampOpt (Supplementary Table B.2).

Next, consider the log-scale interquartile range (IQR) of the running times for the different selected algorithms (Figure 5). We observe that EarlyOrdering has the largest interquartile range, indicating that the EarlyStop module, which tries to

reduce the number of objects evaluated for each feature pair, is very dependent on the object set and feature values. As we discussed in Section 2.3.2, EARLYSTOP has no guarantee on improving the running time. In fact, the algorithm can occasionally be worse than the BASELINE as shown in Figure 5(b) because EARLYSTOP incurs additional overhead for checking the criteria for pruning and early termination when scanning the object list for each feature pair. Similar results for LINCS are shown in Figure 5(b) (see Supplementary Note A.4).

Separability Quality. In Supplementary Figure C.3 (a), we found the the accuracy of the baseline method which computes the Rocchio-based estimate of top-k features to be high. The EARLYSTOP module is deterministic and produces the same top-k feature pairs as the baseline method only with optimized computation. The SAMPLING module, on the other hand, is stochastic and can only provide an approximation of the top-k feature pair ranking. Finally, the TRAVERSAL module is heuristic and may output top-k feature pairs that are very different from the ranking produced by the BASELINE algorithm. and since BASELINE returns the true Rocchio-based separability score of each feature pair, we measured the quality of each selected algorithm by counting the number of common feature pairs returned in the top-100 between the BASELINE and the given algorithm. Figure 6 shows this separability quality comparison.



(a) MSIGDB (b) LINCS

Fig 6. Separability Quality Comparison. Boxplots in the style of Figure 5 comparing the number of feature pairs each method returned from the 100 best feature pairs of the Baseline.

Let us first focus on MSIGDB. EARLYORDERING, as expected, has exactly the same separability quality as the BASELINE. We also observe that the SAMPONLY and SAMPOPT rankings are nearly identical to the top-100 feature pairs of the BASELINE, owing to the probabilistic guarantee described in Supplementary Note A.2. The HORIZSAMPOPT and VERTSAMPOPT algorithms output a median of 92 and 48 feature pairs in common with BASELINE, respectively, because of the heuristic TRAVERSAL module. In the MSIGDB results, HORIZSAMPOPT performs much better than VERTSAMPOPT, with the median much higher and the interquartile range much narrower, as shown in Figure 6(a). This suggests, as we hypothesized, that interesting separating feature pairs exist outside of only the combinations of the top single features as in VERTSAMPOPT. We repeated this quality analysis for LINCS and found that the SAMPLING based algorithms returned identical top-100 feature pairs for all 40 datasets. The quality of the TRAVERSAL based algorithms was again lower, though the performance separation of the HORIZSAMPOPT and the VERTSAMPOPT algorithms was not as large as for MSIGDB.

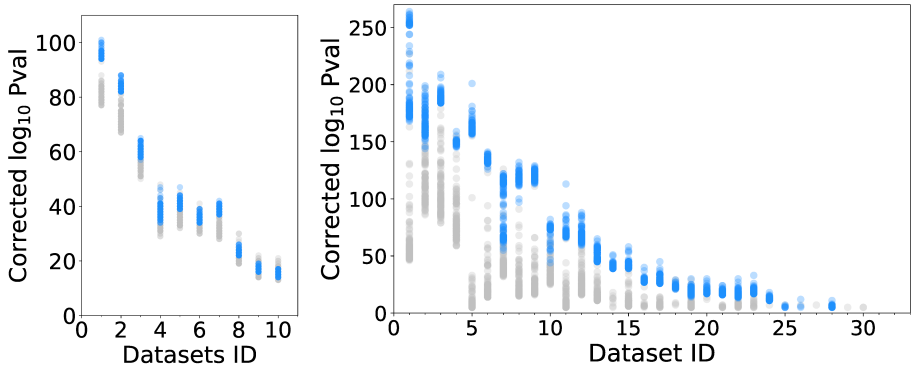
Takeaways. If the accuracy is paramount, SAMPOPT is recommended; if the running time is paramount to the user, HORIZSAMPOPT is recommended.

3.3 Feature Pair vs. Single Feature.

In this section, we quantify the statistical significance of the top ranking results of the selected algorithms. We show that we often find separating feature pairs that are more

significant than the best single separating feature. To assess the significance of a separating feature or feature pair, we first calculate the p-value using the one-sided Fisher's exact test on a 2×2 contingency table. This contingency table is constructed with the rows being the true positive and negative labels, the columns being the predicted positive and negative labels, and the values being the number of objects that belong to each table cell. Using the Fisher's exact test p-value, we assert that feature pairs can provide a better separability compared to single features, i.e., (a) feature pairs have stronger p-values compared to the corresponding individual features even after appropriate multiple hypothesis correction and (b) there exist high-ranked pairs of features that are poorly-ranked on their own as single features.

Single Feature. Finding top-k single features is a special case of finding feature pairs by setting $i = j$. For each single feature obtained, we compute the p-value with Fisher's exact test, denoted as $pval$. Next, we define the Bonferroni corrected p-value as $corrected\_pval = pval \times m \times n$, since there are $m \times n$ possible hypotheses, one for each possible single feature and separating line. We say a selected feature is significant if the corrected p-value is smaller than the threshold 10^{-5}, i.e., $-\log_{10}(corrected\_pval) \geq 5$. In Figure 7, we plot the distribution of the corrected p-value of the top-100 features reported for each dataset in MSIGDB and LINCS. We observe that 10 out of 10 datasets in MSIGDB and 32 out of 40 datasets in LINCS have at least one significant single feature, and will focus on these datasets for further analysis. We observe very small p-values, $\leq 10^{-50}$, in the left part of Figure 7(a) and 7(b), indicating that single features are sufficient to separate the object classes for several datasets well.



(a) MSIGDB (b) LINCS

Fig 7. Single Feature Bonferroni Corrected P-value Distribution vs. Feature Pairs' Corrected P-value Distribution. For each test (x-axis), shows the significance ($-\log_{10}(corrected\_pval)$) of the top-100 best single features (grey dots) and feature pairs (blue dots) for the (a) MSIGDB and (b) LINCS datasets. We order the datasets by their best corrected single feature p-value, and discard the datasets where no single feature has corrected p-value better than 10^{-5}.

Feature Pair. We next build the contingency tables and calculate the p-value for the top-k feature pairs. To correct for m^2 possible feature pairs and the n^2 possible ways to choose the separating lines for each feature pair, we apply a Bonferroni p-value correction to produce the $corrected\_pval = pval \times m^2 \times n^2$. We plot the distribution of the corrected p-values for the top-k feature pairs in Figure 7. Once again, the threshold for defining a significant feature pair is set to 10^{-5}. We find that 10 out of 10 datasets in MSIGDB and 27 out of selected 32 datasets in LINCS have at least one significant feature pair by this metric. Visual comparison of the top-100 single features to the top-100 feature pairs (Figure 7) per dataset reveals several datasets where the corrected p-values of the feature pairs are more significant than those of the best single features, even after accounting for the larger search space. Admittedly, this is not always the

case, e.g., for five LINCS datasets no feature pair was found to be significant at $corrected\_pval \leq 10^{-5}$ while at least one single feature did meet this threshold. Overall, this analysis suggests that rapid discovery of top feature pairs may identify more significant patterns in the given dataset than a traditional single-feature analysis does. In the following, we further illustrate that feature pairs can also provide better and newer insights compared to single features.

Improvement from Single Feature to Feature Pair. Having computed the corrected p-value for each single feature and feature pair in the top-100 for our datasets, we now examine the improvement of each feature pair from its two corresponding single features in terms of p-value. For each feature pair (f_i, f_j), we define the improvement quotient as the ratio between the corrected p-value of (f_i, f_j) and the better one of the corrected p-value of f_i or f_j, i.e., $improv\_quot = \frac{corrected\_pval(f_i,f_j)}{\min(corrected\_pval(f_i),corrected\_pval(f_j))}$. We examined only the *improv_quot* for the top-20 feature pairs for each of the 10 runs in MSigDB and 32 runs in LINCS. We found that on average across these datasets, 9.3 of the top-20 feature pairs in MSigDB and 8 of the top-20 feature pairs in LINCS are more significant than their corresponding single features ($-\log_{10}(improv\_quot) > 5$). The distribution of the *improv_quot* is plotted in Supplementary Figure C.4. Overall, these histograms show that there is a improvement from single features to some feature pairs in terms of the separability significance. Next, we will explore the improved feature pairs more carefully, commenting on their redundancy, reliability, and relevance.

New Insights from Feature Pairs. In order to assess the quality of the top ranking feature pairs, we focused on the LINCS data set where the objects are experimental treatments on the MCF7 breast cancer cell line with the same *drug* and the features are expression values for different *genes*. For the evaluations above, we used object sets for the 40 drugs with the largest number of LINCS experiments. For the following analysis, we refine our list to those that are common drugs and have at least 60 LINCS experiments on the MCF7 cell line. These nine drugs are vorinostat, trichostatin, estradiol, tamoxifen, doxorubicin, gemcitabine, daunorubicin, idarubicin, and pravastatin. For each chosen drug, we ran the SampOpt algorithm of GenVisAGE to rank the top-1000 feature (gene) pairs for separating the LINCS experiments of the drug from all other MCF7 experiments.

For all drugs, except pravastatin, all of the top-1000 ranked feature pairs were found to be significant, i.e. $-\log_{10}(corrected\_pval) > 5$ (see Table 3). As described in the Section 3.3, we are especially interested in feature pairs whose corrected p-value is better than the corrected p-values of their corresponding single features ($-\log_{10}(improv\_quot) > 0$). We found 1070 "improved" feature pairs with larger separability over their single feature among the top1000 of these evaluation drug sets. One drug, trichostatin, had especially strong single features and showed no feature pairs that significantly improved on them. The remaining seven drugs, however, benefited from the feature pair analysis yielding between 9 (tamoxifen) and 369 (doxorubicin) improved feature pairs (Table 3).

Many of the above-mentioned 1070 significantly improved feature pairs are partially redundant, in the sense that they comprise a common best-ranked single feature (gene). An example of this is with the object set for the drug (small molecule) estradiol. We found the gene PRSS23 as the single feature with the highest separability and many feature pairs containing PRSS23 and a second gene as having an improved corrected p-value, for example (PRSS23, RAP1GAP), (PRSS23, TSC22D3), and (PRSS23, BAMBI). We looked for evidence of the relationship between the drug estradiol and these feature pair genes in the Comparative Toxicogenomics Database (CTD) [27] and with our own literature survey. From this search, we found evidence for the pronounced effect of estradiol in increasing expression levels of PRSS23 [28], RAP1GAP [29], and

Drug	NumExprs	Avg Signif	Top1000 Signif	Top1000 Improved
vorinostat	904	235.5	1000	287
trichostatin	689	277.1	1000	0
estradiol	325	166.8	1000	203
tamoxifen	122	105.8	1000	9
doxorubicin	104	28.0	1000	369
gemcitabine	97	52.5	1000	116
daunorubicin	91	40.9	1000	28
idarubicin	78	30.1	1000	58
pravastatin	61	-7.5	0	0
Grand Total		43.1	9068	1070

Table 3. For each chosen drug from LINCS, the number of experiments in MCF7 cell line that were performed with that drug (NumExprs), and statistics for the top1000 feature pairs for that drug including the average $-\log_{10}(corrected\_pval)$ (Avg Signif), number of feature pairs with $-\log_{10}(corrected\_pval) > 5$ (Top1000 Signif), and number with $-\log_{10}(improv\_quot) > 0$ (Top1000 Improved).

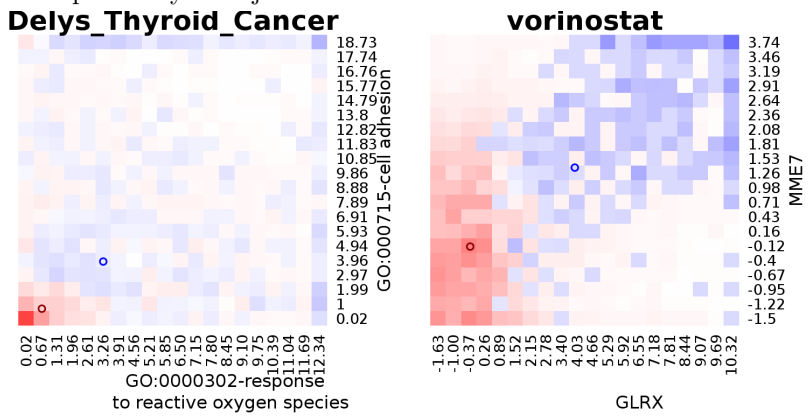
BAMBI [30], and decreasing expression of TSC22D3 [31]. So although the top single feature (gene PRSS23) reoccurred in multiple top feature pairs, each secondary feature gene was also meaningfully related to the administered drug in this case.

We next examined the 1070 improved feature pairs, found over the 9 LINCS datasets, to determine their consistency with existing biological knowledge bases (see Supplementary Note A.5 for details). The interaction networks from these sources covered 23,167 genes and had at least one known interaction between 2.17% of all possible gene pairs. Of the 996 unique feature pairs with significant *improv_quot* where both genes mapped onto the genes covered by the interaction networks, 133 gene pairs (13.4%) were found to have at least one known interaction. This six-fold enrichment demonstrates that GENVISAGE more often finds pairs of genes that have a known relationship than is expected by chance. One example is (GLRX, NME7) that is especially good for separating vorinostat experiments from all others. Not only are both of these genes known to have increased mRNA expression in response to vorinostat [32], [33], but the two genes are annotated by STRING to both be in database pathways of nucleotide biosynthesis, co-express with each other in other model organisms, and mentioned together often in literature abstracts. Later, in Section 3.4, we will demonstrate that the positive objects and negative objects are visually separated under this feature pair, as in Figure 8.

In Supplementary Table B.3, we examine several of the "improved" feature gene pairs reported by GENVISAGE analysis for the LINCS nine drug datasets. Of thirty-nine feature pairs in this table, twelve of them have three types of accompanying evidence: 1) a literature-based relationship between the drug and the first gene, 2) a literature-based relationship between the drug and the second gene, and 3) an interaction network relationship between the pair of genes. Six have two of the three types of evidence and there are only three with no evidence at all. Particularly interesting are the top improved feature pairs in which neither of the single gene features ranked well alone. An example is the gene pair CDKN1A and CEBPB for separating doxorubicin experiments from others. Either gene feature alone is not within the top 600 genes for separating doxorubicin experiments from others. However, the combination of the pair is significant at a corrected p-value of 2×10^{-25} and is the second most improved feature pair for doxorubicin. This feature pair also has all three types of accompanying evidence; doxorubicin is known to increase expression of CDKN1A and CEBPB [34], and the pair of genes are annotated in STRING to have evidence for co-expression and text mining relationships. This feature pair can be used to form an interesting hypothesis for further analysis or experiment. The potential for finding more significant and previously unidentified features is why GENVISAGE is designed to recover top ranking feature pairs instead of just single features.

3.4 Output Visualizations

As discussed in Section 1, the output of GENVISAGE is not simply a ranking of the top feature pairs with their scores, but also a visualization that helps users to interpret the separability. In Figure 8, we depict sample output visualizations from the MSIGDB and LINCS runs. For MSIGDB, we select the feature pair with the highest improved p-value, i.e., *improv_quot*, using the SAMPOPT algorithm. For our LINCS representative, we visualize the gene feature pair (GLRX, NME7) for the drug vorinostat as described in the previous section. For the MSIGDB example (Figure 8(a)), we observe that the feature values for negative objects are clustered around zero, while the genes differentially expressed in papillary thyroid carcinomas from this MSIGDB study have larger values overall, indicating stronger connections to the two Gene Ontology terms features, cell adhesion and response to reactive oxygen species. This is consistent with studies that have highlighted the over expression of important cell adhesion genes in thyroid cancer [35]. For the LINCS example (Figure 8(b)), positive objects mostly have elevated expression for the two reported genes (GLRX and NME7) compared to the negative objects. The direction of this differential gene expression for both genes is consistent with literature for vorinostat experiments [32], [33]. These above two examples illustrate how visualization of significant feature pairs can be a useful way to explain the separability of object sets and understand the data.



(a) MSIGDB (b) LINCS

Fig 8. Visualization Output of GENVISAGE. Heatmap visualization with the pair of top features providing the x and y axes and the name of the run providing the plot title. The relative density of objects determines the color of the heatmap cells with blue indicating a greater proportion of positive objects and red indicating a greater proportion of negative objects. The class centroids are represented by blue (positive class) and red circles (negative class). The two examples shown are representatives from MSIGDB and LINCS datasets.

4 Discussion

The GENVISAGE algorithm with its optimization modules enables researchers to visualize and explore the interplay between important pairs of genomic features rapidly, rather than relying on slow machine learning feature extraction methods or only examining the simple list of top single features. The optimization modules led to a two orders of magnitude speed up in the task of returning the top feature pairs for separating the biological classes in our two benchmark datasets, MSIGDB and LINCS. The quality of these top feature pairs was confirmed by their agreement with literature and interaction databases, and the features are easily understood with intuitive heatmap visualizations. GENVISAGE relies on the Rocchio-based separability measure, which well approximates the best possible linear separator quickly and enables optimizations

like TRANSFORMATION that can pre-compute important quantities from the feature-object matrix before the positive and negative object sets are even provided. One potential downside of the Rocchio-based measure is that because of its dependency on linearity, feature pairs with distinct object class distributions that form complex, non-convex, non-isotropic patterns are potentially very interesting, but will not be well-ranked by GENVISAGE. Finally, in GENVISAGE, the optional SAMPLING module and TRAVERSAL modules make stochastic or greedy decisions in order to estimate the quality of and prune the potential candidate feature pairs for evaluation. While this greatly benefits the amount of time required to find the top ranking pairs, it has the potential to do so at the cost of ranking accuracy. Overall, we observed that for our settings, the sacrifice in accuracy was slight for the SAMPOPT feature pair rankings and more substantial when using the HORIZSAMPOPT and VERTSAMPOPT rankings with the greedy candidate traversal. However, users of GENVISAGE are able to optimize the trade-off with performance and accuracy by modifying the sample size, $|\mathcal{S}|$, used by the SAMPLING module or the number of candidate feature pairs examined, χ, by TRAVERSAL module depending on the needs of their research and dataset.

Funding

This work was supported by the National Institutes of Health Big Data to Knowledge (BD2K) initiative [1U54GM114838, 3U54EB020406-02S1], the National Science Foundation [IIS-1733878], 3M, and Microsoft.

References

1. Tang Z, et al. GEPIA: a web server for cancer and normal gene expression profiling and interactive analyses. Nucleic acids research. 2017;45(W1):W98–W102.

2. Lamb J, et al. The Connectivity Map: using gene-expression signatures to connect small molecules, genes, and disease. science. 2006;313(5795):1929–1935.

3. Liberzon A, et al. The molecular signatures database hallmark gene set collection. Cell systems. 2015;1(6):417–425.

4. Hanczar B, Zucker JD, Henegar C, Saitta L. Feature construction from synergic pairs to improve microarray-based classification. Bioinformatics. 2007;23(21):2866–2872.

5. Geman D, d'Avignon C, Naiman DQ, Winslow RL. Classifying gene expression profiles from pairwise mRNA comparisons. Statistical applications in genetics and molecular biology. 2004;3(1):1–19.

6. Shi P, Ray S, Zhu Q, Kon MA. Top scoring pairs for feature selection in machine learning and applications to cancer outcome prediction. BMC bioinformatics. 2011;12(1):375.

7. Shen R, Luo L, Jiang H. Identification of gene pairs through penalized regression subject to constraints. BMC bioinformatics. 2017;18(1):466.

8. Sinha S, Adler AS, Field Y, Chang HY, Segal E. Systematic functional characterization of cis-regulatory motifs in human core promoters. Genome research. 2008;18(3):477–488.

9. Watkinson J, Wang X, Zheng T, Anastassiou D. Identification of gene interactions associated with disease from gene expression data using synergy networks. BMC systems biology. 2008;2(1):10.

10. Dudoit S, et al. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. Statistica sinica. 2002; p. 111–139.

11. Lai C, et al. A comparison of univariate and multivariate gene selection techniques for classification of cancer datasets. BMC bioinformatics. 2006;7(1):235.

12. Peng J, et al. Regularized multivariate regression for identifying master predictors with application to integrative genomics study of breast cancer. The annals of applied statistics. 2010;4(1):53.

13. Breiman L. Random forests. Machine learning. 2001;45(1):5–32.

14. Basu S, Kumbier K, Brown JB, Yu B. Iterative random forests to discover predictive and stable high-order interactions. Proceedings of the National Academy of Sciences. 2018;115(8):1943–1948.

15. Shah RD, Meinshausen N. Random intersection trees. The Journal of Machine Learning Research. 2014;15(1):629–654.

16. Schwarz DF, König IR, Ziegler A. On safari to Random Jungle: a fast implementation of Random Forests for high-dimensional data. Bioinformatics. 2010;26(14):1752–1758.

17. Stephens ZD, Lee SY, Faghri F, Campbell RH, Zhai C, Efron MJ, et al. Big data: astronomical or genomical? PLoS biology. 2015;13(7):e1002195.

18. Unger G, Chor B. Linear separability of gene expression data sets. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB). 2010;7(2):375–381.

19. Sinha S. Discriminative motifs. Journal of Computational Biology. 2003;10(3-4):599–615.

20. Medin D, Schwanenflugel P. Linear separability in classification learning. Journal of Experimental Psychology: Human Learning and Memory. 1981;7(5):355.

21. Vapnik V, Vapnik V. Statistical learning theory. Wiley New York; 1998.

22. Rocchio J. Relevance feedback in information retrieval. The SMART retrieval system: experiments in automatic document processing. 1971; p. 313–323.

23. Hoeffding W. Probability inequalities for sums of bounded random variables. Journal of the American statistical association. 1963;58(301):13–30.

24. Blatti C, Sinha S. Characterizing gene sets using discriminative random walks with restart on heterogeneous biological networks. Bioinformatics. 2016;32(14).

25. Subramanian A, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proceedings of the National Academy of Sciences. 2005;102(43):15545–15550.

26. Subramanian A, et al. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. Cell. 2017;171(6):1437–1452.

27. Grondin C, et al. Accessing an Expanded Exposure Science Module at the Comparative Toxicogenomics Database. Environmental health perspectives. 2018;126(1).

28. Chan H, et al. Serine protease PRSS23 is upregulated by estrogen receptor α and associated with proliferation of breast cancer cells. PLoS One. 2012;7(1):e30397.

29. Moggs J, et al. Phenotypic anchoring of gene expression changes during estrogen-induced uterine growth. Environmental health perspectives. 2004;112(16):1589.

30. Spink B, et al. Long-term estrogen exposure promotes carcinogen bioactivation, induces persistent changes in gene expression, and enhances the tumorigenicity of MCF-7 human breast cancer cells. Toxicology and applied pharmacology. 2009;240(3).

31. Sengupta S, et al. Molecular mechanism of action of bisphenol and bisphenol A mediated by oestrogen receptor alpha in growth and apoptosis of breast cancer cells. British journal of pharmacology. 2013;169(1):167–178.

32. Qi Y, et al. Systematic analysis of time-series gene expression data on tumor cell-selective apoptotic responses to HDAC inhibitors. Computational and mathematical methods in medicine. 2014;2014.

33. Soldi R, et al. A genomic approach to predict synergistic combinations for breast cancer treatment. The pharmacogenomics journal. 2013;13(1):94.

34. Zhao W, et al. Gene expression profiling identifies the novel role of immunoproteasome in doxorubicin-induced cardiotoxicity. Toxicology. 2015;333:76–88.

35. Gorka B, et al. NrCAM, a neuronal system cell-adhesion molecule, is induced in papillary thyroid carcinomas. British journal of cancer. 2007;97(4):531.

36. Consortium GO. Gene ontology consortium: going forward. Nucleic acids research. 2014;43(D1):D1049–D1056.

37. Finn R, et al. The Pfam protein families database: towards a more sustainable future. Nucleic acids research. 2015;44(D1):D279–D285.

38. Fabregat A, et al. The reactome pathway knowledgebase. Nucleic acids research. 2017;46(D1):D649–D655.

39. Kanehisa M, et al. KEGG: new perspectives on genomes, pathways, diseases and drugs. Nucleic acids research. 2016;45(D1):D353–D361.

40. Altschul S, et al. Basic local alignment search tool. Journal of molecular biology. 1990;215(3):403–410.

41. Szklarczyk D, et al. STRING v10: protein–protein interaction networks, integrated over the tree of life. Nucleic acids research. 2014;43(D1):D447–D452.

42. Croft D, et al. The Reactome pathway knowledgebase. Nucleic acids research. 2013;42(D1):D472–D477.

43. Cerami E, et al. Pathway Commons, a web resource for biological pathway data. Nucleic acids research. 2010;39(suppl_1):D685–D690.

44. Lee I, et al. Prioritizing candidate disease genes by network-based boosting of genome-wide association data. Genome research. 2011;21(7):1109–1121.

45. Chatr-Aryamontri A, et al. The BioGRID interaction database: 2017 update. Nucleic acids research. 2017;45(D1):D369–D379.

46. Orchard S, et al. The MIntAct project-IntAct as a common curation platform for 11 molecular interaction databases. Nucleic acids research. 2013;42(D1):D358–D363.

47. Salwinski L, et al. The database of interacting proteins: 2004 update. Nucleic acids research. 2004;32(suppl_1):D449–D451.

Appendix A Supplementary Notes

A.1 Pre-Transformation Module

Let us review the process of computing the separability $\theta_{i,j}$. Given a feature pair (f_i, f_j) and the corresponding positive and negative centroids, *(i)* we first compute w_0, w_i and w_j for L. Next, for each object o_k, *(ii)* we obtain the predicted label $\eta_{i,j}^{L,k}$ according to Equation 1. This step requires two multiplications and three additions. Finally, *(iii)* we calculate $\theta_{i,j}^k$ and the separability $\theta_{i,j}$ based on formulations in Section 2.2. This whole process is repeated for every feature pair candidate. However, there is massive redundancy across the processing of different feature pairs. For instance, when calculating the separability for two different feature pairs (f_i, f_j) and $(f_i, f_{j'})$ with a common f_i, w_i is in fact shared, and calculation of $w_i \cdot \mathcal{M}_{i,k}$ in Equation 1 is repeated for each object o_k.

Given this, we propose the TRANSFORMATION optimization module which will pre-calculate some common computational components once across different features and reuse these components the separability for each feature pair to eliminate the repeated computation. This TRANSFORMATION module transforms the original $\mathcal{M}_{i,k}$ matrix into another space using our identified common feature pair components and updates the separability score equation accordingly.

For each feature f_i, we find the average values of the positive and negative objects for that feature, $\mathcal{M}_i^+$ and $\mathcal{M}_i^-$ respectively, and then we pre-transform $\mathcal{M}_{i,k}$, i.e., the value of object o_k on the feature i, to $\widehat{\mathcal{M}}_{i,k} = \left((\mathcal{M}_i^+ - \mathcal{M}_i^-) \cdot \mathcal{M}_{i,k} - \frac{(\mathcal{M}_i^+)^2 - (\mathcal{M}_i^-)^2}{2}\right) \cdot l_k$. The basic idea is to decompose Equation 1 into two components, with each one only related to a single feature. This transformation incorporates the class centroids into the matrix values, obviating their integration later for every feature pair that involves the given feature. We also multiplies in the class label of the object, l_k, rather than repeating this multiplication every time the object is evaluated (see example in Supplementary Figure C.1). With this transformation of the feature-object matrix, evaluating whether an object was correctly separated is simplified as: if $sign(\widehat{\mathcal{M}}_{i,k} + \widehat{\mathcal{M}}_{j,k}) = 1$, then $\theta_{i,j}^k = 1$; otherwise, $\theta_{i,j}^k = 0$. Note that this step only involves one addition and one comparison and is performed only once for each feature. Next, we can calculate overall separability score $\theta_{i,j} = \sum_k \theta_{i,j}^k$. Overall, we not only eliminate the steps of computing w_0, w_i and w_j for every feature pair, but also reduce the cost of calculating $\eta_{i,j}^k$ in Equation 1. With the TRANSFORMATION module, we calculate $\widehat{\mathcal{M}}$ as a pre-transformation step, and use it when evaluating feature pairs instead of $\mathcal{M}$ and calculate $\theta_{i,j}$ accordingly.

A.2 Estimation Accuracy in SAMPLING Module

We have proposed SAMPLING for estimating $\theta_{i,j}$ in Section 2.3.3. Next, we formally quantize the sample set size in Theorem 1.

Theorem 1 (Estimation Accuracy)**.** *By considering* $\Omega(\frac{1}{\epsilon^2} \cdot \log(\frac{1}{\delta}))$ *samples, with probability at least* $1 - \delta$*, we have* $|\frac{\theta_{i,j}(\mathcal{S})}{|\mathcal{S}|} - \frac{\theta_{i,j}}{n}| \leq \epsilon$*, i.e.,* $|\tilde{\theta}_{i,j} - \theta_{i,j}| \leq \epsilon n$*.*

We can treat $\log(1/\delta)$ as a constant, e.g., by setting $\delta = 0.05$. Thus, Theorem 1 essentially states that with only $\Omega(\frac{1}{\epsilon^2})$ samples, with probability 95%, the confidence interval for $\theta_{i,j}$ is $[\tilde{\theta}_{i,j} - \epsilon n, \tilde{\theta}_{i,j} + \epsilon n]$.

A.3 Construction of MSIGDB Feature-Object Matrix

To construct the feature-object matrix for MSIGDB, which has gene objects and gene property features, we collected prior knowledge about gene annotations and protein

homology from several databases. Our gene annotations and gene properties were extracted from Gene Ontology terms [36], PFam domains [37], and Reactome [38] and KEGG [39] pathways. We constructed a heterogeneous network with nodes for all 22,210 genes and 21,235 properties from these databases and with edges representing their annotations between genes and properties. We also created weighted homology-based edges in the network between pairs of genes based on their protein sequence similarity as determined by BLAST scores [40]. We used the first phase of the DRaWR algorithm [24] with a restart probability of 0.5 to perform a random walk restarting from each gene node on the heterogeneous network, thereby scoring the connectivity of all nodes in the network to the gene. For each gene-property pair (g, r), we assigned the numeric value from the random walk stationary probability distribution that represents not only whether the gene is annotated with that property, but also whether other genes closely related to gene g are annotated with property r. We thus obtained a feature-object matrix describing each gene (object) as a vector of its strength of association with each property (feature) in light of prior biological knowledge.

A.4 Speedup Analysis for LINCS

In Figure 5(b), we observe over 400× average decrease in the running time of finding the top-k feature pairs that separate the LINCS experiments of a single perturbagen from others. The greatest speedup comes with adding the SAMPLING module, where only $100K$ feature pair candidates, i.e., $|\mathcal{C}|$, are checked out of all $250M$ feature pairs (Supplementary Table B.2). For the selected algorithms with best running times, HorizSampOpt and VertSampOpt, the pre-transformation and feature ordering overhead account for an average of $45 + 35 = 80s$ of the overall 104 and 94 median seconds respectively.

A.5 Gene Interaction Datasets

For our knowledge bases of protein and gene interactions, we downloaded datasets derived from 8 data sources: STRING [41], Reactome [42], Pathway Commons [43], HumanNet [44], BioGRID [45], Intact [46], DIP [47], and BLAST [40] databases. The datasets were downloaded, harmonized, and mapped to Ensembl gene identifiers using the KnowEnG Knowledge Network Builder `https://github.com/KnowEnG/KN_Builder`. The final processed network used in this work can be downloaded from `https://github.com/KnowEnG/KN_Fetcher/blob/master/Contents.md#gene`.

Appendix B Supplementary Tables

B.1 GENVISAGE Method Notation

Notation used in this paper.

Symb.	Description	Symb.	Description
$\mathcal{M}$	feature-object matrix	$\mathcal{F}$	feature set in $\mathcal{M}$
f_i	feature i in $\mathcal{F}$	m	number of features in $\mathcal{F}$
$\mathcal{O}$	object set in $\mathcal{M}$	N	number of objects in $\mathcal{O}$
$\mathcal{O}_+$	positive object set	$\mathcal{O}_-$	negative object set
$\widehat{\mathcal{O}}$	labelled object set	n	number of labelled objects in $\widehat{\mathcal{O}}$
o_k	object k in $\widehat{\mathcal{O}}$	l_k	label of object o_k
ℓ	separating line in 2-D	L	representative line in 2-D
$\eta_{i,j}^{\ell,k}$	predicted label of o_k	$\theta_{i,j}^{\ell,k}$	o_k is correctly separated?
$\theta_{i,j}^{\ell}$	# correctly separated o_k	$\theta_{i,j}$	separability score
$\widetilde{\mathcal{M}}$	$\mathcal{M}$ after transformation	$\tilde{\theta}_{i,j}$	estimated $\theta_{i,j}$

B.2 Running Time Detailed Comparison

For each dataset collection, MSigDB and LINCS, show two statistics FPsChecked and ObjectsChecked, for each optimization module phase (columns). FPsChecked is the number of feature pairs evaluated in the phase, and ObjectsChecked is the average number of sample objects that are evaluated across all feature pairs.

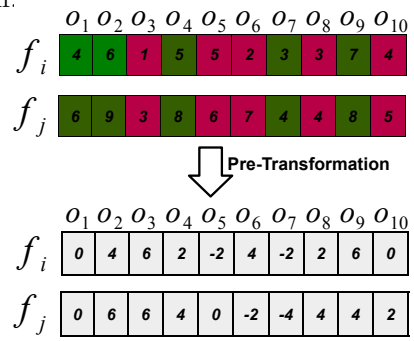
B.3 In-depth Analysis of 'Improved' Feature Pairs

For each drug dataset (col C), we return a limited number of gene feature pairs (cols E,F) that after correction were the most "improved" over their corresponding single feature results either by the change in the corrected_pvalue (cols H,I,J,N,O) or the change in the feature rankings (cols K,L,M,P). Pubmed IDs (cols Q,R) are provided when the relationship between the drug and the gene feature were found in the Comparative Toxicogenomics Database (CTD) or by manual literature search (denoted by *-asterisk). When relationships between the gene features themselves were discovered, the type and strength of the relationships were reported (col S) and the number of relationships quantified (col T).

Appendix C Supplementary Figures

C.1 Example for TRANSFORMATION Module

The TRANSFORMATION module is applied once to the original values in the feature-object matrix $\mathcal{M}$ (above) to produce $\widehat{\mathcal{M}}$ (below). For two features, f_i and f_j. The top half depicts $\mathcal{M}_{i,k}$ and $\mathcal{M}_{j,k}$ before transformation, where green color represents a positive label and red color represents a negative label. In this example, the centroids of the positive and negative objects are $\mu^{+}_{i,j} = (5, 7)$ and $\mu^{-}_{i,j} = (3, 5)$ respectively. Hence, we can rewrite $\widehat{\mathcal{M}}_{i,k} = (2\mathcal{M}_{i,k} - 8) \cdot l_k$ and $\widehat{\mathcal{M}}_{j,k} = (2\mathcal{M}_{i,k} - 12) \cdot l_k$ for features f_i and f_j respectively. After calculation, we can obtain the values for $\widehat{\mathcal{M}}_{i,k}$ and $\widehat{\mathcal{M}}_{j,k}$ shown in the bottom half.

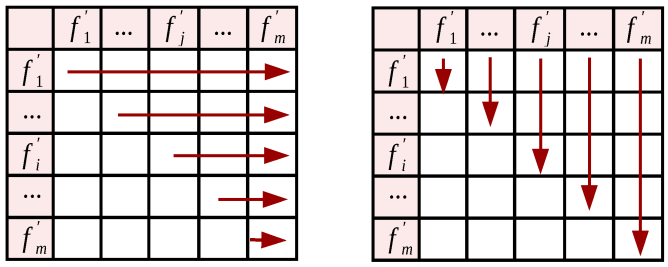


C.2 Illustration for TRAVERSAL Module

We rank individual features based on their single feature separability scores, $\theta_{i,i}$, from best to worst, $\{f_1^{'} f_2^{'}, \cdots, f_m^{'}\}$.

- *Horizontal traversal:* For each feature $f_i^{'}$, pair it with each other feature $f_j^{'}$, where $j \geq i$, to obtain $(f_i^{'}, f_j^{'})$. Repeat for each $f_i^{'}$, where $1 \leq i \leq m$.
- *Vertical traversal:* For each feature $f_j^{'}$, pair it with each other feature $f_i^{'}$, where $i \leq j$, to obtain $(f_i^{'}, f_j^{'})$. Repeat for each $f_j^{'}$, where $1 \leq j \leq m$.

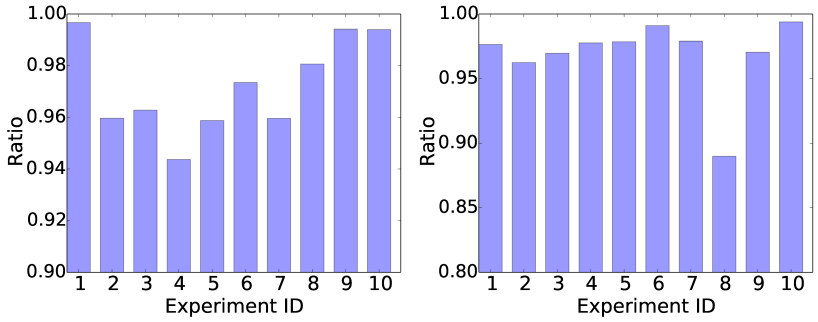
For an example, suppose there are 20,000 features, $m = 2 \times 10^4$. Initially, the number of possible feature pairs is roughly $\frac{m^2}{2} = 2 \times 10^8$. However, if we limit the number of considered feature pairs to $\chi = 10^7$, we reduce our search space to $\frac{1}{20}$ of the



total number of feature pairs. We order the single features by their individual separability scores. In horizontal traversal, only feature pairs with at least one individual feature ranked in the top 500 will be considered; while vertical traversal will consider only feature pairs with both individual features ranked better than 2000.

C.3 Separability Score Comparison

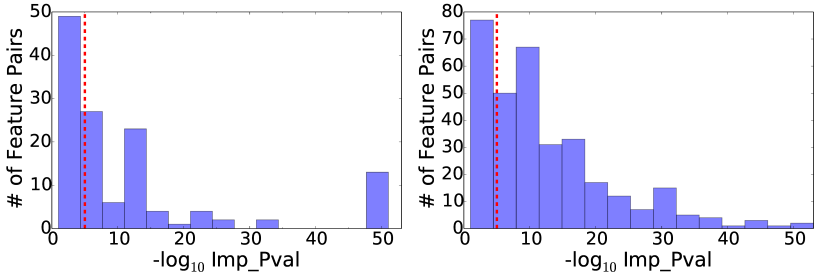
Comparison of Brute Force-based and Rocchio-based separability score. (a) For each of 10 datasets, we display the ratio of the true separability score between the best feature pair chosen by brute force and by the Rocchio-based method. (b) For each dataset, we display the ratio of the true separability score and the Rocchio-based separability score for the best feature pair selected using Rocchio-based method.



(a) Best Feature Pair Comparison (b) Measure Comparison

C.4 Histogram of *improv_quot*

Histogram of *improv_quot*. For the top-20 feature pairs from all runs from the (a) MSigDB and (b) LINCS datasets, distribution of the improvement of the feature pair significance over the corresponding single feature significance. The red line shows the significance threshold of 5.



(c) MSigDB (d) LINCS