Smith *et al. BMC Bioinformatics* (2020) 21:119
https://doi.org/10.1186/s12859-020-3427-8

BMC Bioinformatics

RESEARCH ARTICLE Open Access

Standard machine learning approaches outperform deep representation learning on phenotype prediction from transcriptomics data

Aaron M. Smith[1*], Jonathan R. Walsh[1], John Long[2], Craig B. Davis[3], Peter Henstock[4], Martin R. Hodge[5], Mateusz Maciejewski[5], Xinmeng Jasmine Mu[6], Stephen Ra[2], Shanrong Zhao[2], Daniel Ziemek[7] and Charles K. Fisher[1]

Abstract

Background: The ability to confidently predict health outcomes from gene expression would catalyze a revolution in molecular diagnostics. Yet, the goal of developing actionable, robust, and reproducible predictive signatures of phenotypes such as clinical outcome has not been attained in almost any disease area. Here, we report a comprehensive analysis spanning prediction tasks from ulcerative colitis, atopic dermatitis, diabetes, to many cancer subtypes for a total of 24 binary and multiclass prediction problems and 26 survival analysis tasks. We systematically investigate the influence of gene subsets, normalization methods and prediction algorithms. Crucially, we also explore the novel use of deep representation learning methods on large transcriptomics compendia, such as GTEx and TCGA, to boost the performance of state-of-the-art methods. The resources and findings in this work should serve as both an up-to-date reference on attainable performance, and as a benchmarking resource for further research.

Results: Approaches that combine large numbers of genes outperformed single gene methods consistently and with a significant margin, but neither unsupervised nor semi-supervised representation learning techniques yielded consistent improvements in out-of-sample performance across datasets. Our findings suggest that using l_2-regularized regression methods applied to centered log-ratio transformed transcript abundances provide the best predictive analyses overall.

Conclusions: Transcriptomics-based phenotype prediction benefits from proper normalization techniques and state-of-the-art regularized regression approaches. In our view, breakthrough performance is likely contingent on factors which are independent of normalization and general modeling techniques; these factors might include reduction of systematic errors in sequencing data, incorporation of other data types such as single-cell sequencing and proteomics, and improved use of prior knowledge.

Keywords: Transcriptomics, RNA-seq, Normalization methods, Machine learning, Deep learning, Representation learning, Phenotype prediction, Molecular diagnostics

*Correspondence: drams@unlearn.ai
[1]Unlearn.AI, Inc., San Francisco, CA, USA
Full list of author information is available at the end of the article

Background

The potential to tailor therapies for individual patients rests on the ability to accurately diagnose disease and predict outcomes under various treatment conditions. Predictors based on high-throughput 'omics technologies hold great promise, but a number of technical challenges have limited their applicability [1]. Phenotypes may be complex—involving contributions from large numbers of genes—but 'omics data are so high-dimensional that exploring all possible interactions is intractable. This situation is further complicated by the small sample sizes of typical biological studies and by large systematic sources of variation between experiments [2, 3]. However, recent developments in machine learning have raised hopes that new computational methods integrating data from many studies may be able to overcome these difficulties. Accurate prediction of phenotype or endpoint(s) from 'omics data would usher in an era of molecular diagnostics [4, 5].

Machine learning methods often benefit from large datasets where learning complex relationships is feasible. Although individual biological experiments tend to be small, relatively large amounts of 'omics data are available in public repositories. For example, hundreds of thousands of samples from human RNA sequencing (RNA-seq) experiments are available from the recount2 and ARCHS4 databases [6–8]. Still, these data cover a wide variety of tissues and diseases. Moreover, there are no specific diseases with large numbers of samples and, in many cases, the metadata are not sufficient to determine basic experimental facts like the tissue of origin [9]. As a result, leveraging these data to improve prediction tasks will require machine learning techniques that can learn from large, heterogeneous datasets.

Genes rarely act in isolation, so it is reasonable to expect that combinations of genes may be more effective than individual genes for predicting phenotypes. For example, linear models operating on RNA-seq data create predictors from a weighted combination of gene expression values. However, some of these features could reflect biological processes that are involved in multiple phenotypes. Many previous analyses have explored this possibility by creating complex features that incorporate biological knowledge from gene sets [10, 11], ontologies [12], or interaction graphs [13–15]. More recently, unsupervised machine learning methods incorporating principal components analysis [16], autoencoders [17–20], or other neural network architectures have been developed to discover such features by analyzing large transcriptomics datasets. These attempts are examples of a general program called representation learning in which the purpose of training such unsupervised models is to extract a complex feature embedding from the model [21]. In this setting representation learning holds great promise which is furthermore straightforwardly testable: *If these learned features capture biologically relevant processes, then predictive models built from those features should outperform models built directly from relative transcript abundances.*

In this work, we present a comprehensive analysis of phenotype prediction from transcriptomics data with a particular emphasis on representation learning. Using the recount2 database [7], we systematically explored the impact of normalization techniques, gene sets, learned representations, and machine learning methods on predictive performance for a set of 24 binary and multiclass prediction problems and 26 survival analysis tasks. In total, we analyzed thousands of predictive models using 5-fold nested cross validation to rigorously assess out-of-sample performance. We found that predictors that combined multiple genes outperformed single gene predictors, that logarithmic transformations outperformed untransformed relative expression measurements, and that for survival analyses larger gene sets outperformed smaller gene sets. However, neither unsupervised nor semi-supervised representation learning techniques yielded consistent improvement on out-of-sample predictive performance across datasets. In fact, l_2-regularized regression methods applied directly to the centered log-ratio transform of transcript abundances performed consistently well relative to the other methods. Therefore we recommend treating that particular combination as a baseline method for predictive analysis on RNAseq data. Throughout this text we refer to the combination of l_2-regularized regression methods applied directly to the centered log-ratio transform of transcript abundances as *the recommended model*.

Results

Approach

A high-level description of our quality control, data processing, and machine learning analyses is provided in Fig. 1. Details of the dataset and these steps are provided in the "Methods" section.

Briefly, our dataset is sourced from the recount2 database [7], and contains expression data from Genotype-Tissue Expression (GTEx) project [22], The Cancer Genome Atlas (TCGA) Pan-Cancer Clinical Data Resource [23], and The Sequence Read Archive (SRA). We selected a subset of experiments from recount2 that did not have sparse gene expression data and could be mapped to the same set of tissues covered in GTEx. We assigned the various experiments to "training" (∼37k samples), "validation" (∼4k samples), and "test" sets (∼4k samples). All samples lacking suitable metadata for supervised learning were allocated to the training set. From metadata provided with recount2, the Gene Expression Omnibus [24], and TCGA Pan-Cancer Clinical Data Resource we derived labels for 24 binary and multiclass and 26 survival analysis tasks. Descriptions of these tasks and their assign-

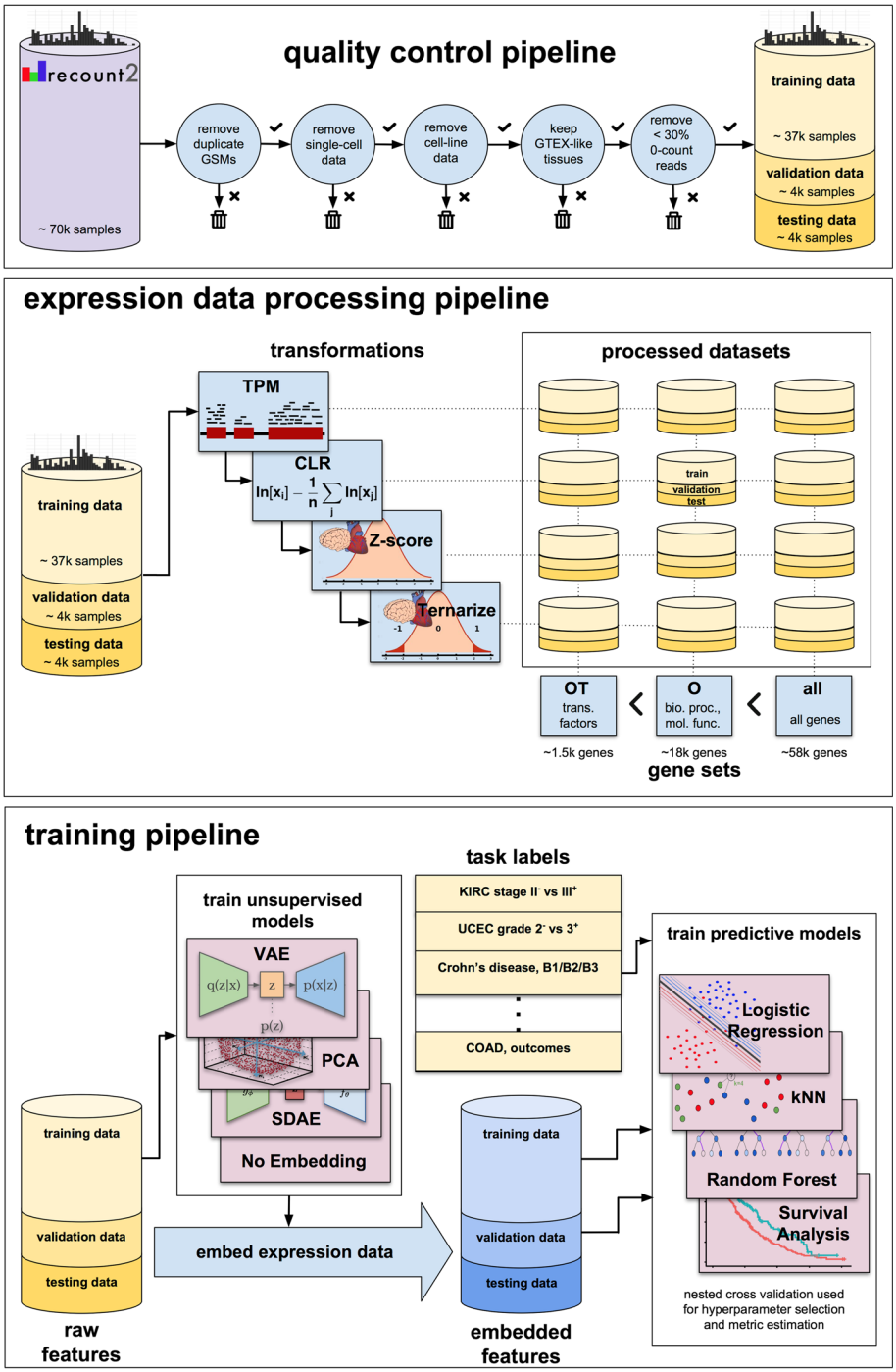

Fig. 1 Schematic. An overview of the quality control, data processing, and training pipelines. Data from recount undergoes several quality checks at the sample and study level, resulting in a dataset of approximately 45,000 samples divided into training, testing, and validation datasets. Twelve different datasets are created from these data, each with a different gene set (**all**, comprising all genes; **O**, comprising key GO categories; **OT**, comprising **O** genes that are known transcription factors) and transform ("TPM", transcripts per million; "CLR", a centered-log-ratio transform of TPM; "Z-score", a Z-score normalization of the CLR data relative to healthy tissue expression levels in GTEx; "Z-ternary", a ternarization of Z-score). The training data is used to train unsupervised models capable of embedding the data (a "no embedding" model is also included, which does not alter the data). These embedded features, along with labels for individual tasks, are used to train a variety of supervised models. The supervised models are trained and evaluated using nested cross validation

ment to the training, validation, and test sets are provided in the "Methods" section (see Tables 1, 2, 3, and 4).

We considered four different normalization methods to correct for variance introduced in the data collection and measurement process. We first converted the samples from counts to Transcripts Per Million (TPM) [25], a normalization which estimates relative molar concentration of transcripts in a sample. Under the operating assumption that relative transcript abundance is determinant of downstream biological function, TPMs should be the baseline quantification to work with from RNAseq. In contrast, raw counts or counts per million (CPMs) contain irrelevant counting bias stemming from variable transcript length. Likewise, the common alternative of fragments per kilobase per million (FPKMs) do not coherently measure relative molar concentration, because they rely on a sample-dependent normalization factor. As such, FPKMs are not a useful measure when processing samples which are not entirely technical replicates of a single tissue sample [25, 26]. Secondly we applied the centered log-ratio transformation (CLR) [27] to the TPM data to address the fact that RNA-seq data quantify relative, rather than absolute, gene expression [28, 29]. The CLR transform has two useful features. First of all, it normalizes compositional data so that samples representing the same relative abundances are equated. Secondly it converts multiplicative relationships between the relative abundances into linear relationships – a feature which allows statistical models to represent fold-differences between expression vectors, and to model the error properly. Since these two normalization methods do not account for the tissue of origin of the sample, we evaluated additional normalization methods based on differential expression with respect to normal tissue. The third normalization method converted the CLR transformed expression data from each sample to a tissue-normalized Z-score by subtracting the mean and dividing by the standard deviation of the associated tissue in GTEx. This mean and standard deviation of the CLR transformed expression data were computed across the GTEx data in recount2 for each annotated tissue type. Therefore the tissue-normalized Z-score expression data measures deviations from normal tissue of each type. As a result, characteristic deviations from normal expression will have the same features after this transformation, even across different tissue types. Finally, a fourth ternarized normalization discretized the Z-scores into down-regulated ($Z < -2$), normal ($-2 < Z < 2$), or up-regulated ($Z > 2$) categories.

For each of these normalization approaches, we also explored three gene sets corresponding to transcription factors [30] (denoted **OT**), protein coding genes annotated as with biological processes or molecular functions in the Gene Ontology 12 (denoted **O**), and all genes provided by recount2 (denoted **all**). The **O** and **OT** gene sets are substantially smaller than the **all** gene set and allow exploration of the dependence on the number of genes. In total, we examined twelve different normalization-gene set combinations for each predictive problem.

Table 1 TCGA binary tasks

Project	Disease	Label	Label type	Group	Samples
TCGA stage tasks					
COAD	colon adenocarcinoma	II- vs. III+	binary	train	505
KIRC	kidney renal clear cell carcinoma	II- vs. III+	binary	train	544
LIHC	liver hepatocellular carcinoma	I- vs. II+	binary	train	374
LUAD	lung adenocarcinoma	I- vs. II+	binary	train	542
SKCM	skin cutaneous melanoma	II- vs. III+	binary	train	249
STAD	stomach adenocarcinoma	II- vs. III+	binary	train	416
THCA	thyroid cancer	I- vs. II+	binary	train	513
UCEC	uterine corpus endometrial carcinoma	I- vs. II+	binary	train	554
LUSC	lung squamous cell carcinoma	I- vs. II+	binary	validate	504
BRCA	breast invasive carcinoma	II- vs. III+	binary	test	1134
TCGA grade tasks					
CESC	cervical squamous cell carcinoma	II- vs. III+	binary	train	306
KIRC	kidney renal clear cell carcinoma	II- vs. III+	binary	train	544
LGG	low grade glioma	II- vs. III+	binary	train	532
LIHC	liver hepatocellular carcinoma	II- vs. III+	binary	train	374
PAAD	pancreatic adenocarcinoma	II- vs. III+	binary	train	179
STAD	stomach adenocarcinoma	II- vs. III+	binary	train	416
UCEC	uterine corpus endometrial carcinoma	II- vs. III+	binary	train	554
HNSC	head-neck squamous cell carcinoma	II- vs. III+	binary	test	504

The 18 binary tasks derived from TCGA used to train supervised models and validate the unsupervised embeddings. The tasks are grouped into two categories, TCGA tumor stage tasks (10), and TCGA tumor grade tasks (8). The project names correspond to those in Fig. 2

Table 2 SRA tasks

SRA tasks					
Project	Disease	Label	Label type	Group	Samples
GSE65832	atopic dermatitis	lesional vs. not	binary	train	40
GSE66207	Crohn's disease	type: B1, B2 or B3	multiclass (3)	train	20
GSE72819	ulcerative colitis	treatment remission	binary	validate	69
GSE47944	psoriasis	lesional vs. not	multiclass (3)	validate	63
GSE50244	diabetes	normoglycemic, impaired, diabetic	multiclass (3)	validate	76
GSE67785	psoriasis	lesional vs. not	binary	test	28

The 8 tasks derived from SRA used to train supervised models and validate the unsupervised embeddings. The project names correspond to those in Fig. 2

Table 3 TCGA overall survival (OS) tasks

TCGA OS tasks					
Project	Disease	Label	Label type	Group	Samples
CESC	cervical squamous cell carcinoma	OS	survival	train	304
COAD	colon adenocarcinoma	OS	survival	train	455
ESCA	esophageal carcinoma	OS	survival	train	184
KIRP	kidney papillary cell carcinoma	OS	survival	train	289
LUAD	lung adenocarcinoma	OS	survival	train	507
OV	ovarian cancer	OS	survival	train	420
PAAD	pancreatic adenocarcinoma	OS	survival	train	178
SARC	sarcoma	OS	survival	train	259
STAD	stomach adenocarcinoma	OS	survival	train	409
UCEC	uterine corpus endometrial carcinoma	OS	survival	train	540
HNSC	head-neck squamous cell carcinoma	OS	survival	validate	501
BLCA	bladder urothelial carcinoma	OS	survival	test	407
LUSC	lung squamous cell carcinoma	OS	survival	test	495

The 13 overall survival tasks derived from TCGA used to train supervised models and validate the unsupervised embeddings. The project names correspond to those in Fig. 2

Table 4 TCGA progression-free interval (PFI) tasks

TCGA PFI tasks					
Project	Disease	Label	Label type	Group	Samples
CESC	cervical squamous cell carcinoma	PFI	survival	train	304
COAD	colon adenocarcinoma	PFI	survival	train	455
ESCA	esophageal carcinoma	PFI	survival	train	184
KIRP	kidney papillary cell carcinoma	PFI	survival	train	288
LUAD	lung adenocarcinoma	PFI	survival	train	507
OV	ovarian cancer	PFI	survival	train	420
PAAD	pancreatic adenocarcinoma	PFI	survival	train	178
SARC	sarcoma	PFI	survival	train	259
STAD	stomach adenocarcinoma	PFI	survival	train	411
UCEC	uterine corpus endometrial carcinoma	PFI	survival	train	540
HNSC	head-neck squamous cell carcinoma	PFI	survival	validate	501
BLCA	bladder urothelial carcinoma	PFI	survival	test	408
LUSC	lung squamous cell carcinoma	PFI	survival	test	496

The 13 progression-free surivival tasks derived from TCGA used to train supervised models and validate the unsupervised embeddings. The project names correspond to those in Fig. 2

We considered four different types of representations of the gene expression data learned by unsupervised models. First, supervised models were trained directly on the normalized expression data without a learned embedding. We also considered representations constructed with Principal Components Analysis (PCA), a Stacked Denoising Autoencoder (SDAE), and a Variational Autoencoder (VAE) trained on the 37k samples in the training set without any supervising information.

For each binary or multiclass prediction task, we trained a k-Nearest Neighbor (kNN) classifier, a Random Forest (RF), and an l_2-regularized multinomial Logistic Regression (LR) on the normalized and transformed data using 5-fold nested cross validation. Using nested cross validation ("Methods" section) is important because it accounts for performance variance that results from different hyperparameter choices (e.g., the number of nearest neighbors, the depth of the trees in the forest, or the strength of the regularization coefficient). An l_2-regularized Cox proportional hazards model was used for all survival tasks, also with 5-fold nested cross validation. Binary tasks were compared using the Area Under the receiver operating characteristic Curve (AUC); multiclass tasks were compared using the accuracy, and survival tasks were compared using the concordance-index (C-index) [31, 32].

Our systematic model search covered four normalization methods, three gene sets, four representations, and three supervised algorithms totaling 144 comparison models for each of the 24 binary and multiclass tasks. For the survival tasks we used the same normalization methods, gene sets, and representations, but considered only one supervised algorithm (Cox proportional hazards). For comparison, we also trained linear predictors using the recommended method that were only allowed to use a single gene. The choice of gene was treated as a hyperparameter and optimized using 5-fold nested cross validation.

Analyses

The predictive performance assessed through 5-fold nested cross validation varied considerably across and within the predictive problems (see Fig. 2). Gene expression data improved predictive performance relative to

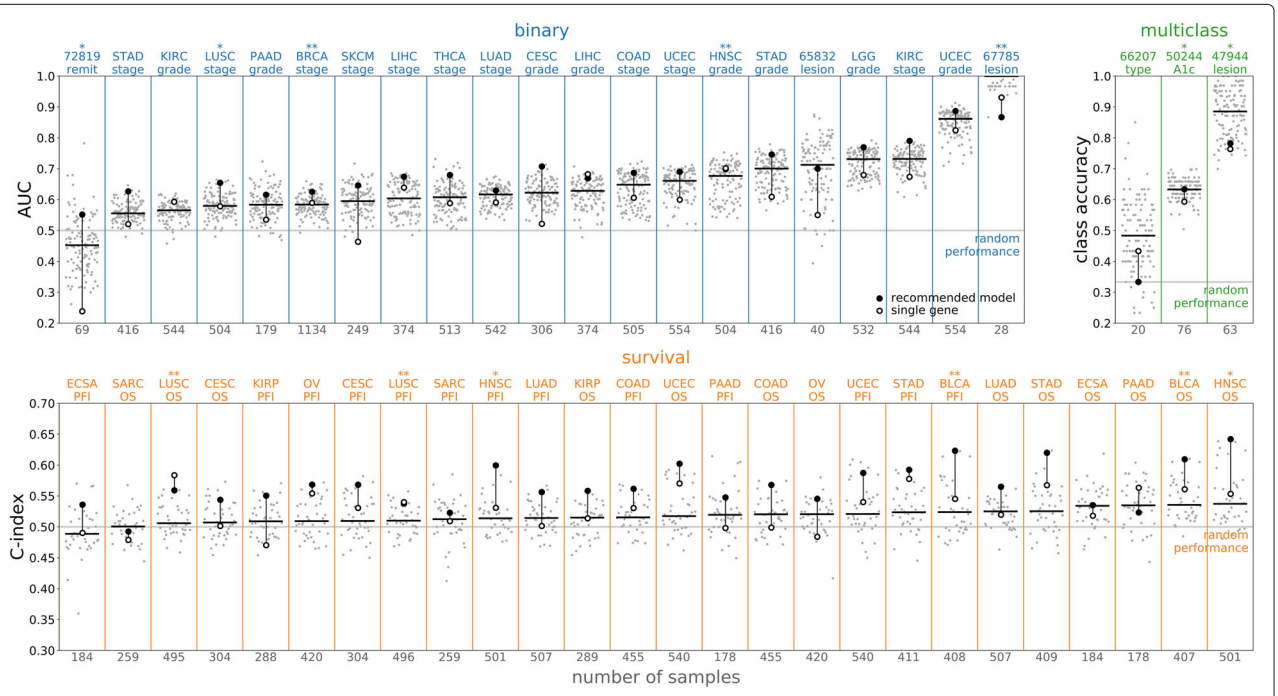

Fig. 2 Performance by predictive task. The performance of all models on each task, ordered by the median performance on each task. The tasks are divided into three groups based on the type of label; the top row shows classification tasks (binary and multiclass) while the bottom shows survival tasks. Each task is labeled by an abbreviation at the top of the plot and the number of samples at the bottom; see the Supplementary material for more details on each task. The task label has one star if the data is in the validation group and two stars if the data is in the test group. For each task, the gray points show the results over the entire set of models and the horizontal line shows their median. The filled black circle shows the performance of the recommended model, while the open black circle shows the performance of the best single gene model. The recommended model uses no embedding, **all** genes, and the CLR transform; the supervised model is logistic regression for the classifier tasks and a Cox proportional hazards model for the survival tasks. The recommended model is often among the best models on a problem and frequently outperforms the best single gene model; the primary exception is the pancreatic adenocarcinoma overall survival (PAAD OS) dataset

random guessing in almost all cases, indicating that RNA-seq data do contain information that is broadly useful for out-of-sample prediction. Moreover, linear predictors that used the expression data from all genes generally outperformed models that only used a single, most predictive gene. It is still common to analyze genes independently in differential expression and regression analyses; our results indicate, however, that linear combinations of genes are significantly more predictive than individual genes. Although there was sizable variance in performance across tasks, predictive performance was not correlated with any obvious dataset characteristics such as the number of subjects.

In order to compare the effects of the gene set size and transformation, it is helpful remove between-task variance and then to aggregate results across tasks. To remove the between-task variance, we defined a shifted statistic in which we subtracted the median value of all models on the same task. For example, the AUC for the random forest classifier on the STAD stage dataset was shifted by subtracting the median AUC for all of the binary classifiers trained on the STAD stage dataset. Averages of the shifted statistics across predictive problems can be easily interpreted: if the value is less than zero then the method underperformed the median, whereas the method outperformed the median if the value is greater than zero.

Within-task variance in predictive performance was partially explained by the choice of gene set and normalization method (see Fig. 3). Because the number of samples in each dataset was much smaller than the number of genes annotated in recount2, we hypothesized that using prior knowledge to select small, biologically relevant gene sets based on the Gene Ontology or transcription factor activity would improve out-of-sample predictive performance by preventing overfitting. However, this hypothesis was not supported by our analyses. The choice of gene set made no difference for the classification problems, whereas the smaller gene sets underperformed on the survival tasks. The log-transformed normalization methods slightly outperformed TPMs, and the Z-score normalization performed the best, on average. Performance improvements of Z-score normalization relative to CLR were small, however, and we do not think that the small gains justify the additional complexity introduced by referencing each sample to an external dataset (i.e., GTEx).

Next, we examined differences in absolute performance between the kNN, RF, and LR models on the classification problems (only a linear Cox proportional hazards model was tested on the survival tasks). As shown in Fig. 4, the kNN classifier consistently underperformed the RF and LR classifiers. The RF was the best performing method for thirteen tasks, LR for nine tasks, and kNN for two tasks, but LR was more consistent than RF and had better average performance.

Gene expression data are very high dimensional, with the number of genes ranging from ∼1.5k in the transcription factor gene set to ∼56k in the gene set consisting of all genes annotated in recount2. In contrast, the supervised task datasets typically consisted of only a few hundred samples. Moreover, it seems unlikely that genes actually coordinate in a linear fashion to generate complex phenotypes. Therefore we hypothesized that predictive performance could be improved by training predictors on lower dimensional representations derived from unsupervised analyses of the ∼37k unlabeled samples in the training set. One could also view these analyses as a type of transfer learning, in which biological knowledge derived from the analysis of one dataset is used to inform the analyses of another.

The first feature representation that we considered was a Principal Components Analysis (PCA) with 512 latent dimensions. These principal components are orthogonal linear combinations of expression values that represent the directions of largest variance in the training set. Together, the 512 principal components we used explained the majority of the variation in the transcriptional datasets (see Supplementary figures). We found that using PCA derived representations as features decreased the out-of-sample performance of downstream predictive analyses (Fig. 5). Therefore, we do not recommend using features derived from PCA of large RNA-seq compendia for predictive analyses.

Training a linear model on top of representations derived from a linear transformation like PCA is equivalent to a regularized linear model trained on the unembedded data. Deep neural network-based architectures like SDAEs and VAEs, by contrast, process an input expression vector through a series of nonlinear transformations to learn more complex features. Therefore, we also trained a 512-dimensional SDAE and VAE on the training set for each gene set-normalization combination and used the representations derived from these neural networks as features for downstream prediction tasks. Nevertheless, we found that preprocessing the expression data using these networks decreased the out-of-sample performance of downstream prediction tasks relative to just using the normalized expression data directly (Fig. 5).

Semi-supervised representation learning

There are a variety of reasons that unsupervised representation learning can fail to discover features that are useful for downstream predictive tasks. For example, a small but consistent difference in the expression of a gene between two groups (e.g., healthy and diseased) can be used to train a highly accurate predictor. However, if this difference is much smaller than the variance in the expres-

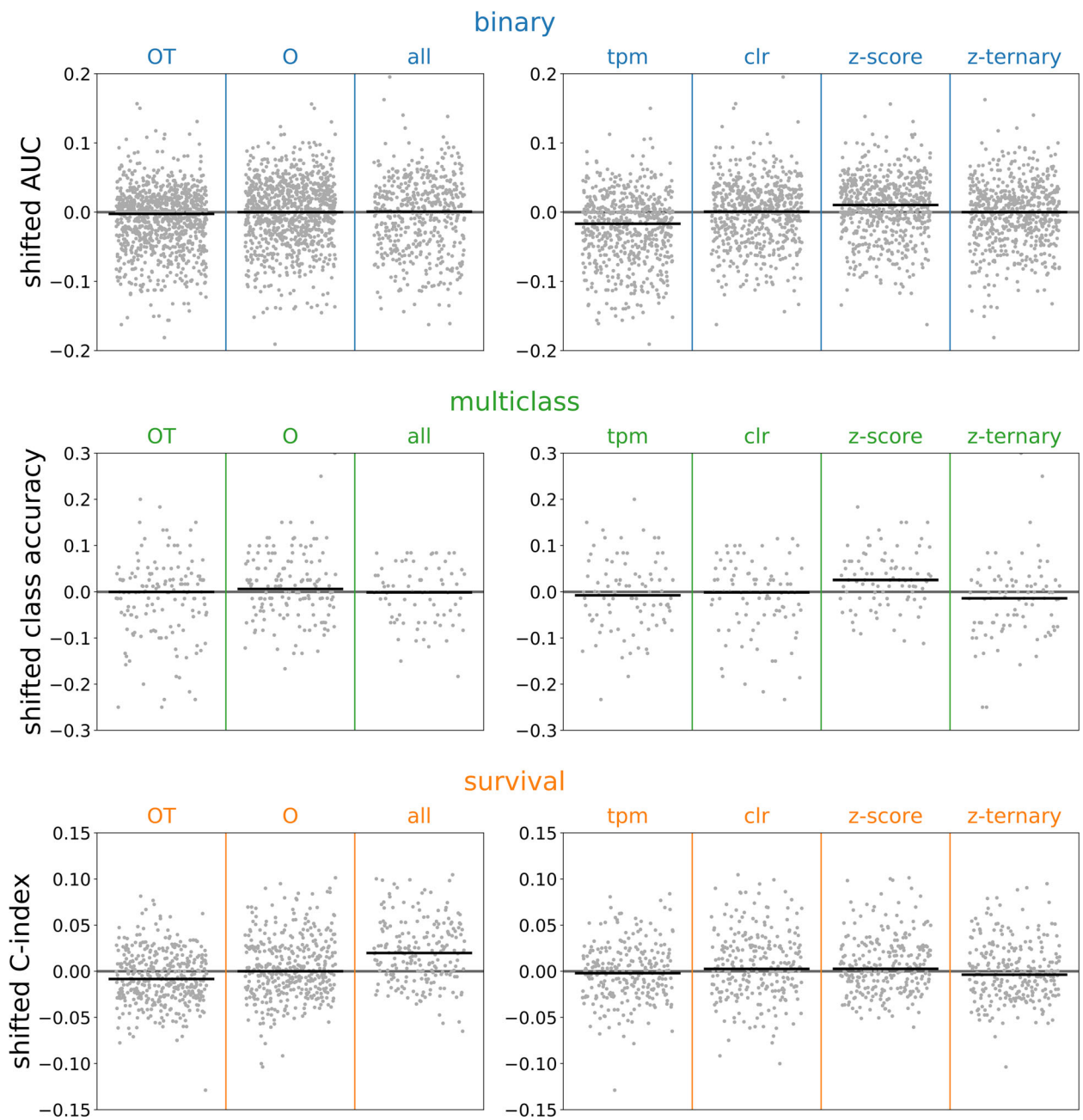

Fig. 3 Performance by gene set and normalization. The performance of all models on each gene set (left column) and transform (right column). The results are divided by row into binary, multiclass, and survival tasks. For each gene set or normalization and task type, the gray points show the shifted statistics computed by subtracting the median of all models trained on the same task as a given model, and the black line is the median taken across all models and tasks

sion of other non-predictive genes, then it will be ignored by most unsupervised representation learning algorithms. One way to avoid this problem is use a semi-supervised method to learn the representation.

The goal of semi-supervised representation learning is to derive a common set of features that are useful for multiple downstream predictive tasks. Our semi-supervised model consists of an autoencoder along with a number of logistic regression classifiers, one for each supervised task involved in the training set. The predictors operate on the 512-dimensional latent space embedding of the autoencoder. For any expression vector the autoencoder contributes a reconstruction loss. Furthermore, if there is a predictive task label associated to the expression

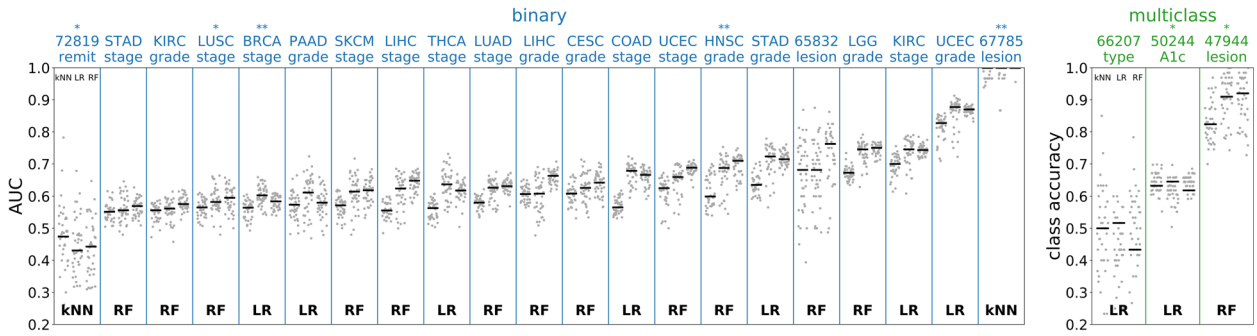

Fig. 4 Performance by supervised model. The classifier tasks are shown in the same format as Fig. 2. In this figure, the model results are divided into groups based on which supervised model is used, kNN (k-nearest neighbors), LR (logistic regression), or RF (random forest). The three horizontal lines for each task show the median result for each of these supervised models in this order (shown in the upper left of the first column plot). For each task, the supervised model with the best median result is shown at the bottom of the plot. While the median over the RF results is most frequently best (for thirteen tasks, compared to nine for logistic regression and two for *k*-nearest neighbors), the best performing logistic regression models are more consistently high performing among models

vector, then the associated linear predictor contributes a classification loss as well. We trained the model to minimize a loss function that was a weighted combination of the autoencoder loss and the supervised loss averaged across each of the predictive tasks. We considered the out-of-sample predictive performance of four representations: the unembedded data, data embedded by a model trained using only autoencoder loss, data embedded by a model trained on the combined autoencoder and supervised losses, and data embedded by a model trained using only the supervised loss. More details are provided in the online "Methods" section.

In order to test the semi-supervised model, we divided the larger labeled training datasets into two halves. The first half of the labeled training datasets were combined with the unlabeled data from the training set and used to train the semi-supervised autoencoder. The second half of the training datasets were held out as validation. We also held out the validation and testing labeled datasets as in the analyses of the representations learned by unsupervised algorithms. This strategy provided two types of validation tasks: those in which the representation was trained on similar data (e.g., from the same study), and those in which the representation had not been trained on similar data. The results are shown in Fig. 6. Using the learned features slightly improved median predictive performance on the divided tasks but did not improve predictive performance on the validation and testing tasks used in the previous analyses.

Discussion

The hypothesis that gene expression measurements can be combined into higher level features that should be useful for predicting phenotypic characteristics has intuitive appeal. Indeed, we believe that genes act together as coordinated pathways that control cellular processes. Moreover, changes in expression at the tissue level could reflect higher level changes due to differences in cellular composition. As a result, one would expect that it is possible to define useful high-level features for expression

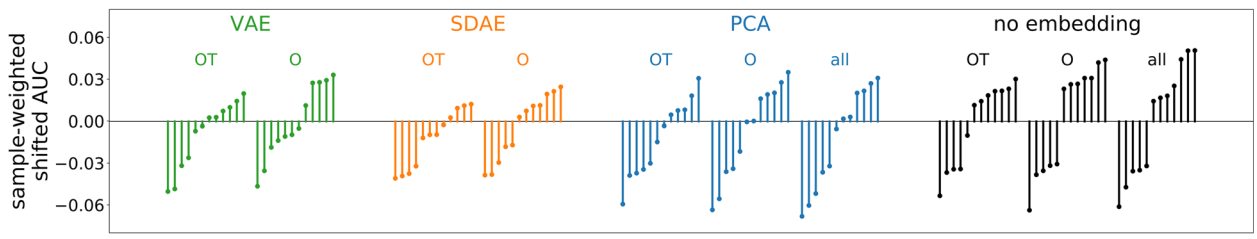

Fig. 5 Performance by unsupervised model and gene set. The binary task performance of each unique model type is shown, grouped by unsupervised model and gene set. A model type is a combination of unsupervised model, supervised model, gene set, and normalization; for example, the recommended model is one model type. Each model type is a single line on this plot. The performance shown is the average of shifted AUCs across binary tasks, weighted by the number of samples in each task to reduce the effect of fluctuations in tasks with fewer samples. There are four unsupervised model types, VAE (variational autoencoder), SDAE (autoencoder), PCA (principal components analysis), and no-embedding (in which the data is unchanged). The best results come from using all genes without an unsupervised embedding

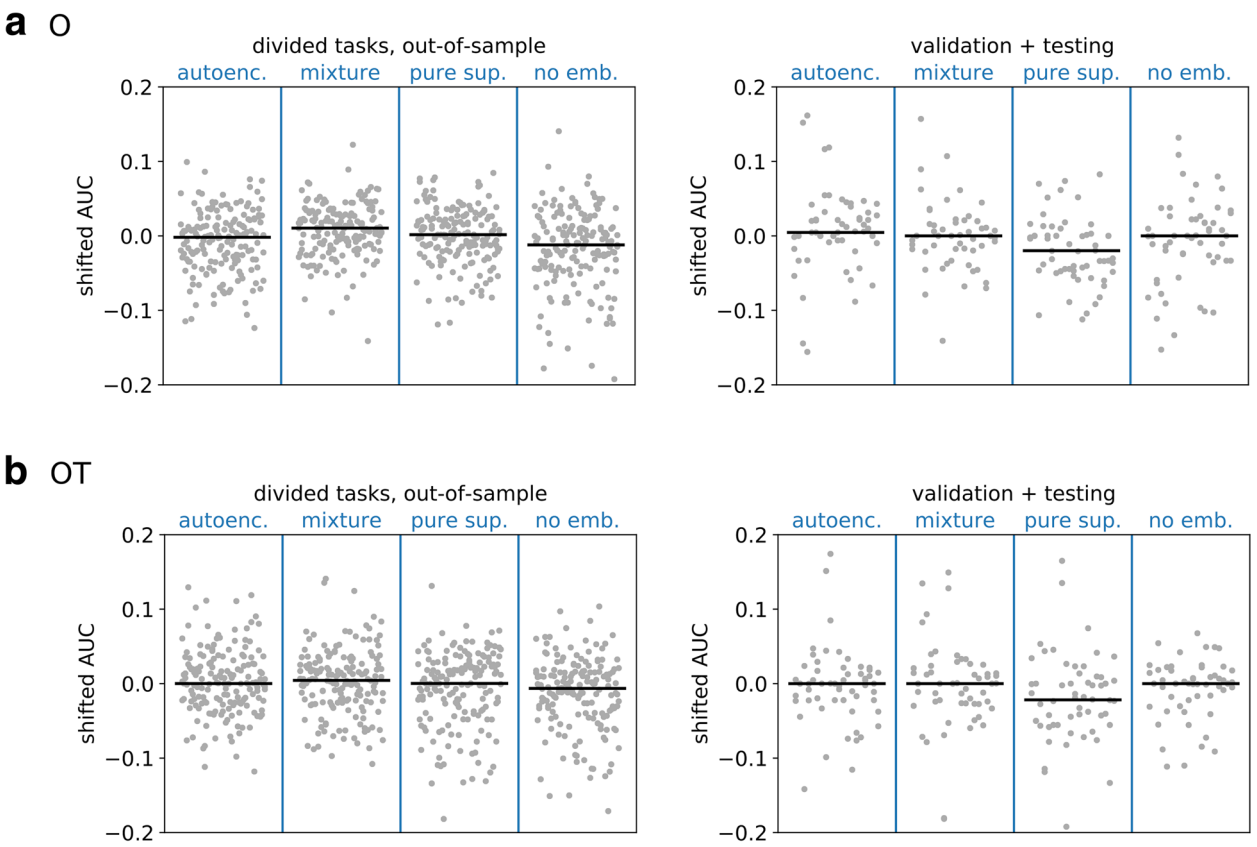

Fig. 6 Performance for semi-supervised models. The binary task performance is shown for four different types of embedding models across two different datasets and two different gene sets. The four models are a purely unsupervised autoencoder (autoenc.), a semi-supervised embedding model (mixture), a purely supervised embedding model (pure sup.), and a no-embedding model (no emb.). To train the supervised component of the embedding models, specific task datasets are divided into two halves, one contributing to the supervised loss in training, and the other held out; the performance of the four models on the held-out halves are shown in the left column. The performance of the same models on the validation and testing tasks (which take no part in training any embedding model) are shown in the right column. Models on the **O** gene set are shown in the upper row, and on the **OT** gene set in the lower row. The gray points show the shifted AUCs on all tasks in each group and all model types, which include all supervised model types and all transforms. The bars show the median score

data; this intuition has driven the development of pathway analyses [33–35], gene set analyses [10, 11], knowledge graphs [14, 15], and cell-type deconvolution approaches [36–38] to analyzing transcriptomics experiments. More recently, a number of studies have introduced deep learning methods that aim to discover useful gene, or transcript, combinations that reflect the underlying biology without imposing particular prior knowledge [4, 5, 39–41]. In theory, these learned representations should provide better predictive performance because they are transferring biological knowledge derived from one dataset to another. In addition, they reduce the dimension of the input data and, as a result, potentially mitigate overfitting. Here, we set out to systematically and rigorously assess the impact of these representations on downstream predictive tasks.

Our key results can be summarized in a few bullet points:

- Multivariate predictors outperformed predictors based on the best single gene.
- Larger gene sets performed better than smaller gene sets.
- CLR and tissue-specific Z-score normalization were better than TPM.
- Logistic regression and random forests performed equally well.

Representations derived from unsupervised or semi-supervised methods did not improve out-of-sample performance for phenotype prediction. Based on these key results, we conclude that l_2-regularized regression applied to the CLR transformed relative transcript abundances is generally the best choice for predictive analyses using transcriptomics data. The Z-score and Z-ternary normalizations generally perform comparably to CLR, but require the GTEx data as a reference and hence CLR is recommended.

Figures 2, 3, 4 and 5 present results for the evaluation of unsupervised models on supervised tasks, studying performance as different aspects of the models change. Figure 2 shows how the performance varies across supervised tasks and demonstrates that the recommended model is nearly always one of the better performing models. Figure 3 presents the relative performance for the choices of normalization and gene set, showing that using larger gene sets improves performance on survival tasks. Figure 4 presents the performance across supervised tasks for different choices of the supervised model, showing that random forest and logistic regression models perform well. Figure 5 shows the relative performance across different unsupervised models, divided by gene set, demonstrating that supervised models on unembedded data for all genes are the best performing. The supervised evaluation results are recorded in an online repository [42] and further visualized in the Supplementary material. Taken together, these motivate the choice of the recommended model.

Our first conclusion, that multivariate predictors outperform predictors based on single gene expression measurements, was the clearest cut. This has some practical consequences when combined with our other conclusion that larger gene sets are better, especially for the fitting of proportional hazards models used for survival analyses. First, using multivariate predictors on large gene sets means that the number of covariates will almost always vastly outnumber the subjects in a study. Therefore, it is absolutely necessary to regularize these models by adding penalties to the coefficients. Moreover, nested cross validation should be used for all performance assessments to mitigate overfitting to hyperparameter choices and to minimize variance in the performance metric. Second, it is often impractical —or even impossible— to fit these models using standard methods on typical computing architectures. For example, open source packages for survival analyses typically use second-order methods to optimize the objective function. This works for a single gene, but fitting the multivariate model requires computing a very large matrix of second derivatives, e.g., 56,000 x 56,000 in this study. As a result, it was necessary to implement first-order optimization methods and perform most of the matrix operations using graphical processing units to make the survival analyses in this study feasible.

Overall, we found that choices such as the normalization method, the gene set, the type of supervised prediction algorithm, and the use of a learned representation made surprisingly little impact on out-of-sample predictive performance. Moreover, we could not identify any clear trends. For example, it is not necessarily better to use smaller gene sets or other lower dimensional representations for studies with smaller sample sizes. In light of these results, it is not clear that features derived from either prior knowledge or from representation learning methods have much value in the analyses of bulk RNA-seq data. If the relationship between bulk gene expression and phenotype is not one-to-one, then there is already a limit on how well one could predict phenotype from gene expression. Relatively simple methods may be already very close to this limit.

Feature importance for the recommended model

This study consists of an analysis of methods for phenotype prediction in which methods are compared against each other according to their performance across a large number of predictive problems. The overarching question is how well different methods perform in finding a predictive signal in the datasets. Once such a method demonstrates that a useful predictive signal is present it may become beneficial to attempt an interpretation of the model features. Such analysis can yield useful insights about biological function, and the presence of the predictive signal provides credence to such interpretations. Although it is beyond the scope of this work to attempt interpretations of model features for each of the model types, we performed an interpretive analysis of the recommended model as it operates on the binary predictive tasks included in this study. We included tables [43] in the figshare repository of the relative predictive importance of genes for each of the binary tasks, and among all three gene sets. A logistic regression model assigns coefficients to each covariate (in this case genes) which describes the strength of influence of that covariate on the outcome variable. In order to compare the importance of different genes for each regression model we ranked the genes according to their regression coefficient values and normalized the ranking by the total number of genes included. Because the predictive models are trained with five-fold cross-validation, there are five different models for each predictive problem, each with different coefficient values. Therefore, the feature importance of a gene on a particular predictive task is the average normalized rank of that gene's regression coefficient across the five cross-validation folds. The most striking observation we made upon reviewing the gene importance rankings was that many genes related to epithelial-to-mesenchymal transition (EMT) were present in the top genes that differentiate stage in various TCGA cancer types. Tumors with EMT features are more likely to metastasize, consistent with the fact that these features distinguish cancer stage. In fact, the ranks of feature importance derived from our models can be further used for gene set enrichment tests to enable exploring biological processes that associate with contrasts or survival. However, we are wary of placing too much emphasis on such an interpretation of l_2 coefficient rankings. When looking at the distribution of coefficient rankings we did not see examples

of obvious outliers. Rather we saw distributions which looked roughly normal, consistent with the imposition of an l_2 prior on the coefficients.

Conclusions

We believe our analysis will have three important effects on the wide community engaged in the project of extracting insights from RNAseq data.

- Our work provides strong justification for recommendations that guide other researchers working on similar problems with RNAseq datasets. A researcher confronted with the problem of predicting phenotypes from RNAseq data ought to feel confident that l_2-regularized linear predictors will yield results which are at or near state-of-the-art.
- Our work provides a collection of curated datasets and benchmarks which can provide terra firma on which to further develop techniques for predictive analyses on RNAseq datasets. Benchmark datasets coupled with standardized testing protocols are used extensively in machine learning research to assess technological improvements. This work provides such a resource to the computational biology community.
- Our work encourages researchers to direct more energy towards the reduction of systematic errors which appear "higher" in the data chain – from improved lab techniques for handling tissue samples to controlling errors in assay technology itself. Because such sources of error are likely just as present in other technologies such as single-cell RNAseq, improvements in these regards effect more than just bulk RNAseq data.

In summary, transcriptomics-based phenotype prediction clearly benefits from proper normalization techniques and state-of-the-art regression approaches. However, *breakthrough* performance is likely contingent on factors such as reduction of systematic errors in sequencing data, incorporation of other data types such as single-cell sequencing and proteomics, and improved use of prior knowledge.

Methods

The analysis presented here and depicted in Fig. 1 is a multi-step procedure, starting from read counts data in the recount2 database and ending at performance metrics for various models. There are principally three stages: dataset preparation, unsupervised model training, and supervised model training.

Dataset preparation

The recount2 database [7] is a repository of transcriptomics data sourced from over 2000 independent transcriptomics experiments. The transcriptomics data from these experiments has been reprocessed using a uniform processing pipeline, forming a single dataset amenable to large scale computational analyses. Such analyses would otherwise be problematic due to systematic differences between the original processing pipelines. The data in recount2 consists of counts of gene reads as well as exon-level quantifications. Our study concerned the gene counts data exclusively.

The data comprising recount2 can be divided into three broad groups according to their sources: GTEx, TCGA, and SRA. The GTEx group was sourced from the Genotype Tissue Expression program and contains 9538 samples from healthy individuals across 30 tissue types. The TCGA group was sourced from the Cancer Genome Atlas project and contains 11284 samples from individuals with cancer across 21 tissue types. In that group samples were taken from tumor sites as well as normal tissue adjacent to tumor (NAT) sites. GTEx and TCGA are each single, large collaboration projects with high quality control standards and protocols for sample processing. Metadata for these projects is extensive. The SRA group contains 49638 samples from 2033 smaller, distinct experiments collected in the Sequence Read Archive. Metadata for experiments in SRA are sparser, with tissue labels occasionally absent.

In total, 70460 samples were available in recount2 when the database was downloaded. However, many of these samples are not ideal for representation learning with transcriptomics data. We developed a quality control (QC) pipeline to remove samples or entire SRA studies. The number of samples remaining after the QC pipeline is 39848. The QC steps are as follows:

- Remove samples in which the reported cell type is a cell line. 9644 samples fit this criterion.
- Remove studies in SRA from single-cell sequencing. Examining metadata from GEO, 38 studies in SRA with 5865 samples in recount2 have single-cell transcriptomic data.
- Remove samples in which the reported tissue does not match any tissue in GTEx. 6824 samples fit this criterion (see Z-score normalization later).
- Remove samples in SRA which have duplicate GEO accession numbers (GSMs). There were 9601 samples that met this criterion.
- Remove samples in which more than 30% of genes listed in the Gene Ontology (GO) 12 under the "biological process" or "molecular function" categories have a counts value of 0. 15390 samples met this criterion.

The number of matching samples in each step are non-exclusive, meaning a sample can match more than one of the exclusion criteria. These effect of these exclusion criteria are depicted in the Supplementary figures. In total we removed 30612 samples, approximately 43% of the total.

No GTEx samples were removed; and only 521 TCGA samples were removed.

It is useful to present some detailed commentary on the duplicate GEO accession number criterion. We observed that several samples have duplicate GSMs, and that many such samples had the same number of reads (a round number, e.g., 8 million). This suggests that the individual samples could be chunks of reads from the same underlying sample. However, we could find no satisfying reason for duplicate GSMs or the round number of read counts for these samples, and therefore excluded them from the dataset.

The QC pipeline determines which samples are admitted to the final dataset; there are also choices to be made about which genes to consider in the analysis, and which normalizing procedures to apply to the expression data.

We considered three different gene sets in the analysis:

- **all** genes: (57992 genes).
- **O** genes: genes in GO under the "biological process" or "molecular function" categories (17970 genes at the time of dataset creation).
- **OT** genes: **O** genes also labeled as transcription factors 29 (1530 genes at the time of dataset creation).

In addition, we considered four different normalizing transformations of the counts data:

- **TPM**: The counts are transformed into transcripts per million (tpm), which account for gene length to normalize reads. The TPM value is determined in terms of the counts as,

$$\text{tpm}_i = \frac{10^6 \cdot (\text{counts}_i/\text{length}_i)}{\sum_j (\text{counts}_j/\text{length}_j)},$$

  in which i and j index genes.
- **CLR**: A centered-log-ratio transform is carried out on the TPM vectors. The CLR value is determined in terms of the TPM values as,

$$\text{clr}_i = \log(\text{tpm}_i) - \frac{1}{N}\sum_j \log(\text{tpm}_j),$$

  in which N is the number of genes.
- **Z-score**: The Z-score transform is carried out on the CLR features. The Z-score is the CLR value standardized by the mean expression of a gene in healthy tissue, determined by the GTEx samples for the same tissue. The Z-score value is determined in terms of the CLR value as,

$$\text{z-score}_i = \frac{\text{clr}_i - \text{mean}(\text{clr, tissue})_i}{\text{std}(\text{clr, tissue})_i}.$$

- **Z-ternary**: The Z-ternary transform is carried out on the Z-score features. The Z-score values are ternarized based on their value, and the ternarization indicates whether the gene's expression is increased, decreased, or unchanged relative to the mean expression in healthy tissue. Since the distribution of Z-score values is expected to be approximately normal for healthy tissue, any Z-score value below -2 is assigned the Z-ternary value of -1; any Z-score value above 2 is assigned the Z-ternary value of 1, and any Z-score value between -2 and 2 is assigned the Z-ternary value of 0.

We made use of the open-source python library *genemunge* [44] for making these normalizations and selecting the gene sets. Each of the normalizations are carried out on the expression data for all genes. Whenever a smaller gene set is used, the values of the features for the selected genes are simply taken from the data for all genes. The three gene sets and four normalizations yield twelve different datasets that are used in the analysis.

Tasks and dataset allocation

The above procedure describes the preparation of the gene expression datasets. In addition to the expression data, some samples have one or more labels suitable for predictive modeling. TCGA has rich metadata with natural label types, available in the TCGA Pan-Cancer Clinical Data Resource [23]; some SRA studies also contain useful metadata in GEO [24] relevant to human disease. From the TCGA and recount2 metadata we selected four categories of predictive tasks: binary labels for the grade of a tumor in various cancer types (8 tasks); binary labels for the stage of a tumor in various cancer types (10 tasks); times for overall survival in various cancer types (13 tasks); and times for progression free interval in various cancer types (13 tasks). From the GEO metadata we selected binary and multiclass labels for various clinical characteristics (6 tasks) [45–50]. In total there are 50 tasks for which supervised models may be built.

We divided the 50 predictive tasks into three groups, "training", "validation", and "test". We then built a "training" gene expression dataset consisting of any samples with a label in the training task group, as well as any samples with no label. This dataset, which has 36794 gene expression samples, was used to train the unsupervised models. A sample's inclusion in this dataset distinguishes the training and validation task groups. Both the training and validation tasks were used in the analysis, whereas tasks in the test group were held out until the end of the project so that no model selection criteria might influence performance on these tasks in any way. The supervised tasks are summarized in Tables 1, 2, 3, and 4.

Unsupervised models

Three different types of unsupervised models were trained on the gene expression datasets: principal components

analysis (PCA), stacked denoising autoencoders (SDAE), and variational autoencoders (VAE). These three types were chosen because they offer a range of sophistication, features, and expressive power. PCA is a simple means of generating a linear feature embedding. SDAEs and VAEs are examples of deep neural networks which have significant expressive power. The SDAE is capable of learning an implicit distribution of feature embeddings of a dataset; the VAE learns an explicit distribution of feature embeddings which can be sampled from.

Principal components analysis (PCA)

The problem of finding the k principal components of a suitably large collection of vectors of dimension n admits an analytic solution. But the computation required to perform this calculation is in $\mathcal{O}(n^3)$, making it intractable in high dimensions. Due to the high dimension of the larger gene sets (>17k), we performed the PCA analysis via stochastic gradient descent following the algorithm introduced by Arora et al. [51] called "Stochastic Approximation." The leading 512 principal components were retained.

Stacked denoising autoencoders (SDAE)

We employed denoising autoencoder architectures of "hourglass" shape with seven layers. The hourglass narrows to a middle layer of 512 dimensions, yielding a 512-dimensional encoder. The details of the architecture are recorded in the Supplementary material. The models were trained with stochastic gradient descent to minimize the mean squared reconstruction loss. We found that pretraining the models layerwise before end-to-end training produced the best results. Therefore these models are best described as stacked denoising autoencoders per the original presentation [52]. The models were regularized by input noise variance and an l_2 weight penalty, with these hyperparameters selected by sweeping a range.

Variational autoencoders (VAE)

We also included a deep generative model among our unsupervised model types, the variational autoencoder [53]. In particular, we employed the methods of Klambauer et al. [54] which make use of self-normalizing units, SNNs, for improved training dynamics and representational capability. We trained the models using the KL-annealing method of Bowman et al. [55] during the first 100 epochs and then let training proceed with the normal loss function for the remaining epochs. The layer dimensions are recorded in the Supplementary material. The latent encodings consist of 512 dimensions for the distributional means and 512 for the distributional log variances. Therefore the trained model's feature encoder is the restriction to the 512 dimensions of the means variables.

The "no-embedding" model

In addition to these unsupervised models, we also employed a kind of control comparator: a "no-embedding" model which does nothing to the expression data. The dimension of the gene expression data is not reduced under the no-embedding model; the features are the normalized gene expression vectors themselves.

Computational constraints

We trained the PCA model on each of the four normalizations for each of the three gene sets. Due to computational constraints we applied the SDAE and VAE models to each of the four normalizations for the **O** and **OT** gene sets excluding the all genes set. The lack of an improvement in performance on smaller gene sets indicated the dataset with all genes was unlikely to provide quality embedding models.

Supervised models

We evaluated the ability of an unsupervised model to learn useful representations across transcriptomics data by assessing the performance of supervised models operating on the learned representations. For each unsupervised model and predictive task, we trained and evaluated supervised models using nested cross validation. The performance of these predictive models gave an indication of how well the learned representation captured features in the data useful for various kinds of phenotype prediction. Before presenting the different kinds of supervised models, we present a small primer on nested cross validation.

Nested cross validation

Nested cross validation is designed to provide a robust estimate of the expected (predictive) model performance on new data, optimizing over a set of hyperparameter values (such as the maximum depth in a random forest). In nested cross validation, there are two loops over the data, the outer and inner loop. The inner loop is used to select an optimal hyperparameter value, and the outer loop is used to estimate the performance of the model with this hyperparameter value. In the outer loop, data is divided evenly into K groups, or folds (we use $K = 5$). For each fold, the data for that fold is held out and the remaining $K - 1$ folds are used for the inner loop. In the inner loop, this data is divided into K folds, and on each fold the data for that fold is held out and the model is trained on the remaining $K - 1$ folds for each hyperparameter value. The held-out fold is used to estimate the model performance for each hyperparameter value, and this performance is averaged over all folds in the inner loop. The best performing hyperparameter value is selected, and the model is re-trained on all data used in the inner loop. The model performance is then evaluated on the held out data from the outer fold. This value is averaged over all folds in the

outer loop, and this final average is the estimated model performance. Note that a different optimal hyperparameter may be selected for each outer fold. Nested cross validation is resistant to hyperparameter overfitting, as the model is evaluated on data completely held out from the process of selecting the optimal hyperparameter. With this robustness comes increased computational complexity —if there are N hyperparameter values tested, nested cross validation requires training $K(KN + 1)$ individual models.

Classification tasks

For classification tasks we applied three different types of supervised models:

- **Logistic regression (LR)**. Logistic regression with an l_2 penalty, trained via stochastic gradient descent. The logistic regression model is a single layer neural network with a softmax activation on the output. The hyperparameter optimized was the l_2 penalty, logarithmically spaced between 10^{-6} and 10^3 in ten steps. The model was implemented in pytorch [56]. Hyperparameters and training notes are provided in the Supplementary material.
- **Random forest (RF)**. Random forest models with 100 trees per forest. The hyperparameter optimized was the maximum depth of the random forest, logarithmically spaced between 2 and 2^7 in seven steps. We relied on the scikit-learn [57] implementation of random forest.
- **K-nearest neighbors (kNN)**. The hyperparameter optimized was the value of k, the number of neighbors used, taking a value of 1, 3, 5, 7, or 9.

Although there are countless types of predictive models, we chose these three because they cover a wide range of features and characteristics of predictive models. LR is the canonical example of a generalized linear model. A random forest is a decision tree-based, non-linear classifier which is known to achieve state-of-the art performance on a large number of difficult classification problems [58]. And finally, the kNN is a non-parametric model, making a contrast with the other two parametric models.

Survival tasks

For survival tasks, in which the overall survival time or the progression free interval time were predicted, we trained a Cox proportional hazard (CPH) model. The standard solvers for CPH models use second-order methods, such as versions of Newton's method, making them unsuitable for use with a large number of features. The computation time required for the 512-dimensional embedding, using nested cross validation, is already immense; training CPH models on data without an embedding is completely impractical. Instead, we implemented a CPH model in pytorch, and trained it via stochastic gradient descent by backpropagating through the Cox-Efron pseudolikelihood[59]. Such models can be trained with a large number of features —even all genes— and can be accelerated with graphical processing units. We regularized these models with an l_2 penalty whose strength, logarithmically spaced between 10^{-6} and 10^3 in ten steps, was optimized in the inner cross-validation loop. Even with this computational speedup, evaluating the survival tasks requires the bulk of compute time. It bears noting that these models were still trained with a fixed initial learning rate which was small enough to guarantee controlled gradient descent across all tasks. It is certain that absolute performance on individual contrasts could be improved by also optimizing the learning rate in the nested cross validation. However, because the study concerns the relative performance of this algorithm across gene sets, embeddings, and normalizations, we avoided this additional multiplier on the computational time.

Single-gene comparators

All of the above supervised models are trained on features from multiple genes. In order to compare our embedding models to single-gene analysis, we also trained a set of models on single genes with no-embedding model. For these models the hyperparameter optimized in the inner cross validation loop is the gene selected for the model. We had no need to run these comparators across all transform/predictor combinations so we restricted these examples to clr-transformed data and used only the univariate logistic regression models for classification tasks. For survival tasks, single-gene CPH models were trained. No regularization term was necessary in either case because these models have so few parameters. These results provide a direct comparison to the multi-gene, CLR-transformed, no-embedding results.

Components of supervised model results

In total, we evaluated a very large number of (multi-gene) supervised models. There are five different characteristics of a single result:

- **Task**. There are 24 classification tasks and 26 survival tasks.
- **Gene set**. There are three gene sets, **all** genes, **O** genes, and **OT** genes.
- **Normalization**. There are four data normalizations, TPM, CLR, Z-score, and Z-ternary.
- **Unsupervised model**. There are four types of unsupervised model, PCA, SDAE, VAE, and no-embedding. SDAE and VAE were only trained on the **O** and **OT** gene sets.
- **Supervised model**. For classification tasks, three different supervised models were trained, LR, RF, and kNN. For survival tasks, a CPH model was trained.

This amounts to 3920 results. In terms of individual models trained during nested cross validation, there are 807600 models.

Semi-supervised models

The semi-supervised models are designed to learn a feature embedding which is co-adapted to the purpose of reconstruction as well as the performance of supervised models operating on the embedding. Each of the semi-supervised models consists of a linear, single-hidden layer autoencoder coupled to a number of logistic regression predictors —one for each supervised task involved in the training dataset. The supervised predictors operate on the autoencoder's 512-dimensional encoding. Both a schematic diagram of the model architecture and the details of the architectures are recorded in the Supplementary figures.

Data preparation

In order to be able to assess the performance of semi-supervised representation learning within-task, we had to further subdivide some of the labeled expression data. In particular, we subdivided into two halves the binary predictive tasks within the "training set" which contained at least 200 samples. The first half was used in the training of the semi-supervised model; the second half was held out for validation. We called these sets the "divided tasks"; they were drawn from the following binary tasks:

{CESC grade, COAD stage, KIRC grade, KIRC stage, LGG grade, LIHC stage, LIHC grade, LUAD stage, SKCM stage, STAD stage, STAD grade, THCA stage, UCEC stage, UCEC grade}.

The rest of the tasks which constituted the original "validation" and "test" sets were used for validation. So the training set for each semi-supervised model consisted of all expression data from the first halves of the fourteen divided tasks along with their associated binary labels.

Training of semi-supervised models

Given any sample expression vector x from the training set we can compute the autoencoder reconstruction loss on that sample, specifically as the squared reconstruction error, $\mathrm{R}(x)_i := (\mathrm{AE}(x)_i - x_i)^2$, in which $\mathrm{AE}(x)$ denotes the action of the autoencoder on x. Supposing that x has a class label $l_x \in \{0, 1\}$ from the j^{th} predictive task, we can also compute a classification error of the associated binary logistic regression classifier P_j,

$$\begin{aligned}\mathrm{C}(x, l_x) &= \mathrm{CrossEntropy}(\mathrm{P}_j, x, l_x)\\ &:= -\log(\mathrm{P}_j(x) l_x + (1 - \mathrm{P}_j(x))(1 - l_x)).\end{aligned}$$

We trained our semi-supervised models (via stochastic gradient descent) to minimize a convex combination of these two error terms. The constant controlling the interpolation of these two losses we called the "predictor strength," π, which ranged from 0 to 1. Our training algorithm allowed different batch sizes for the autoencoder loss and the predictor losses; let these be denoted by BR, and BC, respectively. Let $\{x_m\}, \{x_n, l_{x_n}\}$ be batches drawn randomly from the training data, the first consisting of only expression vectors, the second containing both expression vectors and paired class labels. Our loss term takes the form,

$$\begin{aligned}\mathcal{L}(\{x_m\}, \{x_n, l_{x_n}\}) := &\frac{1-\pi}{\mathrm{BR}} \cdot \sum_{m=1}^{\mathrm{BR}} \mathrm{R}(x_m) + \frac{\pi}{\mathrm{BC}} \cdot \sum_{n=1}^{\mathrm{BC}} \mathrm{C}(x_n, l_{x_n})\\ &+ \lambda_{\mathrm{AE}} \cdot l_2(\mathrm{AE}) + \lambda_P \cdot \sum_{j=1}^{J} l_2(\mathrm{P}_j).\end{aligned}$$

Here, J denotes the total number of predictive tasks. The last two terms are l_2 weight penalties on the model parameters; these are controlled by adjustable constants λ_{AE} and λ_{P}.

We compared three scenarios for the predictor strength in our analysis,

- $\pi = 0.0$, i.e. the model is an autoencoder only.
- $\pi = 0.1$, the model is trained with a mixture of both losses.
- $\pi = 1.0$, i.e. the model is a purely supervised shared-embedding model.

We also compared results to a no-embedding model as a kind of control.

For each gene set, data normalization, and predictor strength scenario, we performed a sweep over all 16 pairs of values for λ_{AE} and λ_{P} in the cartesian product $\{0, 0.1, 0.01, 0.001\}^2$. We selected the l_2 coefficient pair which minimized average error on the held-out half of the divided contrasts.

Finally, we assessed the performance of predictive models (across all three types, LR, RF, kNN) operating on the learned data embedding to compare the effect of semi-supervised representation learning across these three scenarios. Those results are displayed in Fig. 6 in the main text.

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1186/s12859-020-3427-8.

Additional file 1: Supplementary Figures and Tables.

Author contributions

D.Z. and C.K.F. conceived of the analysis. A.M.S., J.R.W., and J.L. carried out the analysis. A.M.S., J.R.W., and C.K.F. wrote the manuscript. D.Z, C.K.F, A.M.S, J.R.W, J.L., C.B.D., P.H., M.R.H, M.M., X.J.M, S.R., and S.Z. participated in the design of the analysis, the interpretation of its results, and revisions to the manuscript. All authors have read and approved the manuscript in its final form.

Abbreviations
ARCHS4: All RNA-seq and CHIP-Seq sample and signature search; AUC: Area under the curve (of the receiver operating characteristic); BLCA: BLadder urothelial CArcinoma; BRCA: BReast invasive CArcinoma C-index: Concordance index; CESC: CErvical squamous cell carcinoma; CLR: Centered log transform; COAD: COlon ADenocarcinoma; CPH: Cox proportional hazards; CPM: Counts per million; EMT: Epithelial-to-mesenchymal transition; ESCA: ESophageal CArcinoma; FPKM: Fragments per kilobase per million; GEO: Gene expression omnibus; GSM: GEO accession number; GTEx: Genome tissue expression; HNSC: Head-neck squamous cell carcinoma; KIRC: KIdney renal clear cell carcinoma; KIRP: KIdney renal papillary cell carcinoma; kNN: k-Nearest neighbors; LGG: Low grade glioma; LIHC: LIver hepatocellular carcinoma; LR: Logistic regression; LUAD: LUng ADenocarcinoma; LUSC: LUng Squamous cell carcinoma; NAT: Normal tussue adjacent to tumor; OV: Ovarian cancer; PAAD: PAncreatic ADenocarcinoma; PCA: Principal components analysis; QC: Quality control; RF: Random forest; SARC: SARComa; SDAE: Stacked denoising AutoEncoder; SKCM: SKin cutaneous melanoma; SRA: Short read archive; STAD: STomach ADenocarcinoma TCGA: The cancer genome atlas; THCA: THyroid CAncer; TPM: Transcripts per million; UCEC: Uterine corpus endometrial carcinoma; VAE: Variational AutoEncoder

Acknowledgements
Not applicable.

Funding
This study was funded by Pfizer inc.

Availability of data and materials
The gene expression data used to train the unsupervised models and the gene expression data and labels used to train and evaluate the supervised models are available on figshare [60]. These data include the gene expression data for all three gene sets and all four normalizations, and the accompanying supervised labels. The code used to process the data and train the unsupervised, supervised, and semi-supervised models is a combination of open source and proprietary, licensable software. The open source software [61], available on GitHub, is a framework to carry out nested cross validation and includes an example to train the recommended model on a particular supervised task. Anyone seeking to test new approaches can adapt their model code to the API of the provided nested cross validation tool in order to rigorously compare their new model to the results of this study.

Ethics approval and consent to participate
All transcriptomics data used in this study is from the recount2 database [7] and is freely available under the CC BY 4.0 license.

Consent for publication
Not applicable.

Competing interests
A.M.S, J.R.W, and C.K.F are affiliated with Unlearn.AI, Inc., a company that creates software for clinical research, and hence may have competing financial interests. J.L., C.B.D., P.H., M.R.H., M.M., J.X.M., S.R., S.Z., and D.Z. are employees of Pfizer and may own stock or stock options in Pfizer.

Author details
[1]Unlearn.AI, Inc., San Francisco, CA, USA. [2]Computational Sciences, Worldwide Research & Development, Pfizer Inc., Cambridge, MA, USA. [3]Oncology Global Product Development, Pfizer Inc., San Diego, CA, USA. [4]Business Technology, Pfizer Inc., Cambridge, MA, USA. [5]Inflammation and Immunology, Worldwide Research & Development, Pfizer Inc., Cambridge, MA, USA. [6]Oncology Research & Development, Worldwide Research & Development, Pfizer Inc., San Diego, CA, USA. [7]Inflammation and Immunology, Worldwide Research & Development, Pfizer Pharma GmbH., Berlin, Germany.

Received: 30 January 2020 Accepted: 21 February 2020
Published online: 20 March 2020

References
1. Madhukar NS, Elemento O. Bioinformatics Approaches to Predict Drug Responses from Genomic Sequencing. Methods Mol Biol (Clifton, N.J.) 2018;1711:277–96. https://doi.org/10.1007/978-1-4939-7493-1-14.
2. Li S, Łabaj PP, Zumbo P, Sykacek P, Shi W, Shi L, Phan J, Wu P-Y, Wang M, Wang C, Thierry-Mieg D, Thierry-Mieg J, Kreil DP, Mason CE. Detecting and correcting systematic variation in large-scale RNA sequencing data. Nat Biotechnol. 2014;32(9):888–95. https://doi.org/10.1038/nbt.3000.
3. ́ Hoen PAC, et al. Reproducibility of high-throughput mRNA and small RNA sequencing across laboratories. Nat Biotechnol. 2013;31(11): 1015–22. https://doi.org/10.1038/nbt.2702.
4. Ching, et al. Opportunities and obstacles for deep learning in biology and medicine. J R Soc Interface. 2018;15(141):20170387. https://doi.org/10.1098/rsif.2017.0387.
5. Mamoshina P, Vieira A, Putin E, Zhavoronkov A. Applications of Deep Learning in Biomedicine. Mol Pharm. 2016;13(5):1445–54. https://doi.org/10.1021/acs.molpharmaceut.5b00982.
6. Frazee AC, Langmead B, Leek JT. ReCount: A multi-experiment resource of analysis-ready RNA-seq gene count datasets. BMC Bioinformatics. 2011;12(1):449. https://doi.org/10.1186/1471-2105-12-449.
7. Collado-Torres L, Nellore A, Kammers K, Ellis SE, Taub MA, Hansen KD, Jaffe AE, Langmead B, Leek JT. Reproducible RNA-seq analysis using *recount2*. Nat Biotechnol. 2017;35:319–21. https://doi.org/10.1038/nbt.3838.
8. Lachmann A, Torre D, Keenan AB, Jagodnik KM, Lee HJ, Wang L, Silverstein MC, Ma'ayan A. Massive mining of publicly available RNA-seq data from human and mouse. Nat Commun. 2018;9(1):1366. https://doi.org/10.1038/s41467-018-03751-6.
9. Ellis SE, Collado-Torres L, Jaffe A, Leek JT. Improving the value of public RNA-seq expression data by phenotype prediction. Nucleic Acids Res. 2018;46(9):54. https://doi.org/10.1093/nar/gky102.
10. Gönen M. Integrating gene set analysis and nonlinear predictive modeling of disease phenotypes using a Bayesian multitask formulation. BMC Bioinformatics. 2016;17(16):0. https://doi.org/10.1186/s12859-016-1311-3.
11. Subramanian A, Tamayo P, Mootha VK, Mukherjee S, Ebert BL, Gillette MA, Paulovich A, Pomeroy SL, Golub TR, Lander ES, Mesirov JP. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. Proc Natl Acad Sci. 2005;102(43): 15545–50. https://doi.org/10.1073/pnas.0506580102.
12. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene Ontology: Tool for the unification of biology. Nat Genet. 2000;25(1):25–29. https://doi.org/10.1038/75556.
13. Zarringhalam K, Degras D, Brockel C, Ziemek D. Robust phenotype prediction from gene expression data using differential shrinkage of co-regulated genes. Sci Rep. 2018;8(1):1237. https://doi.org/10.1038/s41598-018-19635-0.
14. Zarringhalam K, Enayetallah A, Reddy P, Ziemek D. Robust clinical outcome prediction based on Bayesian analysis of transcriptional profiles and prior causal networks. Bioinformatics. 2014;30(12):69–77. https://doi.org/10.1093/bioinformatics/btu272.
15. Kang T, Ding W, Zhang L, Ziemek D, Zarringhalam K. A biological network-based regularized artificial neural network model for robust phenotype prediction from gene expression data. BMC Bioinformatics. 2017;18(1):565. https://doi.org/10.1186/s12859-017-1984-2.
16. Shen Y-J, Huang S-G. Improve Survival Prediction Using Principal Components of Gene Expression Data. Genom Proteomics Bioinforma. 2006;4(2):110–9. https://doi.org/10.1016/S1672-0229(06)60022-3.
17. Lopez R, Regier J, Cole MB, Jordan MI, Yosef N. Deep generative modeling for single-cell transcriptomics. Nat Methods. 2018;15(12):1053. https://doi.org/10.1038/s41592-018-0229-2.
18. Grønbech CH, Vording MF, Timshel PN, Sønderby CK, Pers TH, Winther O. scVAE: Variational auto-encoders for single-cell gene expression data. bioRxiv. 2018. https://doi.org/10.1101/318295.
19. Way GP, Greene CS. Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. Pac Symp Biocomput Pac Symp Biocomput. 2018;23:80–91.
20. Rampasek L, Hidru D, Smirnov P, Haibe-Kains B, Goldenberg A. Dr.VAE: Drug Response Variational Autoencoder. 2017. 1706.08203.
21. Bengio Y, Courville A, Vincent P. Representation Learning: A Review and New Perspectives. IEEE Trans Pattern Anal Mach Intell. 2013;35(8): 1798–828. https://doi.org/10.1109/TPAMI.2013.50.

22. Lonsdale, et al. The Genotype-Tissue Expression (GTEx) project. Nat Genet. 2013;45:580–5. https://doi.org/10.1038/ng.2653.
23. Liu, et al. An Integrated TCGA Pan-Cancer Clinical Data Resource to Drive High-Quality Survival Outcome Analytics. Cell. 2018;173(2):400–41611. https://doi.org/10.1016/j.cell.2018.02.052.
24. Barrett, et al. NCBI GEO: Archive for functional genomics data sets–update. Nucleic Acids Res. 2013;41(Database issue):991–5. https://doi.org/10.1093/nar/gks1193.
25. Wagner GP, Kin K, Lynch VJ. Measurement of mRNA abundance using RNA-seq data: RPKM measure is inconsistent among samples. Theory Biosci Theor Den Biowissenschaften. 2012;131(4):281–5. https://doi.org/10.1007/s12064-012-0162-3.
26. Li B, Dewey CN. RSEM: Accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC Bioinformatics. 2011;12(1):323. https://doi.org/10.1186/1471-2105-12-323.
27. Aitchison J. The Statistical Analysis of Compositional Data. J R Stat Soc Ser B (Methodological). 1982;44(2):139–77.
28. Lovell D, Pawlowsky-Glahn V, Egozcue JJ, Marguerat S, Bähler J. Proportionality: A valid alternative to correlation for relative data. PLoS Comput Biol. 2015;11(3):1004075. https://doi.org/10.1371/journal.pcbi.1004075.
29. Fernandes AD, Reid JN, Macklaim JM, McMurrough TA, Edgell DR, Gloor GB. Unifying the analysis of high-throughput sequencing datasets: Characterizing RNA-seq, 16S rRNA gene sequencing and selective growth experiments by compositional data analysis. Microbiome. 2014;2(1):15. https://doi.org/10.1186/2049-2618-2-15.
30. Chawla K, Tripathi S, Thommesen L, Lægreid A, Kuiper M. TFcheckpoint: A curated compendium of specific DNA-binding RNA polymerase II transcription factors. Bioinformatics (Oxford, England). 2013;29(19):2519–20. https://doi.org/10.1093/bioinformatics/btt432.
31. Mann HB, Whitney DR. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. Ann Math Stat. 1947;18(1):50–60. https://doi.org/10.1214/aoms/1177730491.
32. Harrell FE. Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis: Springer; 2001. https://www.springer.com/gp/book/9781441929181.
33. Fabregat, et al. The Reactome Pathway Knowledgebase. Nucleic Acids Res. 2018;46(D1):649–55. https://doi.org/10.1093/nar/gkx1132.
34. Krämer A, Green J, Pollard J, Tugendreich S. Causal analysis approaches in Ingenuity Pathway Analysis. Bioinformatics (Oxford, England). 2014;30(4):523–30. https://doi.org/10.1093/bioinformatics/btt703.
35. Ozerov IV, et al. In Silico Pathway Activation Network Decomposition Analysis (iPANDA) as a method for biomarker development. Nat Commun. 2016;7:13427. https://doi.org/10.1038/ncomms13427.
36. Zhao Y, Simon R. Gene expression deconvolution in clinical samples. Genome Med. 2010;2(12):93. https://doi.org/10.1186/gm214.
37. Gaujoux R, Seoighe C. CellMix: A comprehensive toolbox for gene expression deconvolution. Bioinformatics. 2013;29(17):2211–2. https://doi.org/10.1093/bioinformatics/btt351.
38. Shen-Orr SS, Tibshirani R, Khatri P, Bodian DL, Staedtler F, Perry NM, Hastie T, Sarwal MM, Davis MM, Butte AJ. Cell type-specific gene expression differences in complex tissues. Nat Methods. 2010;7(4):287–9. https://doi.org/10.1038/nmeth.1439.
39. Gupta A, Wang H, Ganapathiraju M. Learning structure in gene expression data using deep architectures, with an application to gene clustering. bioRxiv. 2015. https://doi.org/10.1101/031906.
40. Dincer AB, Celik S, Hiranuma N, Lee S-I. DeepProfile: Deep learning of cancer molecular profiles for precision medicine. bioRxiv. 2018. https://doi.org/10.1101/278739.
41. Way GP, Greene CS. Evaluating deep variational autoencoders trained on pan-cancer gene expression. 2017. 1711.04828.
42. Supervised results table. https://figshare.com/articles/Supervised_results_table/7817570. Accessed: 17 May 2019.
43. Feature importance for the recommended model. https://figshare.com/articles/Recommended_model_feature_importance_on_binary_predictive_tasks/8980325. Accessed: 24 July 2019.
44. Fisher CK, Smith AM, Walsh JR. Who is this gene and what does it do? A toolkit for munging transcriptomics data in python. bioRxiv. 2018299107. https://doi.org/10.1101/299107.
45. Suárez-Fariñas M, et al. RNA sequencing atopic dermatitis transcriptome profiling provides insights into novel disease mechanisms with potential therapeutic implications. J Allergy Clin Immunol. 2015;135(5):1218–27. https://doi.org/10.1016/j.jaci.2015.03.003.
46. Peck BCE, et al. MicroRNAs Classify Different Disease Behavior Phenotypes of Crohn's Disease and May Have Prognostic Utility. Inflamm Bowel Dis. 2015;21(9):2178–87. https://doi.org/10.1097/MIB.0000000000000478.
47. Tew GW, et al. Association Between Response to Etrolizumab and Expression of Integrin αE and Granzyme A in Colon Biopsies of Patients With Ulcerative Colitis. Gastroenterology. 2016;150(2):477–4879. https://doi.org/10.1053/j.gastro.2015.10.041.
48. Di Meglio P, Duarte JaH, Ahlfors H, Owens NDL, Li Y, Villanova F, Tosi I, Hirota K, Nestle FO, Mrowietz U, Gilchrist MJ, Stockinger B. Activation of the aryl hydrocarbon receptor dampens the severity of inflammatory skin conditions. Immunity. 2014;40(6):989–1001. https://doi.org/10.1016/j.immuni.2014.04.019.
49. Fadista Ja, et al. Global genomic and transcriptomic analysis of human pancreatic islets reveals novel genes influencing glucose metabolism. Proc Natl Acad Sci USA. 2014;111(38):13924–9. https://doi.org/10.1073/pnas.1402665111.
50. Swindell WR, Remmer HA, Sarkar MK, Xing X, Barnes DH, Wolterink L, Voorhees JJ, Nair RP, Johnston A, Elder JT, Gudjonsson JE. Proteogenomic analysis of psoriasis reveals discordant and concordant changes in mRNA and protein abundance. Genome Med. 2015;7(1):86. https://doi.org/10.1186/s13073-015-0208-5.
51. Arora R, Cotter A, Livescu K, Srebro N. Stochastic optimization for PCA and PLS. In: 2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton); 2012. p. 861–8. https://doi.org/10.1109/Allerton.2012.6483308.
52. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res. 2010;11(Dec):3371–408.
53. Kingma DP, Welling M. Auto-Encoding Variational Bayes. 2013. 1312.6114.
54. Klambauer G, Unterthiner T, Mayr A, Hochreiter S. Self-Normalizing Neural Networks. 2017. 1706.02515.
55. Bowman SR, Vilnis L, Vinyals O, Dai AM, Jozefowicz R, Bengio S. Generating Sentences from a Continuous Space. 2015. 1511.06349.
56. Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L, Lerer A. Automatic differentiation in PyTorch. In: Proceedings of Neural Information Processing Systems; 2017.
57. Pedregosa F, et al. Scikit-learn: Machine Learning in Python. J Mach Learn Res. 2011;12:2825–30.
58. Fawagreh K, Gaber MM, Elyan E. Random forests: from early developments to recent advancements. Syst Sci Control Eng. 2014;2(1):602–9. https://doi.org/10.1080/21642583.2014.956265. Accessed 22 July 2019.
59. Efron B. The Efficiency of Cox's Likelihood Function for Censored Data. J Am Stat Assoc. 1977;72(359):557–65. https://doi.org/10.1080/01621459.1977.10480613.
60. Dataset Repository. https://figshare.com/projects/Deep_learning_of_representations_for_transcriptomics-based_phenotype_prediction/60938. Accessed: 17 May 2019.
61. Code repository. https://github.com/unlearnai/representation_learning_for_transcriptomics. Accessed: 17 May 2019.

Publisher's Note