A comparison between fully-supervised and self-supervised deep learning methods for tumour classification in digital pathology data

Elsa Jonsson

Dept. of Computer Science, Electrical and Space Engineering
Luleå University of Technology
Luleå, Sweden

June 4, 2022

Abstract

Whole Slide Images (WSIs) are digital scans containing rich pathology information. There are many available WSI datasets that can be used for a wide range of purposes such as diagnostic tasks and analysis, but the availability of labeled WSI datasets is very limited since the annotation process is both very costly and time consuming. Self-supervised learning is a way of training neural networks to learn and predict the underlying structure of input data without any labels.

AstraZeneca have developed a self-supervised learning feature extractor, the Drug-development Image Model Embeddings (DIME) pipeline, that trains on unlabeled WSIs and produces numerical embedding representations of WSI.

This thesis applies the DIME-embeddings to a binary tumour classification task on the annotated Camelyon16 dataset by using the DIME pipeline as a feature extractor and train a simple binary classifier on the embedding representations instead of the WSI patches. The results are then compared to previous fully-supervised learning approaches to see if the embedding features generated by the DIME pipeline are sufficiently predictive with simple classifiers for the downstream task of binary classification.

The DIME embeddings were trained using Logistic Regression, Multi-Layer Perceptron and Gradient Boosting and the best performing model, a Multi-Layer Perceptron neural network trained on the DIME embeddings produced with an inpainting algorithm achieved a patch-level classification accuracy of 97.3%. This is very competitive results to the fully-supervised algorithms trained on the Camelyon16 dataset, beating some of them, while having a 1.1% gap to the best performing fully-supervised model. In addition to this, the performance of the DIME embeddings on reduced training sets also shows that the features captured in the DIME embeddings are sufficient.

Acknowledgements

First, I want to thank AstraZeneca for providing me with a really interesting and fun thesis project and for assigning me with two incredible supervisors.

I could not give enough thanks to Kajsa Svanlind for giving me the opportunity to join her team to do my thesis project and for her continuous support and engagement. Without your guidance and encouragement this thesis would not be possible.

I would also like to thank Feng Gu for the constant technical help and for always making time to provide support whenever it was needed.

Thanks also goes out to my supervisor at Luleå University of Technology, Kanjar De, for providing valuable insights regarding how a thesis project is executed.

Lastly, a special thanks goes out to my family and friends, especially my parents, for their constant support and interest during this thesis project and during my entire studies at Luleå University of Technology.

List of Figures

List of Tables

Acronyms

AP Average Precision. 11

AUC Area Under Curve. 11

DIME Drug-development Image Model Embeddings. 1

fn False Negative. 10

fp False Positive. 10

GB Gradient Boosting. 10

H&E Hematoxylin and Eosin. 8

LR Logistic Regression. 9

MAP@R Mean Average Precision at R. 7

MLP Multi-Layer Perceptron. 9

MPP Microns Per Pixel. 5

PRC Precision Recall Curve. 11

ROC Receiver Operating Characteristic. 11

TCGA The Cancer Genome Atlas. 3

tn True Negative. 10

tp True Positive. 10

WSIs Whole Slide Images. 1

Contents

1 Introduction

1.1 Background

Whole Slide Images (WSIs) are digital images containing rich pathology information. They are retrieved by scanning microscopic slides with high-resolution scanners enabling a wide area of purposes such as diagnostic tasks and analysis. WSIs are extremely large images with very high resolutions which poses a problem for evaluation at slide-level. The current capacity of computers cannot handle the computation necessary for evaluation of WSIs, which results in that the widely used approach at the moment is to divide the WSIs into smaller patches for use in different tasks, shown in Figure 1. This contributes to the fact that the availability of labeled WSI datasets is very limited since the annotation process is both very costly and time consuming. Since a WSI can contain patches of both healthy tissue and tissue with disease, a label for the entire WSI cannot be used for evaluation at patch-level. To enable fully-supervised learning with a patch-level approach, each pixel of a WSI has to be annotated by a pathologist [1, 2].

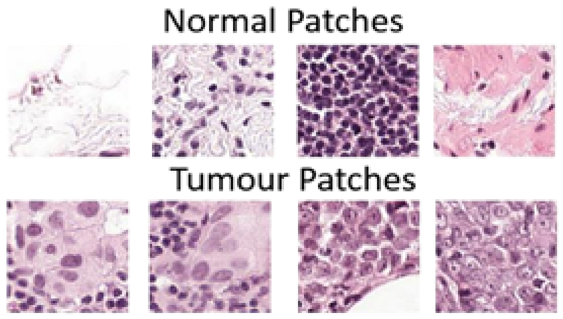


Figure 1: WSI patches from the Camelyon16 dataset

Self-supervised learning is a way of training neural networks to learn and predict the structure of input data without any labels. Unsupervised learning is the more common notation for learning without the use of labeled data, and self-supervised learning can be seen as a sub-part of unsupervised learning that is mainly focused on tasks that are usually solved with supervised learning such as classification tasks. Training with self-supervised learning is done by predicting certain parts of the input data from other parts of that same data. Self-supervised learning creates the ability to train neural networks on huge amount of data, which can create better neural networks that captures more subtle features and more uncommon cases. Self-supervised learning methods have been applied to many different areas with great success, such as in image recovery and colourisation, context filling for texts and voice recordings and video motion prediction [3, 4].

AstraZeneca have created a self-supervised learning pipeline trained on unlabeled WSI to create embedding representations of WSIs. This pipeline is called Drug-development Image Model Embeddings (DIME) and is a way of pre-processing that uses two different self-supervised learning approaches called inpainting and contrastive learning to produce and cluster powerful embeddings that capture high level features of digital pathology data without the need for labeled data [1].

1.2 Motivation

In many different areas of image analysis of pathological data there has been increasing interest in development of machine learning methods. Development of digitized diagnosing enables cost-efficient, fast and high-performing diagnosis to assist pathologists. As well as saving time and cost, it can also increase accuracy [5, 6].

However, the problem is that digital pathological data analysis with the classic fully-supervised methods is not very applicable. Many of the WSI datasets are not annotated and there are therefore many datasets of WSIs that would be rendered useless if only fully-supervised methods were considered. Using self-supervised learning as a pre-processing tool to create the DIME embeddings can be useful for many different downstream tasks. The benefit of producing embeddings of the image-patches before applying to different downstream tasks is that since the embeddings are trained with self-supervised learning they can be trained on a large training set that includes unlabelled data containing a lot of diversity regarding disease, tissue type and other important features. By training on such diverse data, the embeddings will be applicable to many different tasks since high level features are being captured. It can also aid datasets that have a limited amount of data, such as for early stages of clinical trials [1].

Another motivation is the benefits of tumour detection in WSIs. The Camelyon16 dataset consists of WSIs of breast cancer metastases in lymph nodes. Lymph nodes are small glands that are located all over the body. Cancers can spread into the lymph nodes and spread to different parts of the body. Detection of metastases in lymph nodes is one of the most important prognostic variable in breast cancer. Patients tested positive for metastatic cancer in the lymph nodes will receive a more aggressive treatment. Metastases diagnosing is however a very costly and time consuming procedure for a pathologist, and is prone to error. By implementing a tumour classifier to aid in diagnosing this could help save time and money for different diagnosing companies and help with early diagnosing to save lives [7].

1.3 Problem definition

The embeddings produced by the DIME pipeline are numerical representations that extracts features of importance from the image patches created from the WSIs. They contain more powerful information about high-level features than the original images which hopefully will provide better results in different downstream tasks. These embeddings are evaluated with clustering metrics which reward well clustered embedding spaces [1]. Even though this is a good way of evaluating the quality of features captured by the DIME pipeline, it has no indication about the performance of the embeddings in the downstream tasks that they are to be applied to.

One of the downstream tasks that the embeddings can be applied to is histopathologic image classification. By developing a binary classifier for patch-level that trains on the DIME embeddings of the patches instead of the patch images directly, a comparison between fully-supervised methods and self-supervised methods can be done. It also makes it possible to draw conclusions about the performance of the DIME embeddings on classification tasks. The research question of this thesis is:

Are embedding features generated by pre-trained self-supervised learning models sufficiently predictive with simple classifiers when compared to end-to-end fully supervised deep learning methods?

1.4 Delimitations

The biggest delimitation of this project is that the embeddings are provided by AstraZeneca, meaning the thesis report only focuses on the theory behind creating them as well as the use of them in the downstream classification task by implementing a simple classifier and comparing the results to fully-supervised methods.

The thesis report will only implement and evaluate a patch-level classifier since there are results from fully-supervised methods at patch-level. Since the results between patch-level and slide-level classifications varies because of other parameters than the embeddings, this part was not relevant for the evaluation done in this report. A comparison of results of the patch-level classifier should be sufficient to draw conclusions about the performance of the DIME embeddings. This limitation provides more time for the development of the patch-level classifier which could improve the results. However, it makes comparison with other implementations of the Camelyon16 challenge more difficult since some do not report the results of their patch-level classifier.

The DIME embeddings are only trained on The Cancer Genome Atlas (TCGA) dataset and have never been trained on the Camelyon16 dataset. Often in fully-supervised classification tasks the network is trained on the same dataset that is being evaluated. The Camelyon16 dataset and the TCGA dataset can differ fundamentally in ways that influence the result unknowingly. But the TCGA dataset is a big and diverse dataset which hopefully means it captures features important for the Camelyon16 dataset and classification there. One benefit from this limitation is that the performance of the DIME embeddings will be accurate for implementation on other datasets as well. The purpose of the DIME embeddings is that they are pre-trained on bigger, unlabelled datasets and then applied to smaller datasets such as a smaller clinical trial.

Another limitation is that only one dataset will be used to evaluate the embeddings performance in binary patch classification. Datasets in pathology data are typically very different from one another, which means that the results of this will show the performance in breast cancer and lymph nodes, but the results might be different from binary classification on skin cancer and other types of diseases.

This thesis report also only focuses on one downstream task. The embeddings are meant to provide good results results in multiple different downstream tasks, but this thesis will only cover the performance for binary classification of a certain type of tumour.

1.5 Thesis structure

Chapter 1 lays out the problem definition, background, motivation and delimitation of the thesis. Chapter 2 presents a literary review of fully-supervised methods for WSI classification and previous successful implementations of self-supervised learning. Chapter 3 introduces the DIME pipeline and shows the production and performance of the DIME-embeddings. This chapter also covers the data provided by AstraZeneca for this thesis and different binary classifiers and evaluation metrics useful for the experimental study. Chapter 4 presents the experimental implementation details such as the system architecture and technical approach. Chapter 5 shows the results from the experimental study and Chapter 6 discusses the results. Finally Chapter 7 concludes the findings of the thesis and proposes possible future work.

2 Related work

In the area of WSI classification the self-supervised approach is still relatively new. When the Camelyon16 challenge first was deployed a majority of the solutions together with the winner was fully-supervised methods [8].

The winner of the Camelyon16 challenge was Harvard Medical School and MIT [5] where a fully-supervised method was presented. This method used a patch-based approach where the patches were trained on different deep learning algorithms for patch-level classification. Using the probabilities predicted by the patch-classification networks a tumour probability heatmap was created. The probability heatmap was then used further for slide level evaluation. The networks used for patch-level classification were four existing deep learning architectures, and GoogLeNet was the best performing one giving a patch-level classification accuracy of 98.4%.

But since the availability of annotated datasets is scarce, the desire to take advantage of all data available made the interest in developing self-supervised learning approaches strong. Self-supervised learning has been successfully applied in other areas before. The image inpainting algorithm has been applied to the image and video reconstruction, implementing image resolution improvement and image colorization that can be applied to naturalistic images as well as images taken by microscopes, helping improve the results of cellular analysis [9].

Self supervised learning has also previously been implemented to produce representations of an input data. One of the most popular frameworks is the SimCLR framework that uses contrastive learning of visual representations [10]. SimCLR has been applied to not only visual representation with competitive results but also speech representation learning as presented in Speech-SimCLR [11].

3 Theory

3.1 The DIME pipeline

This section explains the DIME pipeline, which is presented in [1]. The text in the section is derived from conclusions and experiments performed by AstraZeneca and the authors of that report. This section is investigating and explaining the methods and results used in the DIME report so that the necessary knowledge of the DIME pipeline is acquired before the method and results of this thesis project is explained.

3.1.1 The TCGA Datasets

The DIME pipeline is trained on two different datasets. Both datasets are constructed using 32 different projects in the National Cancer Institute Data Portal [12], covering 32 different cancer types. They also cover different primary sites, such as breast and skin. A certain disease type at a primary site is called a class, and in total there are 94 different classes in the datasets. The datasets contains similar amounts of each different project to capture diversity and hopefully produce a more general and high-level feature extractor.

The first dataset is called the TCGA dataset and it was constructed using 100 WSIs from each project. If a project consisted of less than 100 WSIs the entire project was included. This resulted in 2962 WSIs. Those were divided into a training and validation set of 2836 WSIs and a test set consisting of 126 WSIs covering 37 classes.

The second dataset is called the TCGA-Diverse dataset, containing 94 WSIs for training which were selected by taking one WSI of each available class, and 20 WSIs for validation, taken from the 20 most common classes. The test set for the TCGA-Diverse dataset was the same test set as the TCGA dataset.

3.1.2 Patch generation & Data Augmentation

The first step of the DIME pipeline was dividing the WSIs into patches since WSIs are too large to process. WSIs are usually stored with multiple levels of zoom, since a pathologist usually needs to look at a slide at different resolutions and zoom-levels to be able to draw accurate conclusions. Since WSIs were made to digitize slides they also have multiple zoom and resolution levels. The level of zoom varies between WSIs, but the two zoom levels closest to 1 and 4 Microns Per Pixel (MPP) were selected from each WSI and patches of size 224x224 pixels were extracted.

A WSI consists of a lot of patches that will not contain any tissue, and some patches will be extracted from the edge of the tissue sample, making the amount of tissue in a patch less than 100%. To avoid wasting computational resources on patches that does not contain enough relevant information to train on, tissue filtering was applied with a cutoff threshold of 10%. Any patches containing less tissue than that was excluded. The reason why the cutoff threshold was set at 10% and not higher was that the edges of a tissue sample can contain a lot of valuable information and by setting a threshold too high and discard all patches containing the edges could mean a loss of important features.

Colour augmentation was also applied to the input patches. Hue factor, saturation factor, contrast factor and brightness factor was applied and there were three different levels of colour augmentation applied as shown in Table 1.

Augmentation	Hue factor	Saturation factor	Contrast factor	Brightness factor
Very Light	0.02	0.1	0.2	0.4 - 0.6
Light	0.06	0.25	0.2	0.4 - 0.6
Aggressive	0.2	0.8	0.8	0.8

Table 1: Colour Augmentation levels in the DIME pipeline

3.1.3 SimCLR

In self-supervised contrastive learning, features are extracted by locating and rewarding similar input data [3]. There are many different approaches to contrastive learning, but for image classification SimCLR [10] is one of the best performing contrastive learning approaches and it is the contrastive learning method used in the DIME pipeline.

In SimCLR the network learns high-level features by maximizing the agreement for a contrastive loss function between input data that has been augmented in different ways. Probabilistic data augmentation is applied to the input data in the form of random cropping, random colour distortions and a random Gaussian blur. An input image is augmented twice, resulting in two different representations of the input data, called a positive pair.

The positive pair is fed into a neural network called a base encoder which creates embedding-representations of each representation of the input. The neural network applied in the DIME pipeline is the ResNet34 architecture [13]. After the input data is transformed into embedding-representations, they are projected into a non-linear space by a small neural network called the projection head where the contrastive loss is calculated. The DIME pipeline use a Multi-Layer Perceptron with one hidden layer as the projection head network. Further the contrastive loss, which is a normalized temperature-scaled cross entropy loss, is calculated. The base encoder network is updated with stochastic gradient decent to minimize the contrastive loss calculated while the projection head network stays the same.

3.1.4 Inpainting

There are other ways of implementing self-supervised learning other than contrastive methods. Inpainting is a technique normally applied in image and video recovery, editing and colourisation. The inpainting approach had not been applied to the histopathology image embedding area before the DIME pipeline that the authors could find [1].

The inpainting approach is built on the principle of reconstruction of input data that has been tampered with. If a network is sufficiently predictive in filling in missing input data from that same data it should capture high-level features and make good embedding representations of the WSI patches. The input data is filled with random black boxes, and the purpose of the inpainting network is to fill in the missing information.

The DIME Inpainting method builds on the U-Net network [14] which has been adapted in a similar fashion as in the Decrappify solution [9]. The U-Net architecture has a contracting path called the encode arm that captures high features of the input data and an expanding path called the decode arm that captures spatial information of those features that are lost during the contracting path and fills in the missing information.

A ResNet34 architecture is used as the encode arm, the same as in the SimCLR base encoder network. For the decode arm, a reverse ResNet34 network is implemented. Both the ResNet34 networks were pre-trained on ImageNet [15] data. The ImageNet dataset does not contain histopathological images, but pre-training on this dataset can still improve results since some features shared between histopathological and naturialistic images exist, such as different figures like circles and lines.

The DIME Inpainting method also added a VGG-16 model [16] pre-trained on the ImageNet dataset to the output of the U-Net network. The VGG-16 network transforms the output of the U-Net network to an output image, and the objective of this network is to get as close to a target image as possible. The target image is the input data before the black boxes are applied. This method enables two more loss functions other than pixel loss, which is the loss used in the standard U-Net architecture. The two additional losses are perceptual loss and style loss. These losses are applied to make the solution better acclimated to histopathology images as the added losses will capture the high-level features present there.

3.1.5 Model variants

The DIME pipeline trained multiple models with the Inpainting and SimCLR methods, training one inpainting method with the entire TCGA dataset and all others with the TCGA-Diverse dataset, shown with D added to the model name. They also trained different models with different level of colour augmentation, resulting in five different DIME embedding representation models seen in Table 2. Models named with C has light colour augmentation applied and models named with CA has aggressive colour augmentation.

Model	Train set	Train algorithm	Data Augmentation
DIME-Inpainting	TCGA	Inpainting	Very Light
DIME-Inpainting-D	TCGA-Diverse	Inpainting	Very Light
DIME-Inpainting-DCT	TCGA-Diverse	Inpainting	Light
DIME-SimCLR-DC	TCGA-Diverse	SimCLR	Light
DIME-SimCLR-DCA	TCGA-Diverse	SimCLR	Aggressive

Table 2: The different models in the DIME pipeline

For the DIME-Inpainting-DCT model the T stands for target adjusted, which means that the target image the network compared the reconstruction to is target adjusted. That is, the target image is the input image before the colour augmentation is applied. The other Inpainting models has the input image after the colour augmentation as the target image.

3.1.6 MAP@R

The evaluation metric Mean Average Precision at R (MAP@R) is a metric that rewards embeddings spaces that are well-clustered. It is a combination between Mean Average Precision and R-precision. It is calculated by taking the Mean Average Precision with R number of nearest neighbors, as shown in equations 1 and 2 [17] .

$$MAP@R = \frac{1}{R}\sum_{i=1}^{R} P(i) \tag{1}$$

$$P(i) = \begin{cases} \text{precision at i,} & \text{if the ith retrieval is correct} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The correctness of the MAP@R calculation is done by checking if a label for an image is correct. In the DIME pipeline a combination of primary site, disease and MPP level was chosen as the label to compare against. This should cover most of the expected clustering done by the DIME pipeline, but there are some draw-backs to consider. There could be unexpected clustering due to the percentage of tissue in a patch. Since the tissue threshold was 10% there could possibly be two different clusters for the same label that should be rewarded because they differ in a fundamental way, but instead the MAP@R will punish the clustering. For a more correct evaluation MAP@R should be used together with a more thorough investigation of the clustering of the DIME pipeline, but MAP@R gives a good overview of the performance of the different model embeddings.

3.1.7 Evaluation of the TCGA test set

MAP@R scores of the embeddings on the TCGA test set produced by the different DIME models were calculated and compared against each other as well as other embedding types. The other embedding types were SimCLR-D embeddings, produced by a pre-trained contrastive model and an embedding type produced by an ImageNet-ResNet34 model, as shown in Table 3.

Embedding type	MAP@R Score (weighted average)
DIME-Inpainting	0.93
DIME-Inpainting-D	0.89
DIME-Inpainting-DCT	0.94
DIME-SimCLR-DC	0.84
DIME-SimCLR-DCA	0.75
ImageNet-ResNet34	0.66
SimCLR-D	0.67

Table 3: MAP@R score for different embedding types on the TCGA test set

The DIME embeddings outperformed other previous methods noticeably and among the DIME embeddings the DIME-Inpainting-DCT embeddings performed the best, showing that the target adjusted implementation of that embedding type performed better. When the datasets were the only difference, as for DIME-Inpainting and DIME-Inpainting-D the embedding type trained on the entire TCGA dataset performed slightly better. Since DIME-Inpainting-DCT outperforms DIME-Inpainting, a conclusion that the TCGA-Diverse dataset was sufficient when training the embedding types can be made. The clustering performance depends more on other factors such as the colour augmentation and if the embedding-type has target adjusted.

The DIME-SimCLR embeddings were not far behind the DIME-Inpainting embeddings, but a comparison between DIME-SimCLR-DC and DIME-Inpainting-DCT can be done as the only thing differing between these embedding types are the self-supervised learning algorithm. This shows that Inpainting slightly outperforms SimCLR, but that the DIME-SimCLR embedding types are still good embedding representations when compared to other methods like ImageNet-ResNet34.

3.2 Application of DIME to the Camelyon16 Dataset

The Camelyon16 dataset contains WSIs of Hematoxylin and Eosin (H&E) stained lymph nodes. It contains annotations of two classes, one tumour class containing metastases and one normal class containing healthy tissue. The dataset contains a training set consisting of 270 WSIs where 110 of those contain tumorous tissue and 160 contains normal tissue only. There is also a test set containing 130 WSIs [18].

AstraZeneca took the Camelyon16 dataset and used the DIME pipeline to create embedding-representations of WSI patches. First they divided each WSI into patches at four different MPP levels. They choose the MPP levels of each WSI that was closest to 0.5, 1.0, 2.0 and 4.0.

For this thesis project, three different embedding types for each patch, DIME-Inpainting-DCT, DIME-SimCLR-DC and DIME-SimCLR-DCA were provided. Each embedding type can be evaluated and applied to binary classification networks. This resulted in three different datasets of DIME embeddings shown in Table 4.

The dataset is not particularly unbalanced when inspecting the slide level WSIs in the Camelyon16 dataset, however the dataset becomes very unbalanced when extracting patches. In each WSI annotated as tumour, there is both patches of normal tissue as well as tumour tissue. A slide often only contains tiny areas of tumour tissue. This means that each tumour WSI will produce normal patches and tumour patches while non-tumour WSIs only produce normal patches.

Embedding type	Train set	Test set
Inpainting	1353990	701967
SimCLR Light	1353990	701967
SimCLR Aggressive	1353990	701967

Table 4: The different Camelyon16 DIME embedding datasets and their sizes

The annotation of the patches is decided by referring to the ground-truth provided by a pathologist. If a patch contains one pixel of tumorous tissue it will be labeled as a tumour patch. The unbalanced dataset is evident when inspecting the training sets where there are 51 551 embeddings of the tumour class and 1 302 439 of the normal class. In the test set, there are 60 214 embeddings of the tumour class and 641 753 embeddings of the normal class.

3.3 Binary Classification Classifiers

3.3.1 Logistic Regression

Logistic Regression (LR) is one of the most common binary classifiers. It is a fast learning model that predicts the probability that an input data belongs to one of the two classes X and Y.

LR uses a linear function to combine input data with weights to predict an output. LR then uses the sigmoid function to map the results to a value between 0 and 1. The training of an LR binary classifier consists of optimizing the weights of the LR function added to the input data by doing a maximum-likelihood estimation that minimizes the error in probabilities predicted by the LR model.

The best weights will push the prediction done by the network very close to either 0 or 1. For classification tasks a yes or no answer is wanted, which is received by setting a threshold-value at 0.5, meaning that any probabilities over 0.5 will mean that the predicted class is the X class, and any predictions under as the other class Y [19].

3.3.2 Multi-layer Perceptron

A Multi-Layer Perceptron (MLP) is a very basic feed-forward artificial neural network. It consists of a series of connected layers of neurons, which are units that receives input signals that are weighted, runs them through an activation function and produces an output signal.

The neurons in hidden layers, which are layers that are not the input layer or the output layer, are fully connected to the previous layer and the next layer. The signals are propagated through the neural network by having each layer perform computations on the input signal that is received. The output signals of one layer will be the input signal to the next. The data is therefore propagated through the neural network. The input layer takes the input data and send it to the hidden layers. Each neural network has at least one hidden layer that is fully connected to the input layer and the output layer, but can also contain multiple hidden layers connected to each other. The output layer for a binary classification problem will output a probability of the input data being of class 1, which with the use of a threshold value such as 0.5 can be turned into predictions.

The activation function of the neurons can be multiple different types, but ReLU or a sigmoid function are typical choices. The weights of the input signals are trained during the training process to minimize the error using stochastic gradient decent [20].

3.3.3 Gradient Boosting

Gradient Boosting (GB) builds on the concept of boosting, which combines the predictions made by multiple weak learners to make a strong model prediction.

A weak learner is a model whose performance is at least slightly better than random chance, meaning that they are very simple classifiers. First all observations of the training data have the same weight and have the same probability of being selected. The boosting algorithm takes training data observations and feeds them into a weak learner, W_1. All the miss-classified observations by that weak learner weights are increased while the correctly classified observations weights are decreased. Then, another weak learner, W_2, will select observations, but now the probability of selecting previously miss-classified data increases. This process continues until all data is correctly predicted by at least one of the models. This means that there will be multiple weak learners, and a decision is made by selecting the class that gets the highest vote from all the weak learners through a majority vote.

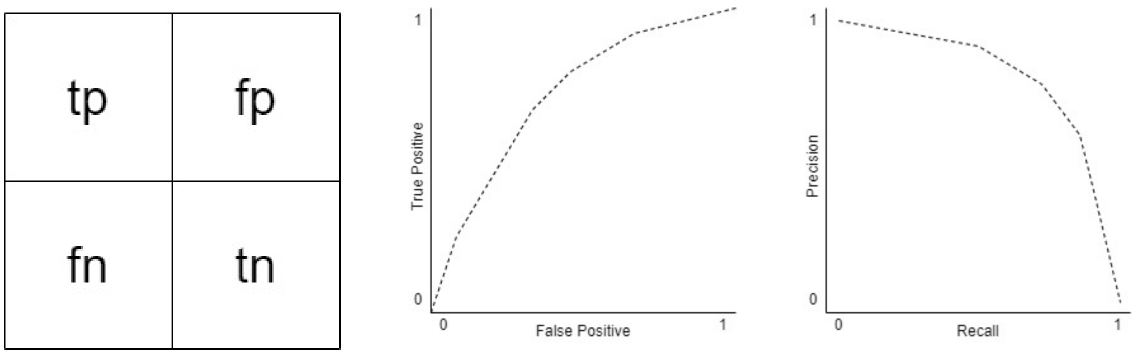
GB implements boosting using decision trees as weak learners and applies new weak learners using a gradient decent procedure. Each new weak learner is added to minimize the gradient by reducing the loss function chosen. In application of GB binary classification a logarithmic loss is a common choice. Once a new weak learner is implemented the old weak learners are not changed further.

The GB algorithm can either stop by having a pre-determined max number of trees or if the loss is minimized enough or no longer improves on the validation set [21, 22].

3.4 Evaluation Techniques

Evaluation metrics are used to evaluate machine learning models performance on unseen data. By using a test set unseen by the model during the training, the results can be evaluated using different evaluation metrics to measure the models ability.

Many of the evaluation metrics used for binary classification are calculated from four different numbers. Those are the True Positive (tp) which is the number of correct predictions that an input data is from the positive class. Then there is the True Negative (tn) which is the number of correct predictions that an input data is from the negative class. Then there are the False Positive (fp) and the False Negative (fn) that are incorrect predictions of each true class [23].



(a) Confusion Matrix (b) ROC Curve (c) Precision Recall Curve

Figure 2: Evaluation Metric Templates

3.4.1 Confusion Matrix

The confusion matrix takes the true positive, true negative, false positive and the false negative predictions and displays those in a 2x2 matrix as in Figure 2a to give a good overview of the models ability to correctly predict for both classes [23].

3.4.2 Accuracy

Accuracy is the most common evaluation metric and it measures the ratio of correct predictions against incorrect predictions shown in equation 3. Accuracy does not take the different classes into account [23].

$$Accuracy = \frac{tp + tn}{tp + tn + tn + fn} \tag{3}$$

3.4.3 AUC & ROC Curve

The Receiver Operating Characteristic (ROC) curve plots the true positive rate against the false positive rate at different classification thresholds, like in Figure 2b. The lower the threshold, the more positive classification results will the model make. The steepness of the curve is important as the true positive rate should be maximised while the false positive rate is minimized [24].

The Area Under Curve (AUC) is the area under the ROC curve. Equation 4 shows AUC defined for a binary classification task, where S_p is the sum of all positive examples ranked while n_p and n_n are the positive and negative examples.

$$AUC = \frac{S_p - n_p(n_n + 1)/2}{n_p n_n} \tag{4}$$

AUC is better than accuracy at evaluating performance with both classes performance included, and gives an overall ranking performance of a classifier.

3.4.4 AP & Precision Recall Curve

Precision measures the correct positive predictions against all the positive predictions, as shown in equation 5. Recall measures the number of positive correct predictions out of all correct classes, as shown in equation 6 [23].

$$Precision = \frac{tp}{tp + fp} \tag{5}$$

$$Recall = \frac{tp}{tp + tn} \tag{6}$$

The Precision Recall Curve (PRC) plots the trade-off between the precision and recall for different classification thresholds, as seen in Figure 2c. The PRC is a very good metric when the classes are unbalanced. Average Precision (AP) is the average precision which summarizes the PRC as it calculates the mean of the precision at each threshold with the increased recall acting as a weight shown in equation 7 [25].

$$AP = \sum_n (R_n - R_{n-1}) P_n \tag{7}$$

3.4.5 Statistical Significance Test

Statistical Significance tests are done to compare machine learning models and decide whether they differ in a statistically significant way. The tests evaluate the results of the models and calculate the probability that the results from the two different models are from the same distribution.

A null hypotheses is presented that the two models are from the same distribution. A significance threshold is also decided, such as $\alpha = 0.001$ where the null hypotheses will be rejected if the probability (P-value) calculated by the statistical significance test is smaller than the significance threshold. If the null hypotheses is rejected the models can be assumed to be statistically different [26].

There are many different statistical tests but the one implemented in this thesis is McNemar's test. McNemar's test compares two binary classifiers by comparing how accurate they are on the same test set. It checks if the models disagree on the same input data or differ on their incorrect predictions. McNemar's test is calculated with a 2x2 contingency table of the two model's predictions. In the contingency table the number of accurate predictions by both models is in the top left corner and the inaccurate predictions by both models is in the bottom right. In the top right corner the number of correct predictions by model 2 while model 1 was incorrect is shown and in the bottom left it is the number of correct predictions by model 1 while model 2 was incorrect, as shown in Figure 3.

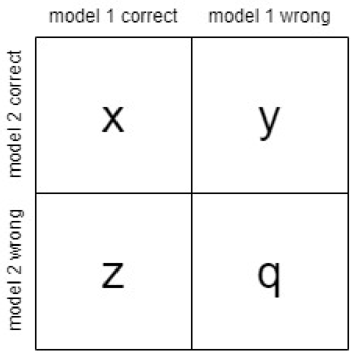

model 1 correct
model 1 wrong
model 2 correct
model 2 wrong
x
y
z
q

Figure 3: Contingency Table Template

The test statistic calculated by McNemar's test is called the chi-squared value and is computed as in equation 8 where the y and z values are the same ones as in the contingency table. The P-value is the probability of observing the chi-squared value in the chi-squared distribution [27].

$$chi^2 = \frac{(y-z)^2}{y+z} \tag{8}$$

3.4.6 Cross-validation

Cross-validation is another way to statistically evaluate the skill of machine learning models. It is often used before application of the finished models as a way of deciding upon the best model to implement regarding hyperparameters and find a model that works well on unseen data.

Cross-validation works by implementing k-fold cross-validation which is a way of dividing up the dataset in k equal subsets. The model is then trained by repeating the training k times, taking one of the k subsets as the test set each time and the other subsets as the training set. After the experiments, the results of all the k runs of the model are summarized and evaluated against other models. The value k is usually 5 or 10 since they have been proven to be good values for k, but can theoretically be any value [28, 29].

4 Implementation

4.1 System architecture

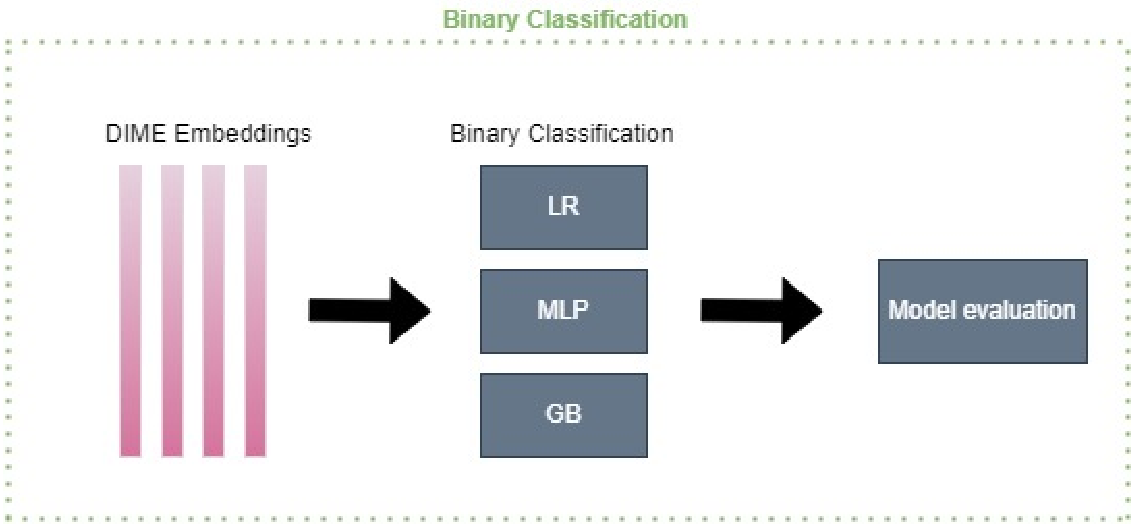


Figure 4: System Architecture of the thesis project

The system architecture of the thesis project is shown in Figure 4. The embedding types produced by the DIME pipeline is fed into different binary classification networks to produce different models that are then evaluated. The implementation code is uploaded on GitHub at `https://github.com/elsajonsson/Master-Thesis`.

4.2 Binary Classification

All binary classifiers were implemented using the scikit learn tools for each model [30, 31, 32]. The models were saved using pickle after each run to be evaluated later.

4.2.1 Hyper-parameter Optimization

A hyper-parameter optimization search was done for the LR networks using the GridSearchCV tool from scikit-learn [29]. The GridSearchCV tool takes in a grid of different hyperparameters and uses cross-validation to apply and evaluate the different hyperparameters on the training set for a model. The hyperparameters that were investigated in the LR networks were the solver, c-value and the penalty and because of the large size of the training set and limited resources of the cloud computing the hyper-parameter optimization was done on 10% of the training set that was randomly picked. The cross-validation applied was a 10-fold cross-validation that was repeated 3 times and the evaluation metric chosen to evaluate and compare models with was the ROC curve and AUC value, since the dataset is very unbalanced and AUC is a better metric than accuracy for unbalanced datasets.

For the MLP networks, no hyper-parameter optimization was implemented. The main reason for not performing a hyper-parameter search was that from initial runs of the MLP networks the change of some hyper-parameters did not show any difference in performance. Since the MLP networks also were noticeably slower regarding training time, spending resources performing a hyper-parameter optimization that showed no promise of improving results was decided against.

One way to speed up the run time could be to use a smaller training set size, but since the LR network already used only 10% of the training set, using any less could give unreliable results. Instead the default hyperparameters were used for the MLP networks.

4.2.2 Reduced training size investigation

For each LR and MLP model trained, a reduced training size investigation was implemented. Each network was trained on 10%, 30%, 50%, 70% and 100% of the training set with the same hyper-parameters and test set. This was done to be able to see how the training size affects the model performance. The metrics that were compared for each training size was the AP and AUC since those are the primary evaluation metrics that will be evaluated in this report because of the unbalanced dataset.

The reduced training sets were produced using the scikit-learn tool train_test_split [33], which when fed a seed gives a reproducible output. Each network was provided the same seed, 42, which made sure that the reduced datasets investigated for each model were the same. The percentage fed into this tool gives back a test set that is that percentage of the input data, which was used as the training set.

4.2.3 Model Evaluation

Each model trained on 100% of the training set were evaluated more thoroughly with more evaluation metrics to enable comparison for patch-based classification with other methods implemented with fully-supervised learning. The models were evaluated with AUC, AP, Accuracy, Confusion Matrices, ROC curves and Precision Recall curves using the scikit-learn tools for each metric [34].

McNemar's test was done between all embedding types comparing the LR and MLP classifiers to see if there was a statistically significant difference between the classifier types for the same embedding type. McNemar's test was also done between each embedding type on the MLP classifier to see if there was a statistically significant difference between the embedding types for a binary classifier. MLP was chosen since it was the best performing binary classifier. The null-hypothesis of the test was that the two models disagree the same amount on the same data. The significance level was set as $\alpha = 0.001$.

5 Results

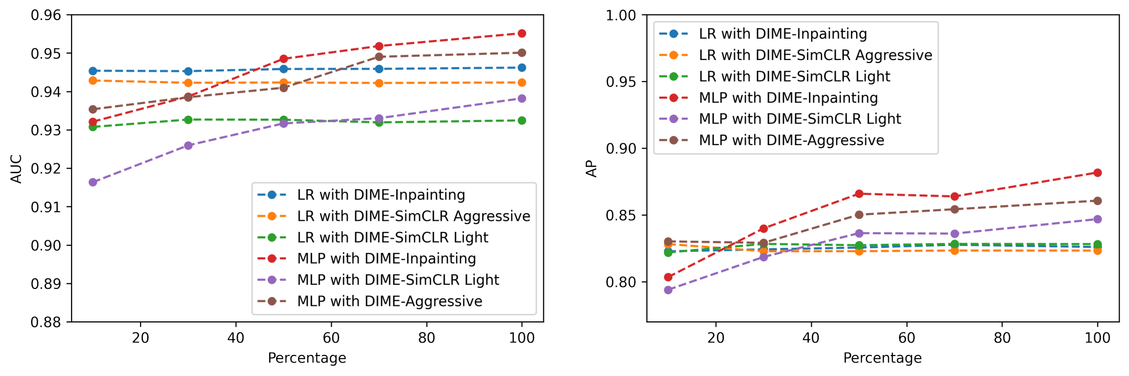
5.1 Hyper-parameter Optimization

Model	Solver	Penalty	C
LR with DIME-Inpainting	saga	l2	100
LR with DIME-SimCLR Light	saga	l2	100
LR with DIME-SimCLR Aggressive	sag	l2	100
Default LR	lbfgs	l2	1.0

Table 5: The optimized hyper-parameters of the LR networks and the default hyper-parameters for a LR network

The result from the hyper-parameter optimization on the LR networks is shown in Table 5. The C-value in LR is the inverse of regularization strength. A smaller C value mean stronger regularization which prevents overfitting. The penalty is the regularization model used in the LR and l2 means that Ridge Regression is implemented. The Solver is the optimization algorithm used and sag and saga are both well suited for large datasets [30].

5.2 Performance of models on reduced training size



(a) Change in AUC (b) Change in AP

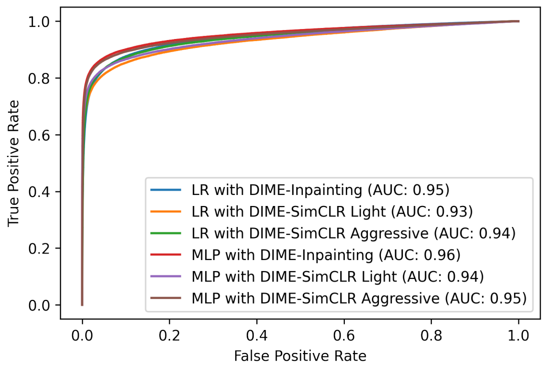
Figure 5: Change in AP & AUC of the DIME-based models for different sized training sets

The different training size performance for each model are shown in Figure 5. The performance of all models stays fairly constant for each different training size, between 0.915 and 0.955 in AUC and 0.8 and 0.88 in AP.

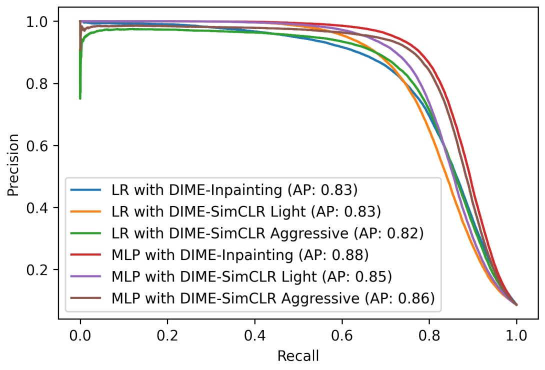
The LR models barely change depending on the training size, staying at a relatively constant AUC and AP value for the models and beating all other network types for lower training sizes. MLP models improve as the training size increases, starting off worse than the LR models and then improving when the training size is increased.

5.3 Model Comparison

5.3.1 Model Evaluation



(a) ROC curve and AUC



(b) Precision Recall curve and AP

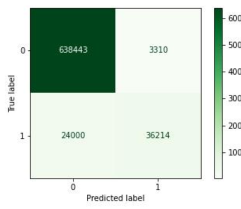
Figure 6: Binary classification performance of each DIME-based model

All models that were trained on 100% of the training set were evaluated thoroughly with additional metrics than for the reduced training size evaluation. The reported accuracy is seen in Table 6, AUC & ROC curve are shown in Figure 6a, Precision Recall curve and AP are shown in Figure 6b. Each models confusion matrix is also shown in Figure 7.

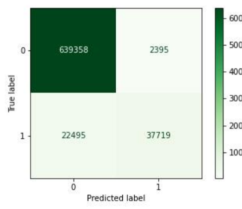
Model	Accuracy
LR with DIME-Inpainting	96.1%
LR with DIME-SimCLR Light	96.5%
LR with DIME-SimCLR Aggressive	96.4%
MLP with DIME-Inpainting	97.3%
MLP with DIME-SimCLR Light	96.9%
MLP with DIME-SimCLR Aggressive	97.1%

Table 6: The accuracy of the DIME-based models on the Camelyon16 test set.

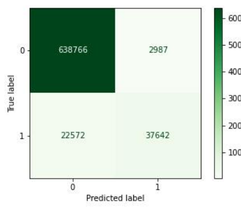
When evaluating the models using all the different metrics the best performing model is MLP with DIME-Inpainting. It has the best performance of all of the models. The other MLP models had some varying performance regarding the AUC, but looking at the confusion matrix for each model the MLP models were better at classifying the tumour class correctly, which is also shown with all the MLP models having the best AP scores and accuracies. They all also have good AUC scores even if the LR with DIME-Inpainting beats the MLP with DIME-SimCLR Light slightly.



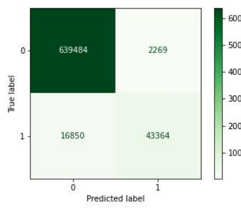
(a) LR with DIME-Inpainting Confusion Matrix



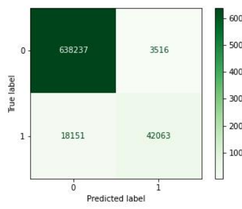
(b) LR with DIME-SimCLR Light Confusion Matrix



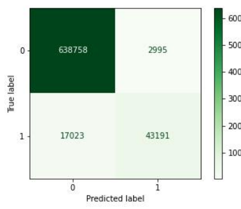
(c) LR with DIME-SimCLR Aggressive Confusion Matrix



(d) MLP with DIME-Inpainting Confusion Matrix



(e) MLP with DIME-SimCLR Light Confusion Matrix



(f) MLP with DIME-SimCLR Aggressive Confusion Matrix

Figure 7: Confusion Matrices for all DIME-based models

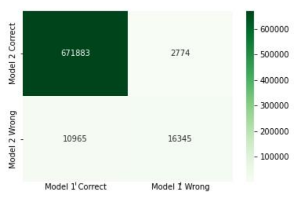
The models trained on DIME-Inpainting show best AUC and AP values for their respective binary classifier type. This is followed by the DIME-SimCLR Aggressive based models and the DIME-SimCLR Light models perform the worst out of all the different embedding type models. For the LR with DIME-SimCLR Light model the confusion matrix and accuracy performs better than the other LR models but the AUC and AP values, showing how well the model separates the two classes, is not as good as the other embedding types. For the MLP-based models the accuracies and confusion matrices follow the results seen regarding AUC and AP for the models.

Model	Accuracy
GoogLeNet [5]	98.4%
AlexNet [5]	92.1%
VGG16 [5]	97.9%
FaceNet [5]	96.8%

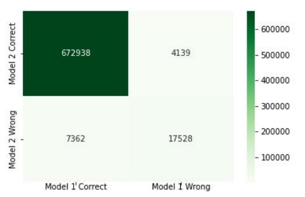
Table 7: The accuracy of the fully-supervised learning models implemented by the Camelyon16 challange winners presented in [5] on the Camelyon16 test set.

The results reported by the winners of the Camelyon16 challange in [5] are fully-supervised methods and are shown in Table 7. There are four different implemented models that differ depending on the neural network architecture that was used for the patch-level classification. All the models implemented are deep learning architectures, containing many layers and trainable parameters. The two best performing architectures, GoogLeNet and VGG16 were the deepest architectures presented. The best performing model was the GoogLeNet architecture that got a patch-level accuracy of 98.4%. This model outperforms all the DIME-based models shown in Table 6 up to 2.3%. The best model presented in this thesis, the MLP with DIME-Inpainting accuracy is only 1.1% lower than the GoogLeNet model and 0.6% lower than the VGG16 architecture. The other fully-supervised networks perform worse than the MLP with DIME-Inpainting model, where the AlexNet model worse than all the models presented in this thesis. The FaceNet model performs worse than all the MLP models but better than the LR models.

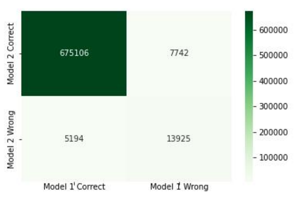
5.3.2 Statistical Significance Test



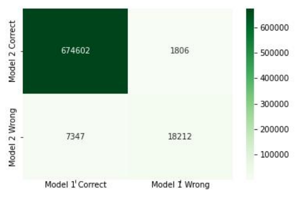
(a) Model 1: MLP with DIME-Inpainting and Model 2: LR with DIME-Inpainting



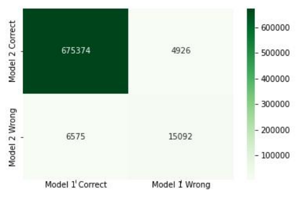
(b) Model 1: MLP with DIME-SimCLR Light and Model 2: LR with DIME-SimCLR Light



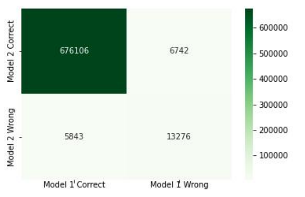
(c) Model 1: MLP with DIME-SimCLR Light and Model 2: MLP with DIME-Inpainting



(d) Model 1: MLP with DIME-SimCLR Aggressive and Model 2: LR with DIME-SimCLR Aggressive



(e) Model 1: MLP with DIME-SimCLR Aggressive and Model 2: MLP with DIME-SimCLR Light



(f) Model 1: MLP with DIME-SimCLR Aggressive and Model 2: MLP with DIME-SimCLR Inpainting

Figure 8: Contingency Tables for all McNemar's tests

Model 1	Model 2	Chi-squared	P-value
MLP with DIME-Inpainting	LR with DIME-Inpainting	4883	0.0
MLP with DIME-SimCLR Light	LR with DIME-SimCLR Light	903	$1.98*10^{-198}$
MLP with DIME-SimCLR Aggressive	LR with DIME-SimCLR Aggressive	3354	0.0
MLP with DIME-SimCLR Light	**MLP with DIME-Inpainting**	502	$3.7*10^{-111}$
MLP with DIME-SimCLR Aggressive	MLP with DIME-SimCLR Light	236	$2.4*10^{-53}$
MLP with DIME-SimCLR Aggressive	**MLP with DIME-Inpainting**	64	$1.1*10^{-15}$

Table 8: McNemar's test results for comparison between models

McNemar's test was applied to compare the difference in the models using the same binary classifier and also the same embedding type. The contingency tables for each comparison are shown in Figure 8 and the calculated values using McNemar's test are shown in Table 8, where the best performing model is highlighted.

All P-values are very small, well below the significance level which means that for each comparison, the null hypothesis can be rejected and it is possible to assume that the models differ in a fundamental way in their proportion of errors, and does not agree on the same data.

As seen in the contingency tables, one model usually has many more correct predictions than the other. Since the models are proven to be statistically different the better performing model is not better by a statistical fluke.

6 Discussion

When investigating the results from the training size investigation in Section 5.2, the most interesting result is that all models are very unaffected by the reduced training size. They perform very well for 10% of the training set. This could be a result from the DIME-embeddings ability to already capture very high-level features and differentiate between tumour and non-tumor well before the binary classification network is applied. This conclusion was made in the DIME report [1] when doing a similar evaluation on another dataset for a clinical study called MYSTIC. The results supports that that is the case for the Camelyon16 dataset as well. This shows how well clustered the embeddings are from the DIME pipeline and speaks to the ability to apply them to many different downstream tasks and would be suitable for smaller datasets such as new clinical studies.

One other possible reason for the similar results for each model in the training size investigation is the randomness included in extracting the reduced training sets. There is a possibility that many tumour patches are selected which could increase the results of the models at each percentage, but since the results are good for every percentage that would mean that the random selected set for each percentage would be selecting more tumour patches than non-tumour patches which would be very unlikely. The randomness argument could also be made to further prove the robustness of the DIME-embeddings since there is a possibility that the randomness selects very little tumour patches, resulting in models that should perform badly on the test set which they do not, further enforcing the idea that the DIME-embeddings have captured very good features.

To be sure that a comparison between the different models could be made and that the models performance difference is not because of a statistical fluke McNemar's test was implemented. McNemar's test was implemented since it was one of the statistical significance tests that did not require further training on each finished model, but rather evaluated the models predictions on the test set against each other. The models, especially the models using MLP as the binary classifier took a long time to train. Using another method for the statistical significance test, such as a cross-validation based test would take too long. The results from the McNemar's tests implemented shown in Section 5.3.2 shows that each model differs from one another. This way we can be certain that while comparing the models performance the difference depends on the actual models and conclusions about each model can be made.

The results from Section 5.3 shows that the MLP networks outperformed the LR networks. A third binary classifier was implemented but never added to the results, Gradient Boosting, to compare to the previous classifiers performance and add more diversity to the thesis. This model took a long time to train, and when training on 10% of the training set this model showed worse performance than LR and MLP, shown in Appendix A. To make sure that the model was not affected more by the reduced training size the GB network with the DIME-Inpainting embedding types, the embedding types that performed best for both other models, had a full reduced training size investigation. The results were still worse so for the other embedding types GB was not implemented other than for 10% of the dataset to secure that the other embedding types would not perform better than the DIME-Inpainting embeddings, which they did not.

When inspecting the confusion matrices and other evaluation metrics it is clear that while both the LR and MLP models are very good at classifying the non-tumour class, the big difference between the models is the performance on the tumour-class. Since the Camelyon16 dataset is very unbalanced towards the non-tumour class it is expected that the models performance would depend on the tumour class. One reason for the MLP models increased performance could be that MLP networks are better for very unbalanced datasets than LR networks as discussed in [35].

It is also clear that the models trained on the DIME-Inpainting embeddings perform the best. In the DIME report the evaluation on the embeddings with MAP@R on the TCGA test set also shows that the DIME-Inpainting embeddings are the best clustered embeddings, which makes a strong argument that they would have the best high-level features captured that would give the best performance on the Camelyon16 dataset as well. One reason that the Inpainting embeddings perform the best could be that the Camelyon16 dataset is very similar in images regarding colour variation, disease and tissue type. The Inpainting method, which learns by filling in missing information, could work better than SimCLR since SimCLR works by augmenting patches and learn not only similarity in the patches but also diversity. The DIME pipeline trains on a very diverse dataset, and the SimCLR embedding types results are very competitive to the Inpainting embedding types, meaning that for other downstream tasks with more diverse data the SimCLR patches could perform better.

The research question of the thesis was to compare the fully-supervised methods against the partially self-supervised models produced by the DIME-embeddings acting as feature extractors together with simple binary classifiers. The only metric available for patch-level classification for the fully-supervised methods was the accuracy. That makes it hard to discuss the fully-supervised models performance regarding the tumour-class which would be more informative because of the unbalanced dataset. Comparing accuracy between models can give deceiving results, since the accuracy can be solely depending on the non-tumour class. But, inspecting the models developed in this thesis, the models with the best performance according to better metrics such as AP, AUC and confusion matrices also have the best reported accuracies. Those models already are very good at classifying the non-tumour class, so any models with higher accuracy should according to those results be better because of the performance on the tumour class. Therefore, only comparing the accuracy will still be enough to make assumptions and comparisons between DIME-based models and fully-supervised models performance.

Comparing the best performing self-supervised based model, the MLP with DIME-Inpainting against the fully supervised models that were the Camelyon16 winners reported in [5], the MLP with DIME-Inpainting model beats both the AlexNet model and the FaceNet model. The best performing fully-supervised model, GoogLeNet, only beats the MLP with DIME-Inpainting with 1.1%. All the fully-supervised models are very deep neural networks, where the GoogLeNet model has 27 layers and more than 6 million parameters, which takes a long time to train. Implementing the simple binary classifiers instead saves a lot of training time. Even if the results are slightly lower, they are still competitive enough to be seen as a very interesting option as by using those models with the pre-trained DIME pipeline would save time and therefore resources when training. Also worth noting is that the fully-supervised networks have seen Camelyon16 during the entire training process. The TCGA dataset that the DIME-pipeline is pre-trained on does not contain any Camelyon16 slides, meaning that the features captured by the DIME pipeline together with a simple classifier are good enough to give comparable results to deep neural networks when applied to the unseen patches in the Camelyon16 dataset.

7 Conclusion & Future Work

7.1 Conclusion

This thesis shows that the DIME embeddings have captured high-level features, and are applicable to binary classification with simple classifiers. The features captured by the DIME pipeline are relevant to this downstream task and aids in situations where the datasets are smaller. This means that the DIME pipeline would work well for smaller clinical studies as a feature extractor to help the model capture high-level features without having a big dataset to train on.

The results of the thesis also shows that even if the MLP with DIME-Inpainting model was the best model and the best contender against the fully-supervised models, all models presented in the thesis give good performance and are competitive against the fully-supervised models. The model in this research with lowest performance is still only 2.3% away from the top-performing fully-supervised method GoogLeNet.

The main conclusion of this thesis is that the embedding features generated by the pre-trained self-supervised learning DIME-pipeline are sufficiently predictive with simple classifiers when compared to end-to-end fully supervised deep learning methods.

7.2 Future Work

In this thesis work the goal was to prove that the DIME pipeline is a good feature extractor for the downstream task of binary classification, which was done on the Camelyon16 dataset. Future work could be to expand upon this thesis project by implementing more binary classifiers and doing more extensive hyper-parameter optimization for each network type. By adding a solution to go from patch-level classification to slide-level classification more extensive comparison could be done against other fully-supervised methods as well as other self-supervised methods.

Future work could also be the application of the DIME pipeline to other downstream tasks, such as quality control, annotation tasks or multi-class classification.

References

[1] Khan Baykaner et al. "Image model embeddings for digital pathology and drug development via self-supervised learning". In: *bioRxiv* (2021). DOI: 10.1101/2021.09.20.461088. eprint: https://www.biorxiv.org/content/early/2021/09/23/2021.09.20.461088.full.pdf. URL: https://www.biorxiv.org/content/early/2021/09/23/2021.09.20.461088.

[2] Pantanowitz L Farahani N Parwani A. "Whole slide imaging in pathology: advantages, limitations, and emerging perspectives". In: *Pathology and Laboratory Medicine International.* 2015, pp. 23–33. URL: https://doi.org/10.2147/PLMI.S59826.

[3] Yann LeCun Ishan Misra. *Self-supervised learning: The dark matter of intelligence.* 2021. URL: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/.

[4] Alamira Jouman Hajjar. *In-Depth Guide to Self-Supervised Learning: Benefits Uses.* 2022. URL: https://research.aimultiple.com/self-supervised-learning/.

[5] Dayong Wang et al. *Deep Learning for Identifying Metastatic Breast Cancer.* 2016. DOI: 10.48550/ARXIV.1606.05718. URL: https://arxiv.org/abs/1606.05718.

[6] D.L. Weaver et al. "Comparison of pathologist-detected and automated computer-assisted image analysis detected sentinel lymph node micrometastases in breast cancer". In: *Modern pathology* (2003). DOI: 10.1097/01.MP.0000092952.21794.AD. URL: https://www.nature.com/articles/3880899.

[7] MultiMedia LLC. *Camelyon16 Challange motivation.* 2016. URL: https://camelyon16.grand-challenge.org/Background/.

[8] Babak Ehteshami Bejnordi et al. "Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer". In: *JAMA* 318.22 (Dec. 2017), pp. 2199–2210. ISSN: 0098-7484. DOI: 10.1001/jama.2017.14585. eprint: https://jamanetwork.com/journals/jama/articlepdf/2665774/jama_ehteshami_bejnordi_2017_oi_170113.pdf. URL: https://doi.org/10.1001/jama.2017.14585.

[9] Jason Antic, Jeremy Howard, and Uri Manor. *Decrappification, DeOldification, and Super Resolution.* URL: https://www.fast.ai/2019/05/03/decrappify/.

[10] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *Proceedings of the 37th International Conference on Machine Learning.* Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 1597–1607. URL: https://proceedings.mlr.press/v119/chen20j.html.

[11] Dongwei Jiang et al. *Speech SIMCLR: Combining Contrastive and Reconstruction Objective for Self-supervised Speech Representation Learning.* 2020. DOI: 10.48550/ARXIV.2010.13991. URL: https://arxiv.org/abs/2010.13991.

[12] National Cancer Institute. *TCGA Dataset portal.* URL: https://portal.gdc.cancer.gov/.

[13] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778. DOI: 10.48550/arXiv.1512.03385.

[14] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation.* 2015. DOI: 10.48550/ARXIV.1505.04597. URL: https://arxiv.org/abs/1505.04597.

[15] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition.* 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[16] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition.* 2014. DOI: 10.48550/ARXIV.1409.1556. URL: https://arxiv.org/abs/1409.1556.

[17] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. *A Metric Learning Reality Check*. 2020. DOI: 10.48550/ARXIV.2003.08505. URL: https://arxiv.org/abs/2003.08505.

[18] MultiMedia LLC. *Camelyon16 Challange dataset*. 2016. URL: https://camelyon16.grand-challenge.org/Data/.

[19] Jason Brownlee. *Logistic Regression for Machine Learning*. URL: https://machinelearningmastery.com/logistic-regression-for-machine-learning/.

[20] Jason Brownlee. *Crash Course On Multi-Layer Perceptron Neural Networks*. URL: https://machinelearningmastery.com/neural-networks-crash-course/.

[21] Jason Brownlee. *A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learnings*. URL: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/.

[22] Anshul Saini. *Gradient Boosting Algorithm: A Complete Guide for Beginners*. URL: https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/.

[23] Mohammad Hossin and Sulaiman M.N. "A Review on Evaluation Metrics for Data Classification Evaluations". In: *International Journal of Data Mining Knowledge Management Process* 5 (Mar. 2015), pp. 01–11. DOI: 10.5121/ijdkp.2015.5201.

[24] scikit-learn. *Receiver Operating Characteristic (ROC)*. URL: https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html.

[25] scikit-learn. *Precision-Recall*. URL: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html.

[26] Jason Brownlee. *Statistical Significance Tests for Comparing Machine Learning Algorithms*. URL: https://machinelearningmastery.com/statistical-significance-tests-for-comparing-machine-learning-algorithms/.

[27] Jason Brownlee. *How to Calculate McNemar's Test to Compare Two Machine Learning Classifiers*. URL: https://machinelearningmastery.com/mcnemars-test-for-machine-learning/.

[28] Jason Brownlee. *A Gentle Introduction to k-fold Cross-Validation*. URL: https://machinelearningmastery.com/k-fold-cross-validation/.

[29] scikit-learn. *Tuning the hyper-parameters of an estimator*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[30] scikit-learn. *LogisticRegression*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

[31] scikit-learn. *MLPClassifier*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html.

[32] scikit-learn. *GradientBoostingClassifier*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html.

[33] scikit-learn. $train_test_split$. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html.

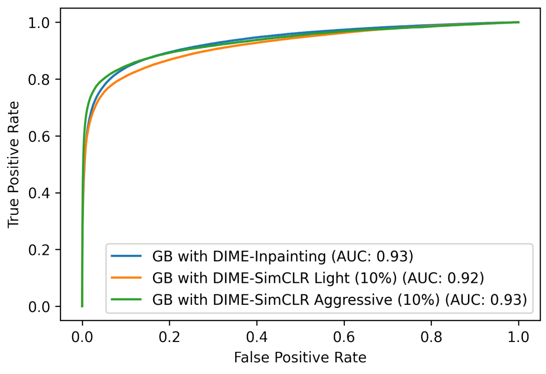
[34] scikit-learn. *Model evaluation: quantifying the quality of predictions*. URL: https://scikit-learn.org/0.15/modules/model_evaluation.html.

[35] Ray Marie Tischio and Gary M. Weiss. "Identifying Classification Algorithms Most Suitable for Imbalanced Data". In: 2019. URL: https://www.semanticscholar.org/paper/Identifying-Classification-Algorithms-Most-Suitable-Tischio-Weiss/3698c77f221bac2e771af3eb661ad04675a0f6b9.

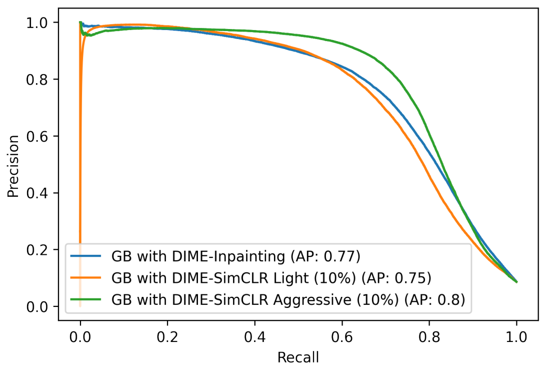
A Gradient Boosting Classifier Results

Model	Accuracy
GB with DIME-Inpainting	95.0%
GB with DIME-SimCLR Light (10%)	95.2%
GB with DIME-SimCLR Aggressive (10%)	95.5%

Table 9: The accuracy of each DIME-based GB models on the Camelyon16 test set

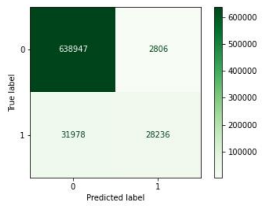


(a) ROC curve and AUC

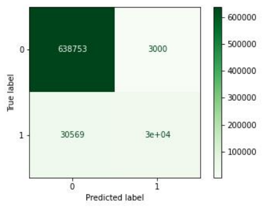


(b) Precision Recall curve and AP

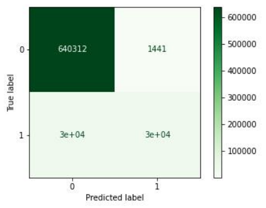
Figure 9: Binary classification performance of each DIME-based GB model



(a) GB with DIME-Inpainting Confusion Matrix (100% of train set)



(b) GB with DIME-SimCLR Light Confusion Matrix (10% of train set)



(c) GB with DIME-SimCLR Aggressive Confusion Matrix (10% of train set)

Figure 10: Confusion Matrices for all DIME-based GB models