Tissue Classification from Gene Expression

(Mohsen Shirkarami)

The code can be found on [my GitHub]

Data

After downloading data from the DepMap portal, we load them into our code environment for further processing.

Task 1: Data Manipulation

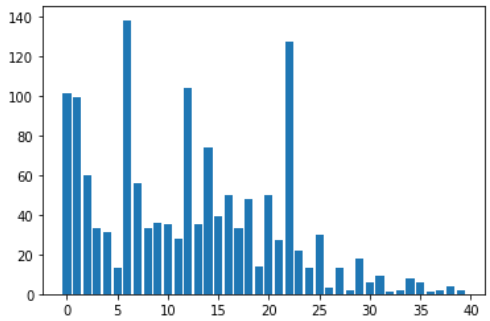
According to the Gene expression (CCLE_expression.csv file), we have 1406 samples, each with 19221 features. First, we find corresponding labels (tissue type) for these samples from the sample_collection_site column of the sample_info.csv file.

To preprocess the data, we do these steps:

- Some samples have no labels (NaN class in the table below). We exclude these samples (4 samples)
- According to the mentioned drug response prediction papers, we exclude the genes that showed a low variability across the cell lines (genes with standard deviation < 0.1)
- We also removed any gene that was not expressed (i.e., value = 0) for more than 90% of the samples
- We perform z-score normalization on each gene across all samples, which makes them zero mean and have one standard deviation
- We encode target labels with values between 0 and n_classes-1
- If we count the samples per class, we can see that the dataset is imbalanced; in other words, we have a different number of samples for different classes. We compute the class weight for each class.
- We have just one sample from two classes, 'sinonasal' and 'salivary_gland.' We exclude these samples from our dataset.

0	lung	101
1	central_nervous_system	99
2	skin	60
3	biliary_tract	33
4	urinary_tract	31
5	abdomen	13
6	pleural_effusion	138
7	soft_tissue	56
8	large_intestine	33
9	endometrium	36
10	kidney	35
11	pancreas	28
12	lymph_node	104
13	fibroblast	35
14	ascites	74
15	bone	39
16	upper_aerodigestive_tract	50
17	ovary	33
18	liver	48
19	cervix	14
20	bone_marrow	50

21	oesophagus	27
22	haematopoietic_and_lymphoid_tissue	127
23	Colon	22
24	stomach	13
25	breast	30
26	pericardial_effusion	3
27	thyroid	13
28	small_intestine	2
29	autonomic_ganglia	18
30	uvea	6
31	eye	9
32	sinonasal	1
33	Embryonal	2
34	prostate	8
35	pleura	6
36	salivary_gland	1
37	spleen	2
38	NaN	4
39	Placenta	2



- Note that after excluding those two classes ('sinonasal' and 'salivary_gland'), 37 classes remain.

After that, we split the dataset into training, validation, and test set (80%, 10%, and 10% of all samples, respectively, in a way that each set include samples from all classes)

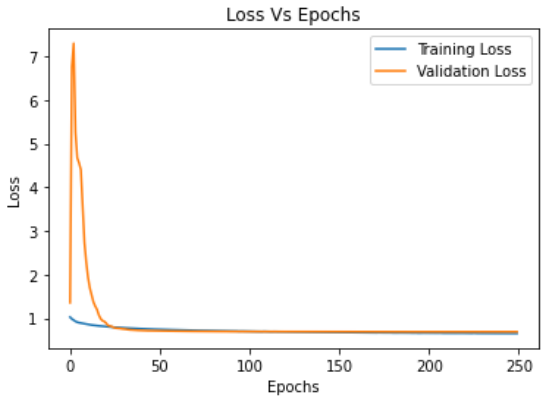
```
number of Training samples: 1120
X_train shape: (1120, 18033)
y_train shape: (1120,)

number of Validation samples: 140
X_val shape: (140, 18033)
y_val shape: (140,)

number of Test samples: 140
X_test shape: (140, 18033)
y_test shape: (140,)
```

Task 2a: Deep Unsupervised Learning

Our data has 18033 features per sample. To reduce the dimensionality of that, we use an Autoencoder. Autoencoder is a neural network that gives a compressed representation from input data. The aim is to train an autoencoder that encodes the gene expression into a vector of size 512.

Due to the small number of training data, the architecture of the autoencoder should not be too complex. Thus, we use one hidden layer in each part of the autoencoder, including the encoder and decoder. We also use the Dropout layer after each Dense layer to avoid overfitting. We compile our model with adam optimizer and mean squared error (mse) loss. Fitting model on (X_train, X_train) and use (X_val, X_val) as validation data. Note that we use X_train as the label here because we aim to reproduce the input data in the output of the autoencoder.

We plot the loss curve for the training and validation set during the training process:



In the last step of this task, we encode all samples (train, val, test) using our trained autoencoder and save the encodings for future use.

Task 2b: Shallow Supervised Learning

In this part, we use the encoded sets as well as the gene expression to train some shallow classifiers (KNN, LDA, SVM, and RF).

- **KNN**:
  - K = list(range(1, 30, 2))

  For different values for K (number of neighbors) as hyperparameter, we train KNN classifiers in two cases:
  1. Train on encoded X_train and apply the model to the encoded X_val
  2. Train on the Gene Expression (X_train) and apply the model to the X_val

  Finally, we save the best value of hyperparameter K (concerning the accuracy of the models on the validation data)

KNN Classifier:

K	Encoded	Gene Expression
1	0.407143	0.392857
3	0.400000	0.414286
5	0.428571	0.435714
7	0.457143	0.478571
9	0.421429	0.457143
11	0.450000	0.457143
13	0.435714	0.457143
15	0.442857	0.450000
17	0.435714	0.457143
19	0.428571	0.471429
21	0.407143	0.450000
23	0.407143	0.457143
25	0.450000	0.457143
27	0.450000	0.464286
29	0.450000	0.457143

- **LDA**:

  Train two models as described above and apply them to the validation data

```
LDA Classifier:

* Using Encoded Data
accuracy: 0.4642857142857143


* Using the Gene Expression
accuracy: 0.5357142857142857
```

- **SVM:**
  - C=[150,120,100,10,1,0.1]

  For different values for C (regularization parameter), train models and apply them to the validation data, and save the best hyperparameter C in each case.

```
SVM Classifier:

      C   Encoded  Gene Expression
150.0  0.485714         0.507143
120.0  0.485714         0.507143
100.0  0.485714         0.507143
 10.0  0.485714         0.507143
  1.0  0.514286         0.485714
  0.1  0.221429         0.171429
```

- **RF:**
  - trees=[200,150,100,50]

  The hyperparameter here is n (number of trees in the forest). We define multiple values for n, train models, and apply them to the validation data.

```
Random Forest Classifier:

  n_trees   Encoded  Gene Expression
      200  0.514286         0.500000
      150  0.478571         0.521429
      100  0.500000         0.485714
       50  0.485714         0.485714
```

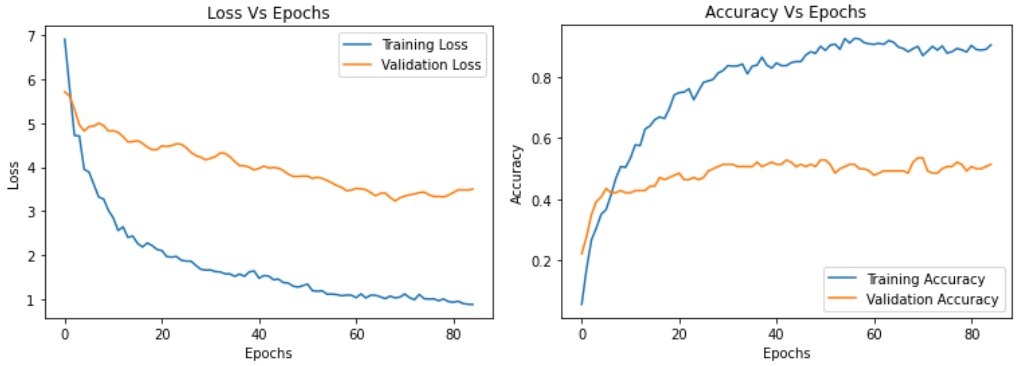
Task 2c: Deep Supervised Learning

We train an MLP that takes the gene expression as an input and classifies the tissue type.

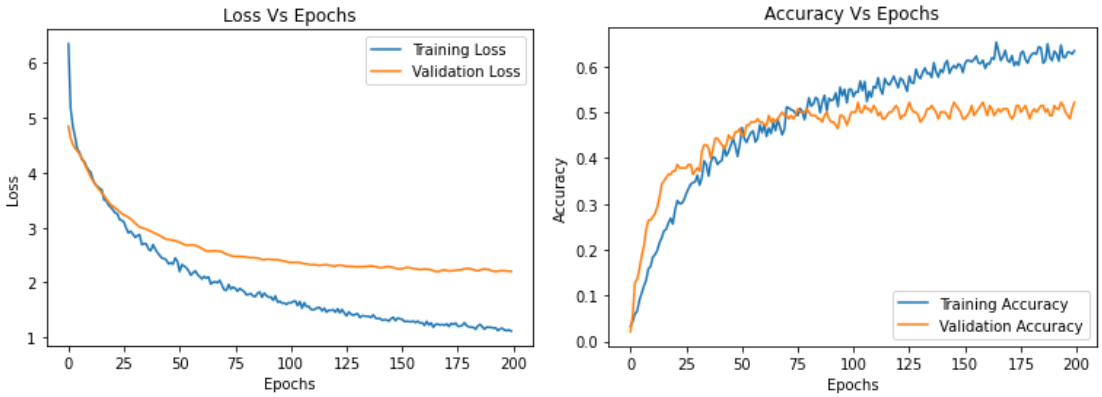
First, we process the labels into one-hot vectors (categorical) needed for the categorical_crossentropy loss function. Using Tensorflow, we define MLP with three hidden dense layers and an output layer with 37 nodes (same as the number of classes). Then we compile and fit a model on the training data. Through the training process, we report loss and accuracy on training as well as validation data. Also, we use the class weights we computed earlier to address the problem of imbalanced data.

Due to the small number of training data, we are prone to overfitting. We use the regularization method to prevent this, including L2 Regularization and Dropout with a rate of 0.5. Using one hidden dense layer and categorical cross-entropy loss, we train the model.

We plot the loss and accuracy curves for the training and validation set during the training process.



We also train another MLP on encoded data (X_train_encoded) and validate it by applying it to the encoded Validation data (X_val_encoded) during training.



Task 3: Model Evaluation

We want to evaluate our models on the test set in this part. With the appropriate hypermeters we have chosen based on the validation data in the previous tasks, we train new models using all available data (aggregation of the training and validation data) and test these models on the test data.

- KNN

```
KNN Classifier:

* Using Encoded Data
accuracy: 0.45714285714285713


* the Gene Expression
accuracy: 0.42857142857142855
```

- LDA

```
LDA Classifier:

* Using Encoded Data
accuracy: 0.45714285714285713


* Using the Gene Expression
accuracy: 0.5
```

- SVM

```
SVM Classifier:

* Using Encoded Data
accuracy: 0.4357142857142857


* Using the Gene Expression
accuracy: 0.45714285714285713
```

- RF

```
Random Forest Classifier:

* Using Encoded Data
accuracy: 0.45714285714285713


* Using the Gene Expression
accuracy: 0.45714285714285713
```

- MLP (trained with Gene Expression aggregated data)

```
MLP (trained with the the Gene Expression):

Test Loss is 3.21848726272583
Test Accuracy is 0.5
```

- MLP (trained with encoded aggregated data)

```
MLP (trained with the the encoded data):

Test Loss is 3.1233761310577393
Test Accuracy is 0.44285714626312256
```

According to the accuracy of models, we can say that using deep methods (such as MLP) for this dataset would overfit the training data since we have limited available data to train the model, and the regularization method did not work well. We should first augment data and then train models with a greater amount of data.

Task 4: Cluster Analysis

We use K-means clustering to cluster our data. To select the optimal number of clusters, there are some methods and metrics, such as the Elbow method, Silhouette method, Gap Statistic, and so on.

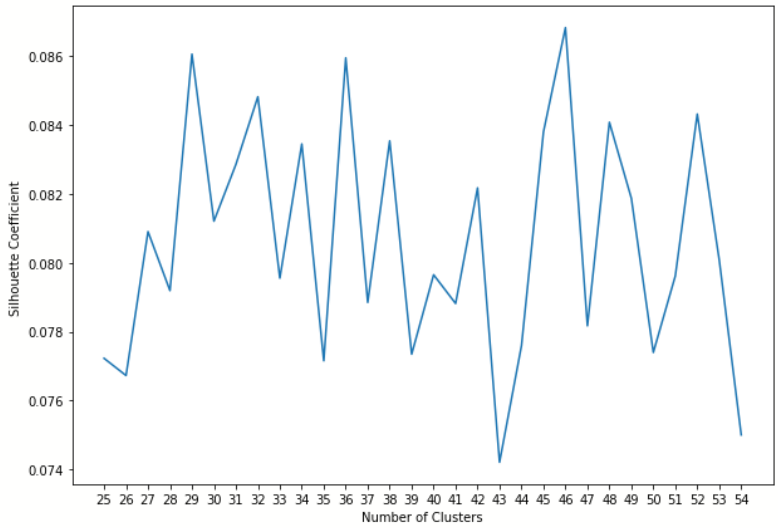
I choose the Silhouette Method. This method calculates the Silhouette coefficient over a range of k. Then it selects the optimal K that produces the largest Silhouette coefficient and assigns data into clusters based on the optimal cluster number K. The Silhouette coefficient is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample.

The Silhouette coefficient for a sample is (b - a) / max(a, b)

In This formula, b is the distance between a sample and the nearest cluster that the sample is not a part of.

Then averaging over all samples, the Silhouette coefficient is computed.

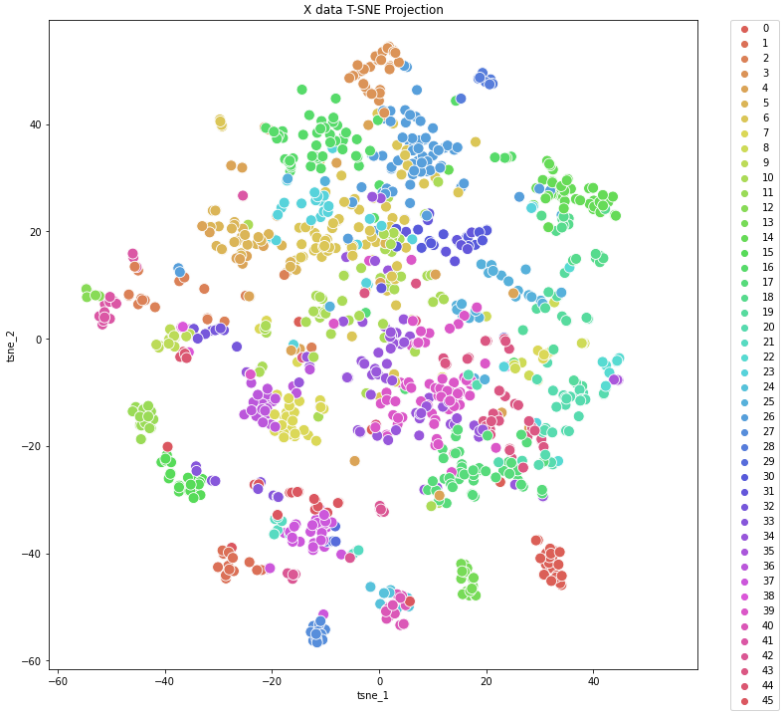
First, we perform K-Means clustering for X_encoded data (due to the relatively small feature vector size) for a range of the number of clusters (for num_cluster in range(25,55)). With respect to the Silhouette method, we plot the Silhouette coefficient to determine the optimal number of clusters:



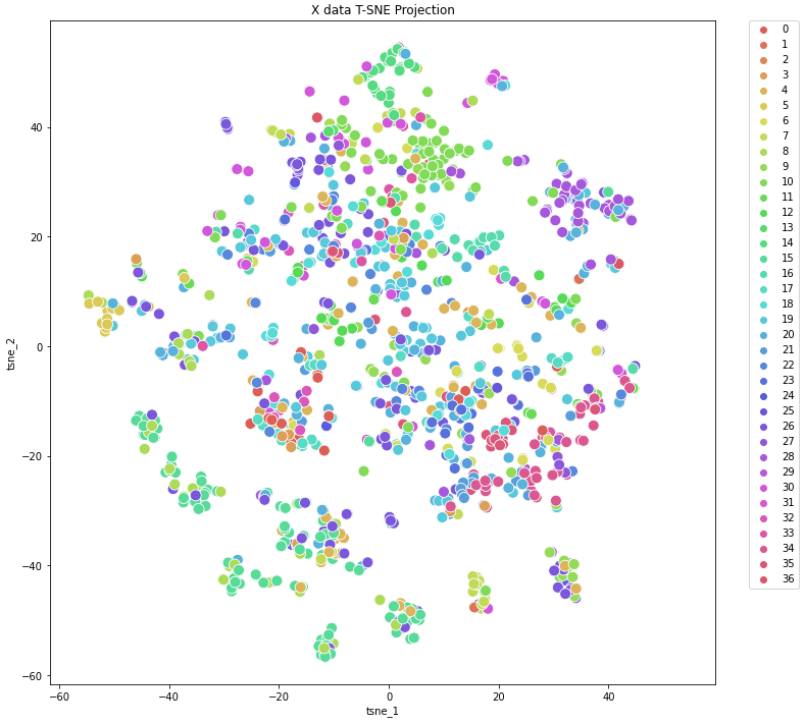
The best number of clusters is 46

Now, we perform clustering on X data (the non-encoded form of the aggregation of training and validation data) with the number of clusters = 46 and save the labels of clustering.

To gain a better understanding of the result, with the T-SNE method, we visualize the data with the result of clustering as label:



We also visualize the data with their labels (ground truth):



To answer the questions asked in Task 4, we need to provide a robust statistical analysis. By defining a table with clusters as its rows and the samples of classes (tissues) in its columns, we count the samples of different classes located in each cluster. Using the data in this table and extracting two dataframes, we can determine the following:

- Which cluster has the most samples from a specific tissue type (class)?
- What is the proportion of the number of tissue samples in that cluster to all the tissue samples in our data?
- Which tissue has the most samples in a specific cluster?
- What is the proportion of the number of samples of that tissue in that cluster to members of that cluster?

The tables are as below:

class(tissue)	maximum occur (? cluster)	% to all samples of the tissue
0	7	0.35
1	13	0.5
2	4	1
3	7	0.166666667
4	10	0.134328358
5	41	0.625
6	43	0.4
7	13	0.342857143
8	1	0.155555556
9	0	0.296296296
10	26	0.617977528
11	43	0.384615385
12	10	0.28125
13	16	0.625
14	3	0.967741935
15	15	0.157894737
16	30	0.419354839
17	7	0.433333333
18	6	0.162790698
19	6	0.208791209
20	14	0.117021277
21	17	0.416666667
22	10	0.2
23	39	0.48
24	0	0.333333333
25	6	0.6
26	0	0.112903226
27	20	0.285714286
28	14	0.5
29	5	0.5

30	16	0.22
31	23	0.5
32	36	0.333333333
33	6	0.333333333
34	20	0.4
35	34	0.214285714
36	18	0.6

cluster	maximum occur (? tissue)	% to num of members of the cluster
0	26	0.482758621
1	15	0.631578947
2	26	0.3
3	14	0.857142857
4	30	0.21875
5	30	0.285714286
6	19	0.240506329
7	17	0.317073171
8	23	0.272727273
9	20	0.428571429
10	4	0.15
11	15	0.736842105
12	5	0.428571429
13	7	0.631578947
14	28	0.6
15	15	0.692307692
16	30	0.186440678
17	34	0.320754717
18	28	0.35
19	4	0.166666667
20	34	0.486486486
21	15	0.666666667
22	16	0.181818182
23	20	0.171428571
24	15	0.555555556
25	4	0.205882353
26	10	0.743243243
27	15	0.714285714
28	30	0.545454545
29	4	0.6
30	16	0.565217391
31	16	0.454545455
32	19	0.444444444
33	15	0.8
34	19	0.23943662
35	9	0.333333333
36	0	0.24137931

37	15	0.545454545
38	20	0.4
39	19	0.185714286
40	15	0.529411765
41	5	0.5
42	15	0.384615385
43	6	0.279069767
44	8	0.4
45	15	0.333333333

For example, according to the second table, the 3rd cluster is mainly composed of the tissue with label 14 ('ascites'). For this tissue type, according to the first table, about 97% of the samples are located in this cluster.