Bioinformatics, 2025, **41(11)**, btaf533
https://doi.org/10.1093/bioinformatics/btaf533
Advance Access Publication Date: 1 October 2025
Original Paper

OXFORD

Gene expression

Self-supervised representation learning on gene expression data

Kevin Dradjat[1,2,*], Massinissa Hamidi[1], Pierre Bartet[2], Blaise Hanczar[1]

[1]IBISC Laboratory, University Paris-Saclay (Univ. Evry), Evry-Courcouronnes 91000, France
[2]ADLIN, Evry-Courcouronnes 91000, France

*Corresponding author. IBISC Laboratory, Evry University, Evry-Courcouronnes 91000, France. E-mail: kevin.dradjat@univ-evry.fr.

Associate Editor: Laura Cantini

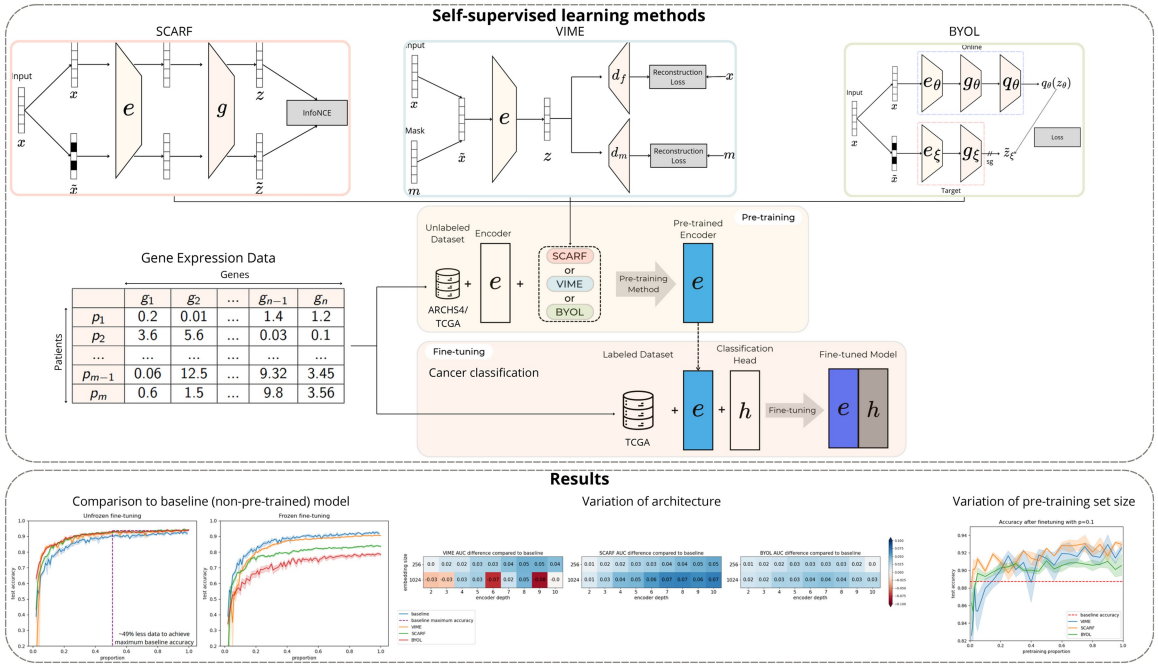
Abstract

Motivation: Predicting phenotypes from gene expression data is a crucial task in biomedical research, enabling insights into disease mechanisms, drug responses, and personalized medicine. Traditional machine learning and deep learning rely on supervised learning, which requires large quantities of labeled data that are costly and time-consuming to obtain in the case of gene expression data. Self-supervised learning has recently emerged as a promising approach to overcome these limitations by extracting information directly from the structure of unlabeled data.

Results: In this study, we investigate the application of state-of-the-art self-supervised learning methods to bulk gene expression data for phenotype prediction. We selected three self-supervised methods, based on different approaches, to assess their ability to exploit the inherent structure of the data and to generate qualitative representations which can be used for downstream predictive tasks. By using several publicly available gene expression datasets, we demonstrate how the selected methods can effectively capture complex information and improve phenotype prediction accuracy. The results obtained show that self-supervised learning methods can outperform traditional supervised models besides offering significant advantage by reducing the dependency on annotated data. We provide a comprehensive analysis of the performance of each method by highlighting their strengths and limitations. We also provide recommendations for using these methods depending on the case under study. Finally, we outline future research directions to enhance the application of self-supervised learning in the field of gene expression data analysis. This study is the first work that deals with bulk RNA-Seq data and self-supervised learning.

Availability and implementation: The code and results are available at https://github.com/kdradjat/ssrl-rnaseq.

Graphical abstract



Received: 6 January 2025; **Revised:** 21 August 2025; **Accepted:** 17 September 2025

1 Introduction

High-throughput sequencing technologies have made it possible to generate large amounts of transcriptomic data, offering a wide range of materials for scientific research (Qin 2019). However, the preprocessing and analysis of these complex data present significant challenges, particularly when it comes to building accurate and generalizable predictive models. In this article, we focus on predicting phenotypes from gene expression data. Phenotype prediction is key in biomedical research, aiding in understanding diseases, finding biomarkers, and developing personalized therapies.

Classical supervised machine learning methods achieve the best performance in this domain (Alharbi and Vakanski 2023) and recent advances have shown that deep learning methods have great potential to outperform classical machine learning methods (Tran *et al.* 2021, Hanczar *et al.* 2022), as deep neural networks can capture non-linear relationships between variables, in exchange of a large training dataset. These methods rely heavily on annotated data, which is costly to obtain. Besides, the available gene expression datasets do not contain as many examples as in other domains, like images or natural language processing where datasets are composed of millions of examples. This leads to a higher risk of overfitting for deep learning models and a limited generalization of the trained models on new datasets.

To overcome these challenges, transfer learning and semi-supervised learning approaches are used to deal with small labeled dataset and their reliability has already been studied for gene expression data (Hanczar *et al.* 2022, Hsu and Lin 2023). More recently, self-supervised learning approaches have emerged as a promising approach (Ericsson *et al.* 2022, Gui *et al.* 2024) and has already been shown to be efficient for related biological data such as single-cell data (Richter *et al.* 2024) or medical images (Azizi *et al.* 2023). Indeed, in contrast to supervised learning methods, self-supervised learning does not rely on annotated data and extracts information directly from the structure of the data to establish meaningful representations that can be applied to the resolution of various downstream tasks. Self-supervised learning enables the utilization of non-annotated datasets unrelated to phenotype prediction, addressing the scarcity of annotated data and improving model performance.

In this study, we propose investigating the performance of state-of-the-art self-supervised learning methods for phenotype prediction from gene expression data. We selected three self-supervised methods from different self-supervised learning categories that were initially created and applied to tabular data and evaluated their ability to generate effective representations for predictive models. Through large scale and systematic experiments using public transcriptomic datasets, we demonstrate the effectiveness of these approaches in terms of predictive accuracy and highlight their limitations. We also provide recommendations for using these methods depending on the case under study.

By providing a comparative analysis of the performance of each selected method, this article aims to highlight the potential of self-supervised learning to improve phenotype prediction while reducing the dependence on annotated data, offering new perspectives for biomedical research and personalized medicine.

2 Methods

2.1 Self-supervised learning

Recent breakthroughs in supervised learning have enabled the development of high-performance models to solve various machine learning tasks. However, supervised learning requires access to a large amount of annotated data, which can be costly and time-consuming in some cases. The availability of large amounts of unlabeled data has led to major advances in different fields, such as natural language processing or the vision domain. These advances are largely due to recent self-supervised methods, which construct a new data representation space with a pre-training step, making prediction tasks easier (Ericsson *et al.* 2022).

Formally, we consider a set of labeled data $\mathcal{D}_l = \{x_i, y_i\}_{i=1}^{|\mathcal{D}_l|}$ of $|\mathcal{D}_l|$ examples, where $x_i \in \mathcal{X} \subset \mathbb{R}^N$ denotes an example and $y_i \in \mathcal{Y}$ its associated label. In the context of supervised learning, we aim to learn a function $f : \mathcal{X} \to \mathcal{Y}$ by minimizing a supervised loss function (e.g. MSE or cross-entropy). With the self-supervised approach, we leverage an unlabeled dataset $\mathcal{D}_u = \{x_i\}_{i=1}^{|\mathcal{D}_u|}$ to learn an encoder function $e : \mathcal{X} \to \mathcal{Z}$, where $\mathcal{Z}$ is the latent space defined by a pretext task. This first step is called pre-training. Generally, $|\mathcal{D}_u| \gg |\mathcal{D}_l|$ since unlabeled data is easier to collect than labeled data. Then, the learned representation space is used to solve various downstream tasks by using a small labeled dataset and attaching a classification head to the pre-trained encoder. This second step is called fine-tuning. The self-supervised training process is described in Fig. 1. This approach allows a generalization, allowing the creation of robust and polyvalent models, called foundation models, which can be adapted to various downstream tasks with small annotated datasets. Self-supervised learning is mainly characterized by the strategy adopted to define the features to be predicted during the pre-training, the so-called pretext tasks. The choice of the pretext task is essential, as it impacts the representations computed by the encoder. The definition of these pretext tasks mainly relies on the exploitation of underlying domain-specific structure of the data, e.g. spatial relations for images or semantic relations for the language. Therefore, the pretext task is tightly linked to the nature of the considered data.

In this study, we focus on self-supervised methods for tabular data, as gene expression data are arranged in the form of high-dimension tabular data, where rows correspond to patients and columns correspond to genes. Besides, the internal structure of tabular data is the most challenging to discover as there are no explicit relationships between variables. The specificity of gene expression data is the high dimensionality (>50k features) and low sample size. In addition to their tabular format, some genes are highly similar and therefore are expressed similarly, which can introduce redundancy in the data.

In the literature on self-supervised learning, few works has been done for tabular data (Wang *et al.* 2024) and no methods were designed or applied on bulk RNA-Seq data. Various approaches have been proposed to advance the exploitation of implicit tabular structures without labels. We distinguish three main types of approach:

- Predictive/Generative Learning: This approach uses a predictive or generative task as a pretext task. The challenge lies on designing predictive tasks that are effective considering the downstream task on which the model will be applied. There is no consensus on the choice of the

predictive task and several methods have been proposed, e.g. reconstructing examples from masked features (Huang *et al.* 2020), by applying perturbations in latent space (Nam *et al.* 2023) or by adopting natural language processing methods (Dinh *et al.* 2022).

- Contrastive Learning: This approach aims to extract information from inter-sample similarities and differences, ensuring that two similar examples also have similar representations in latent space and inversely (Bahri *et al.* 2022, Hajiramezanali *et al.* 2022, Wang and Sun 2022). The difficulty lies in identifying how the examples are being put together or separated, i.e. choosing the adapted loss function. The current popular approach is to use the InfoNCE loss introduced by van den Oord *et al.* (2019).
- Hybrid Learning: This approach consists in using parts of predictive and/or contrastive learning or combines both. It tends to integrate the advantages of both strategies (Somepalli *et al.* 2021).

2.2 Evaluated approaches methods

We chose three widely used and effective self-supervised learning methods tailored for tabular data, each representing one of the main categories of self-supervised learning: contrastive, generative, and hybrid approaches. The first one is SCARF (Bahri *et al.* 2022), an adaptation for tabular data of SimCLR (Chen *et al.* 2020), which is a well-known self-supervised learning method used in computer vision. The second method is VIME (Yoon *et al.* 2020), a modified version of a denoising auto-encoder (Vincent *et al.* 2008) designed for tabular data. The last is BYOL (Grill *et al.* 2020), which is a method initially designed for computer vision based on the principle of teacher–student network.

2.2.1 SCARF

SCARF (Bahri *et al.* 2022) is a contrastive self-supervised learning method for tabular data, described in Fig. 2a. First, considering a batch of examples of the unlabeled training data, a corrupted version $\tilde{x}^{(i)}$ is generated for each example $x^{(i)}$ by randomly selecting a fraction of its features. Each selected feature is replaced by a random draw from the continuous uniform distribution whose support is defined by the minimum and maximum values of that feature in the training set. Then, the pre-training process' objective is to train an encoder e that will define the representation space. The original example $x^{(i)}$ and the corrupted version $\tilde{x}^{(i)}$ pass through the encoder network e to produce representations $z^{(i)}$ and $\tilde{z}^{(i)}$ that will pass through a projector g to obtain projections $q^{(i)}$ and $\tilde{q}^{(i)}$, respectively. Finally, the InfoNCE contrastive loss (van den Oord *et al.* 2019) is applied to ensure that projections $q^{(i)}$ and $\tilde{q}^{(i)}$ are close, for all i, and projections $q^{(i)}$ and $\tilde{q}^{(j)}$ are far apart, for $i \neq j$:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{\exp(s_{i,i}/\tau)}{\sum_{k=1}^{N}\exp(s_{i,k}/\tau)}\right) \tag{1}$$

where $s_{i,j} = q^{(i)^\top}\tilde{q}^{(j)}/(||q^{(i)}||_2.||\tilde{q}^{(j)}||_2)$, for $i,j \in [N]$ and τ is the temperature hyper-parameter. After the pre-training, the encoder network e is kept and attached to a classification head h that takes the outputs of e as its inputs.

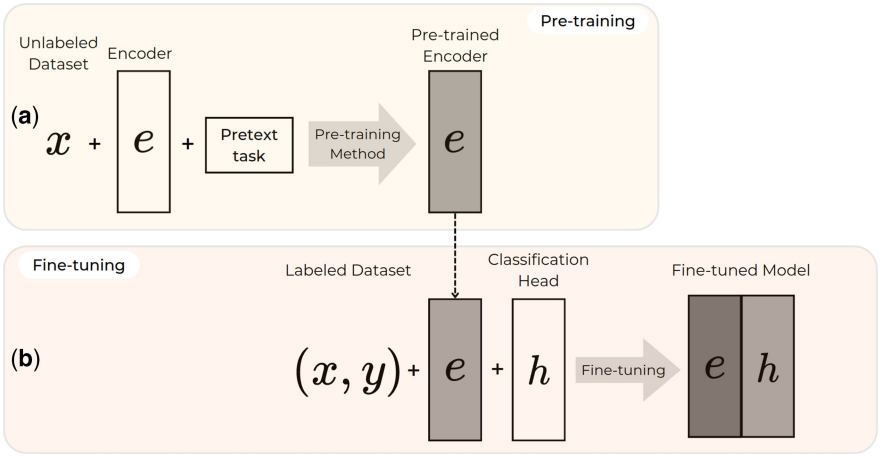
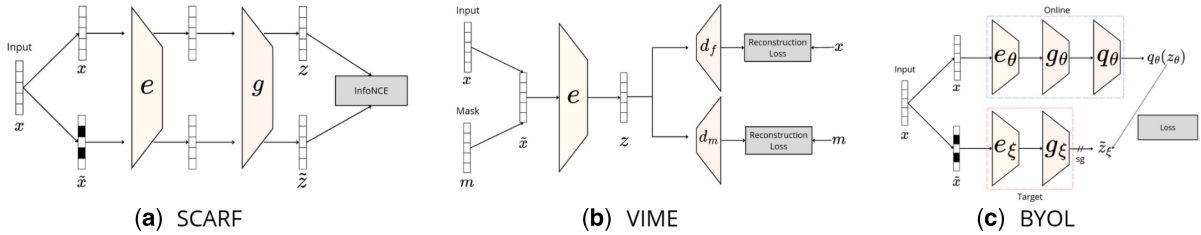


Figure 1. Self-supervised pipeline. Considering an encoder e, the first step consists in pre-training the encoder with unlabeled data (a). Once the pre-training is done, a classification head is attached to the encoder to train the network on a specific downstream task with labeled data (b). This second step is called fine-tuning.



(a) SCARF (b) VIME (c) BYOL

Figure 2. Architectures of the considered SSL pre-training approaches: (a) SCARF, (b) VIME, (c) BYOL.

2.2.2 VIME

VIME (Yoon *et al.* 2020) is a self-supervised generative autoencoder-based learning method for tabular data that relies on two pretext tasks: feature reconstruction and mask reconstruction. First, the pre-training process consists of generating a binary mask vector $m = [m_1, \cdot, m_d]^{\perp} \in \{0,1\}^d$ where $m_j \sim \mathcal{B}(p_m)$, to generate a corrupted version $\tilde{x}$ for each example x with the following transformation: $\tilde{x} = m \odot \overline{x} + (1-m) \odot x$, where the features of $\overline{x}$ are sampled from the empirical marginal distribution of each features.

The corrupted sample $\tilde{x}$ passes through an encoder e that produces a representation $\tilde{z}$, from which a decoder d_f predicts the values of the features that have been corrupted and a decoder d_m predicts which features have been masked. The total loss is a combination of a feature reconstruction loss $\mathcal{L}_f$ and a mask reconstruction loss $\mathcal{L}_m$: $\mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_f$, where $\mathcal{L}_f$ is the mean squared error loss, $\mathcal{L}_m$ is the sum of the binary cross-entropy losses for each dimension of the mask vector and α is a hyper-parameter that adjusts the tradeoff between the two losses. The pre-training process is described in Fig. 2b. After the pre-training process, the encoder part e is conserved and can be fine-tuned.

2.2.3 BYOL

BYOL (Grill *et al.* 2020) (Bootstrap Your Own Latent) is a self-supervised learning method for visual data. We have made some modifications to this method to adapt it to tabular data. BYOL is a pre-training method composed of two neural networks branches: online and target. The online branch is defined by a set of weight θ and composed of three parts: an encoder e_θ, a projector g_θ, and a predictor q_θ. The target branch is defined by a set of weight ξ and is composed of an encoder e_ξ and a projector g_ξ.

For the pre-training process, we choose to generate a corrupted version $\tilde{x}$ of an example x in the same way as VIME. Then, the online branch takes as input the original example x to produce the prediction $q_\theta(g_\theta(e_\theta(x))) = q_\theta(z_\theta)$. Similarly, the *target* branch takes as input the corrupted examples $\tilde{x}$ to produce the projection $g_\xi(e_\xi(\tilde{x})) = \tilde{z}_\xi$. The $l2-$distance between the outputs is then minimized with the following $\mathcal{L}_{\theta,\xi}$ loss:

$$\mathcal{L}_{\theta,\xi} = ||\overline{q_\theta(z_\theta)} - \overline{\tilde{z}_\xi}||_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), \tilde{z}_\xi \rangle}{||q_\theta(z_\theta)||_2 \cdot ||\tilde{z}_\xi||_2} \qquad (2)$$

We use an overline to denote normalized quantities. BYOL is considered as a hybrid approach because it learns from positive pairs only. During pre-training, at each training step, the loss $\mathcal{L}_{\theta,\xi}$ is minimized with respect to θ only, as the *target* branch is on *stop-gradient* mode. On the other hand, the weights ξ are updated by doing an exponential moving average of the online parameters θ. More precisely, at each training step and given a target decay rate $\lambda \in [0,1]$, the following update is applied: $\xi \leftarrow \lambda\xi + (1-\lambda)\theta$. The pre-training process is described in Fig. 2c. After the pre-training, only the encoder e_θ is conserved for the fine-tuning process.

2.2.4 Effects of data augmentation

The objective of applying transformations to the examples before the training process is to diversify the training features. Although these transformations do not have biological meaning, they simulate biological diversity well while preserving the label. Indeed, one risk here is that the transformations alter the examples so much that their corresponding labels change. To support this statement, we conducted an experiment which can be found in Supplementary Material (Fig. S5, available as supplementary data at *Bioinformatics* online). We assume that they do not fundamentally change the underlying information for two main reasons. First, since we consider thousands of features and we perturb only a fraction of them (30%), the transformations' effect is limited. Then, because of redundancy, mentioned in Section 2.1, the applied transformations have less effect on the examples, since the information contained in a given feature can be contained in another feature.

3 Experiments and results

3.1 Datasets

We selected two RNA-Seq datasets, described in detail in Supplementary, available as supplementary data at *Bioinformatics* online, with the preprocessing method (Section A), to conduct our experiments:

- The Cancer Genome Atlas (TCGA): this is a pan-cancer RNA-Seq dataset that contains 9349 samples and 56 902 input genes from 19 different types of cancer, with the majority class representing 12.9% of the dataset and the smallest class representing 2.8%.
- All RNA-Seq and ChIP-Seq Sample and Signature Search (ARCHS4) (Lachmann *et al.* 2018): this is a pan-tissue samples originating from experiences from SRA (Sequence Read Archive) and GEO (Gene Expression Omnibus). It is a dataset that has not yet been widely used in the field of deep learning. It is composed of diverse samples originating from experiences that do not only focus on cancer. It therefore represents a good candidate for use as pre-training dataset. It contains 53 282 samples and 67 128 input genes from 19 different tissue types, with the majority class representing 14.1% of the dataset and the smallest class representing 0.6%.

3.2 Experimental setup

3.2.1 Evaluation settings

The goal is to evaluate the effectiveness of the three selected self-supervised methods with the TCGA and ARCHS4 RNA-Seq datasets. Although self-supervised methods use unlabeled data for the pre-training process, both datasets are labeled. Thus, we divided each dataset into an unlabeled pre-training and a labeled fine-tuning set while conserving initial class proportions to simulate using unlabeled examples during the pre-training. We define three evaluation settings depending on the dataset considered. In each case, we pre-train the model with the unlabeled pre-training set and fine-tune with it with the labeled fine-tuning set on the downstream task, consisting in cancer type identification (multi-class classification):

- TCGA: We divided the dataset into a pre-training set composed of 7291 examples and a fine-tuning set and test set both composed of 1029 examples.
- ARCHS4: We divided the dataset into a pre-training set containing 50 998 examples and a fine-tuning set and test set both containing 1142 examples.
- ARCHS4 to TCGA: We pre-train the model with the pre-training ARCHS4 dataset and fine-tune it with the

fine-tuning TCGA dataset. The goal is to see if the representation learned from the ARCHS4 samples can be transferred to the TCGA downstream task. As the TCGA and ARCHS4 datasets do not feature the same set of genes, we only consider the common genes between the two datasets, representing 55 747 genes.

In each case, we take 15% of both the pre-training and fine-tuning set as validation sets. According to 3.1, a majority class classifier would achieve 12.9% of accuracy on the TCGA and ARCHS4-to-TCGA case, and 14.1% on the ARCHS4 case.

3.2.1.1 Model architecture and training

We tested many encoder architectures e by varying the number and size of hidden layers. We selected the best architecture by training on the fine-tuning training set and taking the one with the best accuracy on the fine-tuning validation set. The hyper-parameters tested are listed in Supplementary Table S4, available as supplementary data at *Bioinformatics* online. The best encoder architecture is composed of four dense layers of dimension 256 and with ReLU activation function and batch normalization.

For SCARF, the pre-training head g is a two layers ReLU network with hidden dimension 256. For VIME, we use the same architecture as the encoder e for both the feature decoder d_f and mask decoder d_m with an output dimensionality of the same size of the input dimensionality. The mean square error reconstruction loss is applied for the feature decoder d_f and the binary cross-entropy loss is applied for the mask decoder d_m. For BYOL, inspired by Grill *et al.* (2020), we choose a linear layer with output size 4096 followed by batch normalization, ReLU and a final linear layer with output dimension 256 as projectors g_θ and g_ξ and predictor q_θ. The pre-training hyper-parameters are listed in Supplementary Table S3, available as supplementary data at *Bioinformatics* online. We use the pre-training validation set for loss tracking during the pre-training step.

For the fine-tuning part, we associate the pre-trained encoder with a classification head composed of one linear layer with embedding size as input dimension and the number of classes as output dimension. We consider two cases: the unfrozen fine-tuning where the layers of the encoder and the classification are trained, and the frozen fine-tuning where we freeze the layers of the pre-trained encoder and only the parameters of the classification head are trained. Supervised fine-tuning uses early stopping with patience 30, using the loss on fine-tuning validation set, with a max number of epochs of 100 and a batch size of 8.

3.2.1.2 Benchmarking

We apply each pre-training method on the defined encoder and compare their performances, after fine-tuning, to the performances of its non-pre-trained version. We call the non-pre-trained model the baseline model. For the fine-tuning part, the training set comprises approximately 1000 examples. We perform fine-tuning with different proportions p of examples used during training, from 0.02 to 1 with a step of 0.01, to highlight the effect of fine-tuning with very few examples. To estimate the variability of the approaches, we train five models for each proportion and each method by randomly initializing the weights of the fine-tuning head in the pre-trained case and by randomly initializing the weights of the whole network for the baseline model.

3.2.1.3 Results of evaluation

The results are presented in Fig. 3. For the two first cases in which pre-training and fine-tuning examples originate from the same dataset (3-first row). We notice a clear difference in performance of the pre-trained models between the frozen and the unfrozen fine-tuning.

Indeed, in the unfrozen case, pre-trained models achieve better performance than the baseline model, especially at low

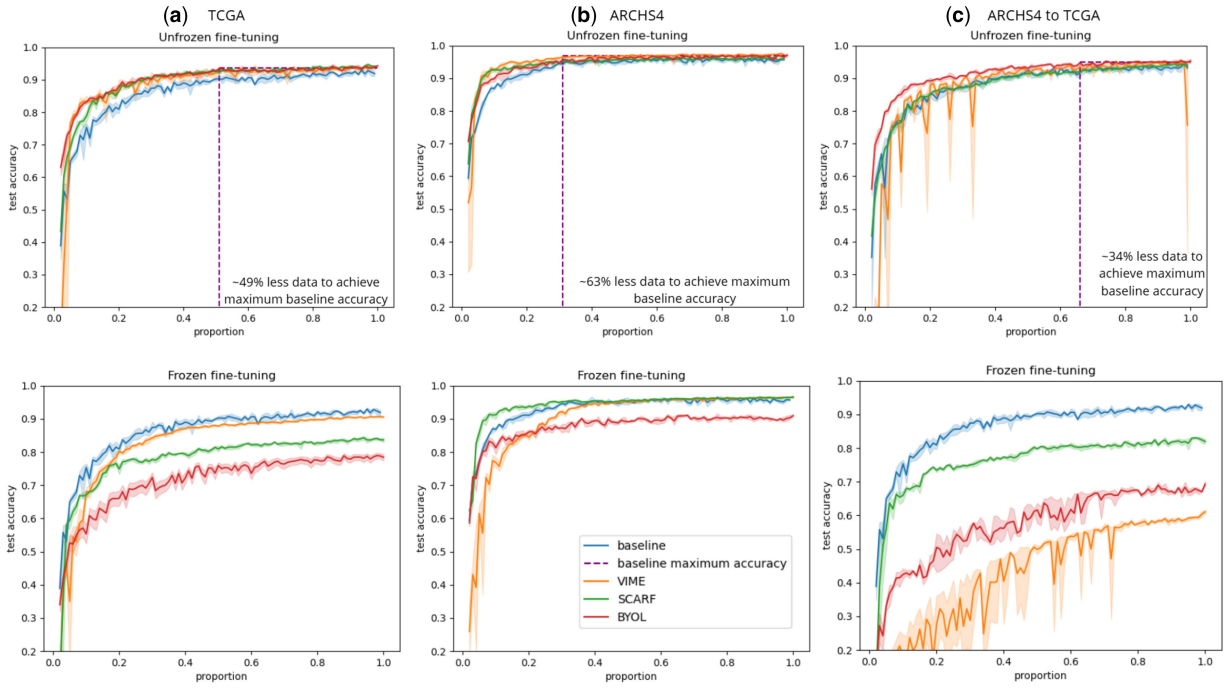


Figure 3. Performance of pre-trained models compared to non-pre-trained model (baseline) for the three different cases: (left column) TCGA, (center column) ARCHS4, and (right column) ARCHS4-to-TCGA.

fine-tuning proportion p for the three experiments. This clearly shows the benefits of self-supervised approaches to improve prediction performances. Besides, all models seem to reach the same accuracy as the proportion increases. However, they seem to reach this common point faster when considering the ARCHS4 dataset. This may be due to the fact that in the case of ARCHS4, the models were pre-trained with more examples than in the TCGA case. In the first two experiments, the three self-supervised methods return the same performance. The third experiment, where pre-training and fine-tuning data come from different datasets, is the most challenging task. We note that BYOL strongly outperforms the other methods.

In the frozen case, the performances are variable. BYOL always produces lower performances. VIME is generally worse than the baseline, but may reach the same performances with a high proportion p. These two approaches seem not to draw benefits from frozen pre-training. SCARF may be worse than the baseline or outperform the baseline, especially for low proportion p (see Fig. 3a and c). These results show that frozen self-supervised learning is possible but is clearly more complex than unfrozen self-supervised learning and requires methods specific to gene expression data.

The selected pre-training methods improve model performance when fine-tuning is done with unfrozen layers and both pre-training and fine-tuning use the same dataset. However, transferring representations from ARCHS4-to-TCGA does not yield better results than training from scratch. In all cases, frozen fine-tuning leads to poor performance, highlighting that the learned representations require adaptation to the downstream task.

By performing the fine-tuning on unfrozen mode, we saw that pre-trained models can outperform the baseline. When this is the case, pre-trained models need less examples to reach the maximum baseline accuracy. For the TCGA case (Fig. 3a) and the ARCHS4-to-TCGA case (Fig. 3c), the maximum baseline accuracy is first achieved by the model pre-trained with BYOL, saving, respectively, 49% and 34% of the TCGA training dataset. For the ARCHS4 case (Fig. 3b), the maximum baseline accuracy is achieved by the model pre-trained with VIME and permits to save approximately 63% of the ARCHS4 training set. Thus, these self-supervised methods permit to produce models that can be effective and can outperform the non-pre-trained model with less examples, which is a great advantage when we consider that annotated RNA-Seq data is time-consuming and expensive to obtain and existing RNA-Seq datasets contain few examples.

The benefits of self-supervised learning can also be demonstrated when we look at the convergence speed (evolution of accuracy as a function of training epochs) of pre-trained models during fine-tuning. Indeed, fine-tuning a pre-trained model takes less training epochs than starting from random weights. Self-supervised methods permit to build models that converges faster to a local minimum by providing a better initialization.

This observation makes us consider the pre-training step as an efficient model initialization, leading to a faster convergence and lesser data consumption in supervised training.

3.2.2 Variation of architecture's size

We want to highlight the architecture-dependent factor of the representation induced by pre-training methods mentioned in Gross *et al.* (2024). We adopt the same evaluation process as mentioned in Section 3.2.1 except that we vary the number of layers and the embedding size of the considered model. The number of layers varies from 2 to 10 and the embedding size varies in the range [256, 1024]. We then pre-train and fine-tune, in unfrozen mode, several architectures with each pre-training methods and compare each one with its corresponding not pre-trained version. To estimate the global performance gap between a pre-trained model and a its non-pre-trained version, we adopt a metric that compares the area under the performance curves (AUPC).

More precisely, we only consider the AUPC for P varying from 0.02 to 0.3, representing 20 to 300 examples, corresponding to the size of a large majority of the real datasets. The corresponding metric is:

$$g(m_1, m_2) = AUPC(m_1)_{[0.02,0.3]} - AUPC(m_2)_{[0.02,0.3]} \quad (3)$$

where m_1 and m_2 represent the accuracy curves of the two models considered. We then obtain a metric so that the model m_1 performs better on average than m_2 if $g(m_1, m_2) > 0$ and inversely if $g(m_1, m_2) < 0$. This metric estimates the performance of a pre-trained model and allows the comparison of different methods on a range of training set size.

3.2.2.1 Results of variation of architecture's size

The results are presented in Fig. 4. When we fix the embedding size at 256, we can see that the three types of pre-training are beneficial regardless of the depth of the encoder. The metric value ranges from 0.00, for VIME with two layers, to 0.07, for SCARF with 10 layers. When we increase the embedding size to 1024, as the model is deeper, the VIME pre-training becomes less effective until reaching a negative value, showing counterproductive pre-training. We can then suppose that the VIME pre-training method can be ineffective in improving models with large embedding size and can produce highly architecture-dependent representations. Concerning SCARF and BYOL, the pre-training is still beneficial with embedding size of 1024, showing that the representations induced by this method are not architecture-dependent.

For a more in-depth analysis, we analysed dimensional collapse-where the model underutilizes parts of the latent space, indicating porr representation learning and potential downstream performance issues (Jing *et al.* 2021). We examined this phenomenon across our pre-trained models for each architecture. We analyse learned representations by collecting the embedding vectors on the validation set. Following (Jing *et al.* 2021), we evaluate the dimensionality by first collecting the embedding vectors on the validation set. We then compute the covariance matrix $C \in \mathcal{M}_{d\times d}(\mathbb{R})$ of the embedding vectors: $C = \frac{1}{N}\sum_{i=1}^{N}(z_i - \overline{z})(z_i - \overline{z})^\top$, where $\overline{z} = \frac{1}{N}\sum_{i=1}^{N} z_i$ is the mean and N is the total number of samples. We then apply the singular value decomposition on this matrix ($C = USV^\top$, where $S = diag(\sigma^k)$) and dispose them in sorted order and logarithmic scale, as shown in Fig. 4. When several singular values collapse to zero, it is a sign of dimensional collapse. Figure 4 shows the singular value spectrum for each architecture.

We can see that models pre-trained with VIME show signs of dimensional collapse, especially with deeper architectures. We can suppose that this collapse is the cause of bad performance on downstream tasks as VIME's gains are unstable

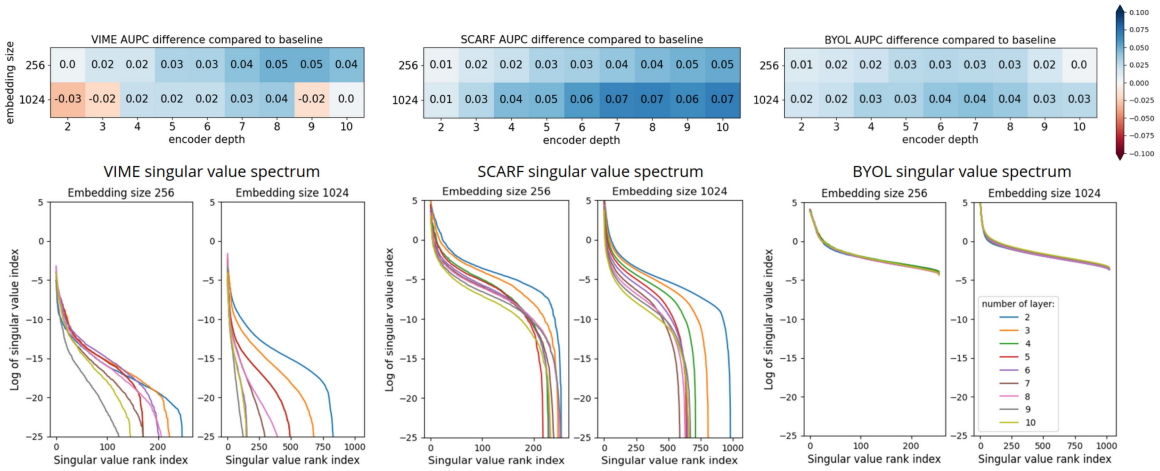


Figure 4. (Top) Average gain of different encoder architecture under the three pre-training methods compared to the associate baseline model coupled to (Down) singular value spectrum for each architecture.

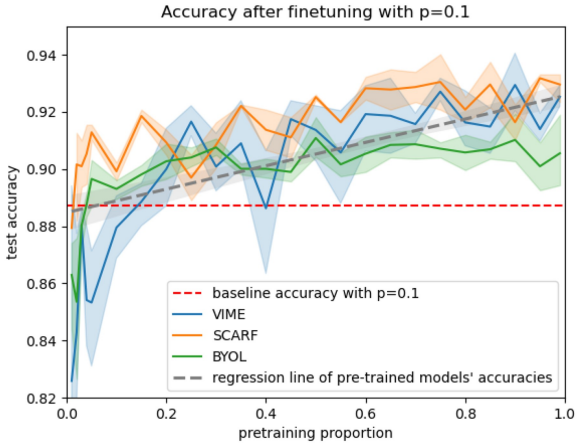


Figure 5. Accuracy of pre-trained models with different pre-training proportion and with fine-tuning proportion $P = 0.1$.

with deeper architectures. For example, with the model composed of nine layers with an embedding size of 1024, approximately only 100 singular values do not collapse to zero and it is associated to a negative performance model in downstream task. In contrast, SCARF and BYOL are more prone to dimensional collapse due to the design of the pre-training process based on the use of positive and/or negative pairs. However, we can see that models pre-trained with SCARF present a less pronounced dimensional collapse: for an embedding size of 1024 up to 50% of singular values collapse to zero and up to 20% collapse to zero for an embedding size of 256. Those pre-trained with BYOL do not present this effect regardless of the number of layer and the embedding size. This confirms the fact that the design of BYOL pre-training is robust to dimensional collapse which is a key point of good performances in our case. Interestingly, SCARF shows a relative effect of dimensional collapse but presents good performances on downstream task.

To conclude, this analysis shows that BYOL permits to build representations that use all the available dimensions, regardless of the architecture, whereas it is not the case for VIME and SCARF. However, dimensional collapse is not a sign of bad representation learning in our case, as even collapsed models can outperform baselines on downstream task.

3.2.3 Variation of number of samples in pre-training set

We also aim to observe the effect of the pre-training set size. For that, we use the ARCHS4 datasets with the initial architecture considered in Section 3.2.1. We vary the number of examples used during pre-training by increasing the proportion q of the pre-training set used from 0.01 to 0.05 with a step of 0.01, then from 0.05 to 1 with a step of 0.05. We train the model for each pre-training proportion q used and for each pre-training method and compare its performance to the non-pre-trained version. For the comparison between two methods, we choose to consider the performance of each model with proportion $P = 0.1$ used in fine-tuning. For this experiment, we set the fine-tuning part on unfrozen mode. The results are presented in Fig. 5.

3.2.3.1 Results on the pre-training set size

The results presented in Fig. 5 show the effect of the size of the pre-training set on the accuracy of the models with a fine-tuning proportion p fixed to 0.1. We observe that the pre-training is more effective as the pre-training proportion increases and can be already effective with a few examples in the pre-training set. More precisely, the SCARF and BYOL pre-training methods provide increasing gains from $q = 0.05$ compared to the baseline, representing approximately 2500 examples. The same goes for VIME which provides increasing gains from $q = 0.2$. We can also see that pre-trained models achieve lower performances than the baseline when the pre-training proportion is small, showing that pre-training can be counterproductive when using an insufficient amount of example during this phase. Note that for BYOL and SCARF, this counterproductive effect occurs at very low pre-training proportion, representing approximately 2000 examples. In contrast, for VIME the effect occurs when there are fewer than 10 000 examples in the pre-training set.

4 Discussion

Results show that self-supervised methods can enhance neural network performance on gene expression data, but only under certain conditions. Indeed, first the fine-tuning process has to be done with unfrozen layers, otherwise the performance of the pre-trained models does not outperform the baseline model performance. It shows that the

representations induced by the three selected pre-training methods cannot be used as they are and must be slightly modified through an unfrozen fine-tuning. Pre-training is most effective when datasets are homogeneous. If this case, the pre-trained models outperform the baseline model, especially at low fine-tuning proportion p. This is a promising result concerning pre-training applied to gene expression data because a low fine-tuning proportion corresponds to a clinical application case for which only a few data are available (<1000 examples). If pre-training and fine-tuning data differ (e.g. due to batch effects), only the BYOL method is effective. SCARF and VIME fail to outperform the baseline in case of dataset heterogeneity even if we ensure that the two datasets are processed in the same way and include the same classes. BYOL is the only method that provides better performances when the pre-training and the fine-tuning set originates from different datasets, showing that the pre-training added qualitative value to the representations. The lack of improvements in this case could be due to the misalignment of the data classes. Indeed, even if the two different datasets follow the same distribution by applying the batch effect correction, those of the TCGA classes are not aligned with the distributions of the ARCHS4 classes.

In addition, the representations induced by the pre-training methods can be highly architecture-dependent. Thus, choosing the right architecture is key to avoid counterproductive pre-training, as seen for VIME with high-layered models. However, SCARF and BYOL appear less architecture-dependent as they only improve the baseline model by improving the performance with a few examples in fine-tuning and no counterproductive cases were encountered.

Finally, the dataset used for the pre-training does not need to contain many examples for this step to be effective. Indeed, we observed improvements for models pre-trained with only a few examples (~ 9000). However, for these experiments, we considered the case where the pre-training and fine-tuning examples come from the ARCHS4 dataset, which can be considered as the most straightforward case because the ARCHS4 dataset contains only 10 classes for the downstream task. Since it can be considered as an easy downstream task, a short pre-training with a few examples can easily improve the model.

5 Conclusion

This study explores the use of self-supervised learning methods on gene expression data for phenotype prediction. We showed self-supervised learning creates high-quality representations from unlabeled data. Self-supervised learning is a cornerstone for the construction of foundation models for gene expression that may significantly improve the performance of classical supervised learning models. The three selected methods succeeded in this task, particularly when the fine-tuning is made with very few data, which is close to a real application case. Thus, these methods reduce reliance on expensive, time-consuming labeled data in the domain of gene expression data.

Besides, to address variability in SSL effectiveness, we provide recommendations for using these methods to observe gains compared to the baseline. Indeed, the fine-tuning should be performed on unfrozen mode. Then, if the pre-training and fine-tuning sets are homogeneous, we recommend using the SCARF method because its pre-training phase lasts less time than that of BYOL. If the sets are not homogeneous, we recommend using BYOL because it is the only method that outperforms the baseline in this case. Practitioners should be cautious with VIME, as its pre-training is longer and the results in fine-tuning are very unstable compared to the two other methods.

Although the results are promising, some research challenges and prospects remain, e.g. the batch effect correction (heterogeneity of pre-training and fine-tuning data) or the design of efficient pretext tasks. We focus our study on bulk gene expression data, but as omics data share similarities, this work should be applied to related data (genomics, proteomics, etc). We therefore hope that this work will encourage future research in the development of self-supervised approaches specific to omics data.

Acknowledgements

The results shown here are in part based upon data generated by the TCGA Research Network, publicly available here: https://www.cancer.gov/tcga. We are grateful to the Institut Français de Bioinformatique for providing computing and storage resources.

Author contributions

Kevin Dradjat (Conceptualization [equal], Data curation [lead], Investigation [lead], Methodology [equal], Project administration [equal], Software [lead], Visualization [equal], Writing—original draft [lead], Writing—review & editing [lead]), Massinissa Hamidi (Conceptualization [equal], Investigation [equal], Methodology [equal], Project administration [equal], Supervision [equal], Validation [equal], Writing—original draft [supporting], Writing—review & editing [supporting]), Pierre Bartet (Conceptualization [equal], Investigation [equal], Methodology [equal], Project administration [equal], Software [supporting], Supervision [supporting], Validation [equal], Visualization [supporting]), and Blaise Hanczar (Conceptualization [lead], Data curation [equal], Funding acquisition [lead], Investigation [equal], Methodology [equal], Project administration [equal], Resources [lead], Software [equal], Supervision [lead], Validation [equal], Writing—original draft [supporting], Writing—review & editing [supporting])

Supplementary data

Supplementary data is available at *Bioinformatics* online.

Conflict of interest: No competing interest is declared.

Funding

This work has been supported by the Paris Île-de-France Région in the framework of Domaine de Recherche et d'Innovation Majeur - Artificial Intelligence for Ile-de-France (DIM AI4IDF) and by a collaboration with ADLIN Science.

References

Alharbi F, Vakanski A. Machine learning methods for cancer classification using gene expression data: a review. *Bioengineering* 2023; **10**:173.

Azizi S, Culp L, Freyberg J *et al.* Robust and data-efficient generalization of self-supervised machine learning for diagnostic imaging. *Nat Biomed Eng* 2023;**7**:756–79.

Bahri D, Jiang H, Tay Y *et al.* SCARF: self-supervised contrastive learning using random feature corruption. In: *Proceedings of International Conference on Learning Representations*. Openreview.net. 2022.

Chen T, Kornblith S, Norouzi M *et al.* A simple framework for contrastive learning of visual representations. In: *Proceedings of the 37th International Conference on Machine Learning*, Vol. 119. PMLR, 2020, 1597–607.

Dinh T, Zeng Y, Zhang R *et al.* LIFT: language-interfaced fine-tuning for non-language machine learning tasks. In: *Proceedings of International Conference on Neural Information Processing Systems*, Vol. **855**. New Orleans: Curran Associates Inc, 2022, 11763–84.

Ericsson L, Gouk H, Loy CC *et al.* Self-supervised representation learning: introduction, advances, and challenges. *IEEE Signal Process Mag* 2022;**39**:42–62.

Grill J-B, Strub F, Altché F *et al.* Bootstrap your own latent - a new approach to self-supervised learning. In: *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., Online, 2020, 21271–84.

Gross B, Dauvin A, Cabeli V *et al.* Robust evaluation of deep learning-based representation methods for survival and gene essentiality prediction on bulk RNA-seq data. *Sci Rep* 2024;**14**:17064.

Gui J, Chen T, Zhang J *et al.* A survey on self-supervised learning: algorithms, applications, and future trends. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*. USA: IEEE Computer Society, 2024, 6.

Hajiramezanali E, Diamant NL, Scalia G. STab: Self-supervised learning for tabular data. In: *Proceedings of International Conference on Neural Information Processing Systems*. New Orleans: Curran Associates Inc., 2022.

Hanczar B, Bourgeais V, Zehraoui F. Assessment of deep learning and transfer learning for cancer prediction based on gene expression data. *BMC Bioinformatics* 2022;**23**:262.

Hsu T-C, Lin C. Learning from small medical data—robust semi-supervised cancer prognosis classifier with Bayesian variational autoencoder. *Bioinform Adv* 2023;**3**:vbac100.

Huang X, Khetan A, Cvitkovic M *et al.* TabTransformer: tabular data modeling using contextual embeddings. arXiv preprint, arXiv:2012.06678, 2020, preprint: not peer reviewed.

Jing L, Vincent P, LeCun Y *et al.* Understanding dimensional collapse in contrastive self-supervised learning. arXiv preprint, arXiv:2110.09348, 2021, preprint: not peer reviewed.

Lachmann A, Torre D, Keenan AB *et al.* Massive mining of publicly available RNA-seq data from human and mouse. *Nat Commun* 2018;**9**:1366.

Nam J, Tack J, Lee K *et al.* STUNT: few-shot tabular learning with self-generated tasks from unlabeled tables. In: *International Conference on Learning Representations*. Openreview.net. 2023.

Qin D. Next-generation sequencing and its clinical application. *Cancer Biol Med* 2019;**16**:4–10.

Richter T, Bahrami M, Xia Y *et al.* Delineating the effective use of self-supervised learning in single-cell genomics. *Nat Mach Intell* 2024;**7**:68–78.

Somepalli G, Goldbium M, Schwarchild A *et al.* Saint: improved neural networks for tabular data via row attention and contrastive pre-training. arXiv preprint, arXiv:2106.01342, 2021, preprint: not peer reviewed.

Tran KA, Kondrashova O, Bradley A *et al.* Deep learning in cancer diagnosis, prognosis and treatment selection. *Genome Med* 2021;**13**:152.

van den Oord A, Li Y, Vinyals O. Representation learning with contrastive predictive coding. arXiv preprint, arXiv:1807.03748, 2019, preprint: not peer reviewed.

Vincent P, Larochelle H, Bengjo Y *et al.* Extracting and composing robust features with denoising autoencoders. In: *Proceedings of the 25th International Conference on Machine Learning*. New York, NY, USA: Association for Computing Machinery, 2008, 1096–103.

Wang W-Y, Du W-W, Xu D *et al.* A survey on self-supervised learning for non-sequential tabular data. *Mach Learn* 2024;**114**:19.

Wang Z, Sun J. TransTab: learning transferable tabular transformers across tables. In: *Proceedings of International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2022.

Yoon J, Zhang Y, Jordon J *et al.* VIME: extending the success of self- and semi-supervised learning to tabular domain. In: *Proceedings of International Conference on Neural Information Processing Systems*, Vol. 33. Red Hook, NY, USA: Curran Associates Inc., 2020, 11033–43.

Bioinformatics, 2025, 41, 1–9
https://doi.org/10.1093/bioinformatics/btaf533
Original Paper