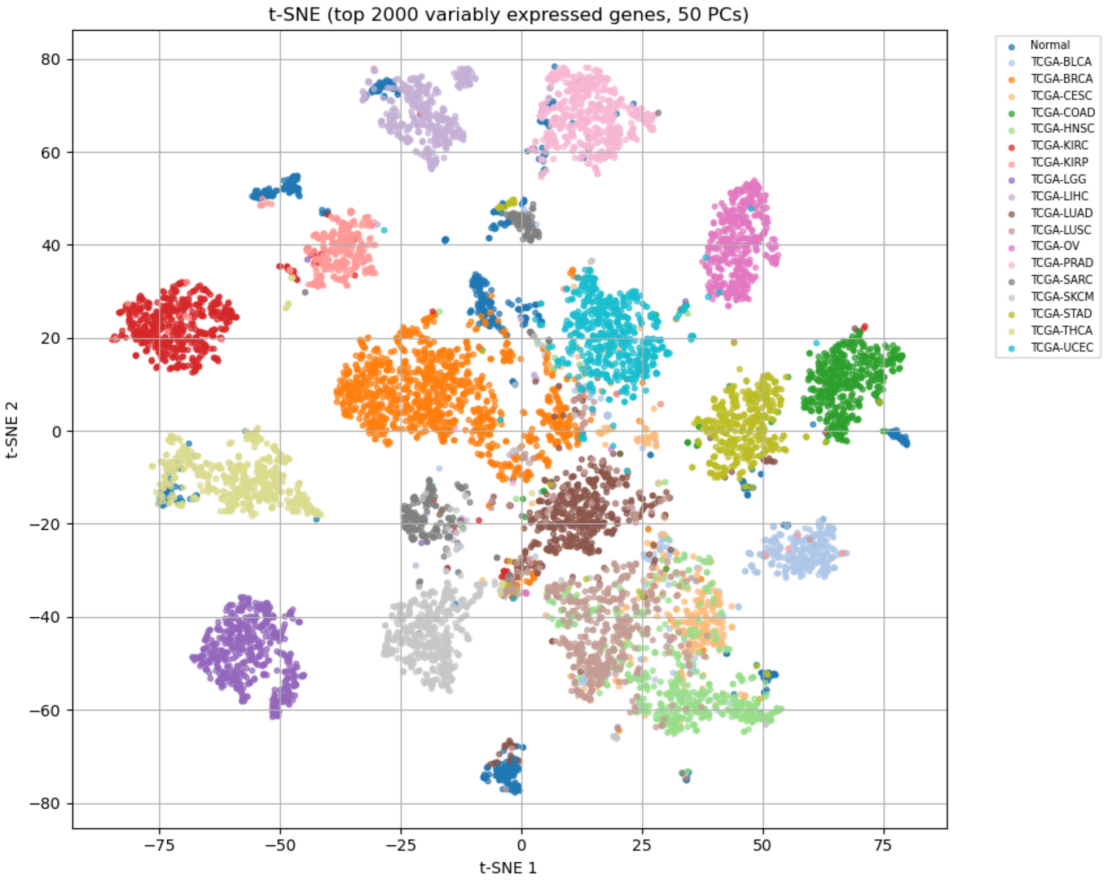
Self-Supervised Learning for Gene Expression Data

Practical Report

Year 2025–2026



Master 2 GENIOMHE

Université d'Evry Paris-Saclay

Student: Svetlana Sannikova

Course: Artificial intelligence and deep learning

Abstract

Gene expression data are characterized by extremely high dimensionality and a limited availability of labeled samples, which makes purely supervised learning approaches difficult to apply effectively. In this project, we investigate the use of self-supervised learning (SSL) for learning meaningful representations of gene expression profiles without relying on class labels. A baseline supervised multilayer perceptron (MLP) classifier is compared with an SSL-based approach that leverages biologically motivated pretraining objectives. The pretrained encoder is subsequently fine-tuned for cancer type classification using varying proportions of labeled data. Experimental results show that self-supervised pretraining leads to improved classification performance, particularly in low-data regimes, demonstrating the potential of SSL for transcriptomic data analysis.

1 Introduction

Gene expression analysis plays a central role in modern bioinformatics and biomedical research. Advances in high-throughput RNA sequencing (RNA-seq) technologies enable the simultaneous measurement of expression levels for tens of thousands of genes across large collections of biological samples. One important downstream task is the classification of samples into clinically or biologically meaningful categories, such as cancer types, based on their gene expression profiles.

Despite the richness of transcriptomic data, machine learning models face several fundamental challenges in this domain. First, gene expression datasets are inherently high-dimensional, typically containing around 20,000 gene features per sample. Second, the number of labeled samples is often limited, as biological annotation and clinical curation are expensive and time-consuming processes. As a consequence, conventional supervised learning models are prone to overfitting and may exhibit poor generalization performance.

Self-supervised learning has recently emerged as a promising paradigm for representation learning in settings where labeled data are scarce but unlabeled data are abundant. Instead of relying on externally provided annotations, SSL methods define auxiliary pretext tasks that encourage models to learn informative and transferable representations directly from the structure of the data. In this project, we explore the application of self-supervised learning techniques to gene expression data and evaluate their impact on downstream cancer type classification.

2 Objectives

The main objectives of this project are the following:

- To construct a fully supervised baseline model for cancer type classification using gene expression profiles.
- To design and implement a self-supervised learning framework for pretraining a neural network encoder on unlabeled gene expression data.
- To fine-tune the pretrained encoder on labeled data for the task of cancer classification.
- To systematically compare the performance of the baseline and SSL-based models across different proportions of labeled training data.
- To analyze the benefits and limitations of self-supervised learning in the context of high-dimensional transcriptomic data.

3 Datasets

All data used in this project are derived from The Cancer Genome Atlas (TCGA) project, which provides large-scale, publicly available molecular profiling data for human cancers [1]. TCGA

contains transcriptomic (RNA-seq) measurements for tens of thousands of tumor and normal samples spanning multiple cancer types.

The original TCGA gene expression dataset consists of more than 20,000 samples, each represented by expression values for approximately 20,000 genes. In this project, a subset of TCGA RNA-seq data was selected and further processed to construct datasets suitable for self-supervised pretraining and supervised classification.

All datasets are represented as tabular matrices in which rows correspond to biological samples and columns correspond to gene expression features. For supervised datasets, an additional column specifying the cancer type (`cancer_type`) is included.

3.1 Pretraining Dataset

The pretraining dataset was constructed to support self-supervised representation learning. It consists of unlabeled gene expression samples and is used exclusively during the self-supervised pretraining stage.

This dataset contains 7,349 samples, each represented by 19,887 gene expression features. No class labels are used during this stage. The primary objective of the pretraining dataset is to expose the model to a large and diverse collection of transcriptomic profiles, allowing it to learn general structure, correlations, and biological patterns present in gene expression data without relying on external annotations.

3.2 Fine-tuning Dataset

The fine-tuning dataset is a labeled subset of the data used to adapt the pretrained encoder to a supervised cancer classification task. It contains 1,000 samples, each represented by the same 19,887 gene expression features as the pretraining dataset, along with an additional column specifying the target label (`cancer_type`).

During fine-tuning, these labeled samples are used to train a classifier on top of the pretrained encoder. Different proportions of this dataset are later used to study model performance under varying levels of label availability.

3.3 Test Dataset

The test dataset is an independent labeled dataset reserved exclusively for evaluation. It also contains 1,000 samples, each represented by 19,887 gene expression features and corresponding `cancer_type` labels.

The test dataset is not used during pretraining, fine-tuning, or model selection. Its sole purpose is to provide an unbiased estimate of the generalization performance of both baseline and self-supervised models.

3.4 Data Split Strategy

Starting from the original TCGA-derived gene expression data, the samples were explicitly divided into three non-overlapping subsets corresponding to pretraining, fine-tuning, and testing.

Approximately 7,000 samples were allocated to the self-supervised pretraining dataset, while 2,000 samples were reserved for supervised learning and evaluation. The labeled portion was split evenly between fine-tuning (1,000 samples) and testing (1,000 samples).

This data split reflects a realistic and common scenario in biomedical machine learning, where large amounts of unlabeled molecular data are available, but high-quality labeled samples are limited due to cost and experimental constraints.

4 Exploratory Data Analysis and Visualization

4.1 Class Distribution

Before training, the class distribution in the labeled data was inspected to understand potential imbalance. Figure 1 shows the number of samples per cancer type. The distribution is not uniform: some classes contain substantially more samples than others, which can bias supervised training if not handled properly. For this reason, later supervised experiments used **stratified sampling** so that each subsample approximately preserves the original class proportions.

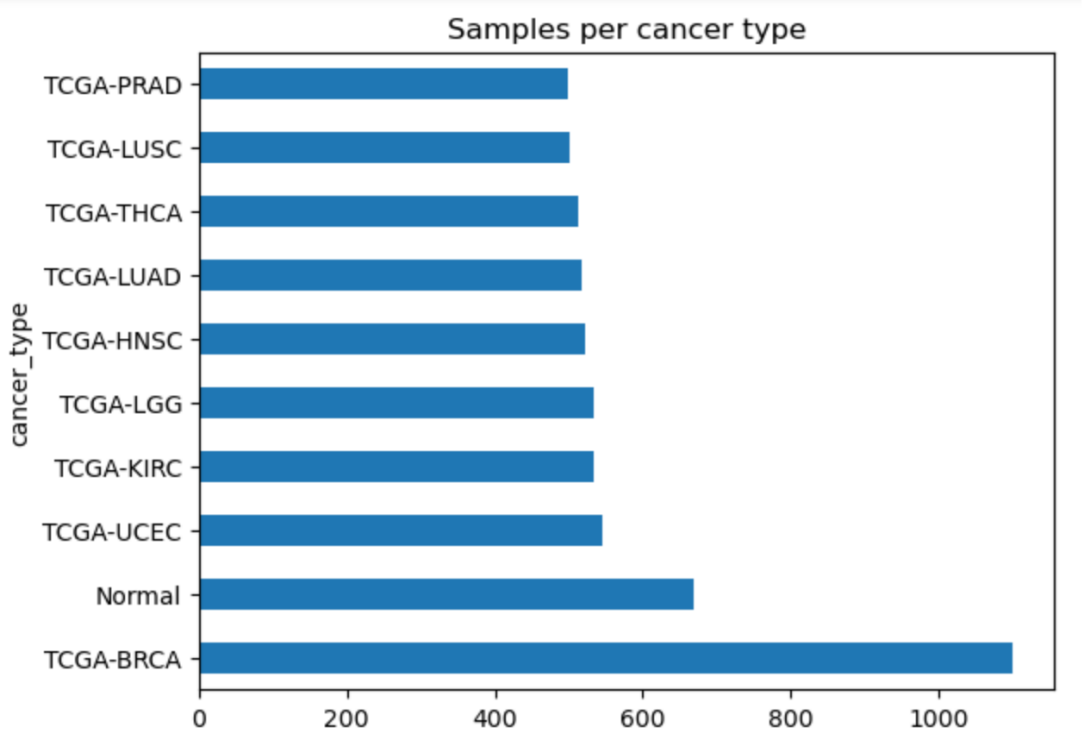


Figure 1: Number of labeled samples per cancer type (`cancer_type`). This visualization was used to identify class imbalance and motivate stratified sampling in supervised experiments.

Additionally, for some comparisons the labels were grouped into a **binary** setting (Normal vs Tumour). Figure 2 shows the corresponding counts, highlighting a strong dominance of tumour samples. This simplified view was used only for exploratory inspection and does not replace the multi-class setting used in the main classification experiments.

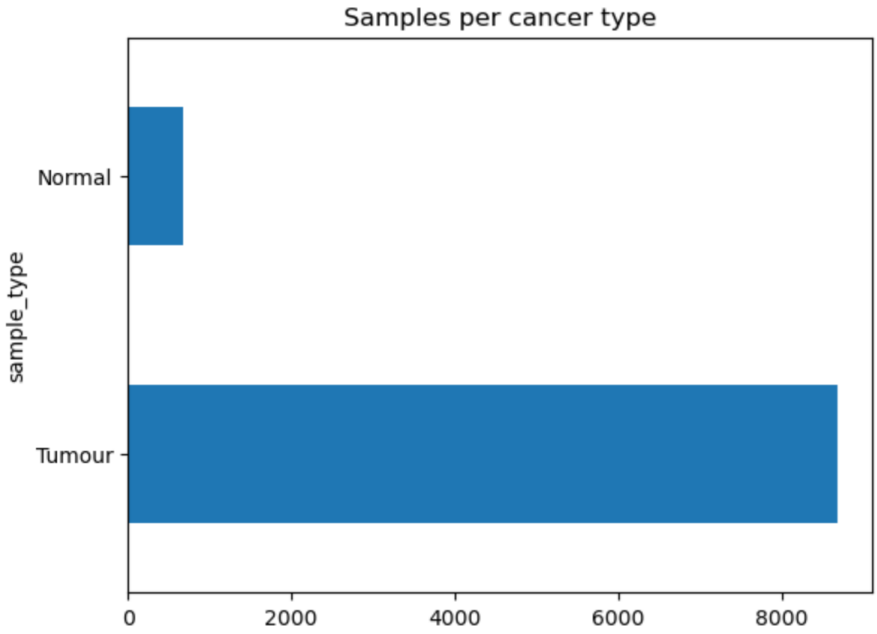


Figure 2: Binary grouping of labeled samples into Normal vs Tumour for exploratory inspection. The imbalance indicates that accuracy alone may be optimistic in binary settings; the main experiments therefore use the original multi-class labels.

4.2 Motivation for Dimensionality Reduction

Gene expression matrices are extremely high-dimensional: here each sample has 19887 gene features. Direct inspection in the original feature space is not feasible. Dimensionality reduction methods allow:

- assessing whether biological structure is visible in the data;
- checking whether samples form clusters related to cancer types;
- detecting outliers and potential batch effects;
- motivating representation learning (e.g., SSL) by demonstrating the complexity of the space.

4.3 Selecting Highly Variable Genes (HVGs)

Rather than using all genes, principal component analysis (PCA) was performed on a subset of genes with the highest dispersion (variance). This is a standard step in transcriptomic analysis because:

1. Many genes show very low variation across samples and contribute mostly noise.
2. PCA explicitly maximizes variance; including thousands of near-constant genes can dilute biologically meaningful variance.
3. Restricting to the top 500–2000 most variable genes typically produces cleaner, more interpretable projections while also reducing computational cost.

In this work, PCA was computed for multiple HVG set sizes (top 500, top 1000, and top 2000 genes) to check whether the observed structure is stable across choices.

4.4 PCA Results

Figures 3 - 5 show the PCA projections for different numbers of top-variable genes. Across all settings, several consistent patterns are observed:

- **Well-separated groups:** certain cancer types form clearly distinct regions, indicating strong tissue-specific transcriptional signatures.
- **Overlapping central cloud:** many tumour types overlap substantially in the center of the PCA space, suggesting shared transcriptional programs (e.g., proliferation, immune infiltration, stromal components).
- **Tissue-driven branches:** several cancer types appear as elongated directions in PCA space, consistent with strong tissue-specific gene expression gradients.
- **Within-class heterogeneity:** some cancer types show broad dispersion, potentially reflecting molecular subtypes, varying tumour purity, immune content, or batch effects.

Importantly, increasing the number of genes from 500 to 2000 does not qualitatively change the global structure, suggesting that the projection is relatively stable and not an artifact of a specific HVG threshold.

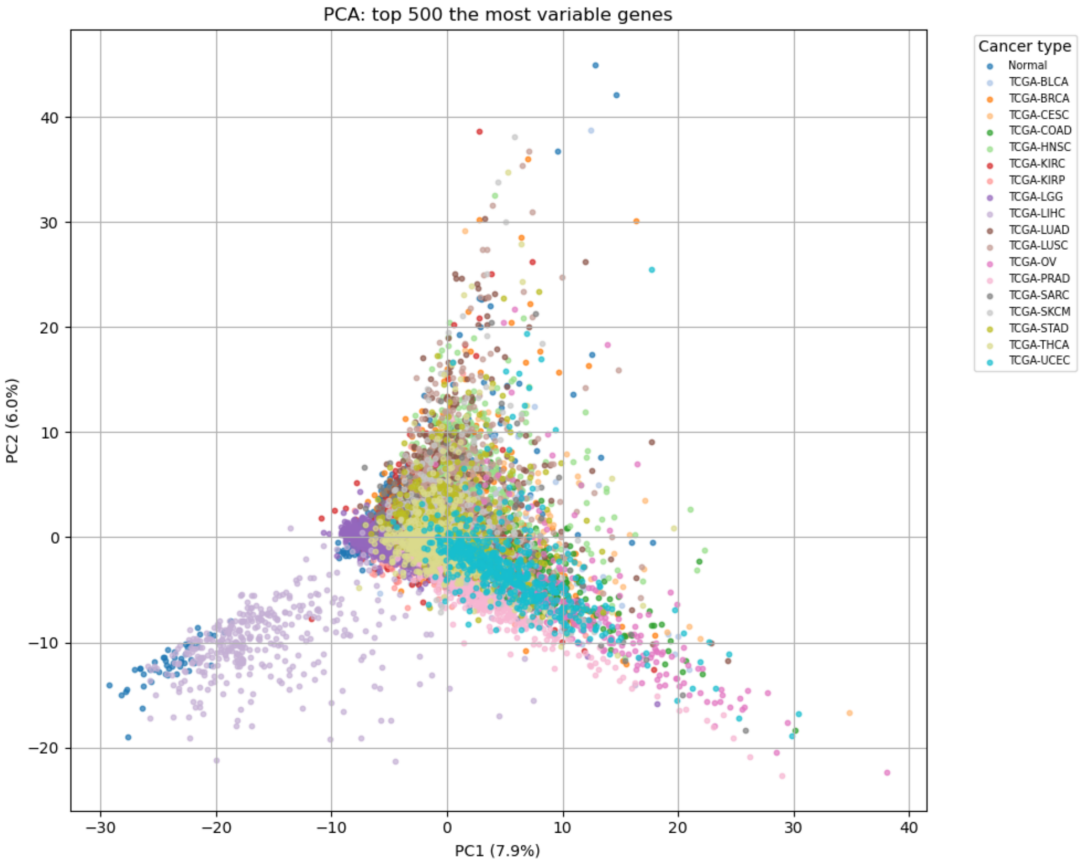


Figure 3: PCA projection using the top 500 most variable genes. Each point is a sample; colors correspond to cancer types.

Figure 4: PCA projection using the top 1000 most variable genes. Cluster structure remains similar to the top 500 case, indicating stability.

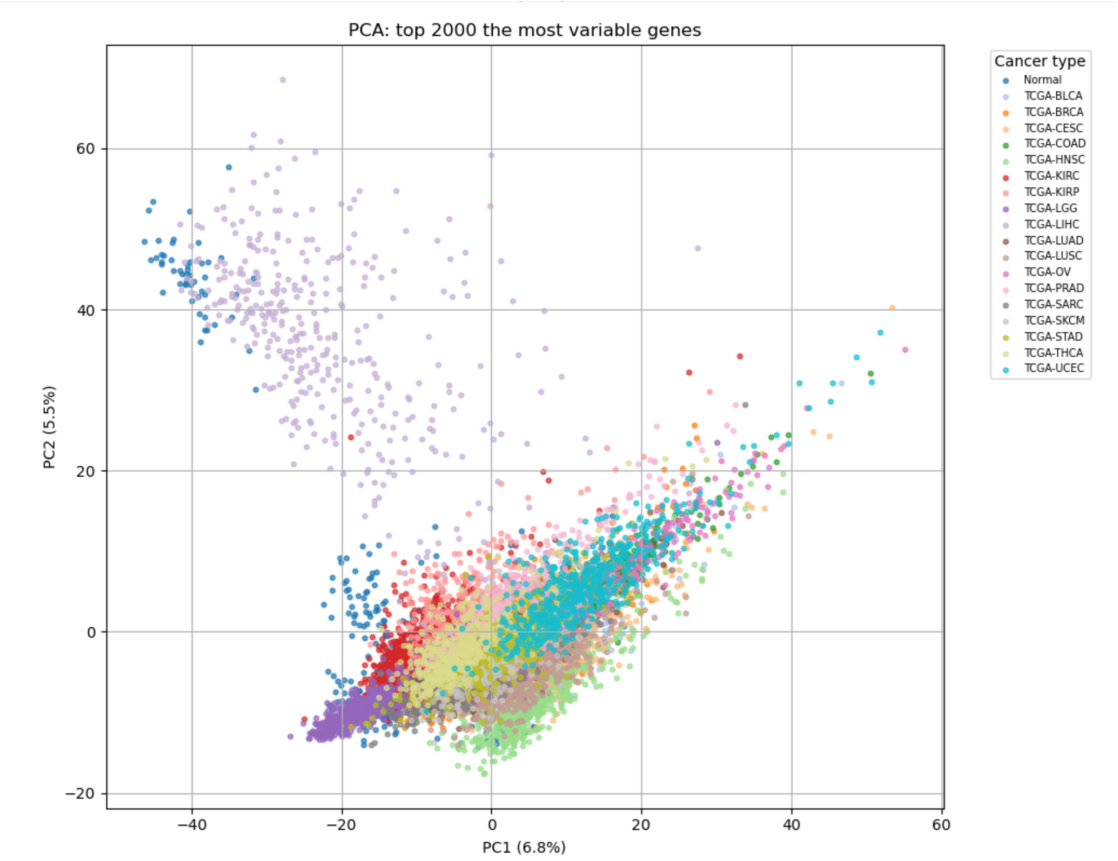


Figure 5: PCA projection using the top 2000 most variable genes. The overall geometry remains consistent; adding more genes does not drastically reshape clusters.

4.5 Explained Variance of PCA Components

Although PCA reveals visually meaningful structure, the proportion of variance explained by the first components is modest. Figure 6 shows the cumulative explained variance of the first 10 components (computed on the HVG subset). In this dataset, 10 components explain only about 0.33 (33%) of the variance, and the curve increases gradually without reaching a clear plateau.

This behaviour is expected for bulk RNA-seq gene expression data:

- Biological variation is distributed across many genes and pathways.
- Technical factors (sequencing depth, batch effects) also contribute variance.
- As a result, the intrinsic dimensionality is high, and no small number of linear components can fully summarize the dataset.

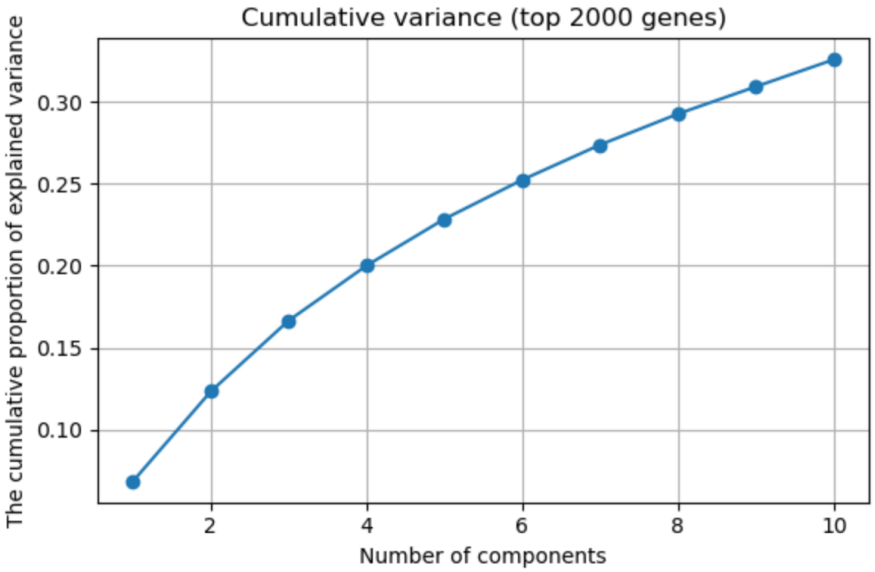


Figure 6: Cumulative explained variance of PCA (top-variable genes). The absence of a plateau indicates high-dimensional structure typical of transcriptomic data.

4.6 Nonlinear Embeddings: t-SNE and UMAP

To complement PCA, nonlinear embeddings were computed using t-SNE and UMAP on a reduced representation (HVGs + a fixed number of PCs). These methods often reveal local neighbourhood structure and cluster separations that linear PCA cannot.

Figure 7 shows the t-SNE embedding. Many cancer types form compact clusters, suggesting that samples from the same tumour type share common expression patterns. However, t-SNE emphasizes local structure and distances between clusters should not be interpreted quantitatively.

Figure 8 shows the UMAP embedding. Compared to t-SNE, UMAP typically preserves more of the global topology; in this case, it produces separated groups and also indicates relationships between cancer types through cluster proximity and bridging trajectories.

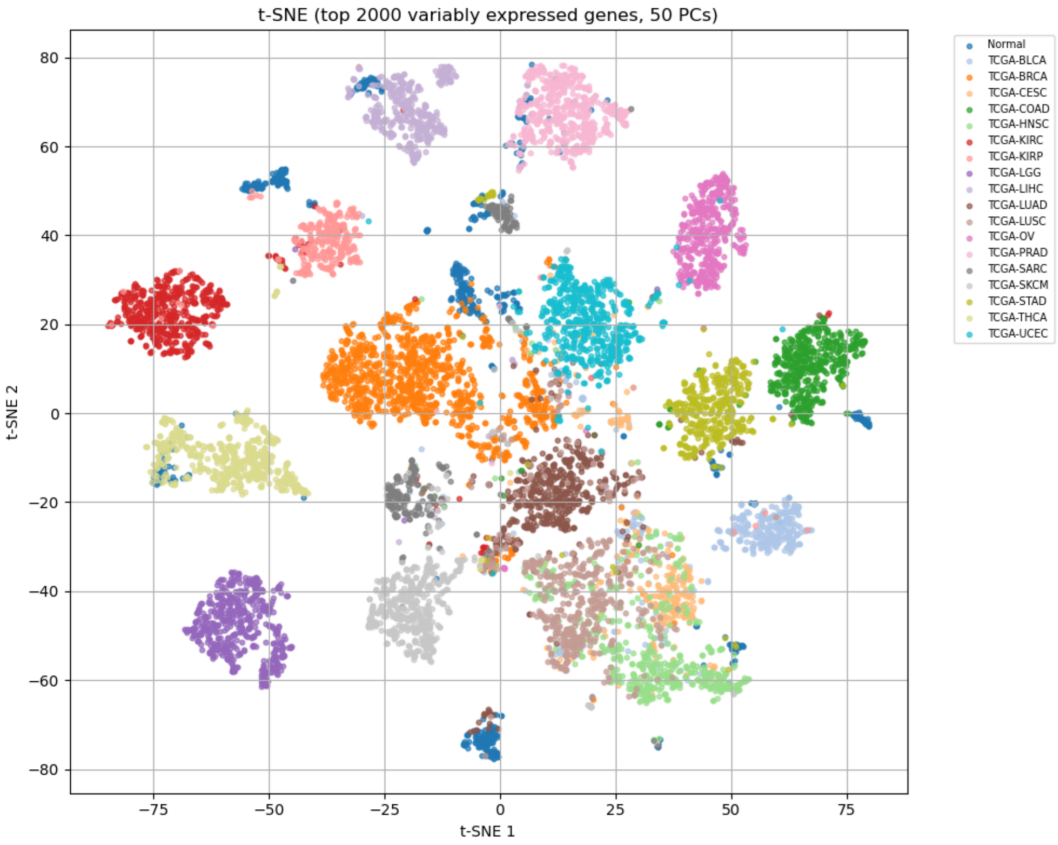


Figure 7: t-SNE embedding computed using top-variable genes and a PCA-reduced representation. Clusters correspond broadly to cancer types, indicating strong biological signal.

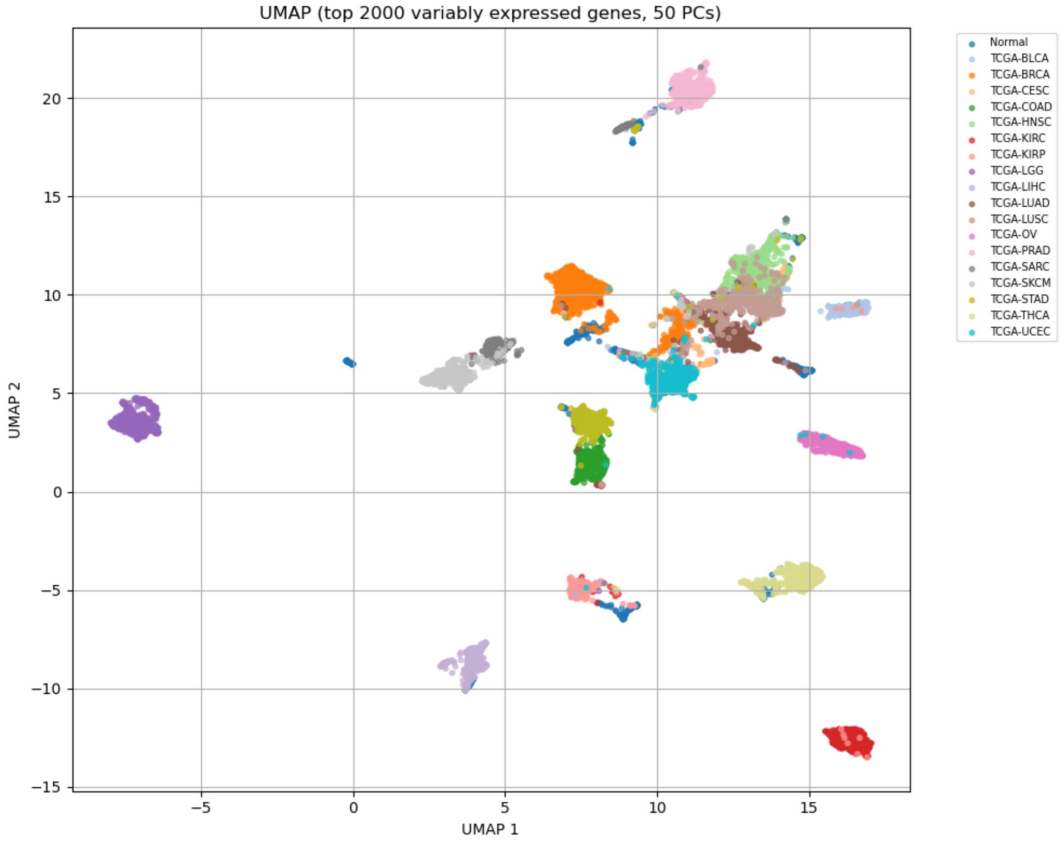


Figure 8: UMAP embedding computed using top-variable genes and a PCA-reduced representation. Clusters are well separated with some visible inter-cluster relationships.

4.7 Unsupervised Clustering on UMAP

Finally, k-means clustering was applied in the UMAP space to explore whether unsupervised clusters align with biological labels. Figure 9 shows the same UMAP embedding colored by k-means cluster assignments. Several clusters correspond closely to single cancer types, while others mix multiple types, consistent with the overlap observed in PCA. This analysis provides additional evidence that:

1. gene expression contains strong tumour-type signal;
2. but the structure is complex and not perfectly separable;
3. representation learning (SSL) is a relevant approach to capture informative features beyond simple linear projections.

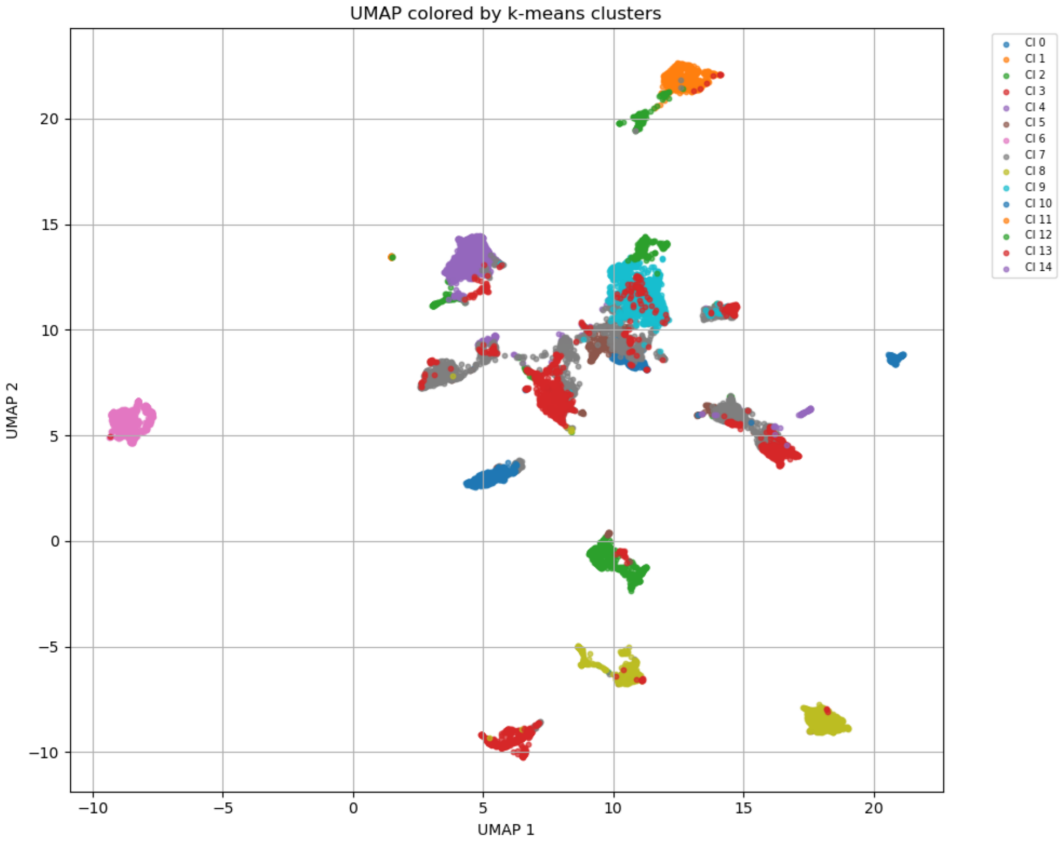


Figure 9: UMAP embedding colored by k-means clusters. Some clusters align strongly with cancer types, while others contain mixed tumour types, reflecting shared transcriptional programs and heterogeneity.

4.8 Summary of the Visualization Stage

The visualization and exploratory analysis lead to three key conclusions that motivated the later stages of the project:

1. **Strong biological signal:** samples cluster by cancer type in PCA/t-SNE/UMAP, indicating that transcriptomic profiles are informative for classification.
2. **High-dimensional complexity:** modest explained variance and broad overlap between several tumour types suggest that meaningful patterns are distributed across many dimensions.
3. **Motivation for SSL:** the combination of limited labeled data, large unlabeled data, and complex structure supports using self-supervised objectives to learn a transferable encoder before supervised fine-tuning.

5 Baseline Supervised Model

5.1 Model Architecture

As a baseline approach, a fully supervised multilayer perceptron (MLP) classifier was implemented. The model takes gene expression vectors as input and produces predictions of cancer types.

The network architecture consists of:

- an input layer corresponding to the gene expression features;

- two fully connected hidden layers with ReLU activation functions;
- an output layer with a number of units equal to the number of cancer classes.

This baseline model does not employ any form of pretraining and relies exclusively on labeled data to learn discriminative representations. As such, it serves as a reference point for evaluating the benefits of self-supervised pretraining in subsequent experiments.

5.2 Training Procedure

The baseline MLP model was trained using varying proportions of the labeled fine-tuning dataset. The size of the training subset was gradually increased from 100 to 1,000 samples in order to assess the effect of labeled data availability on classification performance.

For each training subset size, multiple independent training runs were performed using stratified sampling to preserve the original class distribution across cancer types. This strategy ensures that performance differences are not driven by class imbalance effects.

Model performance was evaluated using classification accuracy on the independent test dataset. For each training size, the mean accuracy and the corresponding standard deviation across runs were computed, allowing both average performance and training stability to be assessed.

5.3 Results

Figure 10 illustrates the relationship between training dataset size and classification accuracy for the baseline MLP model.

The results reveal a clear dependency between the amount of labeled training data and model performance. When trained with only 100 samples, the baseline model achieves relatively low classification accuracy, indicating limited generalization capability in data-scarce settings. As the number of training samples increases, performance improves steadily, with the most pronounced gains observed between 100 and 500 samples.

Beyond approximately 700 training samples, the improvement in accuracy becomes more gradual, suggesting diminishing returns from additional labeled data. This behavior is typical for fully supervised models operating on high-dimensional gene expression data, where increasing sample size alleviates overfitting but cannot fully compensate for the lack of prior representation learning.

The shaded region in Figure 10 represents one standard deviation around the mean accuracy and reflects variability across different stratified training runs. The decreasing variance with increasing training size indicates improved training stability as more labeled samples become available.

Overall, these results highlight the limitations of purely supervised learning in low-label regimes and motivate the use of self-supervised pretraining to improve performance when labeled data are scarce.

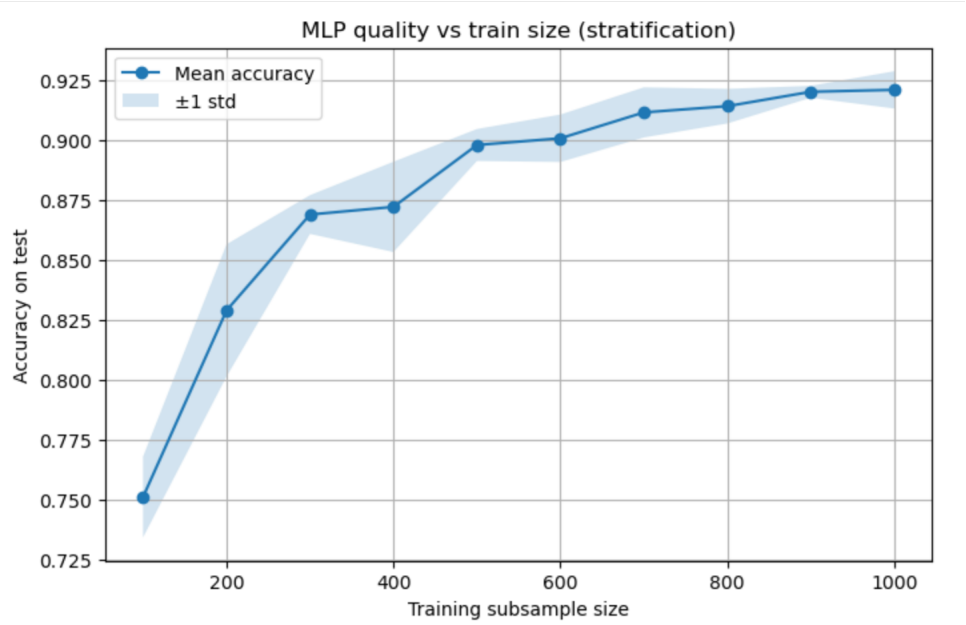


Figure 10: Baseline MLP classification accuracy as a function of training subset size. The curve shows the mean test accuracy across multiple stratified runs, while the shaded area represents ±1 standard deviation.

6 Gene Identifier Mapping (ENSG → Gene Symbol)

The raw transcriptomic matrices were provided with Ensembl gene identifiers (format `ENSG...`) as feature names. However, most biological pathway resources (including KEGG gene sets) are defined using gene symbols (e.g., `TP53`, `EGFR`). Therefore, a mandatory preprocessing step was to map Ensembl IDs to gene symbols and standardize the feature space across all datasets.

Mapping procedure. The mapping was implemented using the `mygene` (`MyGeneInfo`) annotation service. The following operations were applied:

1. **Detection of gene columns.** Columns were split into (i) gene features (`ENSG...`) and (ii) non-gene metadata (e.g., `cancer_type`).

2. **Sample indexing.** If present, the sample identifier column (`caseID`) was moved into the dataframe index to ensure stable sample alignment.

3. **Removal of Ensembl version suffix.** Ensembl identifiers may include a version suffix (e.g., `ENSG00000... .12`). This suffix was removed to make identifiers consistent with annotation databases.

4. **Querying the annotation service.** Each Ensembl ID was queried with `scopes="ensembl.gene"` and `species="human"` to retrieve the corresponding gene symbol.

5. **Handling unmapped genes.** Some Ensembl IDs did not return a valid gene symbol. These unmapped genes were removed from the feature space (option `drop_unmapped=True`) to avoid introducing artificial columns.

6. **Handling duplicate mappings.** Multiple Ensembl IDs can map to the same gene symbol. In such cases, duplicate columns were aggregated by computing the mean expression value (`aggregate_duplicates=True`).

7. **Gene symbol normalization.** All gene symbols were converted to uppercase and stripped of extra whitespace to ensure consistent matching across datasets and pathway resources.

8. **Re-attaching metadata.** After mapping, the non-gene columns (e.g., `cancer_type`) were concatenated back to the mapped gene matrix.

Practical issues observed during mapping. Several issues occurred during this stage and required explicit handling:

- **Unmapped Ensembl IDs:** a small number of gene IDs did not have corresponding symbols and were excluded.
- **Duplicate hits:** the annotation tool sometimes reported duplicates (multiple hits for the same query), requiring aggregation at the gene-symbol level.
- **Dataset-dependent gene sets:** even after mapping, different datasets could contain slightly different gene sets, which later caused dimensionality mismatches.

Resulting feature dimensionality. After ENSG→symbol mapping and cleaning, the number of gene features decreased from 19,887 to 19,706 due to (i) removal of unmapped genes and (ii) aggregation of duplicates. Importantly, the mapping procedure produced consistent shapes across all datasets:

`pretrain_mapped` : (7349, 19706), `finetune_mapped` : (1000, 19706), `test_mapped` : (1000, 19706).

This confirmed that the mapping function produced a unified input space suitable for downstream modeling.

caseID	A1BG	A1CF	A2M	A2ML1	A3GALT2	A4GALT	A4GNT	AAAS	AACS	AADAC	...	ZWILCH	ZWINT	ZXDA	ZXDB	ZXDC	ZYG11A	ZYG11B
TCGA-HQ-A5ND-01A-11R-A26T-07	0.0058	0.0000	24.8118	5.5564	0.0000	14.6656	0.0000	8.9095	3.0487	8.4258	...	8.6626	63.8999	0.6007	1.6964	4.9005	0.2433	6.1036
TCGA-G2-A3IB-01A-11R-A20F-07	0.0000	0.0018	2.6938	20.7405	0.0167	32.5484	0.0096	8.2906	6.8177	2.6333	...	6.1170	18.6896	0.3110	1.1660	2.6759	0.0872	2.7931
TCGA-ZF-AA5N-01A-11R-A42T-07	0.0000	0.0000	6.0953	3.8893	0.0000	50.2704	0.0000	10.5861	2.8862	0.0000	...	3.7234	13.5009	0.3452	0.9348	2.9450	0.2299	2.1526

Figure 11: Gene ID mapping step (ENSG → gene symbol). The output confirms consistent dimensionality across pretraining, fine-tuning and test datasets after removing unmapped genes and aggregating duplicates.

7 Pathway Score Computation (ssGSEA on KEGG Gene Sets)

Self-supervised learning in this project was designed to include biologically informed objectives. In addition to learning from masked-gene reconstruction and contrastive views, the model was trained to predict pathway-level activity profiles. These pathway profiles were computed using **single-sample Gene Set Enrichment Analysis (ssGSEA)**.

7.1 KEGG Gene Set Resource

To compute pathway activity scores, KEGG pathway gene sets were required in `.gmt` format. The gene sets were obtained from the KEGG pathway resource hosted by Genome.jp:

`https://www.genome.jp/`

A KEGG GMT file (e.g., `kegg_2025.gmt`) was downloaded and used as the reference collection of pathways.

7.2 GMT Parsing

The KEGG gene sets were loaded from the GMT file using a custom parser. The GMT format is defined as:

$$\texttt{pathway\_name description gene}_1 \ \texttt{gene}_2 \ \ldots$$

The parser produced a dictionary:

$$\{\texttt{PathwayName} \rightarrow [\texttt{GeneSymbol}_1,\ \texttt{GeneSymbol}_2,\ \ldots]\}.$$

This intermediate representation was then passed directly to the ssGSEA routine.

7.3 ssGSEA Computation using `gseapy`

Pathway activity profiling was implemented with `gseapy.ssgsea`. The key input requirement for ssGSEA is a gene expression matrix with:

- **rows = genes**
- **columns = samples**

Since the data were stored in the usual machine-learning format (rows = samples, columns = genes), the gene matrix was transposed prior to computation.

Batching strategy. Running ssGSEA on thousands of samples can be memory-intensive. Therefore, the computation was performed in **batches of samples** (e.g., `batch_size=200`). For each batch:

1. ssGSEA was applied using rank-based normalization.
2. The output table (`res2d`) was converted into a matrix of normalized enrichment scores (NES).
3. Results from all batches were concatenated into a full pathway-score matrix.

Output format. The resulting pathway matrix was returned as:

$$samples \times pathways,$$

where each entry represents a pathway activity score (NES) for a given sample.

Saving pathway profiles. Pathway activity matrices for pretraining, fine-tuning and test datasets were saved to disk in `.parquet` format using a helper function. This ensured that pathway computation was reproducible and did not need to be repeated during each training run.

Term Name	KEGG_ABC_TRANSPORTERS	KEGG_ACUTE_MYELOID_LEUKEMIA	KEGG_ADHERENS_JUNCTION	KEGG_ADIPOCYTOKINE_SIGNALING_PATHWAY
TCGA-04-1331-01A-01R-1569-13	-0.066531	0.212176	0.257322	0.111193
TCGA-04-1332-01A-01R-1564-13	-0.044394	0.2598	0.273174	0.133795
TCGA-04-1337-01A-01R-1564-13	-0.050204	0.24503	0.264102	0.133747
TCGA-04-1338-01A-01R-1564-13	-0.11324	0.221825	0.286975	0.115488
TCGA-04-1341-01A-01R-1564-13	-0.094715	0.214129	0.241535	0.105869

Figure 12: Pathway profile computation using ssGSEA on KEGG gene sets. First, gene IDs were mapped to gene symbols. Next, KEGG pathways were loaded from a GMT file and ssGSEA was applied to compute pathway activity profiles (NES scores), which were saved for downstream SSL pretraining.

7.4 Motivation for Pathway-Level Targets in SSL

Gene expression measurements are noisy and high-dimensional, while pathway activity provides a more stable biological representation by aggregating signals across functionally related gene sets. Incorporating pathway profiles into self-supervised pretraining provides the model with an additional objective that encourages:

- learning biologically meaningful latent features,
- capturing coordinated gene programs rather than isolated genes,
- improving transferability to downstream classification tasks.

In the SSL framework used in this project, pathway scores served as regression targets in the pretraining stage, complementing masked reconstruction and contrastive learning objectives.

8 Self-Supervised Learning Framework

In this section, we provide a detailed description of the self-supervised learning (SSL) framework used in this project, including the model architecture, pretraining objectives, loss formulation, and hyperparameter choices. Since representation learning is the core contribution of this work, particular emphasis is placed on the motivation behind the architectural and methodological design choices.

8.1 Overview of the Self-Supervised Pipeline

The proposed SSL framework follows a multi-task pretraining strategy designed to learn biologically meaningful representations from unlabeled gene expression data. Given an input RNA-seq sample represented as a high-dimensional gene expression vector

$$\mathbf{x} \in R^G,$$

the model is trained using three complementary self-supervised objectives:

- pathway profile prediction (biologically informed regression task),
- masked gene reconstruction (MAE-style reconstruction),
- contrastive representation learning (SimCLR-style objective).

All three objectives share a single encoder network, encouraging the learned latent space to simultaneously capture biological function, gene-level dependencies, and sample-level similarity structure. After pretraining, the encoder is reused for downstream cancer classification via linear probing or full fine-tuning.

Figure 13 provides an overview of the self-supervised pretraining pipeline, illustrating the shared encoder architecture and the three complementary pretraining objectives.

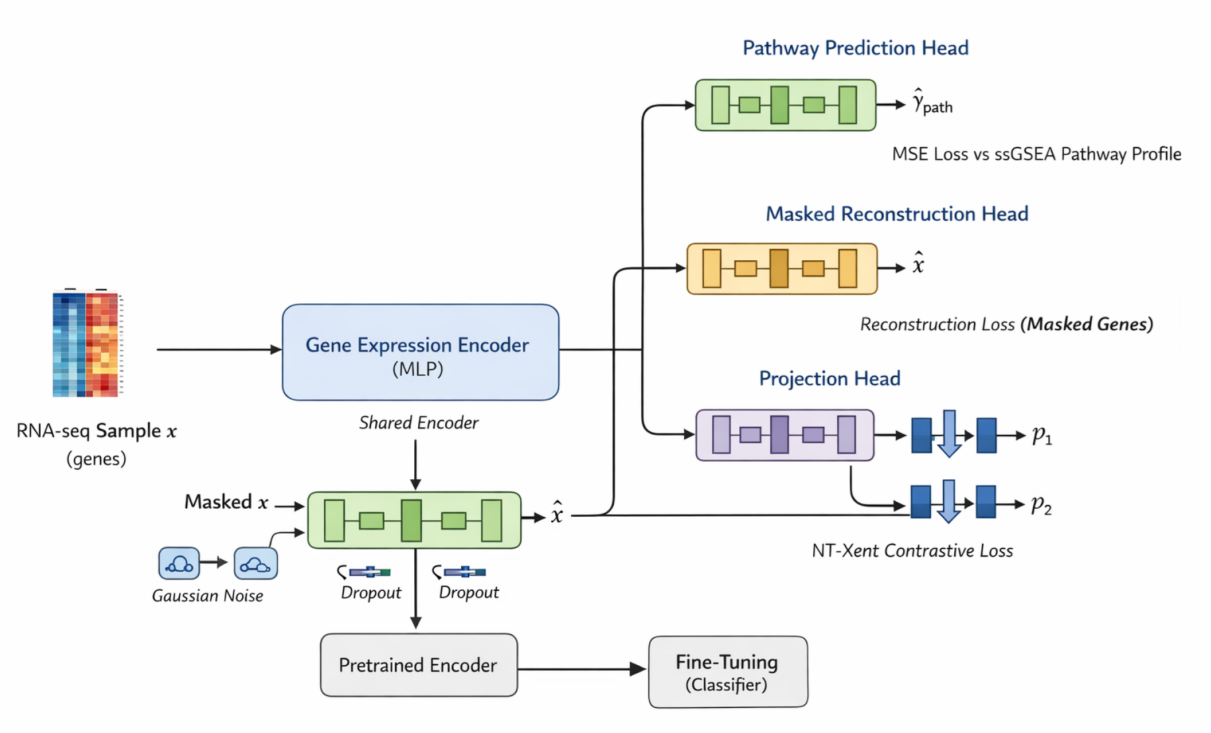


Figure 13: Self-supervised pretraining pipeline for transcriptomic data. An unlabeled RNA-seq sample is processed by a shared encoder and optimized using three complementary self-supervised objectives: pathway profile prediction, masked gene reconstruction, and contrastive representation learning. After pretraining, the encoder is reused for downstream cancer classification via linear probing or full fine-tuning.

8.2 Model Architecture

The SSL model consists of a shared encoder followed by three task-specific heads.

The encoder is implemented as a multilayer perceptron (MLP) that maps gene expression vectors into a latent representation:

$$\mathbf{x} \in R^G Encoder \mathbf{z} \in R^D.$$

It is composed of three fully connected layers with ReLU activations and dropout regularization:

- Linear($G \to 1024$) + ReLU + Dropout,
- Linear($1024 \to 512$) + ReLU + Dropout,
- Linear($512 \to D$) + ReLU.

A layer normalization is applied to the final latent vector to stabilize training and improve contrastive learning behavior.

From the shared latent representation $\mathbf{z}$, three task-specific heads are defined.

The first head predicts pathway activity profiles computed using ssGSEA:

$$\hat{\mathbf{y}}_{path} \in R^P.$$

The second head reconstructs masked gene expression values from a corrupted input:

$$\hat{\mathbf{x}} \in R^G.$$

The third head maps latent representations to a lower-dimensional space used exclusively for contrastive learning:

$$\mathbf{p} \in R^d.$$

This design allows each objective to focus on a specific aspect of representation learning while sharing a common encoder.

8.3 Self-Supervised Objectives

8.3.1 Pathway Profile Prediction Loss

Pathway activity profiles are computed offline using ssGSEA on KEGG gene sets and serve as biologically informed targets. The model is trained to predict these pathway scores using a mean squared error loss:

$$\mathcal{L}_{path} = \hat{\mathbf{y}}_{path} - \mathbf{y}_{path}\,{}_2^2\,.$$

This objective introduces biological prior knowledge into the self-supervised learning process by encouraging the encoder to capture coordinated gene programs rather than individual gene effects.

8.3.2 Masked Gene Reconstruction Loss

To model local gene dependencies, a fraction of gene features is randomly masked (set to zero) for each sample. The model reconstructs the original expression values only at masked positions. The reconstruction loss is defined as:

$$\mathcal{L}_{mae} = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} (\hat{x}_i - x_i)^2\,,$$

where $\mathcal{M}$ denotes the set of masked gene indices. This MAE-style objective promotes robust feature learning and prevents the encoder from relying on a small subset of dominant genes.

8.3.3 Contrastive Learning Loss

To encourage invariance to biologically plausible perturbations, two augmented views of each sample are generated using Gaussian noise injection and random feature dropout. A SimCLR-style NT-Xent loss is applied:

$$\mathcal{L}_{ctr} = NT - Xent\left(\mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right),$$

where $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ are the projected representations of the two augmented views. This loss enforces sample-level consistency and improves global structure in the latent space.

8.4 Combined Loss Function

The total self-supervised loss is defined as a weighted combination of the three objectives:

$$\mathcal{L}_{SSL} = \alpha\,\mathcal{L}_{path} + \beta\,\mathcal{L}_{mae} + \gamma\,\mathcal{L}_{ctr}.$$

The weighting coefficients are chosen to balance biological supervision and generic representation learning. In this work, the following values were used:

- $\alpha = 1.0$ (pathway regression),
- $\beta = 0.3$ (masked reconstruction),
- $\gamma = 0.1$ (contrastive learning).

These values were selected empirically to ensure stable convergence while preventing any single objective from dominating the training process.

8.5 Pretraining Hyperparameters

Self-supervised pretraining was performed using the Adam optimizer. Key hyperparameters are summarized in Table 1.

Hyperparameter	Value
Epochs	50
Batch size	64
Learning rate	1×10^{-3}
Masking ratio	0.4
Latent dimension D	256
Projection dimension d	128
Dropout rate	0.3
Contrastive temperature	0.5
Noise standard deviation	0.1
Feature dropout probability	0.1

Table 1: Hyperparameters used during self-supervised pretraining.

8.6 Motivation for the Architecture

The design of the self-supervised learning framework is motivated by the specific characteristics of transcriptomic data:

- High dimensionality and noise motivate masked reconstruction to improve robustness.
- Biological interpretability motivates pathway-level supervision using ssGSEA scores.
- Heterogeneity across samples motivates contrastive learning to preserve global structure and invariances.

By combining biologically informed and generic self-supervised objectives, the encoder learns representations that are both biologically meaningful and transferable, as demonstrated by strong downstream performance even when the encoder is frozen during fine-tuning.

8.7 Training Dynamics and Pretraining Convergence

Self-supervised pretraining was performed for 50 epochs using the unlabeled pretraining dataset. Figure 14 shows the evolution of the average training loss during pretraining.

The loss decreases steadily over epochs and converges smoothly, indicating stable optimization and successful learning of latent representations. The absence of oscillations or divergence suggests that the combination of self-supervised objectives is well balanced and does not lead to conflicting gradients. The gradual reduction of loss reflects the model's increasing ability to capture biologically meaningful structure in the gene expression data.

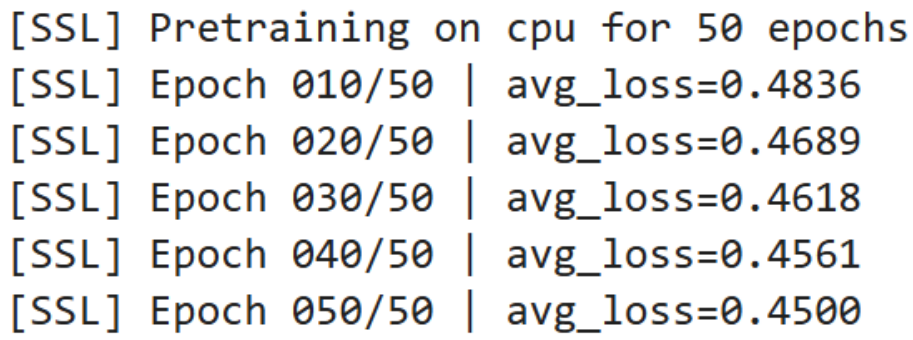


Figure 14: Self-supervised pretraining loss as a function of training epoch. The steady decrease indicates stable convergence of the SSL objectives.

8.8 Fine-Tuning Results

After self-supervised pretraining, the learned encoder was evaluated on the downstream cancer classification task using two fine-tuning strategies: frozen encoder (linear probing) and unfrozen encoder (full fine-tuning). Performance was assessed across different proportions of labeled training data, ranging from 10% to 100% of the fine-tuning dataset.

Figure 15 presents the mean classification accuracy as a function of the labeled data proportion for both strategies.

Several important observations can be made. First, self-supervised pretraining yields strong performance even in low-data regimes. With only 10% of labeled data, the frozen encoder already achieves accuracy above 0.83, substantially outperforming a purely supervised baseline trained from scratch.

Second, freezing the encoder consistently provides competitive performance across all data proportions. This indicates that the representations learned during self-supervised pretraining are highly informative and transferable, allowing effective classification with only a linear classifier on top.

Third, full fine-tuning of the encoder provides additional performance gains when sufficient labeled data are available. At higher data proportions (above 50%), unfreezing the encoder leads to slightly higher peak accuracy, reaching approximately 0.93. This suggests that while the pretrained representations are strong, task-specific adaptation can further refine them when enough labeled samples are provided.

```
Freeze_encoder=True
Prop 0.1 → Mean Acc = 0.8420
Prop 0.2 → Mean Acc = 0.8934
Prop 0.3 → Mean Acc = 0.9036
Prop 0.4 → Mean Acc = 0.9144
Prop 0.5 → Mean Acc = 0.9142
Prop 0.6 → Mean Acc = 0.9206
Prop 0.7 → Mean Acc = 0.9200
Prop 0.8 → Mean Acc = 0.9246
Prop 0.9 → Mean Acc = 0.9266
Prop 1.0 → Mean Acc = 0.9258
Freeze_encoder=False
Prop 0.1 → Mean Acc = 0.8360
Prop 0.2 → Mean Acc = 0.8946
Prop 0.3 → Mean Acc = 0.9002
Prop 0.4 → Mean Acc = 0.9118
Prop 0.5 → Mean Acc = 0.9130
Prop 0.6 → Mean Acc = 0.9216
Prop 0.7 → Mean Acc = 0.9216
Prop 0.8 → Mean Acc = 0.9260
Prop 0.9 → Mean Acc = 0.9256
Prop 1.0 → Mean Acc = 0.9278
```

	proportion	mean	std	run1	run2	run3	run4	run5
0	0.1	0.8360	0.012602	0.838	0.842	0.836	0.813	0.851
1	0.2	0.8946	0.003555	0.894	0.900	0.890	0.892	0.897
2	0.3	0.9002	0.005706	0.908	0.899	0.899	0.904	0.891
3	0.4	0.9118	0.006046	0.916	0.903	0.917	0.906	0.917
4	0.5	0.9130	0.004858	0.921	0.911	0.913	0.914	0.906
5	0.6	0.9216	0.003929	0.928	0.918	0.917	0.922	0.923
6	0.7	0.9216	0.004224	0.925	0.917	0.924	0.916	0.926
7	0.8	0.9260	0.001414	0.927	0.924	0.925	0.928	0.926
8	0.9	0.9256	0.003382	0.930	0.927	0.924	0.920	0.927
9	1.0	0.9278	0.000748	0.928	0.927	0.928	0.929	0.927

Figure 15: Classification accuracy as a function of labeled training data proportion for self-supervised pretraining followed by fine-tuning. Results are shown for frozen (linear probing) and unfrozen (full fine-tuning) encoder settings.

9 Comparison with Supervised Baseline

A direct comparison between the purely supervised baseline model and the proposed self-supervised learning (SSL) framework was conducted in order to quantify the benefits of representation learning from unlabeled gene expression data. The comparison focuses on classification accuracy, data efficiency, and robustness across different proportions of labeled training data.

Figure 16 presents the mean test accuracy as a function of the fraction of labeled training samples for three models: a fully supervised baseline MLP, an SSL-pretrained encoder with frozen weights (linear probing), and an SSL-pretrained encoder with full fine-tuning.

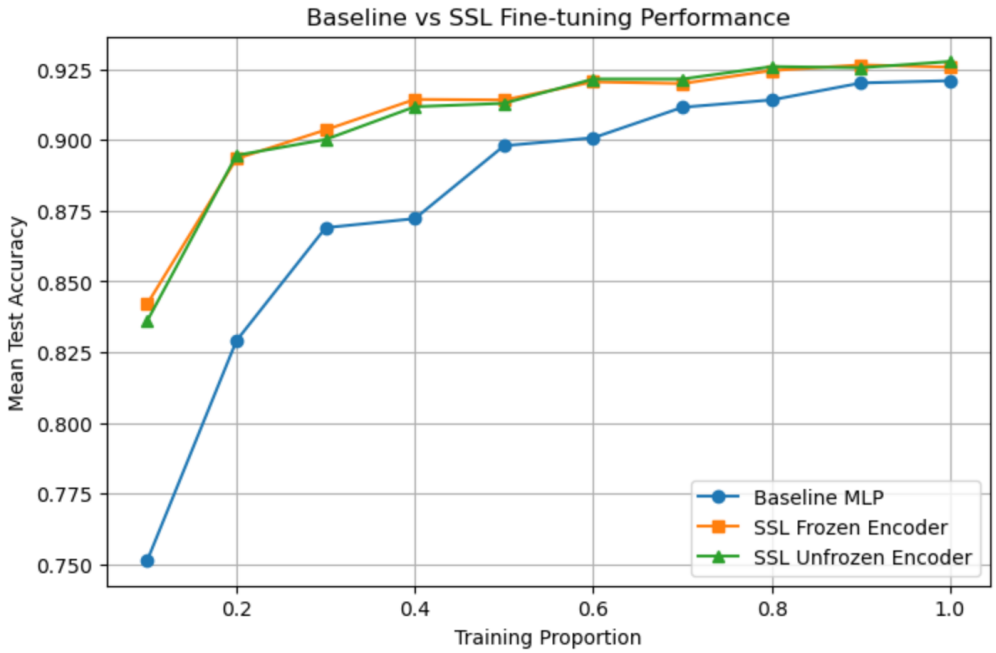


Figure 16: Comparison of classification performance between the supervised baseline MLP and SSL-based models. Mean test accuracy is reported as a function of the labeled training data proportion.

Several important observations can be drawn from these results.

First, in the low-data regime, the supervised baseline model exhibits substantially lower performance. When only 10–20% of the labeled data are available, the baseline MLP struggles to generalize, reflecting the well-known difficulty of training high-capacity models on high-dimensional transcriptomic data with limited supervision. In contrast, both SSL-based models achieve significantly higher accuracy under the same conditions, demonstrating the strong benefit of self-supervised pretraining.

Second, even when the pretrained encoder is kept frozen and only a linear classifier is trained on top, the SSL approach consistently outperforms the fully supervised baseline. This indicates that the representations learned during self-supervised pretraining capture biologically meaningful structure that is directly transferable to downstream cancer classification tasks.

Third, allowing full fine-tuning of the pretrained encoder leads to the best overall performance across nearly all training proportions. The improvement over the frozen-encoder setting is most pronounced when moderate to large amounts of labeled data are available, suggesting that fine-tuning enables the model to further adapt its representations to task-specific decision boundaries while still benefiting from the SSL initialization.

As the amount of labeled training data increases, the performance gap between the baseline and SSL-based models gradually decreases. This behavior is expected, as fully supervised learning becomes more effective when sufficient labeled samples are available. Nevertheless, the SSL

models remain competitive even in the high-data regime, achieving equal or slightly superior performance compared to the baseline.

To better contextualize these results, the architectural complexity of the models should also be considered. The supervised baseline MLP contains approximately 10 million trainable parameters, whereas the SSL encoder contains over 20 million parameters due to its deeper architecture and normalization layers. Despite this increased capacity, the SSL model does not suffer from overfitting in low-data regimes, highlighting the regularizing effect of self-supervised pretraining.

Overall, this comparison clearly demonstrates that self-supervised learning provides a strong inductive bias for gene expression analysis. By leveraging unlabeled data and biologically informed pretraining objectives, the SSL framework improves data efficiency, robustness, and generalization performance compared to a purely supervised baseline.

10 Discussion

The results of this study highlight the critical role of representation learning in the analysis of high-dimensional gene expression data. Due to the extremely large feature space and limited availability of labeled samples, purely supervised learning approaches are prone to overfitting and often fail to generalize well. Self-supervised pretraining addresses this limitation by allowing the model to learn informative structure from unlabeled data before being exposed to class labels.

A key insight from the experiments is that biologically informed self-supervised objectives, such as pathway activity prediction, provide an effective inductive bias. By incorporating prior biological knowledge through pathway-level supervision, the model is encouraged to learn representations that reflect functional relationships between genes rather than purely statistical correlations. This significantly improves downstream classification performance and model stability.

The combination of multiple self-supervised tasks further enhances representation quality. Masked gene reconstruction promotes the learning of local dependencies between genes, while contrastive learning encourages global structure preservation and invariance to perturbations. Together, these objectives enable the encoder to capture complementary aspects of gene expression variation, including biological function, co-expression patterns, and sample-level similarity.

Another important observation is the strong performance of the frozen encoder during fine-tuning. Even without updating the encoder weights, SSL-pretrained representations consistently outperform a fully supervised baseline. This indicates that the learned representations are robust and transferable, which is particularly valuable in biomedical settings where labeled data are scarce and expensive to obtain. At the same time, full fine-tuning of the encoder provides additional performance gains when sufficient labeled data are available, offering increased flexibility and task adaptation.

Finally, the experiments emphasize the importance of careful data preprocessing and strict feature alignment. Consistent gene mapping, removal of ambiguities in gene identifiers, and stable feature spaces across datasets were essential to ensure reliable training and fair comparison between models.

11 Conclusion

In this project, a complete self-supervised learning pipeline for gene expression data analysis was designed, implemented, and evaluated. A neural network encoder was pretrained on unlabeled transcriptomic data using a combination of biologically motivated and generic self-supervised objectives, and subsequently fine-tuned for cancer type classification.

The experimental results demonstrate that self-supervised pretraining substantially improves classification performance, particularly in low-data regimes. Compared to a purely supervised

baseline, the SSL-based approach achieves higher accuracy, improved robustness, and better data efficiency. Even with a frozen encoder, SSL representations provide a strong foundation for downstream tasks, while full fine-tuning yields the best overall performance when labeled data are available.

Overall, this work shows that self-supervised learning is a powerful and practical framework for transcriptomic data analysis. By leveraging large amounts of unlabeled data and incorporating biological prior knowledge, SSL offers a promising direction for future applications in bioinformatics and precision medicine, especially in scenarios where labeled data are limited.

References

[1] National Cancer Institute. *The Cancer Genome Atlas (TCGA)*. Available at: https://portal.gdc.cancer.gov/. Accessed: 2025-01.

[2] Kanehisa, M., Furumichi, M., Sato, Y., Ishiguro-Watanabe, M., & Tanabe, M. *KEGG: integrating viruses and cellular organisms*. Nucleic Acids Research, Volume 49, Issue D1, 2021, Pages D545–D551. Available at: https://www.genome.jp/. Accessed: 2025-01.

[3] gseapy developers. *GSEApy: Gene Set Enrichment Analysis in Python*. Available at: https://gseapy.readthedocs.io/en/latest/gseapy_example.html#Single-Sample-GSEA-example. Accessed: 2025-01.

[4] MyGeneInfo project. *mygene.py: Python client for MyGene.info API*. Available at: https://docs.mygene.info/projects/mygene-py/en/latest/. Accessed: 2025-01.

[5] Radjat, K. *SSRL_RNAseq*. GitHub repository. Available at: https://github.com/kdradjat/SSRL_RNAseq. Accessed: 2025-01.

[6] Barbie, D. A., Tamayo, P., Boehm, J. S., et al. *Systematic RNA interference reveals that oncogenic KRAS-driven cancers require TBK1*. Nature, 2009. (This is often cited for ssGSEA context — include if relevant.)

[7] Le-Khac, P. H., He, X., & Smeaton, A. F. *Contrastive Representation Learning: A Critical Review and Comparisons*. arXiv:2009.08317.