

Olli Hantula

**PREDICTING SPATIALLY RESOLVED
GENE EXPRESSION CLUSTERS FROM
PROSTATE CANCER TISSUE IMAGES
WITH CONVOLUTIONAL NEURAL NET-
WORKS**

Bachelor's Thesis
Faculty of Medicine and Health Technology
Instructor: Antti Kiviaho
February 2024

ABSTRACT

Olli Hantula: Predicting Spatially Resolved Gene Expression Clusters from Prostate Cancer Tissue Images with Convolutional Neural Networks
Bachelor's Thesis
Tampere University
Biotechnology and Biomedical Engineering
February 2024

Gleason grading is the standard evaluation method of prostate cancer severity. However, analysis of prostate cancer's gene expression can lead to a more accurate and deeper understanding of cancer. New technologies, such as spatial transcriptomics, enable novel gene expression analysis methods. Machine learning techniques, such as dimensionality reduction and convolutional neural networks, introduce even more analysis approaches.

In this thesis, spatial transcriptomics data is classified into clusters based on gene expression and spatial proximity. Then, deep convolutional neural networks are trained to predict the clusters from hematoxylin & eosin-stained tissue images. This method introduces new insights into the histological differences associated with gene expression.

Spatial transcriptomics and convolutional neural networks are the most relevant concepts underlying the work and are introduced in the theoretical background of the work. In addition, the main performance metrics relevant to the work are presented in a comprehensive way to make the results understandable and easier to analyse.

The results of the study are examined using performance metrics such as accuracy and confusion matrixes, as well as visual interpretation. The results show that some of the clusters have clear histological features, making them easier to distinguish from the images by convolutional neural networks. Other clusters do not have clear histological features, or have features similar to those of other clusters, and are often confused by the convolutional neural networks. Further analysis of the gene expression differences between clusters could provide insight into the reasons behind histological differences, but this is outside the scope of this work.

However, it should be noted that only one sample is used for the cluster prediction in this thesis. Using multiple samples may result in greater histological differences between samples, even if the gene expressions are similar. This could confuse the convolutional neural networks, which would further decrease the performance of cluster prediction.

The cluster prediction method demonstrated in this thesis can be integrated into various gene expression analyses. This could provide an understanding of the consensus between prior results and histological observations.

Keywords: cluster prediction, clustering, spatial transcriptomics, convolutional neural networks, machine learning, gene expression, prostate cancer

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

TIIVISTELMÄ

Olli Hantula: Spatiaalisesti selvitettyjen geeniekspressioklustereiden ennustaminen eturauhassyövän kudostyypistä konvoluutiohermoverkkojen avulla
Kandidaatin tutkielma
Tampereen yliopisto
Bioteknologian ja biolääketieteen tekniikan tutkinto-ohjelma
Helmikuu 2024

Gleason-luokitus on standardi eturauhassyövän vakavuuden arviointimenetelmä. Eturauhassyövän geeniekspression analyysi voi kuitenkin johtaa vielä tarkempaan ja syvällisempään syövän ymmärrykseen. Uudet teknologiat, kuten spatiaalinen transkriptomiikka, mahdollistavat uudenlaiset geeniekspression analyysimenetelmät. Koneoppimistekniikat, kuten ulotteisuuden vähentäminen ja konvoluutiohermoverkostot, tuovat vielä lisää mahdollisia analyysimenetelmiä.

Tässä tutkielmassa spatiaalinen transkriptomiikkadata luokitellaan klustereihin geeniekspression ja spatiaalisen läheisyyden perusteella. Tämän jälkeen syviä konvoluutiohermoverkkoja koulutetaan ennustamaan klusterit hematoksyliini- ja eosinivärjätyistä kudostyypistä. Menetelmä tuo uutta tietoa geeniekspression liittyvistä histologisista eroista.

Spatiaalinen transkriptomiikka ja konvoluutiohermoverkot ovat olennaisimmat käsitteet työn taustalla, ja ne esitellään työn teoreettisissa taustatiedoissa. Lisäksi tärkeimmät työhön liittyvät suorituskymittarit esitellään kattavasti, jotta tulokset olisivat ymmärrettäviä ja helpommin analysoitavissa.

Tutkimuksen tuloksia tarkastellaan suorituskymittareilla, kuten täsmällisyydellä ja konfuusiomatriiseilla, sekä visuaalisella tulkinnalla. Tuloksista selviää, että osalla klustereista on selkeät histologiset piirteet, jolloin myös konvoluutiohermoverkot erottavat ne helpommin kuvista. Toisilla klustereilla taas ei ole histologisesti selkeitä piirteitä, tai piirteet ovat samankaltaisia muiden klusterien kanssa, jolloin myös konvoluutiohermoverkot sekoittavat klusterit usein keskenään. Klusterien välisten geeniekspressioerojen lisäanalyysi voisi antaa tietoa histologisten erojen syistä, mutta sitä ei käsitellä tässä työssä.

On kuitenkin huomioitava, että työssä käytetään ainoastaan yhtä näytettä klusterien ennustamiseen. Useampaa näytettä käytettäessä näytteiden väliset histologiset erot voivat olla suurempia, vaikka geeniekspressiot olisivat samankaltaiset. Tämä saattaisi sekoittaa konvoluutiohermoverkkoja, mikä edelleen heikentäisi klusterien ennustamisen suoriutumista.

Työssä esitelty klusterien ennustamismenetelmä voidaan sisällyttää erilaisiin geeniekspressioanalyysihin. Näin voidaan tutkia, miten aiemmat tutkimustulokset sopivat näytteen histologiaan.

Avainsanat: klusterien ennustus, klusterointi, spatiaalinen transkriptomiikka, konvoluutiohermoverkot, koneoppiminen, geeniekspressio, eturauhassyöpä

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck –ohjelmalla.

PREFACE

This thesis was made as part of Bachelor of Science Degree in Biotechnology and Biomedical Engineering at Tampere University. The thesis was also done as part of a research project in Professor Matti Nykter's Computational Biology research group.

I would like to thank Professor Matti Nykter for the chance to start my research career at his research group. I am grateful to Antti Kiviaho for the guidance of this thesis and throughout my work at the research group.

Tampere, 26.2.2024

Olli Hantula

CONTENTS

1. INTRODUCTION	1
2. THEORETICAL BACKGROUND	1
2.1 Spatial Transcriptomics	1
2.2 Machine Learning	2
2.2.1 Neural Networks	2
2.2.2 Convolutional Neural Network	4
2.2.3 <i>k</i> -Fold Cross Validation	5
2.2.4 Performance Metrics	5
3. MATERIALS & METHODS	8
3.1 Data & Preprocessing	8
3.2 Clustering	9
3.3 Cluster Prediction	9
4. RESULTS AND DISCUSSION	10
4.1 Clustering	10
4.2 Cluster Prediction	11
5. CONCLUSIONS	15
REFERENCES	16

ABBREVIATIONS

PCa	prostate cancer
CRPC	castration-resistant prostate cancer
AI	artificial intelligence
ML	machine learning
H&E	hematoxylin and eosin
mRNA	messenger RNA
NN	neural network
CNN	convolutional neural network
CM	confusion matrix
TP, TN, FP, FN	true positives, true negatives, false positives, false negatives
TPR, FPR	true positive rate, false positive rate
ROC curve	receiver operating characteristic curve
AUC	area under the ROC curve
OvR, OvO	one vs rest, one vs one
anndata	Annotated data
PCA	principal component analysis

1. INTRODUCTION

The severity of prostate cancer (PCa) is evaluated based on morphological examination and assigned a Gleason grade annotation by pathologists. Due to the increasing amount of manual labor on pathologists and the subjectivity of Gleason grading, artificial intelligence (AI) systems have been developed to automate Gleason grading, whilst also reducing the subjectivity of individual pathologists [1].

However, analyzing the gene expression of PCa can lead to a more accurate and deeper understanding of cancer. Detecting various gene expression profiles opens the possibility to develop more targeted anticancer treatments. Clustering cells together based on gene expression can reflect different components of the tissue, such as stroma, inflammation, and cancer. These different components might not be histologically visible. [2]

In this thesis, machine learning (ML) models are trained to predict gene expression clusters from hematoxylin and eosin (H&E) -stained images. This could lead to a better understanding of which gene expression profiles can be detected histologically from H&E-stained images.

2. THEORETICAL BACKGROUND

2.1 Spatial Transcriptomics

Spatial transcriptomics allows transcriptomics analysis while maintaining the spatial context of the tissue and was first introduced by Ståhl et. al in 2016. Their method included 1007 identifiable locations, also known as spots, with a diameter of 100 μm on a glass slide. Each spot contains unique DNA-barcoded surface probes with information including the spatial barcode and messenger RNA (mRNA) capture region. To capture spatial transcriptomics data, the sectioned tissue sample is placed on a glass slide, stained, and imaged. The tissue is then permeabilized to allow mRNA molecules from the tissue cells to hybridize with the mRNA capture region. [3] The structure of surface probes and localization of spots are shown in Figure 1.

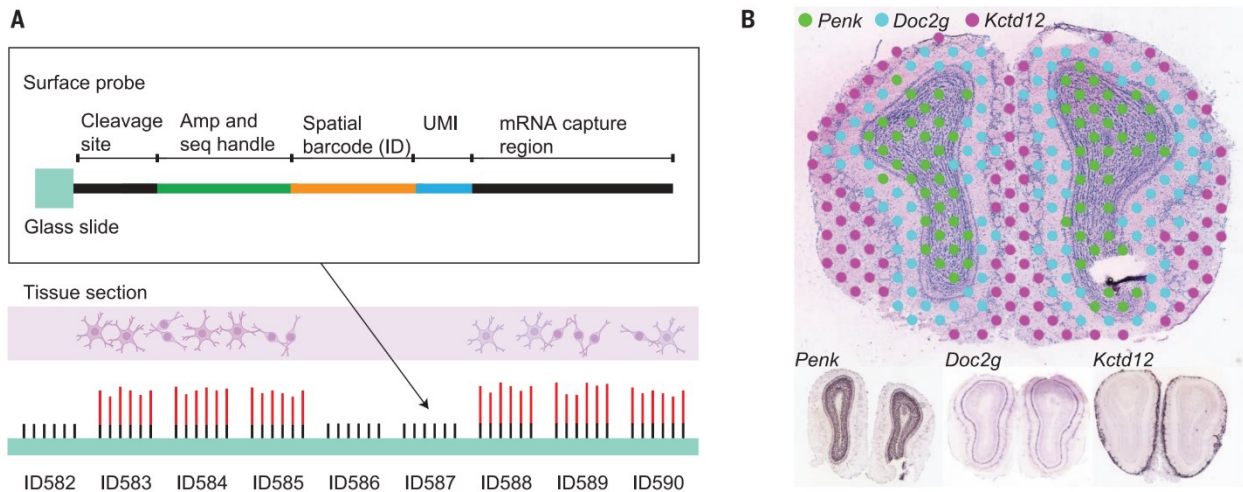


Figure 1: (A) Structure of surface probes and (B) spots shown on the tissue. [3]

The commercialized fresh frozen method from 10x Genomics Visium Spatial Gene Expression [4] was used to obtain the spatial transcriptomics data from prostate cancer tissue samples for this thesis.

2.2 Machine Learning

Machine learning (ML) is a multidisciplinary field that solves various problems by training algorithms that can generalize to new unseen data. Successful applications of ML include recognizing spoken words, driving autonomous vehicles, and playing backgammon. [5]

ML can be divided into supervised and unsupervised learning. In supervised learning, the data is already labeled, and the ML algorithm attempts to find underlying differences in the training data to replicate the correct labeling. In unsupervised learning, the data is not labeled, and it is the ML algorithm's task to discover similarities and structures in the given data.

In this thesis, both unsupervised and supervised learning are used. First, spatially weighted dimension reduction is used to label spatial transcriptomics spots to clusters based on their gene expression and spatial proximity. Then images from each spot and its corresponding label are fed into a deep convolutional neural network to find visually differentiating factors among the clusters.

The focus of this thesis is on cluster prediction. The unsupervised ML methods used in cluster labeling are mentioned in 3.2 but not further explained. The principles behind cluster prediction are described in the following sub-chapters.

2.2.1 Neural Networks

Neural networks (NNs) are ML algorithms that are based on the biological neural networks found in animals (Figure 2) [6].

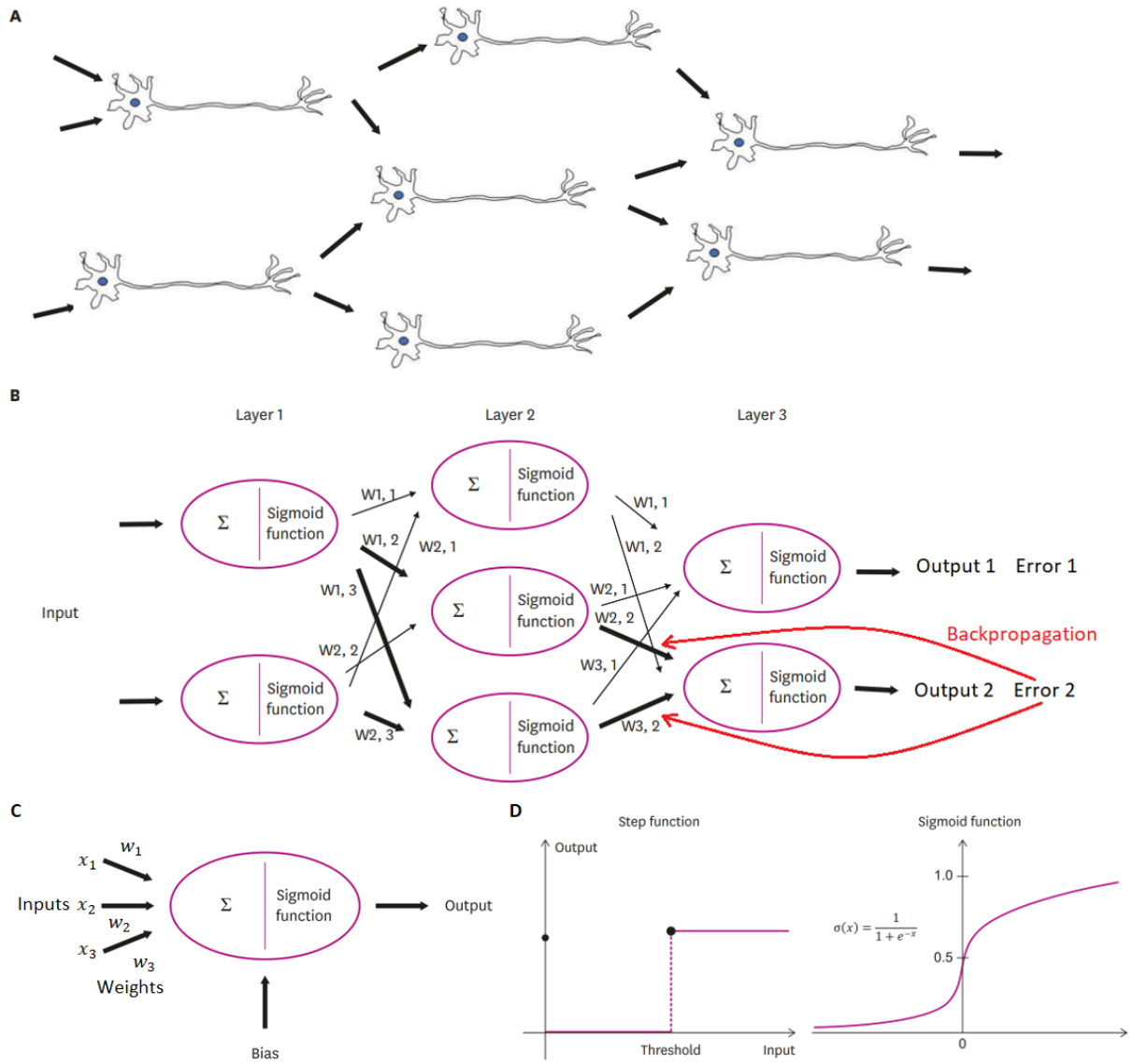


Figure 2: (A) A biological neural network, (B) an artificial neural network (NN), (C) a single neuron, and (D) two activation functions. Similar to their biological counterparts, NNs are composed of multiple interconnected neurons. The connections include weights that may strengthen or weaken the output of a neuron. A single neuron sums its weighted inputs and bias and provides an output according to its activation function. NNs learn by adjusting weights and biases in a process called backpropagation. Figure modified from [6].

A single neuron and two activation functions are depicted in Figure 2, C, and D. The main components of a neuron are inputs (x), weights (w), bias (b), and an activation function ($a = f(z)$). Neurons calculate a weighted sum (z) based on its inputs, weights, and bias according to equation (1).

$$z = x_1w_1 + x_2w_2 + \dots + x_nw_n + b \quad (1)$$

The final output of a neuron is then calculated according to its activation function. For example, a step function could be defined as $a = f(z) = \begin{cases} 1, & z > 1 \\ 0, & z \leq 1 \end{cases}$, in which case the output of the neuron would be 1 if z is over 1 and 0 otherwise. This output would then be one of the next neuron's inputs.

[6]

The final component of NNs is backpropagation, denoted in red in Figure 2B. Backpropagation is the learning mechanism of NNs. The difference between an output in the final layer and its true value is called error. Multiple weights and biases correspond to each output so they must be adjusted to minimize errors. However, calculating the best weights and biases directly is impossible because there are too many variables. Instead, they are adjusted iteratively in small steps with an algorithm called the gradient descent. [6]

The whole learning process of NNs consists of feeding training data through the network (forward propagation), and tweaking the weights and biases of neuron connections (backpropagation) until the NN converges to a point where the errors do not improve anymore. [6]

2.2.2 Convolutional Neural Network

Image classification tasks, such as cluster prediction, are ineffective with traditional NNs. If NNs are trained with the pixel values of images, everything from the image background to object orientation would directly affect the classification. By going through the image, multiplying pixels and their periphery with different numerical matrixes called kernels, distinct structures can be found in the images, such as vertical or horizontal edges. This process is called image convolution and results in an abstract representation of the image. [7]

In addition to convolution layers, CNNs consist of pooling layers which reduce the dimensions of CNNs by outputting only one value from a group of pixels. For example, a max pooling layer outputs the maximum value of a 2x2 pixel grid. [7] Optionally, augmentation layers can also be added to training data to artificially increase the amount of training samples. Some examples include random rotation, contrast, and brightness alterations. Augmentation layers are especially beneficial in limited datasets. [8]

The neurons in NNs are replaced by these convolution, pooling, and augmentation layers. This way the CNNs can learn to affiliate different patterns and structures to the classes it needs to predict and adjust the weights and biases between layers, similar to regular NNs, to reduce errors. Figure 3 depicts a typical structure of a CNN for clarification. [7]

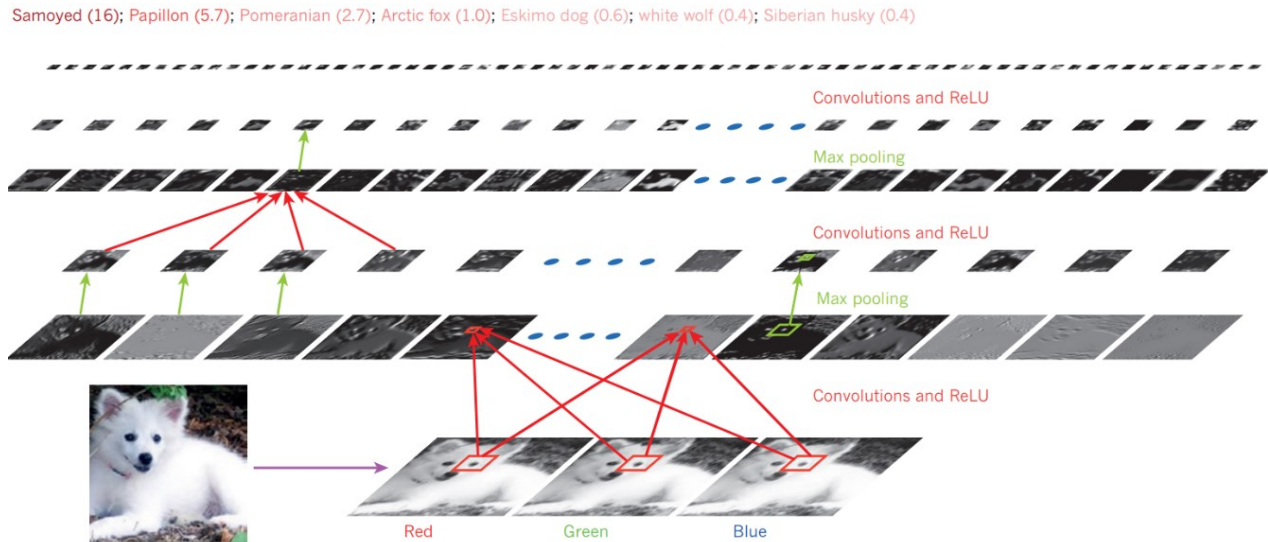


Figure 3: A typical CNN structure. The image consists of three color channels (red, green and blue). Each channel is fed through convolution layers to find varying patterns from the image. Rectified linear unit (ReLU) is used as an activation function. Max pooling layers take the maximum value from a grid of pixels (typically 2x2) to reduce image dimensions. The final layers are fully connected and 1-dimensional. A score is computed for each class from the final output values. The example scores are at the top of the figure. The class “Samoyed” has the highest score and thus is the predicted class. [7]

2.2.3 *k*-Fold Cross Validation

In ML problems, at least two, and preferably three different datasets are needed. A training set, to train the ML algorithm, a validation set (optional), to evaluate how the algorithm performs on unseen data throughout training, and lastly, a test set, to evaluate the final performance of the algorithm. Usually, more training samples lead to better results. Therefore, the train set might be 80% of the available data, leaving only 10% of the data for validation and test sets each.

For limited datasets, a method called *k*-fold cross-validation is often used. It allows recycling of the data by randomly dividing the dataset into *k* unique sets or folds with similar sizes and number of instances from each class. Then the ML algorithm can be trained *k* times, taking one of the folds as a test set each time. This way, the whole dataset can be used as a train, validation, and test set and the performance can be evaluated on the results from all the test sets. [9]

2.2.4 Performance Metrics

The field of ML is wide and suitable evaluation metrics may vary across studies. [10] For example, an ML algorithm that detects fraudulent ATM transactions might have a 99% accuracy, even if it classifies every transaction as non-fraudulent. The high accuracy is due to the minuscule fraction of fraudulent transactions. Thus, accuracy would not be a viable performance metric on its own.

In this thesis, the performance metrics consist of accuracy, recall, precision, F1 score, ROC curve, and AUC score. Hicks et al. [11] highlight the importance of reporting multiple metrics to gain an overall understanding of the ML model's performance and making the results more comparable to other models. They also explain the metrics used in this thesis for binary classification. However, the metrics can be calculated slightly differently in a multi-class setting. The differences are explained in [12]. The Receiver Operating Characteristic (ROC) curve and Area Under the ROC Curve (AUC) score are explained in [13], [14].

The first four metrics can be calculated from confusion matrixes (CM). In a multi-class setting, CMs can be done in two ways, which are depicted in Figure 4. The first method is to create binary CMs for each class (Figure 4A) and the second method is to create a single multi-class CM for all the classes (Figure 4B). The metrics are easier to calculate with binary CMs and the CMs of each class can be compared to each other. However, the distribution of false negatives to each class remains in multi-class CM, which can be used in analyzing how the ML model confuses different classes together.

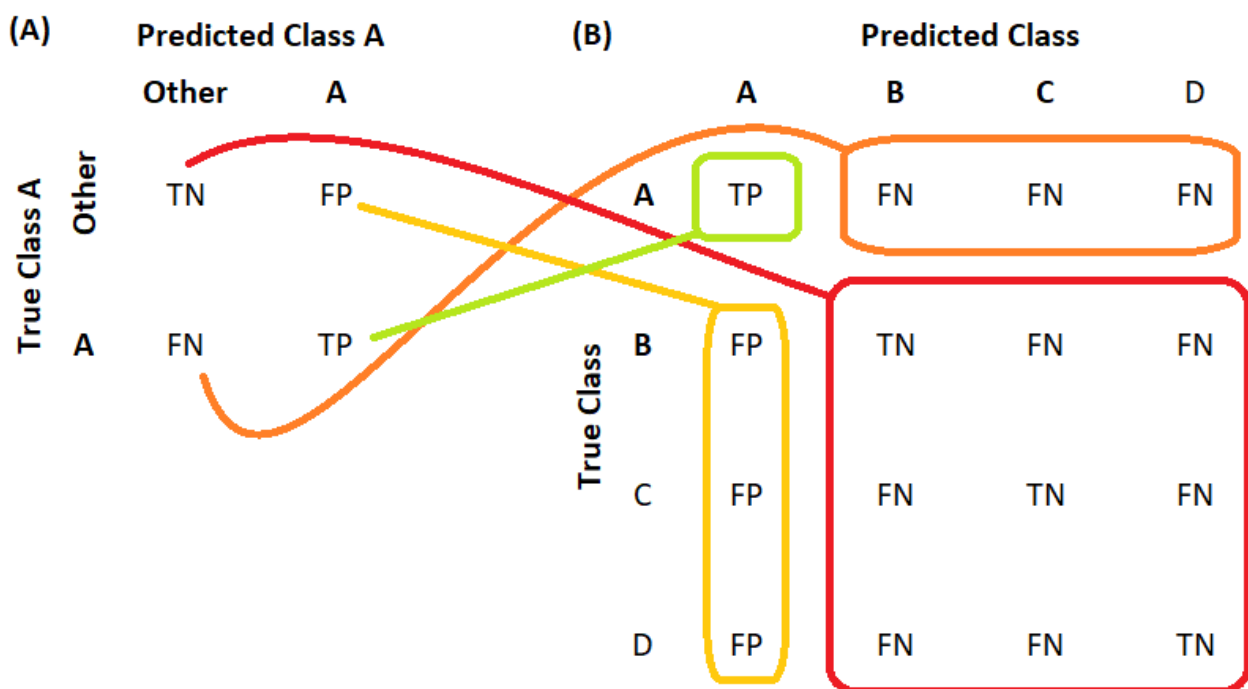


Figure 4: Confusion matrixes (CM) in multi-class classification. (A) A binary CM provides a concise “One vs Rest” depiction for individual classes. (B) A multi-class CM shows the true classes (rows) against the predicted classes (columns) for each class. Both CMs are from class A perspective. The colors indicate how the binary CM values are distributed to multi-class CM. Loss of information occurs especially with the binary CM’s true negatives (TN) marked in red.

A CM has four distinct values: the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). In a multi-class CM, the TNs, FPs, and FNs split into each class as depicted in Figure 4. For instance, consider a binary confusion matrix (CM) for class 3 in cluster prediction. If a spot from cluster 3 is correctly predicted as 3, it is a TP. However, if it is predicted

as anything other than 3, it is a FN. On the other hand, if a spot from any other cluster is incorrectly predicted as 3, it is a FP. Lastly, all other samples that are predicted as anything other than 3, regardless of their actual class, are considered TNs.

Accuracy is the number of correct predictions divided by the total number of predictions. This should not be calculated from a binary CM in multi-class classification, as the FNs of other classes are depicted as TNs. However, it can be calculated from the multi-class CM as

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

Recall, also known as True Positive Rate (TPR), is the fraction of instances correctly classified in a class out of all instances in that class, while precision is the fraction of instances correctly classified to a class out of all instances predicted to that class. Recall measures the model's ability to identify all instances of a particular class, while precision measures the model's ability to identify instances of a particular class correctly. Recall and precision are defined in equations (3) and (4)

$$recall_{class A} = \frac{TP_{class A}}{TP_{class A} + FN_{class A}} \quad (3)$$

$$precision_{class A} = \frac{TP_{class A}}{TP_{class A} + FP_{class A}} \quad (4)$$

Finally, the F1 score is the harmonic mean of precision and recall, thus it symmetrically represents both precision and recall in one metric. Similar to recall and precision, it is also calculated separately for each class. The F1 score is defined in equation (5).

$$F1\ score_{class A} = 2 \cdot \frac{precision_{class A} \cdot recall_{class A}}{precision_{class A} + recall_{class A}} \quad (5)$$

Accuracy, recall, precision, and F1 score are all bounded to [0,1], where a higher score expresses better performance. [11], [12]

The ROC curve and AUC score differ from the other metrics as they show the ML model's performance across different classification thresholds. These values cannot be calculated directly from the CMs. Instead, they are calculated from the prediction probabilities of the ML model.

The ROC curve is a plot of the TPR (or recall) against the False Positive Rate (FPR) at varying classification thresholds. If a binary classification model is expected to have a certain TPR or FPR threshold, for example, a minimum number of FNs (high TPR), the appropriate classification threshold can be chosen from the ROC curve. In multi-class classification, the predictions are typically made based on the highest probability among all classes. The probability could be tweaked for certain classes based on the ROC curves.

AUC score provides a single number to summarize the ML model's performance across all classification thresholds. Like all the other metrics, AUC is also bounded to [0,1]. An AUC score of 0.5 would indicate random guessing. Typically, a score above 0.8 is considered good and a score above 0.9 is considered great. [13]

As with recall and precision, ROC curves and AUC scores are also calculated for each class individually in this thesis. There are two ways of calculating them for multi-class classification, the "One vs Rest" (OvR), and "One vs One" (OvO). Similar to when transferring from binary CM to multi-class CM, the information between class pairs is lost when transferring from OvO to OvR. However, the OvO method would produce $C \times (C-1)$ curves, C being the number of classes, while OvR produces only C curves. As C is high in this thesis, only OvR ROC curves and AUC scores are calculated. Should there be major confusion between some of the classes in the multi-class CM, the OvO ROC curves could be produced for additional analysis. [14]

3. MATERIALS & METHODS

All programs used in this thesis have been uploaded to a public GitHub repository [15]. *clustering.py* includes initial data loading, preprocessing, and clustering, described in 3.1 and 3.2. Cluster prediction, described in 3.3, is done with *cluster_prediction.py* and the results in Chapter 4 are calculated with *results.py*.

3.1 Data & Preprocessing

The data processing is based on methods from the Python packages scanpy and squidpy [16], [17]. The spatial transcriptomics data is first loaded with `scanpy.read_visium()` -function from 10x Genomics output folder. The resulting data is an Annotated data (anndata) object [18] with the number of observations and variables ($n_{\text{obs}} \times n_{\text{vars}}$) being 3937 x 33538. The observations consist of spatial barcode indexing, spot coordinates, and a Boolean value of 1 or 0, indicating whether the spot is in the tissue area or not. The variables consist of gene name indexing, Ensembl gene identifiers, feature types (gene expression), and genome (GRCh38). The high- and low-resolution tissue images are provided in unstructured data (uns).

Data preprocessing consists of quality control and normalization which is important in removing cells with low quality or low complexity that would likely add noise to the analysis. In preprocessing, rare genes that appear in less than 5 cells and cells that have less than 500 gene counts are filtered out. Then each cell is normalized by total counts over all genes, so that every cell has the same

total count after normalization. The preprocessing was done using a custom function “qc_and_normalize” [19]. After preprocessing the number of gene counts (n_counts) is added to observations and the number of cells in which each gene is expressed (n_cells) is added to variables.

3.2 Clustering

Clustering was also done using a custom function “spatially_aware_clustering” [19]. First, the data is scaled to zero mean and unit variance. From the scaled features, 2000 most variable genes are selected for downstream analysis. A dimensionality reduction is calculated using principal component analysis (PCA) [20] and a neighborhood embedding is calculated based on the PCA values. Next, spatial neighbors are marked for each spot and a joint adjacency is calculated by summing the gene expression-based neighborhood connectivities and the spatial neighbors multiplied by a proximity weight parameter. The final clustering is done with the Leiden community detection algorithm [21] based on joint adjacency and a given resolution.

The reason a proximity weight is included is that the resolution of the 10x Genomics Visium system is not on a single-cell level. Instead, a single spot with a 55 μm diameter can have mRNAs from tens of different cells. The proximity weight compensates for the random distribution of mRNAs and includes spatial information in dimension reduction to create spatially coherent structures. The resolution parameter given to the Leiden algorithm affects the number of different clusters detected with higher resolution yielding more clusters.

For this thesis, two clusterings are made with different weights and resolutions. This is done to illustrate the differences these parameters cause and to compare them in cluster prediction performance.

3.3 Cluster Prediction

In this thesis a pre-trained deep CNN model called ResNet-50 [22] is trained further (also called transfer learning) to predict cluster labels from H&E-stained tissue images. The method is modified from the squidpy tutorial “Predict cluster labels spots using Tensorflow” [23].

First, the data resulting from clustering and a high-resolution image of the tissue are loaded. The data is split using 10-fold cross-validation, with 81% training, 9% validation, and 10% test samples.

Datasets are created with *create_dataset*-function, which combines each cluster label with the corresponding image crop from the high-resolution tissue image. The datasets are created in batches of 64 to preserve memory. Processing layers resize the images to 128 x 128 pixels and rescale them to pixel values between 0 and 1 (from the initial 0 to 255). Additionally, the validation dataset is shuffled randomly, and the training dataset is shuffled and augmented. Augmentation layers

include randomly flipping the image horizontally and vertically, randomly rotating the image, and randomly increasing or decreasing the image contrast.

The ResNet-50 model is loaded with pre-trained weights and processing layers, augmentation layers (for the training dataset), and a dense (fully connected) output layer are added. The model is trained for 50 to 150 epochs (or rounds). The training stops after 50 epochs if the validation accuracy does not increase in 10 consecutive epochs and the best result is restored. The learning rate is set to 0.00001 with an Adam optimizer. Categorical cross-entropy is used as the loss function.

After training, the test set prediction probabilities are saved for ROC curve generation. The final predictions based on the maximum probabilities are added to the original anndata observations as 'cluster_predictions' and the anndata is saved. The results are calculated with *results.py*.

4. RESULTS AND DISCUSSION

4.1 Clustering

In total 36 clusterings were computed with spatial weights varying from 0.0 to 0.5 and Leiden algorithm resolution varying from 0.6 to 1.1, both with 0.1 intervals. From these, the two spatially most coherent results were chosen, which are illustrated in Figure 5. Sample 1 in the middle has a weight of 0.1 and a resolution of 0.7, while sample 2 on the right has a weight and resolution of 0.3 and 1.0 respectively. It is important to note that some similar areas belong to different clusters in these samples, for example, the distinct cluster on the right is cluster 1 in sample 1 but cluster 0 in sample 2. This is because the clustering function arranges the clusters according to their sizes, with cluster 0 having the greatest number of spots.

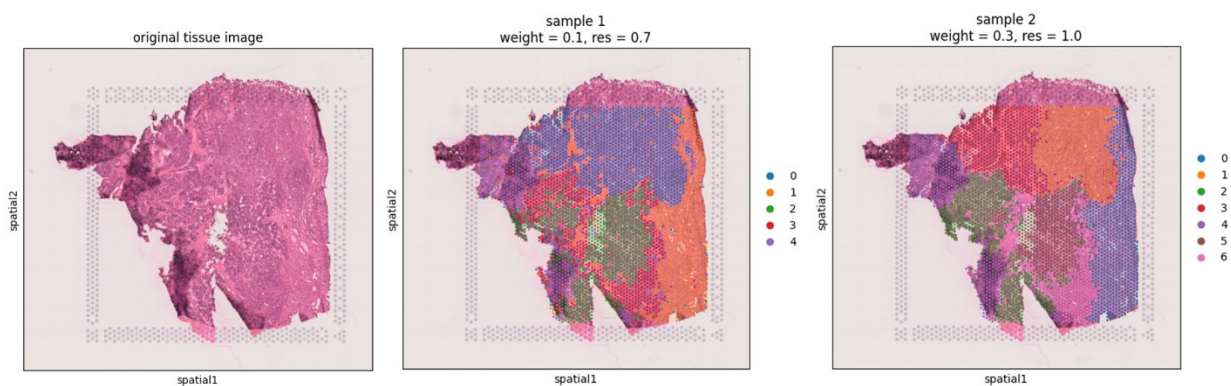


Figure 5: The original tissue image (left) and the two spatially most coherent clustering results chosen for further analysis. Similar structures can be distinguished between the underlying histology and gene expression clusterings.

A higher weight produces more spatially distinct clusters, but excessive weight may hide important smaller details in the sample. For example, some parts of clusters 0 and 1 overlap, revealing some histologically distinct structures in sample 1.

A higher resolution produces more clusters, with sample 1 having only 5 clusters, while sample 2 has 7. It seems that cluster 0 in sample 1 has split into two similar size clusters 1 and 3 in sample 2 and cluster 2 in sample 1 has split into clusters 2 and 5 in sample 2. A good aim would be to have as many clusters as possible that still have significant differences in gene expression. Without further analysis between the clusters, it is impossible to know if the gene expression differences are significant or not. However, that is outside the scope of this thesis, and instead, the focus is on the differences in cluster prediction performance between the two samples.

Table 1 shows the number of spots belonging to each cluster in both samples. In sample 1, cluster 0 is almost twice as big as cluster 4, in sample 2 cluster 0 is about 1.69 times bigger than cluster 6. However, a class imbalance this small should not have a significant impact on prediction.

Table 1: Number of spots in each cluster.

	cluster 0	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5	cluster 6
sample 1	1120	841	780	630	562	-	-
sample 2	714	639	590	585	539	442	424

4.2 Cluster Prediction

The cluster prediction results are introduced one by one for both samples to make comparing them easier. The original clusterings and cluster predictions for both samples are illustrated in Figure 6. Despite the apparent disorder in the cluster predictions of sample 2 compared to those of sample 1, the overall accuracies are similar. Specifically, sample 1 has an accuracy of 0.6428, while sample 2 has a slightly lower accuracy of 0.6206.

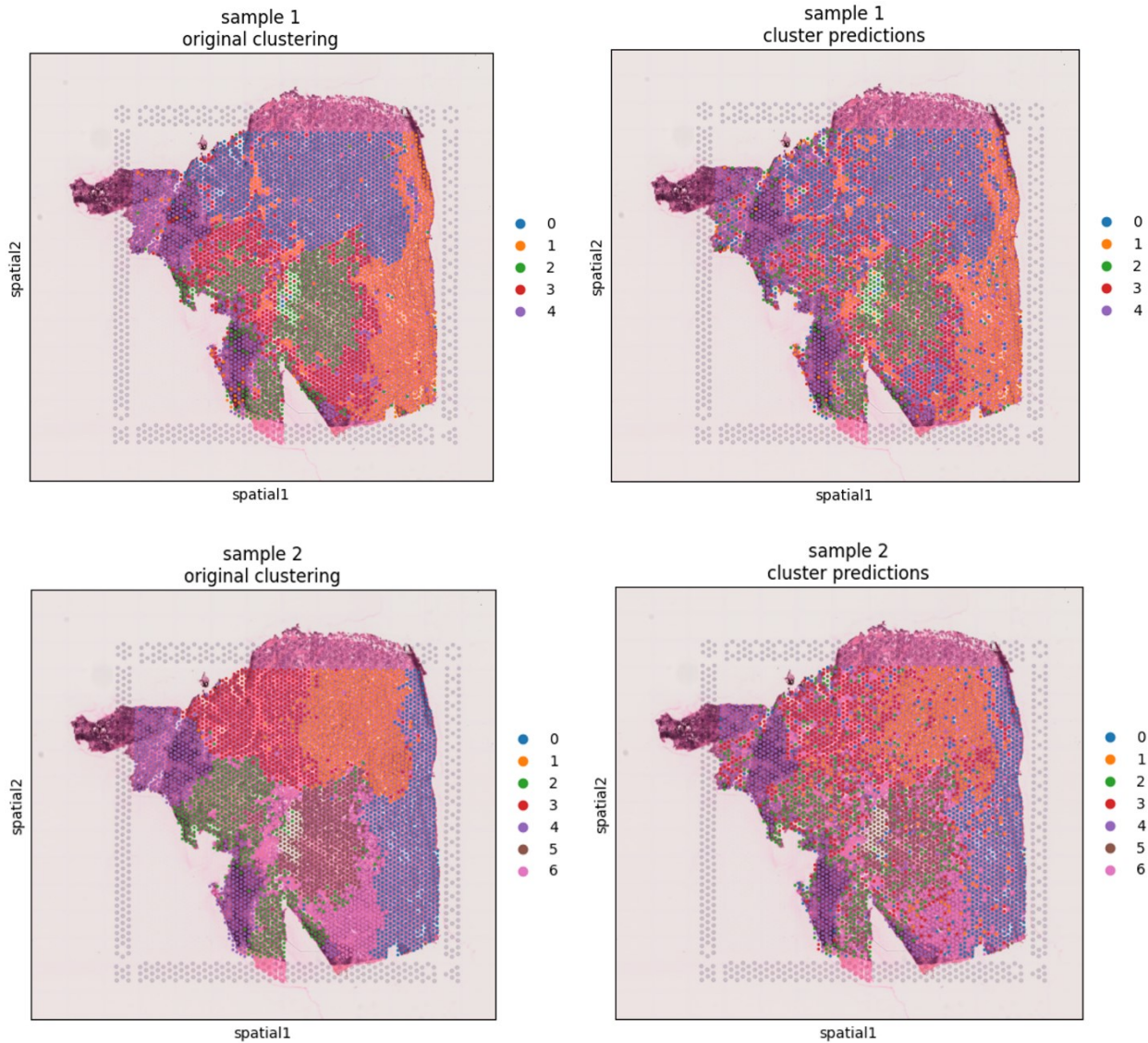


Figure 6: Original clusterings and cluster predictions.

The original, plain tissue sample is illustrated in Figure 7 with some areas of interest marked. The 1st area marked with light blue has stroma which is lighter than the rest of the sample and as such more easily recognized by the CNNs. The 2nd area marked with green is also visually distinguishable and recognized by the CNNs. The 3rd area marked with navy blue is clustered together in cluster 4 with the darker areas in both samples. However, it is visually very different and thus predicted falsely in both samples.

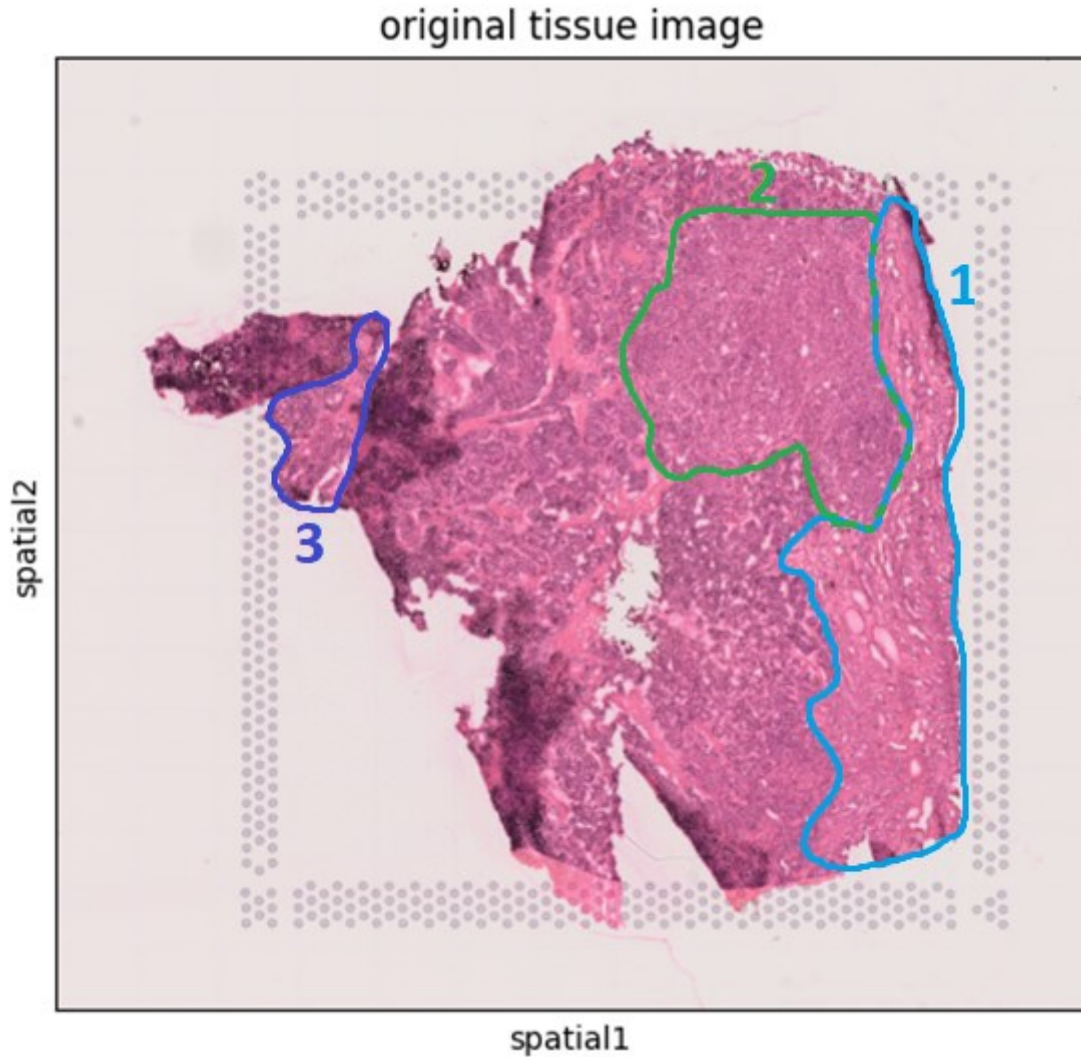


Figure 7: Plain tissue image with some areas of interest marked.

The binary confusion matrixes and all performance metric values for both samples are collected into Tables 2 to 4. The metric values support that the 1st and 2nd areas in Figure 7 were predicted with the highest and second-highest accuracy, respectively. Figure 6 suggests that cluster 4 is predicted rather accurately. However, the visually differing 3rd area complicates the predictions which can also be seen from the slightly lower performance values in the tables. Cluster 3 had the lowest prediction performance in sample 1. Likewise, in sample 2 the lowest prediction performances were related to clusters 2, 3, and 6.

Table 2: Sample 1 binary confusion matrixes

confu- sion matrix		cluster 0		cluster 1		cluster 2		cluster 3		cluster 4	
TN	FP	2393	420	2909	183	2941	212	2979	324	3105	266
FN	TP	265	855	190	651	334	446	390	240	226	336

Table 3: Sample 1 performance metric values

metric	cluster 0	cluster 1	cluster 2	cluster 3	cluster 4	mean
precision	0.6706	0.7806	0.6778	0.4255	0.5581	0.6225
recall	0.7634	0.7741	0.5718	0.3810	0.5979	0.6176
F1 score	0.7140	0.7773	0.6203	0.4020	0.5773	0.6182
AUC	0.8892	0.9441	0.8958	0.7660	0.8600	0.8710

Table 4: Sample 2 binary confusion matrixes

confu- sion matrix		cluster 0		cluster 1		cluster 2		cluster 3		cluster 4		cluster 5		cluster 6	
TN	FP	3091	128	3120	174	3042	301	2944	404	3226	168	3357	134	3326	183
FN	TP	138	576	189	450	344	246	249	336	161	378	198	244	213	211

Table 5: Sample 2 performance metric values

metric	cluster 0	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5	cluster 6	mean
precision	0.8182	0.7212	0.4497	0.4541	0.6923	0.6455	0.5355	0.6166
recall	0.8067	0.7042	0.4169	0.5744	0.7013	0.5520	0.4976	0.6076
F1 score	0.8124	0.7126	0.4327	0.5072	0.6968	0.5951	0.5159	0.6104
AUC	0.9637	0.9425	0.8139	0.8470	0.9365	0.9300	0.8704	0.9006

The multi-class confusion matrixes are illustrated in Figure 8. For sample 1, cluster 3 is predicted worst and it is most often confused with clusters 0 and 2. For sample 2 the worst performing cluster 2 is most often falsely predicted as cluster 3, 5, or 4. These results can also be observed in Figure 6.

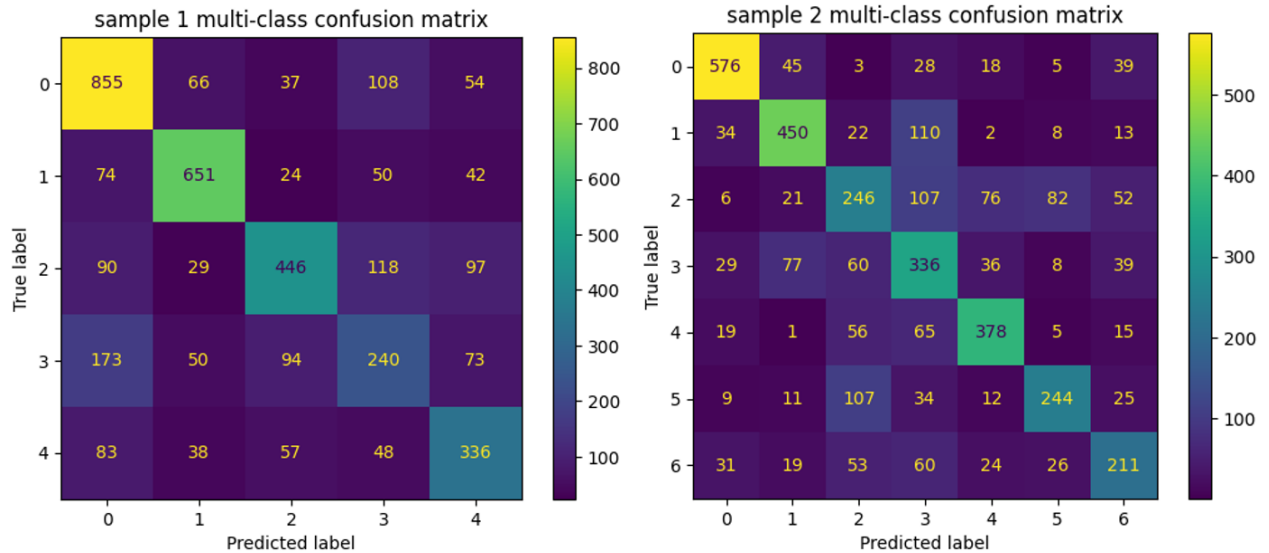


Figure 8: Multi-class confusion matrixes.

Finally, the ROC curves for all clusters in both samples are depicted in Figure 9. For sample 1, cluster 3 is again performing the worst. For sample 2 clusters 2, 3, and 6 are performing the worst which is also supported by previous results.

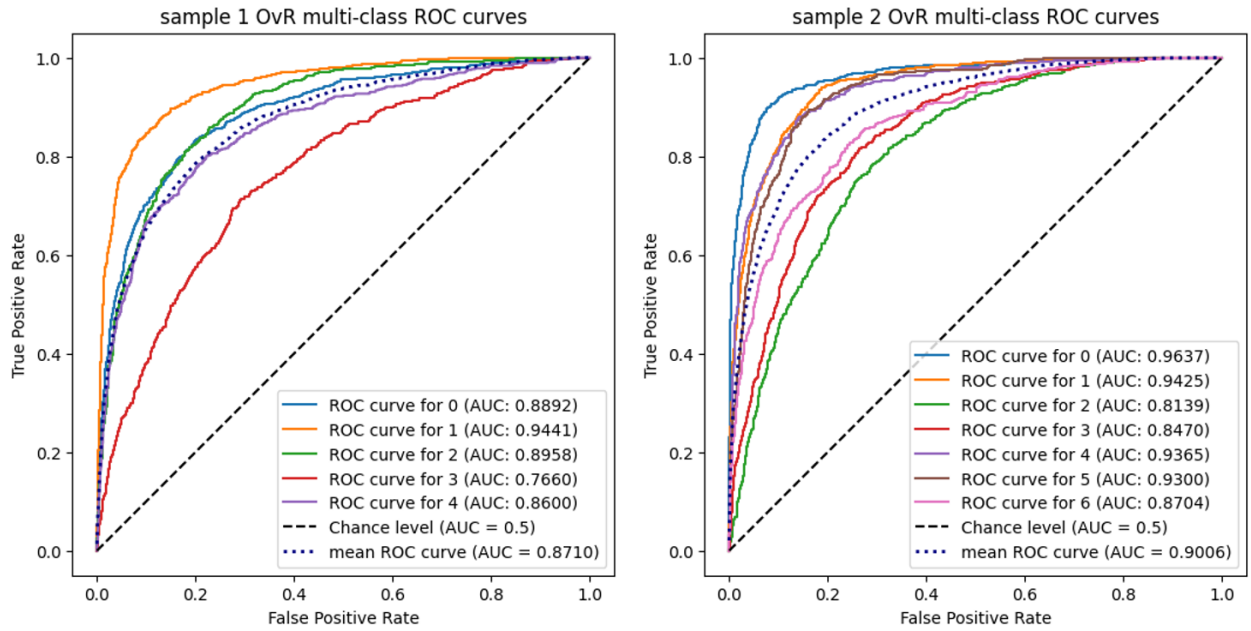


Figure 9: Multi-class ROC curves.

Based on these results, the lowest-performing clusters do not have histologically distinct features and as such are hard to distinguish from other clusters. On the other hand, the best-performing clusters have a clear, distinct histology, which is not easily confused with other clusters.

5. CONCLUSIONS

Overall, the CNNs performed well in cluster prediction and the results revealed the worst and best-performing clusters. Further analysis of the clusters' gene expressions could provide insight into the reasons behind the histological differences, or the lack of them.

However, it must also be highlighted that the cluster prediction was performed with only one sample. With multiple samples, the histological differences between samples could be higher, even though the gene expressions were similar. This could confuse the CNNs, thus lowering the performance of cluster prediction.

The cluster prediction approach introduced in this thesis can be easily integrated into various gene expression analyses. It can provide additional information on the sample histology and whether it generalizes well with other results such as unsupervised gene expression clustering.

REFERENCES

- [1] P. Ström *et al.*, “Artificial intelligence for diagnosis and grading of prostate cancer in biopsies: a population-based, diagnostic study,” *Lancet Oncol.*, vol. 21, no. 2, pp. 222–232, Feb. 2020, doi: 10.1016/S1470-2045(19)30738-7.
- [2] E. Berglund *et al.*, “Spatial maps of prostate cancer transcriptomes reveal an unexplored landscape of heterogeneity,” *Nat. Commun.*, vol. 9, no. 1, Art. no. 1, Jun. 2018, doi: 10.1038/s41467-018-04724-5.
- [3] P. L. Ståhl *et al.*, “Visualization and analysis of gene expression in tissue sections by spatial transcriptomics,” *Science*, vol. 353, no. 6294, pp. 78–82, Jul. 2016, doi: 10.1126/science.aaf2403.
- [4] “Spatial Gene Expression for Fresh Frozen - Official 10x Genomics Support,” 10x Genomics. Accessed: Nov. 16, 2023. [Online]. Available: <https://www.10xgenomics.com/support/spatial-gene-expression-fresh-frozen>
- [5] T. M. Mitchell, *Machine Learning*. in McGraw-Hill series in computer science. New York: McGraw-Hill, 1997. Accessed: Jan. 24, 2024. [Online]. Available: <https://www.cs.cmu.edu/~tom/mlbook.html>
- [6] S.-H. Han, K. W. Kim, S. Kim, and Y. C. Youn, “Artificial Neural Network: Understanding the Basic Concepts without Mathematics,” *Dement. Neurocognitive Disord.*, vol. 17, no. 3, pp. 83–89, Sep. 2018, doi: 10.12779/dnd.2018.17.3.83.
- [7] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, Art. no. 7553, May 2015, doi: 10.1038/nature14539.
- [8] A. Mikołajczyk and M. Grochowski, “Data augmentation for improving deep learning in image classification problem,” in *2018 International Interdisciplinary PhD Workshop (IIPHDW)*, May 2018, pp. 117–122. doi: 10.1109/IIPHDW.2018.8388338.
- [9] T.-T. Wong and P.-Y. Yeh, “Reliable Accuracy Estimates from k-Fold Cross Validation,” *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 8, pp. 1586–1594, Aug. 2020, doi: 10.1109/TKDE.2019.2912815.
- [10] G. S. Handelman *et al.*, “Peering Into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods,” *Am. J. Roentgenol.*, vol. 212, no. 1, pp. 38–43, Jan. 2019, doi: 10.2214/AJR.18.20224.
- [11] S. A. Hicks *et al.*, “On evaluation metrics for medical applications of artificial intelligence,” *Sci. Rep.*, vol. 12, no. 1, p. 5979, Apr. 2022, doi: 10.1038/s41598-022-09954-8.
- [12] “Accuracy, precision, and recall in multi-class classification.” Accessed: Nov. 13, 2023. [Online]. Available: <https://www.evidentlyai.com/classification-metrics/multi-class-metrics>
- [13] “How to explain the ROC AUC score and ROC curve?” Accessed: Dec. 08, 2023. [Online]. Available: <https://www.evidentlyai.com/classification-metrics/explain-roc-curve>
- [14] “Multiclass Receiver Operating Characteristic (ROC),” scikit-learn. Accessed: Nov. 13, 2023. [Online]. Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html
- [15] “Bachelor-s_thesis/ at main · OlliHantula/Bachelor-s_thesis.” Accessed: Jan. 08, 2024. [Online]. Available: https://github.com/OlliHantula/Bachelor-s_thesis/tree/main
- [16] F. A. Wolf, P. Angerer, and F. J. Theis, “SCANPY: large-scale single-cell gene expression data analysis,” *Genome Biol.*, vol. 19, no. 1, p. 15, Feb. 2018, doi: 10.1186/s13059-017-1382-0.
- [17] G. Palla *et al.*, “Squidpy: a scalable framework for spatial omics analysis,” *Nat. Methods*, vol. 19, no. 2, Art. no. 2, Feb. 2022, doi: 10.1038/s41592-021-01358-2.
- [18] I. Virshup, S. Rybakov, F. J. Theis, P. Angerer, and F. A. Wolf, “anndata: Annotated data.” bioRxiv, p. 2021.12.16.473007, Dec. 19, 2021. doi: 10.1101/2021.12.16.473007.
- [19] “spatial/scripts/utils.py at master · akiviaho/spatial,” GitHub. Accessed: Dec. 14, 2023. [Online]. Available: <https://github.com/akiviaho/spatial/blob/master/scripts/utils.py>
- [20] I. T. Jolliffe and J. Cadima, “Principal component analysis: a review and recent developments,” *Philos. Trans. R. Soc. Math. Phys. Eng. Sci.*, vol. 374, no. 2065, p. 20150202, Apr. 2016, doi: 10.1098/rsta.2015.0202.
- [21] V. Traag, L. Waltman, and N. J. van Eck, “From Louvain to Leiden: guaranteeing well-connected communities,” *Sci. Rep.*, vol. 9, no. 1, p. 5233, Mar. 2019, doi: 10.1038/s41598-019-41695-z.
- [22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition.” arXiv, Dec. 10, 2015. Accessed: Sep. 01, 2023. [Online]. Available: <http://arxiv.org/abs/1512.03385>
- [23] “Predict cluster labels spots using Tensorflow — squidpy documentation.” Accessed: Dec. 14, 2023. [Online]. Available: https://squidpy.readthedocs.io/en/latest/notebooks/tutorials/tutorial_tf.html