Improving Deconvolution Methods in Biology through Open Innovation Competitions: an Application to the Connectivity Map

Andrea Blasco (Harvard) Ted Natoli (Broad) Michael G. Endres (Harvard)

Rinat A. Sergeev (Harvard) Steven Randazzo (Harvard) Jin H. Paik (Harvard)

Max Macaluso (Broad) Rajiv Narayan (Broad) Karim R. Lakhani (Harvard)

Aravind Subramanian (Broad)

Last updated: Jan 05, 2020

Abstract

The Connectivity Map (CMap) project has generated millions of perturbational gene expression profiles using a new assay, called L1000. Using computational deconvolution methods, the L1000 platform characterizes the expression of 1,000 genes from composite measurements obtained by 500 analytes. This procedure greatly lowers cost allowing CMap's massive scale-up. However, the current deconvolution algorithm is slow (~5 minutes per experimental plate) and susceptible to errors. Here we describe new methods obtained via an open innovation challenge. The challenge used a novel dataset of 2,200 L1000 profiles of perturbation experiments for evaluation and attracted 294 competitors from 20 countries. The top-nine performing methods ranged from machine learning (Convolutional Neural Networks and Random Forests) to more traditional solutions (Gaussian Mixtures and k-means). These approaches were faster and more accurate than the benchmark and likely have applications beyond gene expression.

Keywords: connectivity map; open innovation competitions; crowdsourcing; deconvolution; gene expressions; machine learning; transcriptome.

1 Introduction

Deconvolution problems are commonplace in many areas of science and engineering. In the context of biomedical research, a recurring issue is how to isolate signals of distinct populations (cell types, tissues, and genes) from composite measures obtained by a single analyte or sensor. This problem often stems from the prohibitive cost of profiling each population separately [1, 2] and has important implications for the analysis of transcriptional data in mixed samples [3, 4, 5, 6], single-cell data [7], and the study of cell dynamics [8], but it also appears in the analysis of imaging data [9].

Existing computational deconvolution approaches work well in many specific settings [10] but they might be suboptimal in more general applications. Machine learning techniques can potentially improve upon current methods. A typical advantage is the ability to capture complicated patterns that can be hard to model otherwise, especially in complex and massive datasets as those frequently used in biomedical research. However, introducing machine learning in the field presents several challenges; some of which are validation, adaptation to complex datasets, and identification of the best machine-learning approaches to specific problems.

Here we describe how we addressed these challenges in the context of the Connectivity Map (CMap). The CMap project is a catalog of over 1.3 million human gene-expression profiles of genetic or pharmacologic perturbation. This resource enables rapid hypothesis development in multiple areas of biomedical research, including drug discovery and development [1]. CMap has achieved its massive scale by using a new assay, called L1000. This assay uses a 500-plex fluorescence-based flow cytometry system to measure a reduced representation of the transcriptome comprised of 1,000 "landmark" genes that largely capture the cell's transcriptional state. L1000 achieves a significant cost reduction compared to more traditional methods, such as RNA-sequencing. However, a key technical challenge is to deconvolute the 500 measurements into separate 1,000 gene expression values. The current approach, a k-means algorithm called "dpeak", is slow and can be susceptible to errors. Below we report new methods, obtained via an open innovation challenge, that represent improvements to the current algorithm.

For the evaluation of these new methods, we generated a novel experimental dataset of L1000 profiles for 122 different perturbagens (shRNA and compounds) and several replicates for a total of over 2,200 gene expression

experiments. We varied the detection mode for acquiring L1000 data between the current dual-detection procedure (DUO) that obtains a raw composite measure of two genes per analyte color, and a more expensive uni-detection procedure (UNI) that measures one gene per analyte color. The deconvolution algorithm processes the DUO data and assigns the correct expression level to each of the two genes whose transcripts bind to beads of the same analyte color. This procedure is not needed with the UNI data. Hence, the UNI data served as "ground truth" that enabled us to evaluate different deconvolution methods applied to the DUO data.

Leveraging this data set, we then explored different deconvolution approaches through an open innovation competition [11, 12]. We ran the contest on Topcoder (Wipro, India), a popular crowdsourcing platform. The contest challenged participants to use the novel dataset to improve the deconvolution algorithm utilized by the L1000 platform. The contest drew about 300 competitors from 20 different countries and resulted in a diversity of approaches. The top approaches included machine-learning methods, such as Random Forests and Convolutional Neural Networks (CNNs), as well as more traditional Gaussian-mixture models and k-means. These approaches significantly performed better than the L1000 benchmark in various measures of accuracy and computational speed, and likely have application beyond gene expression.

2 Methods

We provided competitors with a problem statement, access to training and testing data, and a well-defined scoring function. Figure 1 shows a schematic illustration of these three elements.

The problem statement described L1000's deconvolution task and the current solution. The key insight was that the distribution of the composite measurements of two differentially expressed genes should have two peaks and the size of each peak should reflect the proportion of measurements for the corresponding gene. L1000 takes full advantage of this fact by pairing genes optimally, trying to maximize the average difference in their expression levels, and by mixing genes in a 2:1 proportion to enable the assignment of the correct expression levels to each gene within each pair [see 1, for the details]. Then it uses the dpeak deconvolution algorithm to detect the peaks within the composite distribution. This procedure is based on a k-means clustering algorithm that partitions the composite measurements for each profile into k clusters by minimizing the within-cluster sum of squares. It then associates the largest cluster to the gene with higher bead proportion, and the smallest cluster to the gene with lower bead proportion, assigning their median values to the corresponding gene.

The training and testing datasets are publicly available (S1 Data). These data consisted of six 384-well perturbagen plates, each containing mutually exclusive sets of compound and short-hairpin (shRNA) treatments (S1 Table and S2 Table show a complete list of the perturbagens). Multiple cell lines and perturbagen were used to avoid potentially over-fitting to any one. The compound and shRNA perturbagen plates were arbitrarily grouped into pairs, and to avoid any potential 'information leakage' each pair was profiled in a different cell line. The resulting lysates were amplified by Ligation Mediated Amplification (LMA, Subramanian et al. [1]). The amplicon was then split and detected in both UNI and DUO detection modes, resulting in three pairs of data generated under comparable circumstances. The training data was available for all the contestants to develop and validate their solutions offline. The testing data was used for submission evaluation during the contest and to populate a live leaderboard. The holdout data was for final evaluation,

thus guarding against over-fitting. Prizes were awarded based on performance on the holdout dataset.

The scoring function combined measures of accuracy and computational speed (S1 Appendix). The accuracy metric was the product of two different metrics. The first was the average genewise Spearman's rank correlation between the deconvoluted expression values and the ground truth. The second was the Area Under the receiver operating characteristic Curve (AUC) in the prediction of extremely modulated genes. Speed was measured by executing each submission on comparable multi-core machines, thus allowing competitors to employ multithreading techniques, and the corresponding score was the average runtime in units of the benchmark runtime.

The contest lasted 21 days. A prize purse of $23,000 in cash was offered to competitors as an incentive to be divided among the top 9 submissions.

3 Results

The contest attracted 294 participants who made 820 submissions using a variety of different methods (S1 Table). We report the top four methods based on the holdout data.

The winning solution (by a competitor from the United States with a degree in Physics from the University of Kansas) used a random forest algorithm. The algorithm combines predictions from 10 different trees trained on 60 derived data features. These features include a combination of low-peak and high-peak estimates for each gene pair and aggregate measures that are sensitive to systematic bias at the perturbagen, analyte, and plate level.

The second solution (by a competitor from Poland with a Master's degree in Computer Science from Lodz University of Technology) used the Expectation-Maximization (EM) algorithm to fit a mixture of two log-normal models to each gene pair's data. This algorithm does not assume any a priori probability (the 2:1 ratio) of assignment to clusters, but learns it from the data by fitting a plate-wide distribution of cluster sizes.

The third solution (by a competitor from India with a bachelor's degree in Computer Science) used a fast k-means algorithm with a random initialization procedure that tends to avoid local minima and is more robust to extreme outliers.

The fourth solution (by a competitor from Ukraine with a bachelor's degree in Computer Science from the Cherkasy National University) used a Convolutional Neural Network (CNN). This algorithm first filters and transforms the data into a 32-bin histogram for each pair of genes. Then, it uses the U-net architecture [13], comprising a contracting path to capture context and a symmetric expanding path, to provide adequate representation of the data. The output of this network is then used to label bins into one of the two genes for each pair, and to predict the exact value within the bin. This second step uses two subnetworks with the same architecture and a mean squared error loss function.

3.1 Clustering by method and perturbagen type

We evaluated the top-nine performing methods on the accuracy of their predictions, as well as their speed. Using the holdout dataset, we generated the contestants' deconvolution data (DECONV) and the corresponding

differential expression (DE) values [as in 1]. We then compared the results using a two-dimensional t-SNE projection, run once on each of the entire DECONV and DE datasets [14].

The DECONV data clustered well by pertubagen type and less by algorithm type (Figure 2, A and B), although with some notable commonalities in the predictions generated by similar approaches (Figure 2, C). For example, the decision tree regressor (DTR) algorithms have similar 'footprints' in the projection, as do the k-means and Gaussian mixture model (GMM) algorithms. After the transformation to DE data, however, the t-SNE projection was more homogenous, with no particular clustering by perturbagen and algorithm type (Figure 2 D), which was reassuring given the downstream analysis normally concentrates on DE values.

3.2 Correlation accuracy

To evaluate the deconvolution accuracy, we used the genewise Spearman rank correlation (ρ) between the UNI data and the values obtained by the competitors through the deconvolution of DUO data. We did so for the shRNA and compound perturbagens separately, comparing the results between the subsets of 488 genes in high and low bead proportion.

The winner's cumulative distribution of ρ's was significantly shifted towards higher values compared to the benchmark's ($p < 0.001$; Figure 3, a and b) with an average improvement that was twice as large with genes in low bead proportion compared to those in high bead proportion (5 and 2 percentage points, respectively).

The other competitors showed similar improvements on average (Figure 3, c and d), although the average improvement was generally smaller compared to the winning method and, for the genes in low bead proportion, insignificantly different from the benchmark (Figure 3, c and d).

To evaluate the extent to which the winning algorithm outperformed the others, we ranked the top-nine algorithms by the average correlation metric for each genes (1 = highest, 9 = lowest). We then computed the percentage of genes for which a given algorithm was ranked first. The winner was ranked first for 30% of the genes, followed at some distance by the second-placed gaussian-mixture method (20%), and by the CNN method (13%). Thus, the top two submissions combined outperformed the rest for about half of the genes in our sample. Even so, all but a few algorithms were the best performers for at least 5% of the genes, suggesting some complementarity between these algorithms.

3.3 Detection of extreme modulations

To evaluate the accuracy at the DE level, we used the detection accuracy of extreme modulations (genes notably up- or down-regulated by perturbation). We used the UNI data with DE values above a threshold as the ground truth; and we evaluated the detection accuracy of each solution by computing the corresponding AUC for each perturbagen type. The detection accuracy of extreme modulations was generally high for both shRNA and compound samples (AUC > 0.87 and AUC > 0.91, respectively), with the competitors achieving notable improvements over the benchmark (Figure 4, a). Compared to the benchmark, the winning solution detects about 4 thousand less extreme modulations (40.8 and 44.2 thousand, respectively), thus being more conservative. However, when we restricted the comparison to extreme modulations detected by UNI as well (thus controlling for detection precision), the winning solution detects about 1.5 thousand more EMs than

the benchmark (27.1 and 25.6 thousand, respectively), representing a sensible 6% increase in "true" EM detection.[1]

We complemented the above analysis by using targeted gene knockdown (KD) experiments as the ground truth for a subset of data. These are experiments in which a landmark gene was targeted by an shRNA, and hence we expect to observe a significant decrease in expression for the targeted gene. We evaluated the KD detection accuracy of each solution by computing the corresponding percentage of successful KD genes identified or recalled by the algorithm (defining a successful KD as one gene in which the DE value and the corresponding gene-wise rank in the experiment are less than a given threshold, -2 and 10 respectively). We computed the percentage recall for the UNI data as well, which yielded an estimate of the maximum achievable recall of 0.80. Relative to this level, nearly all algorithms achieved a good recall and precision, with values that were higher than the benchmark solution for all but two methods (Figure 4, b).

3.4 Reduced variation across replicate samples

To evaluate the reproducibility of the results, we leveraged the several replicate samples for each shRNA and compound experiment in our dataset (about 4 and 10 replicates, respectively). We computed the mean gene-wise coefficient of variation (CV) for each method, which is a measure of inter-replicate variability computed as the average ratio between the interquartile range and the median value across all the replicates. Using this measure, we found all solutions achieved significant improvements over the benchmark (Figure 5); and the winning method, which was the most accurate on average, also achieved the lowest inter-replicate variation overall.

3.5 Computational speed

The speed improvements over the benchmark were substantial. While dpeak took about 4-5 minutes per plate, the fastest algorithm took as little as 5 seconds per plate (more than a 60x speedup compared to the benchmark) and the slowest was well below one minute. These speed improvements are not directly attributable to the use of multiple cores, since both the benchmark and contestant algorithms leverage multi-core techniques. We observed no particular trade-off between speed and accuracy.

3.6 Ensembles

Lastly, we assessed the performance of ensembles combining the predictions of different computational methods by taking the median value. By focusing on the subset of the data with shRNA experiments (ignoring the data with compound experiments), the performance in both Spearman correlation and the AUC metrics of the ensemble tended to increase with the number of models involved (Figure 6). However, the maximum performance in both metrics tended to plateau (or even decrease) after combining the results of 3 or more models. This result suggested limited gains from having ensembles, although it may be worth exploring more sophisticated aggregation approaches.

[1] These results are for the dataset with shRNA experiments. We expect similar results for the dataset of compound experiments.

4 Discussion

We created a novel dataset of L1000 profiles for over 120 shRNA and compound experiments with several replicates for a total of 2,200 gene expression profiles of genes measured independently, and in tandem. This dataset constitutes now a public resource (S1 Data) to all the researchers in this area who are interested in testing their deconvolution approaches.

Using an open innovation competition, we collected and evaluated multiple and diverse deconvolution methods. The best approach was based on a random forest, which is a collection of decision tree regressors. This method achieved: (i) the highest global correlation between the ground-truth and the corresponding deconvoluted data, (ii) the lowest inter-replicate variation, and (iii), compared to the benchmark, was able to detect more than a thousand additional extremely modulated genes, while reducing the false positives at the same time. Our analysis further showed that these gains are considerable when the gene populations were sampled in different proportions (here, genes in high and low bead proportions), with the k-means benchmark approach being systematically less accurate because it does little to mitigate the discrepancy in variability between the genes measured with high and low bead proportion.

In addition, the random-forest approach achieved these improvements with only 10 trees on 60 features. Thus, the algorithm is also relatively fast and easy to implement. By comparison, the fastest approach used a more traditional Gaussian mixture model (with plate-level adjustments), which turned out to be less accurate. Hence, and overall, our analysis provided evidence of the tremendous potential of using random-forest methods for deconvolution problems in biology.

5 Tables and Figures

Figure 1

Schematic illustrating the computational problem, generated data, and scoring function. Panel (**a**) provides a schematic description of the L1000 DUO detection mode and the associated deconvolution problem to be addressed by the contestants. Panel (**b**) shows the data generated for the contest comprising 6 different sets of perturbational experiments with 3 plates of compound (CP) and shRNA treatments (KD) each. Each plate was detected using DUO (2 genes per analyte) and UNI (one gene per analyte) with UNI serving as the ground truth. Contestants were given two plates of data for training their models offline; a second set of two plates was used during the contest for testing and to populate the live leaderboard; and the third set of two was used as holdout to determine the final contestant placements. Panel (**c**) illustrates the accuracy component of the scoring function that was used to evaluate the solutions submitted by the competitors. A solution's overall accuracy score was the product of the genewise Spearman rank correlations with ground truth (DECONV data) and the area under the curve AUC of extreme modulations (DE data).

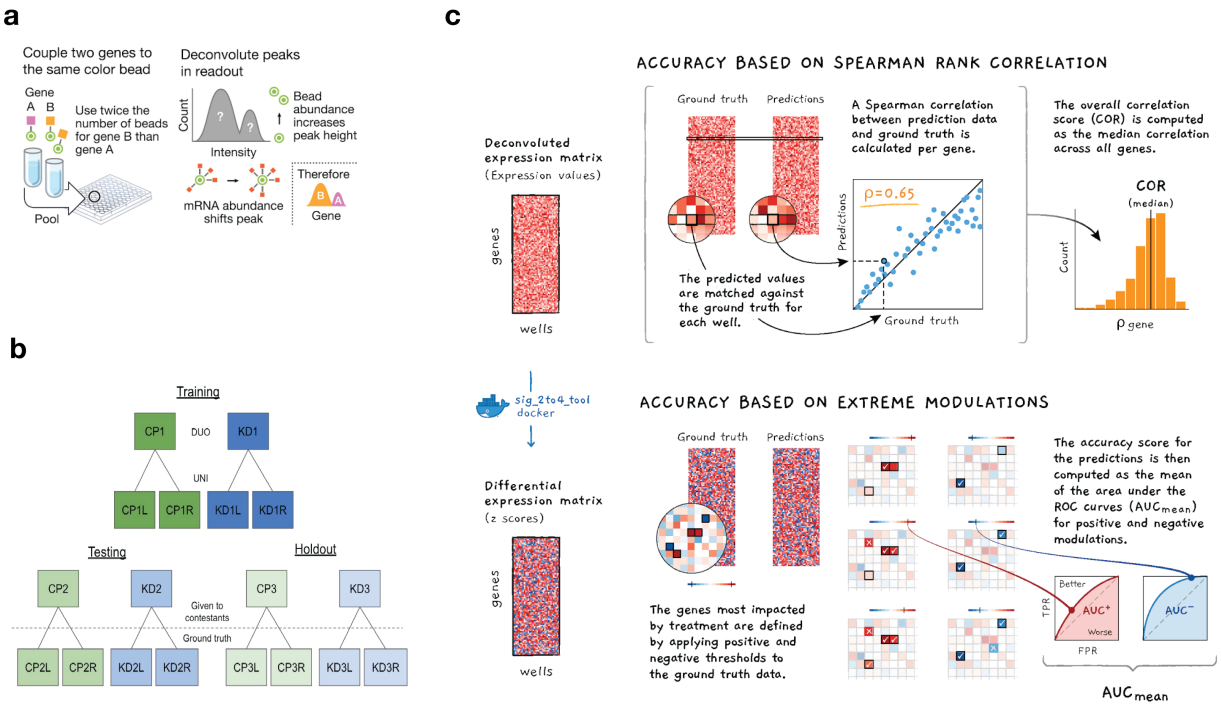


Figure 2

Clustering of solutions. Each point represents the two-dimensional projection of a sample generated by UNI ground truth (GT) or by applying a deconvolution algorithm to DUO data. t-SNE was run on the 2 plates of holdout data, one each containing compound and shRNA treatmens. t-SNE was run once on all DECONV data and once on all DE data. The resulting projections were colored and subset to generate the following panels: DECONV data colored by perturbagen type (**a**) and algorithm type (**b**). DECONV (**c**) and DE (**d**) data colored by algorithm type and stratified by each individual implementation.

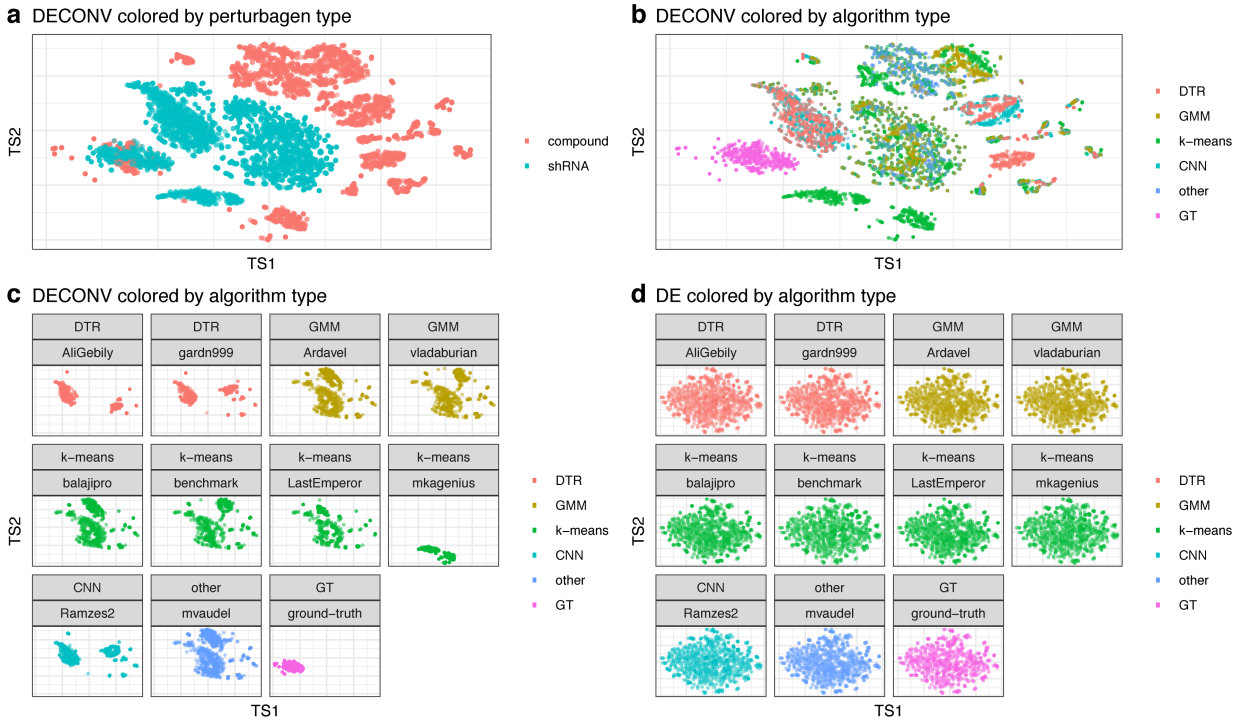


Figure 3

Correlation between ground-truth and deconvolution samples. This figure shows the distribution of the genewise Spearman correlation coefficients between the ground-truth (UNI) and the deconvolution data (DUO) achieved by the winning random-forest method against the k-means benchmark (dpeak*) for the subset of genes in low bead proportion (**a**) and in high bead proportion (**b**) for the shRNA and compound perturbagen types. It also shows the average and standard error of the correlation coefficients for the top-nine performing methods and the benchmark for the genes in low bead proportion (**c**) and those in high bead proportions (**d**) for both compound and shRNAs experiments. Asterisks indicate statistical significance at *** 0.001, ** 0.05, and * 0.1 level of a Wilcoxon-rank-sum test of location difference between the competitor's distribution and the corresponding distribution of the benchmark.

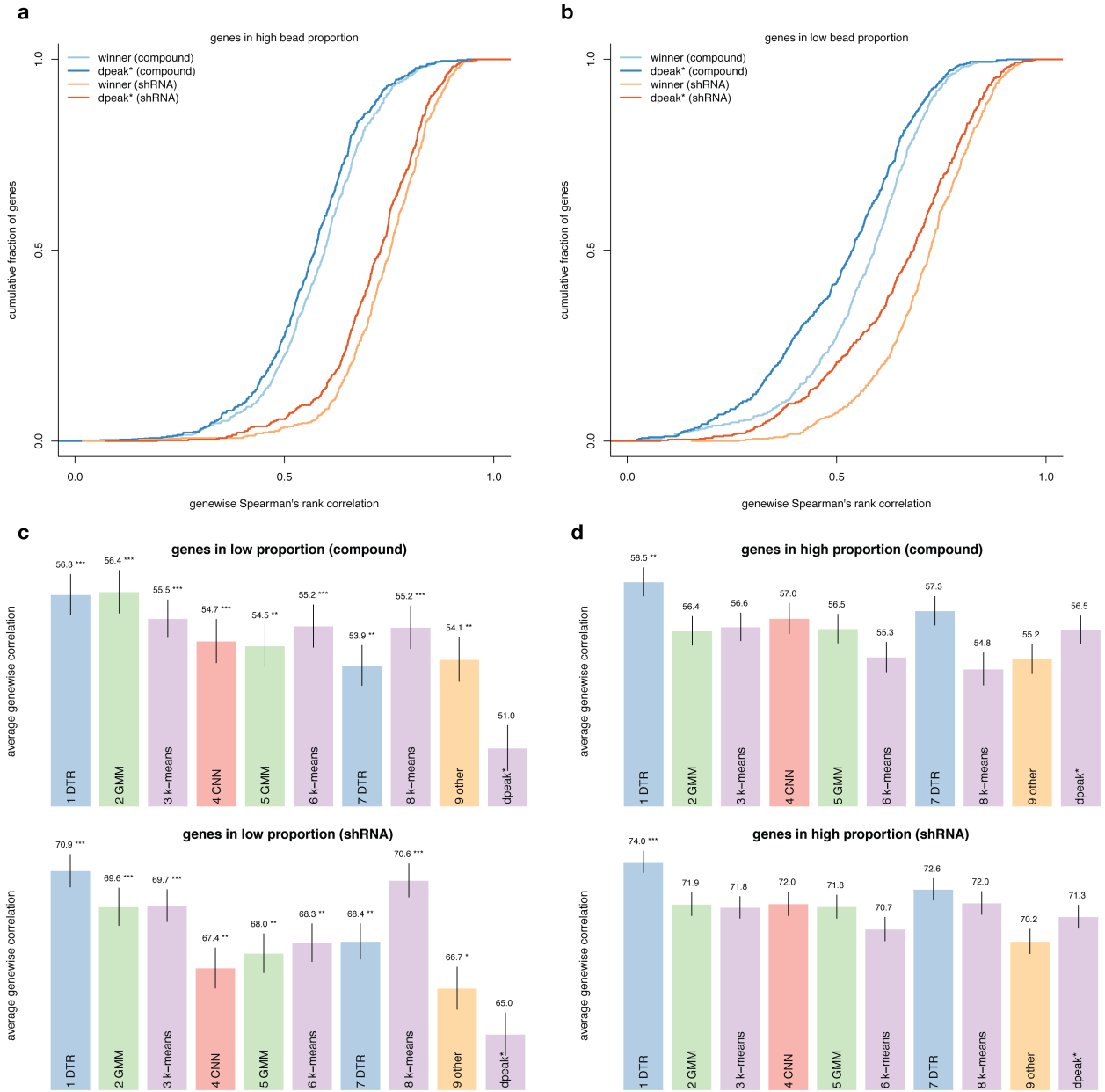


Figure 4

Detection of extreme modulations and targeted knockdown genes. This figure shows the AUC for the predicted differential expressions (DE data) obtained by the top-nine performing methods and the k-means benchmark (dpeak) and the corresponding ground-truth extreme modulations (as detected in the UNI data) both for the shRNA and compound samples (**a**). It also shows the computed recall and precision of the top-nine methods and the benchmark for the detection of the targeted knockdown genes for a subset of shRNA experiments (**b**).

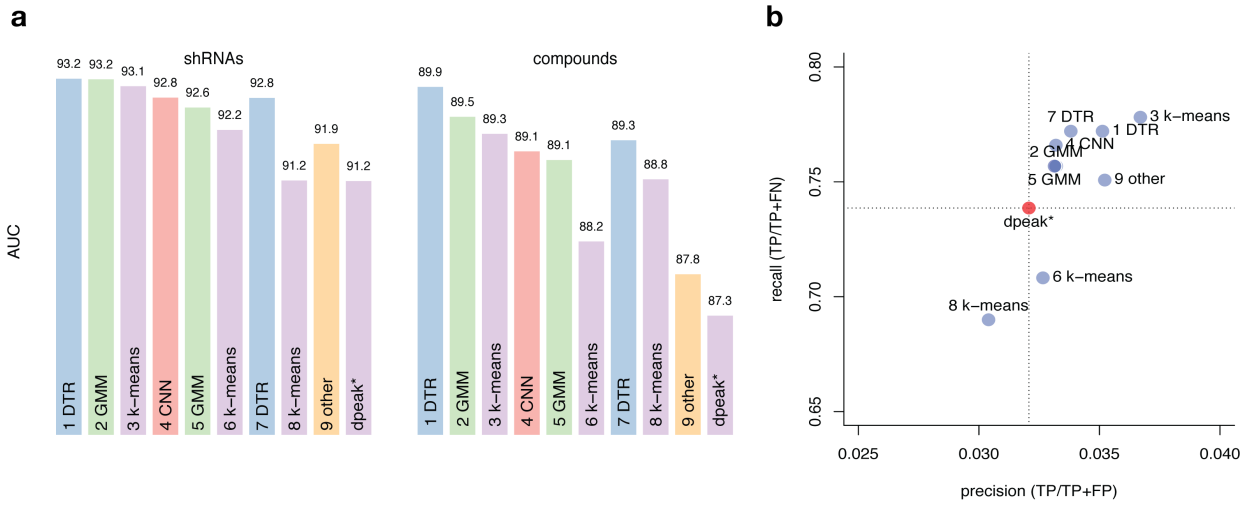


Figure 5

Variation across replicates for the top-four performing methods and the (dpeak) benchmark. This figure shows the mean coefficient of variation for each compound perturbagen in our sample. The mean coefficient of variation is computed as the average of the gene-wise percent ratio between the interquartile-range and the median of the deconvoluted values across 10 replicates. Results are stratified by the genes in low bead proportions (**a**) and those in high bead proportions (**b**). The perturbagens on the x-axis are ordered by increasing mean coefficient of variation of the benchmark.

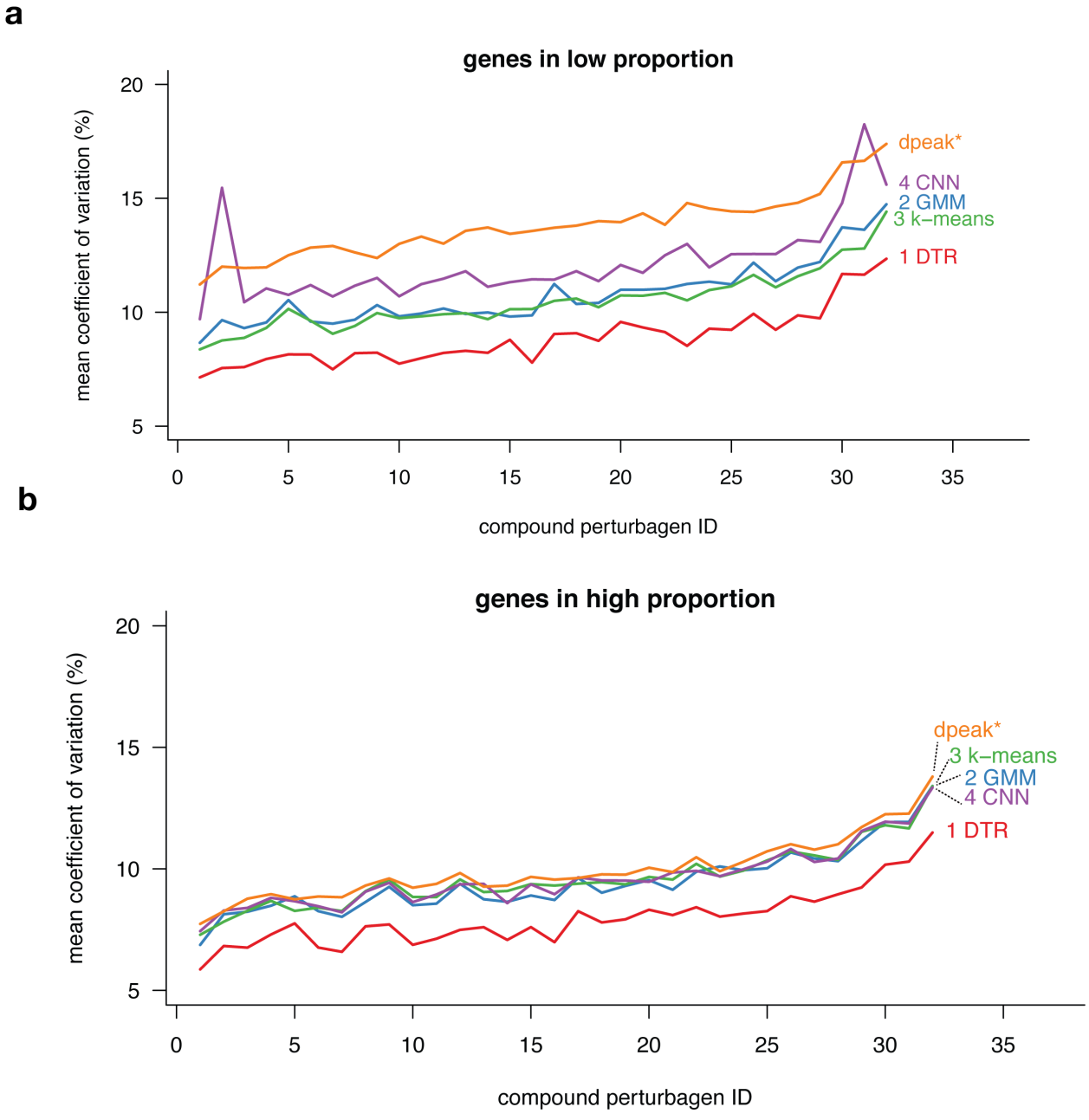
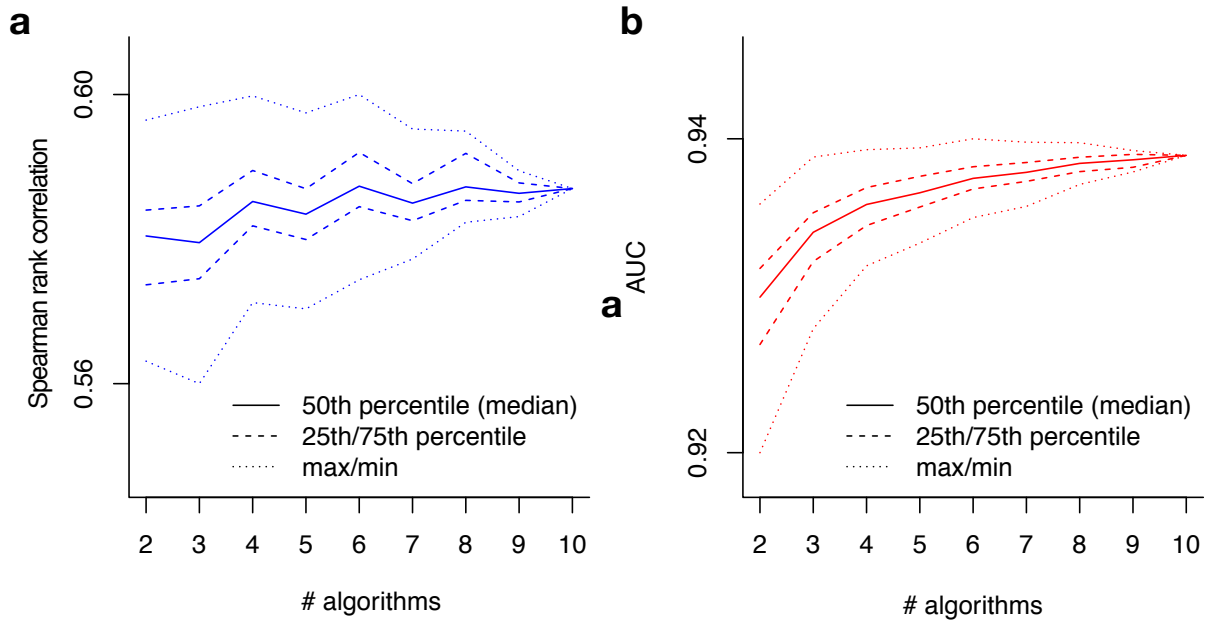


Figure 6

Performance of ensembles. This figure shows the performance in the (**a**) correlation metric and (**b**) AUC metric of the ensemble based on the median prediction of all possible combinations of a given size of the top 9 algorithms plus the benchmark. The median performance of the ensemble tends to increase with its size. However, the maximum performance in both metrics tends to plateau (or even decrease) after the ensamble reaches a size equal to 3.



Supporting information

S1 Table

Top-nine performing solutions. This table lists the top-nine solutions and the languages and algorithms each used, as well as the average speedup per plate relative to the k-means benchmark.

rank	handle	language	method	category	speedup
1	gardn999	Java	random forest regressor	DTR	17x
2	Ardavel	C++	Gaussian mixture model	GMM	62x
3	mkagenius	C++	modified k-means	k-means	24x
4	Ramzes2	Python/C++	ConvNet	CNN	10x
5	vladaburian	Python/C++	Gaussian mixture model	GMM	7x
6	balajipro	Python/C++	modified k-means	k-means	21x
7	AliGebily	Python	boosted tree regressor	DTR	5x
8	LastEmperor	Python	modified k-means	k-means	7x
9	mvaudel	Java	other	other	55x

S2 Table

Compound perturbagens descriptives. This table shows componud perturbagen names (`pert_iname`), unique id (`pert_id`), time of treatment (`pert_itime`), dose (`pert_idose`), and number of replicates (`num_replicates`).

pert_iname	pert_id	pert_itime	pert_idose	num_replicates
abiraterone(cb-7598)	BRD-K50071428	24 h	10 um	11
acalabrutinib	BRD-K64034691	24 h	10 um	11
afatinib	BRD-K66175015	24 h	10 um	11
artesunate	BRD-K54634444	24 h	10 um	11
azithromycin	BRD-K74501079	24 h	10 um	11
betamethasone dipropionate (diprolene)	BRD-K58148589	24 h	10 um	11
CGS-21680	BRD-A81866333	24 h	10 um	11
chelidonine	BRD-K32828673	24 h	10 um	11
clobetasol	BRD-K84443303	24 h	10 um	11
digoxin	BRD-A91712064	24 h	10 um	11
disulfiram	BRD-K32744045	24 h	10 um	10
emetine hcl	BRD-A77414132	24 h	10 um	10
eplerenone	BRD-K19761926	24 h	10 um	11
epothilone-a	BRD-K71823332	24 h	10 um	9
flumetasone	BRD-K61496577	24 h	10 um	11
fluocinolone	BRD-K94353609	24 h	10 um	11
genipin	BRD-K28824103	24 h	10 um	11
hydrocortisone	BRD-K93568044	24 h	10 um	10
hyoscyamine	BRD-K40530731	24 h	10 um	11
indirubin	BRD-K17894950	24 h	10 um	10
L-745870	BRD-K05528470	24 h	10 um	10
nTZDpa	BRD-K54708045	24 h	10 um	11
oligomycin-a	BRD-A81541225	24 h	10 um	11
PRIMA1	BRD-K15318909	24 h	10 um	11
RITA	BRD-K00317371	24 h	10 um	11
spironolactone	BRD-K90027355	24 h	10 um	11
tanespimycin	BRD-K81473043	24 h	10 um	11
tretinoin	BRD-K71879491	24 h	10 um	10
UB-165	BRD-A14574269	24 h	10 um	11
ursolic-acid	BRD-K68185022	24 h	10 um	11

pert_iname	pert_id	pert_itime	pert_idose	num_replicates
WAY-161503	BRD-A62021152	24 h	10 um	11
ZM-39923	BRD-K40624912	24 h	10 um	11

S3 Table

Short-hairpin (shRNA) perturbagens descriptives. This table shows shRNA perturbagen names (`pert_iname`), unique id (`pert_id`), and number of replicates (`num_replicates`).

pert_iname	pert_id	num_replicates
ABCB6	TRCN0000060320	4
ADI1	TRCN0000052275	4
ALDOA	TRCN0000052504	4
ANXA7	TRCN0000056304	4
ARHGAP1	TRCN0000307776	4
ASAH1	TRCN0000029402	4
ATMIN	TRCN0000141397	4
ATP2C1	TRCN0000043279	4
B3GNT1	TRCN0000035909	4
BAX	TRCN0000033471	4
BIRC5	TRCN0000073718	4
BLCAP	TRCN0000161355	4
BLVRA	TRCN0000046391	4
BNIP3L	TRCN0000007847	4
CALU	TRCN0000053792	4
CCDC85B	TRCN0000242754	4
CCND1	TRCN0000040038	4
CD97	TRCN0000008234	4
CHMP4A	TRCN0000150154	4
CNOT4	TRCN0000015216	4
DDR1	TRCN0000000618	4
DDX10	TRCN0000218747	4
DECR1	TRCN0000046516	4
DNM1L	TRCN0000001097	3
ECH1	TRCN0000052455	4
EIF4EBP1	TRCN0000040206	4
EMPTY_VECTOR	TRCN0000208001	15
ETFB	TRCN0000064432	4
FDFT1	TRCN0000036327	4
GALE	TRCN0000049461	4
GFP	TRCN0000072181	16

pert_iname	pert_id	num_replicates
GRN	TRCN0000115978	4
GTPBP8	TRCN0000343727	4
HDGFRP3	TRCN0000107348	4
HIST1H2BK	TRCN0000106710	4
IKBKAP	TRCN0000037871	4
INPP4B	TRCN0000230838	4
INSIG1	TRCN0000134159	4
ITFG1	TRCN0000343702	3
JMJD6	TRCN0000063340	4
LBR	TRCN0000060460	4
LGMN	TRCN0000029255	4
LPGAT1	TRCN0000116066	4
LSM6	TRCN0000074719	4
MAPKAPK2	TRCN0000002285	4
MAPKAPK3	TRCN0000006154	4
MAPKAPK5	TRCN0000000684	4
MIF	TRCN0000056818	4
MRPL12	TRCN0000072655	4
NT5DC2	TRCN0000350758	4
NUP88	TRCN0000145079	4
PARP2	TRCN0000007933	4
PLCB3	TRCN0000000431	4
POLE2	TRCN0000233181	4
PPIE	TRCN0000049371	4
PRKAG2	TRCN0000003146	4
PSMB10	TRCN0000010833	4
PTPN6	TRCN0000011052	4
RAB11FIP2	TRCN0000322640	4
RALB	TRCN0000072956	4
RHEB	TRCN0000010425	3
RNF167	TRCN0000004100	4
RPN1	TRCN0000072588	4
SLC25A4	TRCN0000044967	4
SNX11	TRCN0000127684	4

pert_iname	pert_id	num_replicates
STK25	TRCN0000006270	4
STUB1	TRCN0000007525	4
STXBP1	TRCN0000147480	4
SYPL1	TRCN0000059926	4
TATDN2	TRCN0000049828	4
TM9SF3	TRCN0000059371	4
TMEM110	TRCN0000127960	4
TMEM50A	TRCN0000129223	4
trcn0000014632	TRCN0000014632	4
trcn0000040123	TRCN0000040123	4
trcn0000220641	TRCN0000220641	4
trcn0000221408	TRCN0000221408	4
trcn0000221644	TRCN0000221644	4
TSKU	TRCN0000005222	4
UGDH	TRCN0000028108	4
USP14	TRCN0000007428	4
USP6NL	TRCN0000253832	4
VAT1	TRCN0000038193	4
VDAC1	TRCN0000029126	4
WIPF2	TRCN0000029833	4
YME1L1	TRCN0000073864	4
ZW10	TRCN0000155335	4

S1 Data

LINK TO DATA REPOSITORY on http://CLUE.io WILL BECOME AVAILABLE AFTER PUBLICATION

S1 Appendix

Scoring function. This appendix describes the scoring function used in the contest to evaluate the performance of the competitors' submissions.

Submissions were scored based on a scoring function that combines measures of accuracy and computational speed. Accuracy measures were obtained by comparing the contestant's predictions, which were derived from DUO data, to the equivalent UNI ground truth data generated from the same samples.

The scoring function combines two measures of accuracy: correlation and AUC, which are applied to deconvoluted ($DECONV$) data and one to differential expression (DE) data, respectively.

DE is derived from DECONV by applying a series of transformations (parametric scaling, quantile normalization, and robust z-scoring) that are described in detail in Subramanian et al. [1]. The motivation for scoring DE data in addition to $DECONV$ is because it is at this level where the most bilogically interesting gene expression changes are observed. Of particular interest is obtaining significant improvement in the detection of, so called, "extreme modulations." These are genes that notably up- or down-regulated by pertubation and hence exhibit an exceedingly high (or low) DE values relative to a fixed threshold.

The first accuracy component is based on the Spearman rank correlation between the predicted $DECONV$ data and the corresponding UNI ground truth data.

For a given dataset p, let $M_{\mathrm{DUO},p}$ and $M_{\mathrm{UNI},p}$ denote the matrices of the estimated gene intensities for $G = 976$ genes (rows) and $S = 384$ experiments (columns) under DUO and UNI detection. Compute the Spearman rank correlation matrix, ρ, between the rows of these matrices and take the median of the diagonal elements of the resulting matrix (i.e., the values corresponding to the matched experiments between UNI and DUO) to compute the median correlation per dataset,

$$\mathrm{COR}_p = median(diag(\rho(M_{\mathrm{DUO},p}, M_{\mathrm{UNI},p}))).$$

The second component of the scoring function is based on the Area Under the receiver operating characteristic Curve (AUC) that uses the competitor's DE values at various thresholds to predict the UNI's DE values being higher than 2 ("high") or lower than -2 ("low").

For a given dataset p, let $\mathrm{AUC}_{p,c}$ denote the corresponding area under the curve where $c = \{\mathrm{high}, \mathrm{low}\}$; then, compute the arithmetic mean of the area under the curve per class to obtain the corresponding score per dataset:

$$\mathrm{AUC}_p = (\mathrm{AUC}_{p,\mathrm{high}} + \mathrm{AUC}_{p,\mathrm{high}})/2.$$

These accuracy components were integrated into a single aggregate scores:

$$\mathrm{SCORE} = \mathrm{SCORE}_{\max} \cdot (\max(\mathrm{COR}_p, 0))^2 \cdot \mathrm{AUC}_p \cdot \exp(-T_{\mathrm{solution}}/(3 \cdot T_{\mathrm{benchmark}})),$$

where T_{solution} is the run time for deconvoluting the data in each plate, and $T_{\mathrm{benchmark}}$ is the deconvolution time required by the benchmark dpeak implementation.

S2 Appendix

L1000 Experimental Scheme The L1000 assay uses Luminex bead-based fluorescent scanners to detect gene expression changes resulting from treating cultured human cells with chemical or genetic perturbations [Subramanian 2017]. Experiments are performed in 384-well plate format, where each well contains an independent sample. The Luminex scanner is able to distinguish between 500 different bead types, or colors, which CMap uses to measure the expression levels of 978 landmark genes using two detection approaches.

In the first detection mode, called UNI, each of the 978 landmark genes is measured individually on one of the 500 Luminex bead colors. In order to capture all 978 genes, two detection plates are used, each measuring 489 landmarks. The two detection plates' worth of data are then computationally combined to reconstruct the full 978-gene expression profile for each sample.

By contrast, in the *DUO* detection scheme two genes are measured using the same bead color. Each bead color produces an intensity histogram which characterizes the expression of the two distinct genes. In the ideal case, each histogram consists of two peaks each corresponding to a single gene. The genes are mixed in 2:1 ratio, thus the areas under the peaks have 2:1 ratio (see Figure 1), which enables the association of each peak with the specific gene. The practical advantage of the DUO detection mode is that it uses half of the laboratory reagents as UNI mode, and hence *DUO* is and has been the main detection mode used by CMap. After *DUO* detection, the expression values of the two genes are computationally extracted in a process called 'peak deconvolution,' described in the next section.

References

[1] Aravind Subramanian et al. "A next generation connectivity map: L1000 platform and the first 1,000,000 profiles". In: *Cell* 171.6 (2017), pp. 1437–1452.

[2] Brian Cleary et al. "Efficient generation of transcriptomic profiles by random composite measurements". In: *Cell* 171.6 (2017), pp. 1424–1436.

[3] Shai S Shen-Orr et al. "Cell type–specific gene expression differences in complex tissues". In: *Nature methods* 7.4 (2010), p. 287.

[4] Yi Zhong and Zhandong Liu. "Gene expression deconvolution in linear space". In: *Nature methods* 9.1 (2012), p. 8.

[5] Aaron M Newman et al. "Robust enumeration of cell subsets from tissue expression profiles". In: *Nature methods* 12.5 (2015), p. 453.

[6] Konstantin Zaitsev et al. "Complete deconvolution of cellular mixtures based on linearity of transcriptional signatures". In: *Nature communications* 10.1 (2019), p. 2209.

[7] Yue Deng et al. "Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning". In: *Nature methods* 16.4 (2019), p. 311.

[8] Peng Lu, Aleksey Nakorchevskiy, and Edward M Marcotte. "Expression deconvolution: a reinterpretation of DNA microarray data reveals dynamic changes in cell populations". In: *Proceedings of the National Academy of Sciences* 100.18 (2003), pp. 10370–10375.

[9] Stephan Preibisch et al. "Efficient Bayesian-based multiview deconvolution". In: *Nature methods* 11.6 (2014), p. 645.

[10] Shai S Shen-Orr and Renaud Gaujoux. "Computational deconvolution: extracting cell type-specific information from heterogeneous samples". In: *Current opinion in immunology* 25.5 (2013), pp. 571–578.

[11] Karim R Lakhani et al. "Prize-based contests can provide solutions to computational biology problems". In: *Nature biotechnology* 31.2 (2013), p. 108.

[12] Andrea Blasco et al. "Advancing computational biology and bioinformatics research through open innovation competitions". In: *PLOS ONE* 14.9 (Sept. 2019), pp. 1–17. DOI: 10.1371/journal.pone.0222165. URL: https://doi.org/10.1371/journal.pone.0222165.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[14] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.