KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Information and Network Engineering

Second cycle, 30 credits

Identifying Complications of Impacted Canines in Panoramic Radiographs Using Deep Learning Techniques

An investigation of methods for a small-sized dataset

LAURA BRIFFA

Stockholm, Sweden, 2024

Identifying Complications of Impacted Canines in Panoramic Radiographs Using Deep Learning Techniques

An investigation of methods for a small-sized dataset

LAURA BRIFFA

Degree Programme in Electrical Engineering
Date: December 12, 2024

Supervisors: Antoine Honoré, Elena Borsci
Examiner: Saikat Chatterjee
School of Electrical Engineering and Computer Science
Host organization: Karolinska Institutet
Swedish title: Djupinlärning med ortopantomogram för identifiering av komplikationer från retinerade hörntänder
Swedish subtitle: En undersökning av lämpliga metoder för en liten datamängd

Abstract

Maxillary canine impaction occurs when the canine tooth in the upper jaw fails to emerge as anticipated. Addressing impacted canines presents significant challenges, including potential complications arising from delayed treatment. While 2-dimensional panoramic radiographs are commonly employed for initial diagnostics of canine impaction, this modality has inherent limitations. For more complex cases, cone beam computed tomography (CBCT) provides detailed 3-dimensional imaging. Compared to the panoramic radiograph, CBCT offers a higher diagnostic accuracy for detection of root resorption and localization of the canine. However, the use of CBCT is associated with significantly higher radiation exposure to the patient. This thesis aims to explore the application of machine learning techniques to predict the need for CBCT evaluation for patients with impacted canines. Utilizing a dataset of 109 panoramic radiographs, baseline metrics are established based on measured features derived from the radiographs. These baseline results are compared to deep learning image classification techniques. The study examines a feature learning approach employing convolutional autoencoders and investigates a transfer learning strategy using a pre-trained Vision Transformer classifier. The comparative study showed that the best baseline classifier achieved an F1 score of 0.40 and an AUROC of 0.56. In the deep learning component of the study, two autoencoder models and two instances of fine-tuning a pre-trained Vision Transformer were investigated. Among these, one autoencoder implementation outperformed the baseline, achieving an F1 score of 0.60 and an AUROC of 0.81. This particular model employed an encoder that reduced the image input space by half and was paired with a linear Support Vector Machine for classification. The findings suggest potential in these methods, yet underscore the need for a larger dataset to more thoroughly assess their suitability for the task.

Keywords

Impacted maxillary canines, Panoramic radiography, Autoencoders, Convolutional Neural Networks, Vision Transformers, Deep learning.

Sammanfattning

Retinerade hörntänder uppkommer främst hos barn, och innebär att den permanenta hörntanden inte trängt fram när den ska. Behandlingen av retinerade hörntänder är invecklad, där en försenad behandling kan innebära komplikationer. Panoramaröntgen, en sorts 2D-bildtagning, är vanligt förekommande för diagnosering av retinerade hörntänder. Denna modalitet har däremot begränsningar som kan försvåra diagnostiken. I mer komplicerade fall kan datortomografi av tänder (CBCT) användas för att erhålla en 3D-volymbild. Jämfört med panoramaröntgen kan CBCT erbjuda en högre diagnostisk noggrannhet för rotresorption och lokalisering av hörntanden. Däremot avger CBCT en betydligt högre strålningsdos till patienten, och dess användning ska vara väl motiverad. Denna avhandling utforskar möjligheten att använda maskininlärningsmetoder för att förutse om en patient med retinerade hörntänder behöver CBCT-utvärdering givet en panoramaröntgen. Med en datamängd bestående av 109 panoramaröntgenbilder, skapas först baslinje-modeller tränade på numeriska värden uppmätta från dessa panoramabilder. Dessa baslinje-modeller jämförs sedan med djupinlärningmetoder som utnyttjat metoder inom bildklassificering. Närmare bestämt utforskar denna avhandling en metod som involverar autokodare uppbyggda av faltningsnät, samt en metod som innefattar finjustering av en tidigare tränad visionstransformator. Den jämförande studien visar att den bästa baslinje-modellen uppnår ett F1-tal på 0.40 och ett AUROC tal på 0.56. I djupinlärningsdelen av studien undersöktes två instanser av autokodare, och två instanser av finjustering av en visionstransformator. Bland dessa kunde en implementation av en autokodare uppnå högre resultat än basnivån, med ett F1-tal på 0.60 och en AUROC på 0.81. Denna implementation hade en kodare som halverade inmatningsstorleken, som parades ihop med en linjär stödvektormaskin för klassificering. Resultaten indikerar en potential i de undersökta metoderna, men understryker behovet av en större datamängd för att noggrant utvärdera deras lämplighet för uppgiften.

Nyckelord

Retinerade hörntänder, Panoramaradiografi, Autokodare, Faltningsnätverk, Visionstransformator, Djupinlärning.

Acknowledgments

I am deeply grateful to my supervisors, Antoine Honoré and Elena Borsci, and wish to express my sincerest thanks for their unwavering support, invaluable feedback, and enthusiasm throughout this project. Their guidance was central to its successful completion. Additionally, I extend my heartfelt appreciation to my examiner, Saikat Chatterjee, for his encouragement and insightful feedback, and for making this project a possibility.

Furthermore, I had the pleasure of collaborating with Pia Appelquist, and I am thankful for her meticulous feedback on the report and her moral support throughout this project.

Lastly, I am grateful for my friends and family who have shown unconditional support during my time at KTH.

Stockholm, December 2024
Laura Briffa

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

AE	autoencoder
AI	artificial intelligence
AUPR	area under precision-recall curve
AUROC	area under receiver operator characteristic curve
CBCT	cone beam computed tomography
CNN	convolutional neural network
DL	deep learning
FN	False Negative
FP	False Positive
ML	machine learning
MLP	multilayer perceptron
PAN	panoramic radiograph
PR	precision-recall
ReLU	rectified linear unit
RFC	random forest classifier
ROC	receiver operator characteristic
SDG	Sustainable Development Goal
SVM	support vector machine
TN	True Negative
TP	True Positive
UN	United Nations
ViT	Vision Transformer

Chapter 1

Introduction

The topic of artificial intelligence (AI) is ever so prominent with its ongoing research and practical applications [1]. AI refers to the capabilities of machines or computer systems to have human-like learning, problem-solving, or reasoning. Within the field of AI exists the sub-domain of machine learning (ML), which is the study of making computers learn through data, mathematical models, and algorithms [2]. Advances in these domains enable us to construct systems capable of numerous tasks, from routine labor automation, to speech recognition and synthesis, and sophisticated tasks such as making medical diagnoses [1]. The part of ML concerned with systems recognizing input data as belonging to a specific category is called classification.

Traditional ML algorithms find success with many important problems. However, simple ML algorithms have been insufficient for more advanced problems in AI that require generalization, such as image recognition. To tackle these challenges with generalization, the field of deep learning (DL) developed. DL was designed to work with the challenges that come with high-dimensional data, and its methods can learn more complex representations of data [1]. In this thesis, traditional ML and conventional DL techniques will be evaluated, to study the role of ML in the diagnosis and treatment of the dental condition of impacted maxillary canines.

This chapter gives an introduction and context to the specific problem that this thesis aims to address. Moreover, the goals and delimitations of this thesis project are determined, and an outline of the thesis structure is provided.

1.1 Background

The canine tooth is the third tooth from the center of the mouth, and it is a pointed tooth useful for tearing food. Humans have four canine teeth, and the two located in the upper jaw are called the maxillary canines [3]. Maxillary canine impaction is a condition where a canine tooth in the upper jaw fails to erupt when expected [4]. The prevalence of impacted maxillary canines is reported to be in the range of 1.1 to 4.7%, depending on the population [5]. Treatment of canine impaction is non-trivial, and complications can arise if it is not treated in a timely manner. Diagnosis of canine impaction is based on clinical and radiographic evaluation. In some cases, panoramic radiography is utilized to localize the impacted canine [4, 6]. A panoramic radiograph (PAN) is a 2-dimensional X-ray image depicting all the teeth [7]. If further information about the exact position of the impacted canine is needed, cone beam computed tomography (CBCT) can be used. However, CBCT is not routinely used as it exposes the patient to a higher radiation dosage [4].

Research concerning the application of ML in dental imaging has expanded in the last few years [8]. Several studied use cases for ML in dental imaging are presented in [8], such as the identification of oral cancers, landmark identification in cephalometrics, and detection of dental caries. A recurring choice of method is the utilization of convolutional neural networks (CNNs), a type of DL architecture [8]. One challenge of research in the field is the small number of data points a research group has available, often having around 1000 X-rays per group, or in some cases, in the hundreds [8]. This might explain the popularity of fine-tuning pre-trained CNNs in the conducted research.

As for using DL methods for impacted canine-related problems in particular, there is much to fill in as research for this application is in its infancy. At the moment of writing, three different research articles have been found using the keywords ”deep learning” and ”impacted canines”. Different pre-trained CNNs architectures are used together with a dataset of PANs in [9], to classify two types of canine impaction. The use of generative AI is studied in [10], for 3D-reconstructions of PANs, to assess localization of impacted maxillary canines. Finally, in [11], automated image segmentation of impacted maxillary canines in CBCT-scans is performed using CNN architecture.

1.2 Problem

Today, there is a limited ability to predict if a case of canine impaction will be more complicated, requiring greater attention and a longer treatment time. Early indicators of a complicated condition would be beneficial for both the patient and the practitioner. Maxillary canine impaction is a complex condition with its treatment requiring multidisciplinary care. Difficult complications can arise and the treatment outcome has both aesthetic and functional consequences that need consideration. With this as background, the Department of Dental Research at Karolinska Institutet has assembled a dataset from 109 patients with impacted maxillary canines. The dataset contains PANs of each patient, as well as angular and linear measurements made by radiologists from these PANs. There is also some information about the treatment, such as if a CBCT was performed for a specific patient.

1.2.1 Original problem and definition

To define the original problem, two research questions were specified.

R1 Can the features available in the given dataset be used to construct simpler baseline classification models, and can they predict if a case of maxillary canine impaction is complicated?

R2 Can we do better than the baseline classification models, by using PANs together with more intricate DL image classification approaches?

1.2.2 Scientific and engineering issues

The research questions in 1.2.1 do not specify the exact classification task, but clarification is necessary for scientific research. No such label as ”complication degree” is available in the dataset, and there is no clear reason to choose one label over another. A decision was made in accordance with researchers at Karolinska Institutet, where labels that can be an indication of case complexity were considered. It was decided that the classification task of the ML models was to classify if a patient needs CBCT referral or not.

A further consideration is the extensive amount of approaches for image classification tasks in general, while there also does not exist a favorable type of model for the kind of images in the dataset. Therefore, it is interesting to study different approaches and compare these.

1.3 Purpose

The purpose of this thesis is to study different ML methods for the task of finding complications of impacted canines. As stated in 1.2, researchers and practitioners within the field are interested in predicting the complexity of a patient's condition. Therefore, this thesis aims to contribute to research on practical methods for a condition that mainly affects children. By completing the goals of this thesis, we would be one step closer to making a system able to assist dentists with the decision of CBCT referral.

Moreover, there is also the purpose of contributing research to an application of DL that is not particularly deeply researched. It is of interest to study what approaches finds success for this particular area and type of dataset. Furthermore, due to the limited dataset, the thesis explores the issue of data-limitation. The thesis aims to examine what kinds of ML methods are suitable with these data constraints.

1.3.1 Sustainability

The thesis addresses sustainability aspects defined by the United Nations (UN) Sustainable Development Goals (SDGs). Primarily, the thesis contributes to SDG 3: Good Health and Well Being. The thesis aims to develop a predictive system that assists clinicians in identifying cases where CBCT imaging is necessary for impacted canines. By accurately predicting the need for CBCT, the system can improve clinical decision-making, which improves treatment outcomes and patient health. Predicting the need of CBCT can also minimize unnecessary radiation exposure for patients. This is particularly significant in the case of impacted canines, where many patients are children who are more sensitive to radiation. The aspect of utilizing CBCT only when truly needed also touches on SDG 12: Responsible Consumption and Production, as clinically justified use of CBCT conserves energy and reduces material consumption.

1.4 Goals

The goal of this project is to establish a comparison of models for classifying the need for CBCT referral given a PAN. This has been divided into the following sub-goals:

1. Train and evaluate the baseline models, utilizing the linear and angular

features measured from PANs.

2. Train and evaluate the DL approaches: the autoencoder and transfer learning implementations. These models use the PANs as input data.

1.5 Research Methodology

The research methodology relates to retrospective cohort studies, as imaging data from past patients with impacted maxillary canines received from Karolinska Institutet at the Department of Dental Medicine will be utilized. This data will be used to predict the necessity of CBCT imaging for patient treatment. The methodology involves studying existing measurement-based decision rules for further treatment, and implementing classifiers to establish a baseline utilizing these measured features. Following the established baseline, the method will comprise of design and evaluation of advanced models based on autoencoders and transfer learning approaches. A benchmark comparison of various methods will be performed to analyze the results.

1.6 Delimitations

The project is limited with respect to the dataset as it will only be used in its given form. No further work with labeling, segmentation, or cropping of images by researchers and dentists from Karolinska Institutet is possible. This results in the project being limited to image classification tasks. Moreover, it is not possible to collect more data, thus, the implementation utilizes the data of 109 patients with available images and measurement-based features.

Furthermore, in studies that employ transfer learning approaches such as fine-tuning pre-existing models, it is common to compare several such trained models. This thesis limits itself to one pre-existing model. The focus of the thesis is to compare different ML approaches, not models within the transfer learning domain.

1.7 Structure of the thesis

Chapter 2 presents relevant background information about the diagnosis of maxillary canine impaction, as well as ML techniques for image classification. Chapter 3 presents the methodology and methods used to solve the problem. In Chapter 4, details about the implementation of the project are provided.

The results and metrics of the various approaches are presented in Chapter 5. Chapter 6 provides an analysis of the results and a reflection on the method. The conclusions are presented in Chapter 7.

Chapter 2

Background

This chapter provides background information about the condition of impacted maxillary canines. Additionally, the topic of deep learning will be described through the lens of medical image classification. Lastly, descriptions of related work will be provided.

2.1 Diagnosis of Impacted Maxillary Canines

Usually, the maxillary canines erupt between the ages of 10.8 and 11.6 years [12]. If the eruption of the canine is delayed compared to the development of the other teeth, the canine is considered impacted [6]. An effective treatment of canine impaction involves combining the specialties of orthodontics and oral and maxillofacial surgery [6].

An early diagnosis is essential for timely treatment, reduced costs, and a decreased risk of complications [13]. Some of the most common complications include root resorption in adjacent teeth, bone loss, and gum recession around the treated teeth [4]. Further, in [14], it was found that the patient's age had an impact on the treatment outcome. An older age was linked to suboptimal esthetic results, prolonged treatment times, and the need for re-intervention [14], supporting the necessity for early diagnosis.

Root resorption in adjacent teeth is notably difficult to treat [15]. In more severe cases involving a sustained loss of tooth tissue, a permanent tooth affected by root resorption may not be possible to salvage [16]. In that case, the affected tooth requires extraction [15].

Diagnosis of canine impaction is based on clinical and radiographic

evaluation [4], although specific guidelines vary across countries [12]. In Sweden, the guidelines in some counties state that the canines should be checked through palpation (the dentist uses their fingers to check the area physically) from the age of 9, and if no bulge from the canine is found it is recommended to perform a new examination within a year. If it is suspected that there is a disturbance to the eruption, radiographic imaging evaluation should be performed [12]. Further, the precise localization of the impacted canine is central for diagnosis and treatment planning [17], which may be achieved through radiographic imaging.

2.1.1 Evaluation through Panoramic Radiography

Panoramic radiography is an extraoral imaging method that gives a two-dimensional image of all the teeth, mandible (lower jaw), and parts of the maxilla (upper jaw) [7]. An example of a PAN is shown in Figure 2.1. Panoramic radiography is a one-time imaging method useful for the diagnosis and treatment planning of various conditions, such as impacted teeth [7]. Additionally, in orthodontics, it is a standard diagnostic tool for pre-operative diagnosis [17]. In clinics, 2D imaging methods such as the PAN have been the most common modality used not only for the localization of impacted canines but also for treatment planning and treatment result evaluation [17].

However, the use of panoramic radiography for evaluating impacted canine-related issues has its limitations. For instance, the radiographs are sometimes difficult to interpret, leading to a poor localization of the impacted canine. This is due to different factors such as distortion, and the fact that the maxilla is projected into 2D [17]. Furthermore, PANs may not be sufficient for detecting root resorption in adjacent teeth [15, 18, 19].

2.1.2 Evaluation through Cone Beam Computed Tomography

CBCT is an imaging technique that provides three-dimensional volumetric data [20]. The technique involves taking x-ray projection images while the CBCT scanner rotates around the region of interest. These images are then combined to form a 3D reconstruction through the help of algorithms [20]. CBCT has many clinical applications [20], and can be used for localizing impacted canines, as well as for detecting root resorption [17]. Compared to PANs, CBCT scans have improved diagnostic capabilities in cases of impacted canines. Regarding localization and detection of root resorption, it is reported

Figure 2.1: A panoramic radiograph from the dataset.

that CBCT has a higher diagnostic accuracy [17, 21]. However, CBCT is not routinely used due to its cost and the radiation exposure inflicted on the patient [4]. Principles and guidelines state that the use of CBCT should be well motivated and used with caution for specific patients where conventional radiography is not adequate for a diagnosis [22].

Therefore, algorithms able to predict the need for CBCT from previous radiographic evaluations could be useful to help clinicians motivate their choice.

2.2 Machine Learning for Image Classification

Machine learning is a field of study concerned with the usage of data to construct methods able to detect patterns. These learned patterns can then be used to perform different kinds of decision-making tasks on unseen data [23]. This thesis has a primary focus on supervised learning, which is an approach where the given input training data has known outputs (labels), and the aim is to learn a mapping from input to label [24]. When the output is categorical and has a limited amount of possible values (classes), the problem at hand is called a classification task [23]. More formally, it means that we assume there exists a function f such that, $f(\mathbf{x}) = y$, where $y \in (1, \ldots, C)$ is the output with C possible classes. The input data $\mathbf{x}$ is frequently denoted as ”features”. The supervised learning task is to estimate the function f given labeled training

data, to make predictions $\hat{y} = \hat{f}(\mathbf{x})$ on unseen data [23]. Within the field of ML, there exist other learning approaches such as unsupervised learning. Unsupervised learning does not require the use of labels, and tasks can include distribution estimation and clustering [23].

ML techniques can be utilized with medical data for specific tasks such as classifying patients as having a specific condition or not [25]. The classification is based on relevant attributes, or features, of the patient. For instance, when classifying cardiovascular disease, dietary patterns, demographics, and lifestyle attitudes are examples of possible features [25]. Classification tasks may be approached with a variety of algorithms, including logistic regression, support vector machines (SVMs), Decision Trees, and Artificial Neural Networks [25]. Some of these models are examined in this thesis, and readers interested in further descriptions of these models are referred to [25].

With the advances in computational capabilities, it has become possible to use large amounts of data to train complex models [24]. For instance, this has allowed the design of models that learn to classify diseases directly from imaging data [24]. The methods found in recent medical imaging literature employ a class of methods called deep learning. Deep learning is a subset of ML, which can learn more complex representations of the input data to accurately predict an output [1].

2.2.1 Models for deep learning: Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN)

In [1], I. Goodfellow, Y. Benigo, and A. Courville describe the multilayer perceptron (MLP) as the "quintessential deep learning model". MLPs are chains of more elementary functions composed of a linear transform followed by a non-linearity. This simple yet powerful design, allows the MLP to learn complex mappings between input and output data. MLPs are the simplest form of feedforward neural networks, where the output of a layer is directly fed as input to the next layer, and the output is not fed back into the model. The non-linearity is primarily introduced through the activation functions of the hidden layers. A common choice of activation function is the rectified linear unit (ReLU), defined as $g(z) = \max\{0, z\}$. The ReLU function yields a non-linear transformation, but since it is piecewise linear, it keeps some favorable properties of linear models [1].

During training, the MLP (and other deep learning algorithms) learn a function approximation of the mapping from the input $\mathbf{x}$ to the output y. More

formally, the approach aims to find the model parameters θ that minimize a chosen loss function. Gradient-based optimization algorithms, such as stochastic gradient descent, are commonly used for the minimization task. As such, the calculation of the gradient with respect to the model parameters θ is required. The back-propagation algorithm allows for the estimation of the gradient, by letting the loss flow backward through the network [1].

CNNs are a kind of neural network that has shown success in practical applications [1]. They are used for processing data with grid-like topology, such as image data. While other types of neural networks use general matrix multiplications, the CNN utilizes the convolution operation. With a two-dimensional image I as input, and a 2-dimensional kernel K, the convolution over two axes is obtained through,

$$(I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n).$$

By making the kernel smaller than the input, the CNN structure allows for so-called sparse connectivity. This means that a smaller number of connections are used to calculate outputs in the network. This improves efficiency in memory management as fewer model parameters are needed, as well as reducing the number of operations to calculate outputs, which reduces runtime [1].

2.2.2 Attention and The Vision Transformer

A certain type of DL architecture, the Transformer, has shown great success for natural language processing tasks [26]. Introduced in 2017 [27], this model is based entirely on a self-attention mechanism. The self-attention mechanism is inspired by how humans pay attention to parts of information in a larger context. The mechanism calculates weights of tokens/words to find the most important regions [28]. For further insights, refer to [27]. In 2020, the transformer architecture was modified for computer vision tasks, resulting in the creation of the Vision Transformer (ViT) [26]. Here, the input image is divided into patches, that are applied to the transformer the same way as tokens/words in natural language processing settings [26]. A positional embedding is added to maintain the positional information of the patches [26]. Compared to the state-of-the-art CNNs the ViT has obtained excellent results when trained on larger benchmarking datasets [26].

The question of whether the ViT is better than convolutional architectures for medical image classification is unanswered. Nonetheless, the architecture

shows signs of potential for application in the field. One advantage of the ViT is its ability to capture both local and global patterns. This can be beneficial when diagnosing medical conditions, as the surrounding areas of the medical concern are also of interest [28]. Further, the ViT has a reduced inductive bias, *i.e.,* the model makes fewer assumptions about the input data compared to CNN-based models. For instance, the sizes of the input images are flexible, and only the positional embedding of the image patches is manually entered [28]. However, the weaker inductive bias does come with the downside of requiring a larger dataset for effective training [28].

2.2.3 Feature Learning and Autoencoders

Feature learning is a part of ML concerned with techniques for representing data. These techniques can for instance be used for constructing features from raw data, and for denoising data. It is common practice to reduce raw data as it often contains redundancies that can diminish performance. More formally, approaches in feature learning aim to find a mapping $f : R^n \rightarrow R^m$, where n-dimensional data is transformed to a set of m-dimensional labels, where $m < n$. As such, the approaches can be described as techniques for dimensionality reduction [29].

Autoencoders (AEs) are a type of neural network with an encoder-decoder structure that learns to output a reconstruction of the input from a compressed latent variable. There are different types of AEs, but dimensionality reduction is a traditional application. The encoder of the AE learns to reduce the input into latent variables, which have a smaller dimension than the input. The decoder aims to reconstruct the original input from the latent variables [30]. The training procedure is unsupervised, meaning the data is used without labels, often using the reconstruction error for learning. After training of the AE, the encoder-part can be utilized for dimensionality reduction of the raw data, and the encoded latent variables can be used as features for a downstream task such as a classification task [29].

2.2.4 Transfer Learning

Transfer learning stems from the idea that knowledge about a certain domain can be utilized to improve performance on new tasks in another domain. A common transfer learning practice is to select a DL model that is pre-trained with a large dataset, and fine-tune it with new data, perhaps from another domain. Fine-tuning can be performed by freezing existing layers in the

network, so their parameters are not updated when the model trains with the new data [31].

Within medical image analysis and classification, there exists a problem of data unavailability. This stems partly from the high costs of using experts for annotating and labeling data. Data scarcity is a problem when working with DL models, such as deep CNNs, as a large amount of data is preferable. Approaches within transfer learning aim to overcome this problem, as less data is typically needed. Consequently, it has made a major contribution to the area of medical image classification [31].

2.3 Related work

2.3.1 Feature learning methods for dental imaging applications

A method using variational autoencoders to predict patient age, using segmented PANs from 910 patients was employed by Joo, Jung, and Eel Oh [32]. The variational autoencoder is a type of AE where the encoder maps the input to a probability distribution of the latent variables. For training, the reconstruction error as well as the Kullback-Leibler divergence is used. The method described by [32] utilizes encoders and decoders composed of 5 convolutional layers. The dataset consists of grayscale images with size 256×256 pixels, depicting single teeth manually segmented by dentists from PANs. After training, the latent variables are used to train a linear regression model to estimate the chronological age.

2.3.2 Transfer learning approaches in related areas

Four different pre-trained CNN architectures are utilized by Aljabri, Aljameel, *et al.* [9] to classify two types of canine impaction according to the Yamamoto classification. The CNN-models used in the study were DenseNet, VGG, Inception V3, and ResNet-50. Their original dataset consisted of 416 images of canine teeth from PANs, labeled by expert dentists. The image preprocessing steps are clearly described, and among other actions, the images were rescaled to a size of 224×224 pixels that depicted a canine and the immediate region. To have an equal distribution of the two classes, random undersampling was performed, which resulted in a second dataset consisting of 268 images, with 134 samples of each type. These two datasets were used for two different experiments to evaluate the models, and the datasets were

split into 80% training and 20% testing sets. The best-performing model was the Inception V3, which with the imbalanced dataset yielded an accuracy of 0.8095, precision of 0.7143, recall of 0.4545, and F-score of 0.5555. With the balanced dataset, where undersampling had been performed, the results improved for the Inception V3 model, achieving an accuracy of 0.9259, precision of 0.9355, recall of 0.9355, and F-score of 0.9355 [9].

2.3.3 Non-DL approaches for predicting outcomes of impacted canines

The use of panoramic radiographs for statistical analysis and prediction of canine impaction-related problems has been studied in the last decade. Some of these methods, such as by [13, 22, 33], have relied on using angular and linear measurements from panoramic radiographs as features and constructing multivariate logistic regression models. The measurements from the radiographs are measured by radiologists using software tools.

For instance, Alqerban, Storms *et al.* [33], studies different linear and angular measurements taken from panoramic radiographs for predicting canine impaction. The three most statistical discriminant parameters were determined through the calculation of the AUC of Mann-Whitney U-tests. Then, these most discriminant parameters were used as features to construct a logistic regression model. The model achieved an AUC of 0.97, with 90% sensitivity and 94% specificity [33], although the description of the test data is ambiguous. The objective of Margot, Cadenas De Llano-Pérula *et al.* [13] was to expand upon the approach presented in [33] and use linear and angular measurements of a larger sample size to predict canine impaction, as well as validate the model presented in [33]. While the former logistic regression model from [33] now achieved an AUC of 0.594, the new prediction model with a new set of parameters yielded an AUC of 0.750 [13].

While [33] and [13] studied the task of predicting the occurrence of canine impaction at an early stage, Alqerban, Jacobs *et al.* [22] studied the use of panoramic radiography for predicting root resorption. In [22], panoramic radiographs are used to obtain measurements and parameters similar to [33]. These parameters are then used to construct a multivariate logistic regression model to predict the presence of root resorption in adjacent teeth. Using leave-one-out cross-validation, the model yielded an AUC of 0.71. The metrics on a validation set consisting of 55 canines yielded an AUC of 0.687, sensitivity of 50%, and a specificity of 84.6% [22].

Chapter 3

Method

This chapter explains the research process and the general procedure for the implementation. Details about the given experimental setup and dataset will be provided. Lastly, the metrics used for comparisons of the approaches will be described.

3.1 Research Process

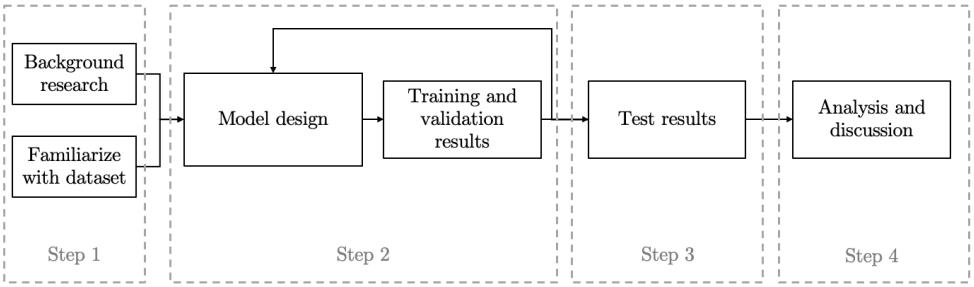


Figure 3.1: An overview of the research process and procedure

The research process can be summarized with the following steps,

Step 1 Background research and familiarize with measured features and image dataset.

Step 2 Design classification models (baseline, AE implementation, fine-tune vision transformer) iteratively.

Step 3 Collect models' performances on test set.

Step 4 Analyze and discuss the results.

The steps of the research process are also illustrated in Figure 3.1, and shows that the first step is to conduct background research in parallel with working with the dataset. Step 2 and Step 3 will be repeated for each approach to the classification task. The first approach is to use the measured features to construct a classifier denoted as the baseline in this project. The second approach will use feature learning-based methods, utilizing convolutional AEs. The third approach is based on the transfer learning-based method where a pre-trained ViT model will be fine-tuned. For all three approaches, the procedure of Step 2 is to improve upon the models iteratively, using training and validation sets. In Step 3, the models' performances on the test set will be recorded using the evaluation metrics presented in section 3.5, to lay grounds for analysis and discussion in Step 4.

3.2 Data Collection

The data provided by the research group at Karolinska Institutet had passed selection criteria and quality assessment. The target study group was patients between the ages of 9 - 23 who had been diagnosed with failure of maxillary canine eruption. An initial group of patients came from a pedodontics clinic, found through referral lists for surgical exposure of canines. The patients' journals were inspected, and in order to qualify for the study, they had to be free of maxillofacial abnormalities and must have had at least one PAN. Selection based on quality assessment of the PANs was also conducted. The final dataset contained data from 109 patients. Information about CBCT referral was found in the patients' journals, where the number of scans and dates of the scans were recorded in the dataset. 28 of the patients in the dataset had at least one CBCT during treatment. This CBCT information paired together with dates of the scans were used as a basis for labeling the dataset. See Section 4.1.1 for exact label statistics. Information about canine localization was also available in the dataset.

3.2.1 Panoramic Radiographs

The PANs were evaluated by the researchers at Karolinska Institutet and a selection was made to ensure the quality of the data. Imported PANs, PANs taken using another machine, and PANs with another format than the DICOM format, were excluded. Then, quality assessment of the PANs was conducted

using an evaluation protocol, to exclude PANs with positioning errors and insufficiencies for diagnostics.

As each patient had *at least* one PAN, the total number of provided DICOM images was 161. All patients had a pre-orthodontic treatment radiograph, but some patients also had one or more radiographs taken later during treatment. Still, 109 images are used for the implementation of this project, to have directly comparable results to the baseline model, see Section 4.1.1. The images vary slightly in size, with a height of around 1200px and a width of around 2500px.

3.2.2 Measured features

The dataset also contains the measured features: angulations, tooth depth, and horizontal and vertical sectors, which were calculated by the OnyxCeph™ software. Three different operators, with one being a resident in Oral and Maxillofacial Radiology, manually marked anatomical points in each PAN, and the OnyxCeph™ software calculated the measured features. Figure 3.2 depicts how OnyxCeph™ displays information in the PAN when manually marking anatomical points. The measurements were performed on the 109 pre-orthodontic treatment PANs.

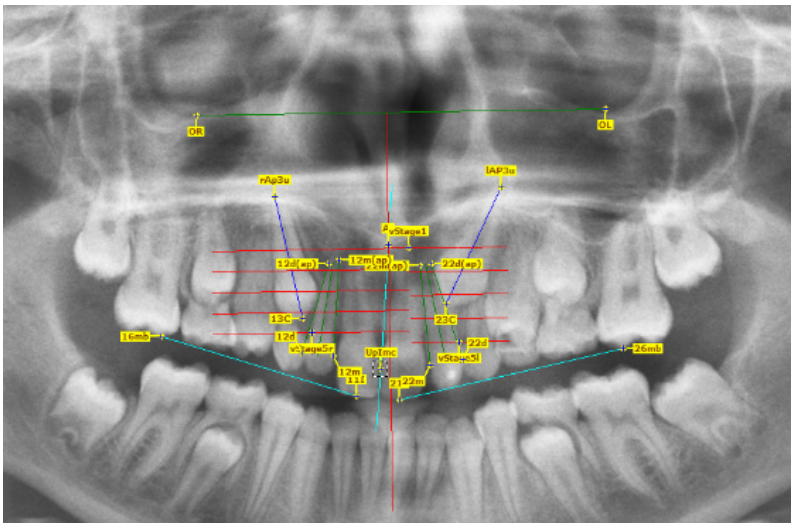


Figure 3.2: A panoramic radiograph in the OnyxCeph™ software. The markings show landmarks and different measurements used to determine the measured features.

For each patient, the measured features in Table 3.1 were calculated for

Feature name	Short description
Tooth depth	A measure of the canine tooth depth in millimeters.
α-angle 1	The angulation of the canine with respect to the midline drawn between the central incisors.
α-angle 2	The angulation of the canine with respect to the midline with a central point on the spina nasalis anterior.
Overlap	Categorical feature with range $\{1, \ldots, 4\}$. Measures how much the canine overlaps the adjacent incisor.
Vertical	Categorical feature with range $\{0, \ldots, 5\}$. Measures the vertical placement of the canine.

Table 3.1: The measured features calculated for both canines of each patient.

both maxillary canines utilizing OnyxCeph™Therefore, there are 10 measured features for each patient in the dataset.

3.2.3 Ethics

Several ethical considerations are needed as the dataset consists of medical data. Firstly, the study has an approved ethical application with reference number Dnr. 2022-01284-01. Another consideration is that the measured features can be pseudo-anonymized, but PANs cannot be completely anonymized and could be used to identify individuals. Therefore, data security is important to protect from unauthorized access. Data security has been ensured by not sharing data with any third parties, and keeping the data and experimental setup local. Furthermore, it is important to consider who benefits from this project, and to make sure that the patient's well-being is prioritized.

3.3 Experimental Design

The overarching goals mention three approaches that will be studied in this project. The first experiment will be concerned with using the measured

features only. The second and third experiments will focus on the DL techniques for image classification: the feature learning and transfer learning approaches. These latter experiments will not use the measured features, and will instead use the labeled PANs.

3.3.1 Classification Task Using Measured Features

The models utilizing the measured features as input data are termed the baseline models. The reasoning behind the name is that similar methods have previously been researched, see Section 2.3.3, and there is interest in comparing the results of such methods with the DL results.

Various ML classifiers will be trained using the measured features. These models are logistic regression, SVM, random forest classifier (RFC), and the MLP, all widely employed for classification tasks. These models are chosen to experiment with different model complexities, and to study if a particular model is especially suitable. The model hyperparameters are chosen through a grid search. Furthermore, some experimentation will be conducted for using the MLP classifiers with data oversampling, *i.e.,* copies of minority class samples are added until an even ratio of positive and negative samples is obtained. A graphical overview of the implemented methods is shown in Figure 3.3

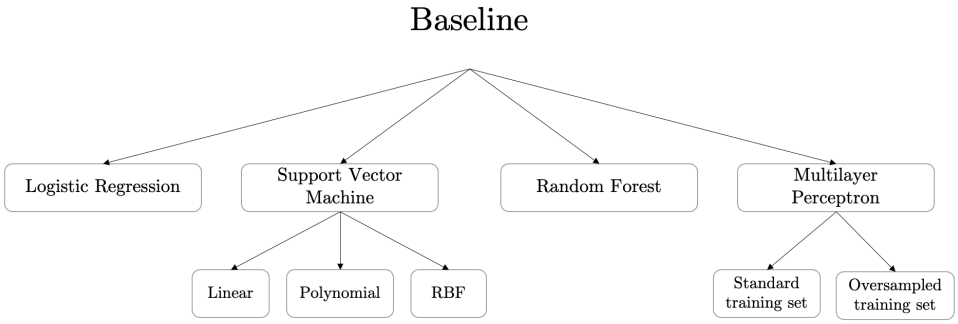


Figure 3.3: Overview of all examined baseline models. Some specific hyperparameters, such as the SVM kernels, are highlighted.

3.3.2 Classification Task Using PANs

For the feature learning approach, different autoencoder structures will be considered. A couple of autoencoder structures are chosen, where the encoder

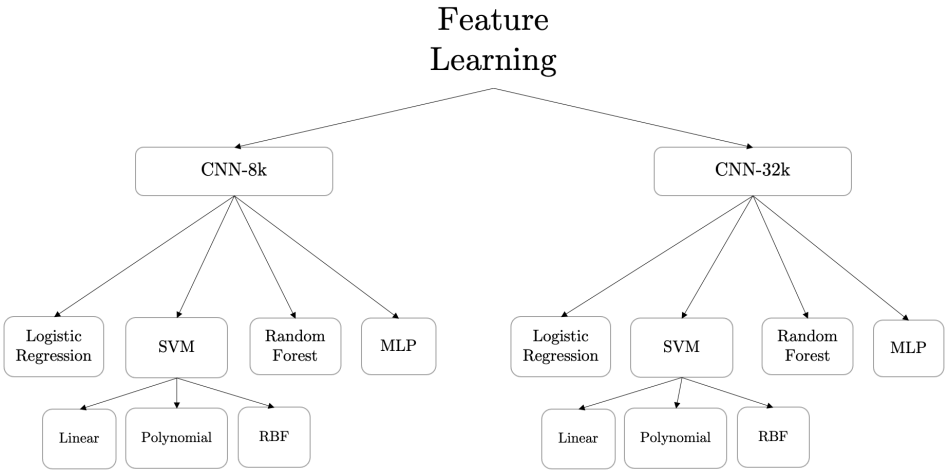


Figure 3.4: Overview of the feature learning implementation.

part will act as a feature extractor, taking in a PAN and outputting a vector with reduced dimension. These extracted features are then used as inputs to logistic regression, SVM, RFC, and the MLP models. The hyperparameters are again chosen through a grid search. An overview of the models used for the feature learning approach is shown in Figure 3.4. The two examined autoencoder structures are named CNN-8k and CNN-32k, and are presented in Section 4.3.

For the transfer learning approach, the pre-trained vision transformer "ViT-B/16" [26] will be utilized. There are different Vision Transformer models proposed in [26], where the smallest version "ViT-B/16" stands for a "Base" version that has a 16×16 input patch size. It was pre-trained with 224×224 images from ImageNet-21k, a dataset containing 14 million non-medical images and 21843 classes. In the implementation of this thesis, "ViT-B/16" is fine-tuned with the PAN dataset. A couple of pre-processing techniques will be studied, where two different image input sizes will be used for training. Furthermore, experimentation will be performed for training duration, where an early-stopping procedure is examined. A graphical representation of the transfer learning method is provided in Figure 3.5.

3.3.3 Average Results

There is an additional experiment where the models with the final hyperparameters are further evaluated. This further evaluation utilizes 20 random train-test splits to re-do the train and test procedure, to obtain average results. The reasoning is that the results for just one random train-test split

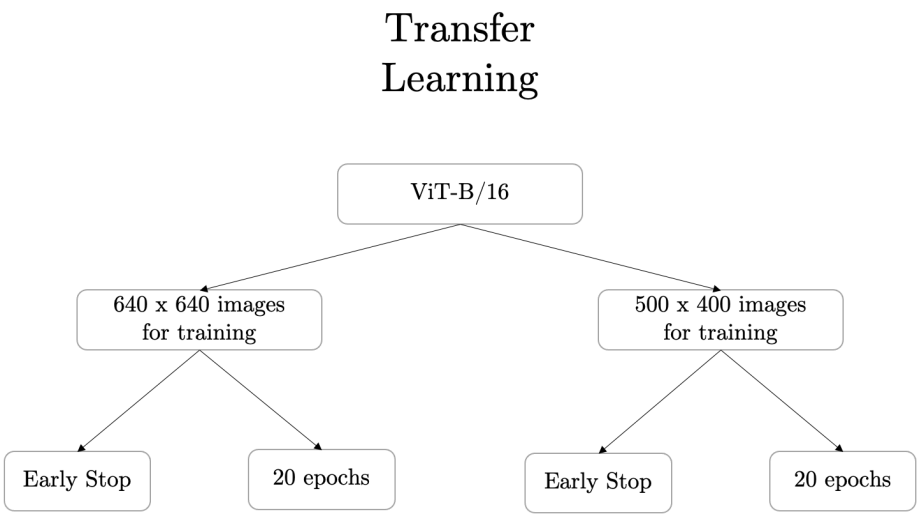


Figure 3.5: Overview of the transfer learning implementation.

might be unreliable, see Section 3.4.2.

3.3.4 Experimental Setup

The project is implemented with the help of several Python libraries. For dataset-splitting, non-DL based models, grid-search functionality, and model evaluation, Scikit-learn [34] is utilized. For image preprocessing, and the DL implementation and training loop, PyTorch [35] is used. Integration of ”ViT-B/16” is enabled through PyTorch Image Models [36]. Reading DICOM files is enabled through pydicom [37]. As for hardware, a computer with an 8GB Nvidia RTX3070 is utilized during training.

3.4 Reliability and validity of the dataset and methods

3.4.1 Data validity

Several things can be examined about the dataset’s validity *i.e.,* the accuracy and truthfulness of its representation of the patients. First of all, an extensive quality assessment was performed. One reason for the assessment was to discard samples of patients that was affected by underlying maxillofacial abnormalities that can skew the features. Furthermore, the dataset contains

radiographs with the same type of image format, taken from the same type of machine, and the quality assessment made sure that the placement of the mouth was consistent throughout the dataset. As such, there is consistency in the data and approaches for unbiased data, promoting its validity.

On the other hand, some data practices might affect the dataset validity negatively. For instance, the labeling of the data, described in Chapter 4, was not straightforward in some cases. This might lead to selection bias even though consultation was made with researchers, as one might be prone to label true or false for any reason. This impacts the validity of the data, as there is a question of whether a data point is truthful in its labeling.

Lastly, there is a question of whether the classification of CBCT based on PAN is an appropriate task, as the radiologist might have more information than just the PAN when a CBCT referral is made. In that case, the data does not contain all the information needed for the decision, which lessens its validity as it does not represent the underlying concept. However, the fact that the PAN lays grounds for CBCT referral in practice, supports the use of this dataset. Also, for patients with CBCT scans, the reasons behind the CBCT were checked, to verify that the CBCT referrals were made with the impacted canines and the PAN in mind, and not some other condition that needed CBCT evaluation.

3.4.2 Reliability of data

It is necessary to evaluate the reliability of the data as well. Here, reliability is concerned with how consistent the results are with repeated trials under the same conditions. A primary limitation of the dataset is its small size. When splitting the data into different train-test splits, the results can vary depending on what cases of a training or testing set contains. This limitation is recognized, and is one reason for reporting average metrics as part of the project results.

A further aspect to the reliability of the data, is how consistent the measured features are between the three operators. With three different measurements representing the same construct, it is important that they do not vary a lot from each other as it would not only question its reliability but validity as well. Here, the research team at Karolinska Institutet found that the three operators had a very strong correlation for each measurement.

3.4.3 Validity of method

With many image classification techniques available, with no type established as particularly appropriate for this image dataset, it is difficult to know the validity of a chosen method beforehand. This thesis aims to study different approaches for this image classification task, to contribute to the discussion about what is suitable.

3.4.4 Reliability of method

For the implementation of the project, all random seeds used for random number generation are saved. This is to ensure repeatability and therefore reliability. However, some models will not have exactly the same outcome if trained again even with the same training set. This applies especially for the autoencoder training and the fine-tuning of the pre-trained ViT. The hyperparameters of the models are however reported in this report for transparency.

3.5 Evaluation Metrics

To evaluate and compare the results of the binary classifiers, a group of performance metrics is used. A description and motivation of the chosen metrics for this project are provided in the following sections.

3.5.1 Confusion Matrix

The confusion matrix gives information about how the predicted class compares to the true class of a patient. In the confusion matrix, see table 3.2, the number of True Positives (TPs) is the amount of correctly predicted cases of the true class, while False Negative (FN) is the number of cases that are positive but were predicted as negative. The confusion matrix also gives information about the amount of True Negatives (TNs) and False Positives (FPs) [38], see table 3.2.

3.5.2 Accuracy

Accuracy is a commonly used metric, which is the proportion of correctly classified cases among all cases. Using the same terminology found in the

	Predicted Negative	Predicted Positive
Actual Negative	TN	FP
Actual Positive	FN	TP

Table 3.2: A confusion matrix in the binary case.

confusion matrix, the formula for accuracy is,

$$\text{Accuracy} = \frac{\text{TN+TP}}{\text{TN+TP+FN+FP}}. \tag{3.1}$$

However, accuracy can be misleading when working with an imbalanced dataset [38]. Therefore in this project, accuracy is considered less important.

3.5.3 Precision and Recall

Precision, also known as the positive predictive value, is used to evaluate how accurate the model is at predicting positive values. Precision is the proportion of TPs over all positive predictions,

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}.$$

Recall, also called sensitivity, is a measure of the model's ability to predict positive outcomes [38]. It is the proportion of actual positive cases that are predicted as positive and it reflects on how many of the positive predictions are relevant [39]. Recall has the formula,

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}.$$

Both precision and recall are valuable metrics that focus on the positive class, but there is often a trade-off between them, and one often aims to improve the recall without affecting the precision [38]. In medical problems in general, recall is essential, as there is a desire to find all actual positive cases [39]. In the context of this project, the positive class represents that CBCT referral is necessary for the patient. Finding the patients that need CBCT referral is important, which makes the recall score important. However, one can also

argue that the precision score is crucial, as we want to limit the number of FP so that children are not unnecessarily exposed to the radiation of a CBCT scan.

3.5.4 The F1-score

The F1 score combines the precision and recall scores of a model. It is defined as the harmonic mean of the two,

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

3.5.5 The ROC-curve and AUROC

The receiver operator characteristic (ROC) curve is also used for evaluation in this project. It is calculated by plotting the false positive rate versus the true positive rate for different classification thresholds. It can be made into a numerical score ranging between 0 and 1 by calculating the area under the curve, yielding the area under receiver operator characteristic curve (AUROC) [38], allowing for straightforward comparison between methods. The AUROC can also be interpreted as the probability of the classifier ranking a randomly chosen positive case higher, *i.e.,* higher predicted probability than a randomly chosen negative case [40]. A classifier with no predictive power that makes random predictions gives a 45-degree diagonal in the ROC curve, and an AUROC of 0.5 [40].

3.5.6 The PR-curve and AUPR

The precision-recall (PR) curve plots precision as a function of recall, for different classification thresholds. For a PR curve, a classifier making random predictions yields a horizontal line at the proportion of true cases in the dataset. The area under precision-recall curve (AUPR) can be calculated for a numerical representation.

In the presence of class imbalance, the PR curve may be preferable over the ROC curve. For instance, if it is deemed important that the predicted positive samples are mainly actual positives. Another reason for using the PR curve over the ROC curve is if there is a class imbalance in the dataset, as it often leads to the AUROC being inflated [40]. A further explanation on this matter is found in [40].

3.5.7 Loss curve

For the DL models implemented in this project, the cross-entropy function [1] is used to perform backpropagation during training. The loss curve displays the loss of the classifier per epoch, during training and validation. From the loss plot, one can interpret how the model is adapting to the training data. It may be possible to make conclusions, such as if the model is overfitting on the training data. This type of curve is essential during model construction and hyperparameter tuning of the deep learning models.

Chapter 4

Implementation

In this chapter, the implementation of the project is described and design choices are explained. Additional details about the execution of the described experimental design in 3.3 are provided.

4.1 Data preprocessing

This section describes how the data was processed, and the steps taken for converting the given dataset into passable inputs to the models.

4.1.1 Data labeling

The 109 data samples used for the implementation needed to be labeled given the patient information about CBCT referral. The PANs and measured features belonging to patients who did not have CBCT scans, could directly be labeled as CBCT Negative. However, further consideration was needed for the positive cases. For these cases, the recorded dates of the CBCT scans were compared to the date of the PAN scan, also making sure to use the same PAN used for calculating the measured features. The procedure for labeling these samples is shown in Table 4.1. The labeling procedure resulted in a dataset consisting of 83 Negatives and 26 Positives. The dataset was then split into a training and testing set, with the testing set receiving 20% of all samples. To execute the split, a random seed was utilized and saved for replicability. Table 4.2 shows the resulting number of positives and negatives in each part of the dataset.

There was a possibility to have a larger dataset for the DL approaches which used the images as input data, as there were more than 109 images available. These extra images were however not utilized, as there was a desire

CBCT date	CBCT Label
within 2 years after PAN date	Positive
more than 2 years after PAN date	Negative
within 1 year before PAN date	Positive
more than 1 year before PAN date	Negative

Table 4.1: How the CBCT labeling was determined for patients with CBCT referral.

Part of dataset	No. Positives	No. Negatives	Positives %
Training set	20	67	23%
Testing set	6	16	27%

Table 4.2: Distribution of positives and negatives in training and testing sets.

to have directly comparable metrics to the baseline. The baseline could not utilize these extra images, as the measured features only existed for the 109 pre-orthodontic treatment PANs.

4.1.2 Combining measured features

For each image used for the implementation, there existed 10 measured features from three different operators. Subsequently, these three sets of measured features relating to the same image were combined into one set of 10 features. The tooth depth, α-angle 1, and α-angle 2 were combined by taking the mean of the three operators' measurements. As for the categorical features Overlap and Vertical, the mode, or majority vote, was utilized to obtain a single measurement for each image. These categorical features were also one-hot encoded, increasing the feature space of the baseline implementation to 25 features. Lastly, the features were standardized using the mean and standard deviation of the training set.

4.1.3 Image preprocessing

For the DL-based approaches, the PAN images were prepared to work as input to the models. The first step was converting the images from a DICOM format to tensors. Following this, the images were cropped and resized, where the size parameters were different for the feature and transfer learning approaches. For the feature learning approach, the initial cropping procedure produced a

1200×1200 image, where the crop was performed from the center. This resulted in a square image, which was then resized to 256×256. Smaller input sizes for the autoencoder were examined, but initial experimentation found that a smaller input size negatively impacted the validation loss. For the transfer learning approach, two different image input sizes were used. The first entailed cropping to a 1200×1200 image, followed by scaling to 640×640, the size inspired by the method by [41]. The second input size came from center cropping to 1000×800 and scaling to 500×400 (width = 500px, height = 400 px).

Resizing and cropping were followed by normalization and standardization of the image dataset. To perform the standardization, the mean and standard deviation of the training set were calculated. Finally, the images were expanded into three channels for the transfer learning approach. The reason is that the pre-trained ViT-Base/16 used in the implementation expects a three-channel input.

4.2 Baseline

After labeling and preprocessing the measured features, the training set described in Section 4.1.1 and Table 4.2 was used for hyperparameter tuning of the classifiers: logistic regression, SVM, RFC and MLP. The final hyperparameters were found through grid searches using Sci-kit learn GridsearchCV. For every classifier, a parameter space was determined, detailed in Appendix A. The grid searches found the best combination of hyperparameters in the parameter space using stratified 3-fold. Stratified k-fold is a cross-validation procedure, where the given training set is split into k folds, where the percentage of positives and negatives in the folds are kept. One fold is selected at a time to act as a validation set, while the rest $k - 1$ folds are used for training the model. By default, GridsearchCV utilizes accuracy as a metric to compare performance and decide the hyperparameters. In this implementation, the F1-score was primarily used for scoring performance instead of accuracy. When several combinations of hyperparameters performed equally on the validation sets, the AUPR metric could be used as a tie-breaker.

4.2.1 Oversampling dataset for baseline MLP

The Sci-kit learn MLP class does not contain a class weight hyperparameter that considers the class distribution. To investigate if the model performs

better when the class imbalance is considered, a smaller experiment was performed. Here, the class imbalance was addressed by performing random upsampling of the training set. This meant duplicating the positive cases in the training set at random until achieving an equal amount to the negative class. Then, hyperparameter tuning was performed through grid search, using the upsampled training set.

4.3 Feature learning

The feature learning implementation began with initial experimentation of different image input sizes, and types of layers in the autoencoders. Early trials found that structures similar to the convolutional autoencoder in [32] seemed promising. A validation set consisting of 10% of the initial training set was used. To narrow down the feature learning setup, two autoencoder structures were selected by considering the validation losses, as well as choosing model sizes and latent variable spaces that appeared sensible. Both models have a 256×256 input size. The first model, named CNN-32k, has five convolutional layers, a latent variable space of 32768, and roughly 3.1 million parameters. With an image input size of $256 \times 256 = 65536$, the CNN-32k halves the feature space dimension. A second model was constructed where a smaller latent variable space was desired. An extra convolutional layer was added, which caused an increase in the number of parameters. The second autoencoder is named CNN-8k, for its 8192 latent variable space. It has around 7.8 million parameters. Both implementations use ReLU activation layers throughout, except for the final activation layer in the decoders, which employ a sigmoid function. The binary cross entropy loss function was used for both autoencoders, implemented with the PyTorch `BCELoss` function. The optimization of parameters was performed using the Adam algorithm, with a learning rate of 10^{-3} and an L2 regularization penalty of 10^{-5}. The autoencoders were trained for 12 epochs, with a batch size of 3.

After completing the training of the two autoencoders, all the images in the dataset were used as input for the encoder to output the latent variables. These latent variables were used as input features for various ML models: logistic regression, SVM, RFC, and MLP. To determine each model's optimal hyperparameters, grid search was utilized, with parameter spaces described in Appendix A. The SVM was divided into three models, to have a grid search for each kernel: linear, polynomial, and radial basis function (RBF). The MLP was also divided into two separate grid searches, to individually study the performance when consisting of 1 hidden layer and 2 hidden layers.

4.4 Transfer learning

The transfer learning implementation involved fine-tuning the ”ViT-B/16” model in [26]. In the subsequent parts of this thesis, it is named ViT-Base. For fine-tuning, the last feedforward layer of the model was modified to output the number of classes. All layers except the last layer were frozen, *i.e.,* they did not update during training. This set-up resulted in 1538 trainable parameters. Two different image resolutions were tested, 640×640 and 500×400. Similarly to the feature learning implementation, the training set was sub-splitted into a validation set consisting of 10% of the initial training set. This validation set of 9 samples had 3 Positive cases and was used to study the effect of changing the hyperparameters.

Minor hyperparameter tuning was performed for the transfer learning approach, as grid search was not utilized. Instead, two components were studied: the optimization algorithm and the learning rate. For the choice of optimization algorithm, stochastic gradient descent and Adam were considered. The learning rate was initially set to 0.001 and then increased/decreased in steps based on the training and validation metrics and loss plots. The hyperparameter tuning was based on the 500×400 version of the model. Cross entropy loss was utilized as a loss function, and class weights were used, scaling the positive class with 0.3 and the negative class with 0.7. A batch size of 10 was applied.

An early stopping procedure was employed as an attempt to limit overfitting. Both models, the 500×400-version and the 640×640-version were trained for 20 epochs. Furthermore, for each epoch, the validation loss was calculated. At the end of the 20 epochs, the model parameters from the epoch with the lowest validation loss were also kept.

4.5 Collecting results

After training the models, they were tested with the testing set described in Section 4.1.1 and Table 4.2. These results are presented in Chapter 5. Note that precision is ill-defined when the classifier does not make a single positive prediction. In those cases, the precision is set to 0. Furthermore, a random classifier has an AUROC of 0.5, and an AUPR of 0.27 for the test set described by Table 4.2.

Lastly, the average metrics were computed. Here, 20 different 80%-20% train-test splits were established. All models were trained and tested 20 times,

once for each split. The evaluation metrics were recorded for all splits to calculate each metric's mean and standard deviation.

Chapter 5

Results

In this chapter, the results of the different approaches to the classification task are presented. The results are presented in terms of the metrics described in section 3.5. First, for each approach, the model metrics with the single specific training and testing dataset are presented. At the end of the chapter, in Section 5.5, the average metrics over 20 different dataset splits for all models are presented.

5.1 Baseline

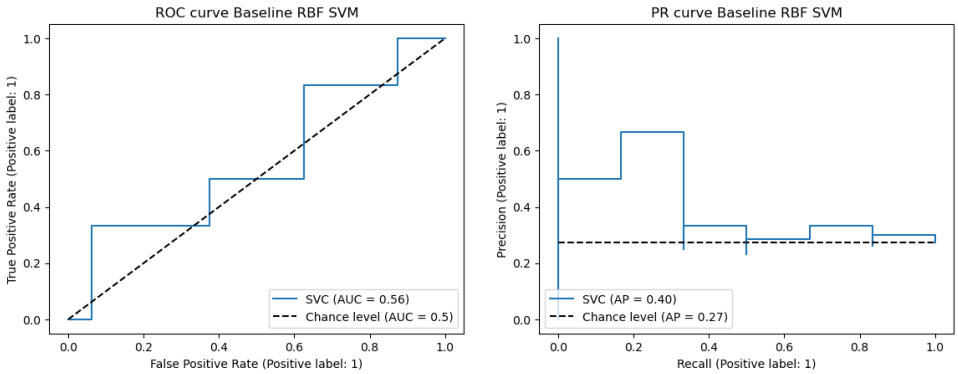
This section presents the baseline results for comparison of the different classifiers using the same training and test set. Details about the hyperparameters of each classifier are available in the Appendix, Table B.1. Table 5.1 and 5.2 present the test and training set results of the final classifier models after hyperparameter tuning. The SVM implementation is divided into three different classifiers in Table 5.1 and 5.2, one for each kernel type: linear, polynomial, and RBF kernels. Table 5.1 shows that the highest F1 scores were achieved by the MLP and the SVM with an RBF kernel. The ROC and PR curves for the MLP are shown in Figure 5.2a and Figure 5.2b, respectively. The ROC and PR curves for the SVM with the RBF kernel are shown in Figure 5.1a and Figure 5.1b, respectively. The confusion matrices for the RBF SVM and the MLP are shown in Appendix C. The calibration curves of the baseline classifiers are available in the Appendix, Figure D.1.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	**0.73**	0	0	0	0.36	0.29
Linear SVM	**0.73**	0	0	0	0.51	0.36
Poly SVM	0.68	0.40	**0.33**	0.36	0.49	0.34
RBF SVM	**0.73**	**0.50**	**0.33**	**0.40**	**0.56**	0.40
Random Forest	**0.73**	0	0	0	0.40	0.27
MLP	**0.73**	**0.50**	**0.33**	**0.40**	0.54	**0.48**

Table 5.1: Baseline classifiers and their metrics on the *test set*.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	0.79	**1.0**	0.10	0.18	0.76	0.58
Linear SVM	0.79	**1.0**	0.10	0.18	0.67	0.60
Poly SVM	**0.94**	**1.0**	**0.75**	**0.86**	0.97	**0.97**
RBF SVM	0.92	**1.0**	0.65	0.79	0.97	**0.97**
Random Forest	0.82	**1.0**	0.20	0.33	0.88	0.79
MLP	0.93	0.94	**0.75**	0.83	**0.99**	**0.97**

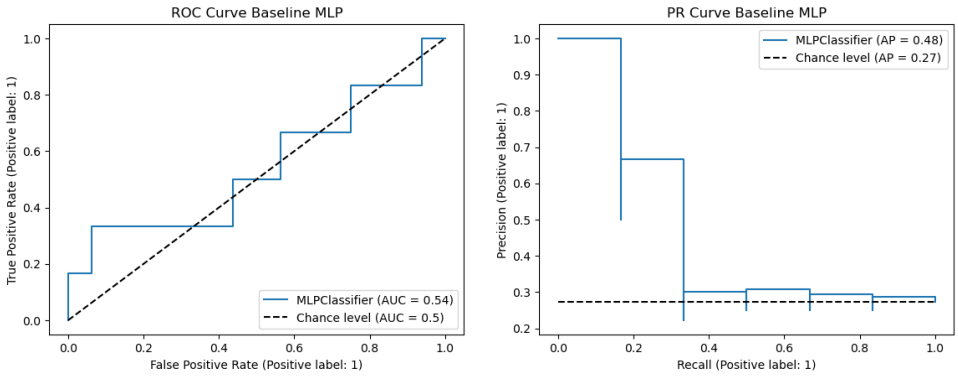
Table 5.2: Baseline classifiers and their metrics on the *training set*.



(a) ROC curve for the baseline SVM classifier with an RBF kernel.

(b) PR curve for the baseline SVM classifier with an RBF kernel.

Figure 5.1: The ROC and PR curves for the SVM classifier with an RBF kernel that yielded an F1 score of 0.40 on the test set.



(a) ROC curve for the baseline MLP classifier.

(b) PR curve for the baseline MLP classifier.

Figure 5.2: The ROC and PR curves for the MLP classifier that yielded an F1 score of 0.40 on the test set.

5.1.1 Baseline MLP with data oversampling

This section presents a comparison of the best-performing MLP trained on the oversampled dataset and the standard, non-oversampled, dataset. Hyperparameter tuning with the oversampled dataset found that setting the hidden layer size to (12,4) gave the best performance of the grid search. The best performing MLP of the standard dataset, see Section 5.1 and Table 5.1, had a hidden layer size of (11,). These two sets of MLP hyperparameters

are compared in Table 5.3, where the two models were trained with the oversampled dataset, and tested with the non-oversampled test set. Table 5.4 presents these classifiers' training set metrics.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
MLP hidden layers: (12,4)	**0.59**	**0.29**	0.33	0.31	0.34	**0.38**
MLP hidden layers: (11,)	0.50	0.27	**0.50**	**0.35**	**0.49**	0.31

Table 5.3: Test set results from the experiment with dataset oversampling of the Positive class.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
MLP hidden layers: (12,4)	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
MLP hidden layers: (11,)	0.94	0.93	0.96	0.94	0.99	0.99

Table 5.4: Training set results from the experiment with dataset oversampling of the Positive class.

5.2 Feature Learning

This section is divided into two parts, one for each AE: AE-CNN-8k and AE-CNN-32k.

5.2.1 Results from CNN-8k

The training and validation loss per epoch during training of CNN-8k is shown in Figure 5.3. The fine-tuned hyperparameters of the different classifiers taking the 8k latent variables as input are found in Table B.2. The test metrics of these classifiers are shown in Table 5.6, and the training metrics are shown in Table 5.5. Classifiers with perfect training metrics results, *i.e.,* scoring 1.0 on all metrics, are excluded from Table 5.5. In Table 5.6, it is shown that the SVM classifier with a polynomial kernel performed highest across all metrics. The

kernel of this SVM was a polynomial of degree 2, see Table B.2, and its ROC and PR curves are displayed in Figures 5.4a and 5.4b, respectively. Confusion matrix results of the SVM classifier are shown in Figures C.3a and C.3b.

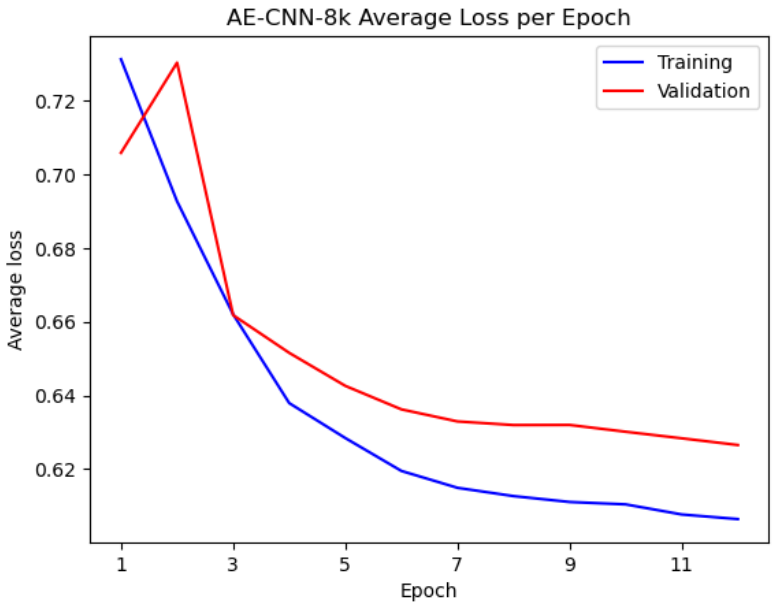


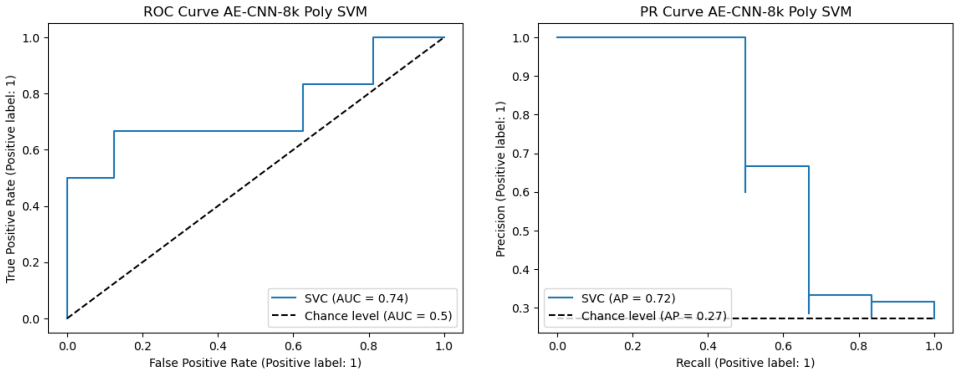
Figure 5.3: Training and validation loss for CNN-8k.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Linear SVM	0.95	**1.0**	0.80	0.89	**1.0**	**1.0**
RBF SVM	**0.97**	**1.0**	**0.85**	**0.92**	**1.0**	**1.0**
Random Forest	0.82	0.7	0.35	0.46	0.66	0.40

Table 5.5: Non-perfect *training set* results of classifiers trained on the latent variables from the CNN-8k encoder.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	0.68	0.33	**0.17**	0.22	0.61	0.47
Linear SVM	0.73	0.50	**0.17**	0.25	0.60	0.47
Poly SVM	**0.77**	**1.0**	**0.17**	**0.29**	**0.74**	**0.72**
RBF SVM	0.73	0	0	0	0.65	0.57
Random Forest	0.54	0	0	0	0.375	0.27
1-layer MLP	0.68	0	0	0	0.66	0.42
2-layer MLP	0.68	0	0	0	0.56	0.34

Table 5.6: *Test set* results of classifiers trained on the latent variables from the CNN-8k encoder.



(a) Test set ROC curve for the Poly SVM classifier.

(b) Test set PR curve for the Poly SVM classifier.

Figure 5.4: The ROC and PR curves for the SVM classifier paired with a polynomial kernel of degree 2, with an 8k latent variable input.

5.2.2 Results from CNN-32k

The training and validation losses during the training of CNN-32k are displayed in Figure 5.5. The resulting parameters of the hyperparameter

tuning of the different classifiers taking the 32k latent variables as input, are detailed in Table B.3. The test and train metrics are presented in Table 5.7 and Table 5.8, respectively. Classifiers with perfect train metrics results are excluded from Table 5.8. In Table 5.7, it is shown that the Linear SVM obtained an F1-score of 0.60 on the test set, and the logistic regression classifier an F1-score of 0.55. The ROC and PR curves of the logistic regression classifier are shown in Figure 5.6a and Figure 5.6b, respectively. The ROC and PR curves of the SVM with a linear kernel are shown in Figure 5.7a and Figure 5.7b, respectively. The classifiers' confusion matrices are found in Appendix C.

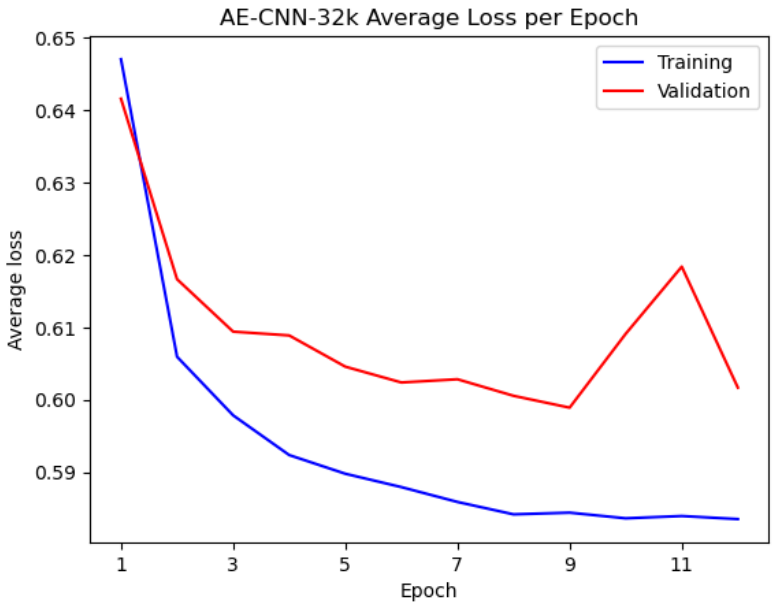


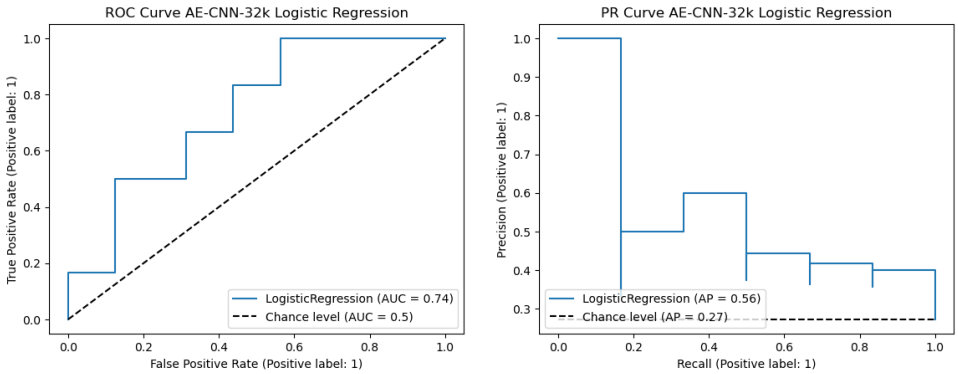
Figure 5.5: Training and validation loss for CNN-32k.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	0.77	0.60	**0.50**	0.55	0.74	0.56
Linear SVM	**0.82**	**0.75**	**0.50**	**0.60**	**0.81**	0.64
Poly SVM	0.73	0	0	0	0.74	0.61
RBF SVM	0.73	0	0	0	0.79	**0.73**
Random Forest	0.55	0	0	0	0.38	0.27
1-layer MLP	0.68	0	0	0	0.72	0.43
2-layered MLP	0.68	0	0	0	0.67	0.41

Table 5.7: *Test set* results of the classifiers trained on the latent variables from the CNN-32k encoder.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Poly SVM	**0.99**	**1.0**	**0.95**	**0.97**	**1.0**	**1.0**
Random Forest	0.97	0.63	0.75	0.68	0.81	0.53
1-layer MLP	**0.99**	**1.0**	**0.95**	**0.97**	**1.0**	**1.0**

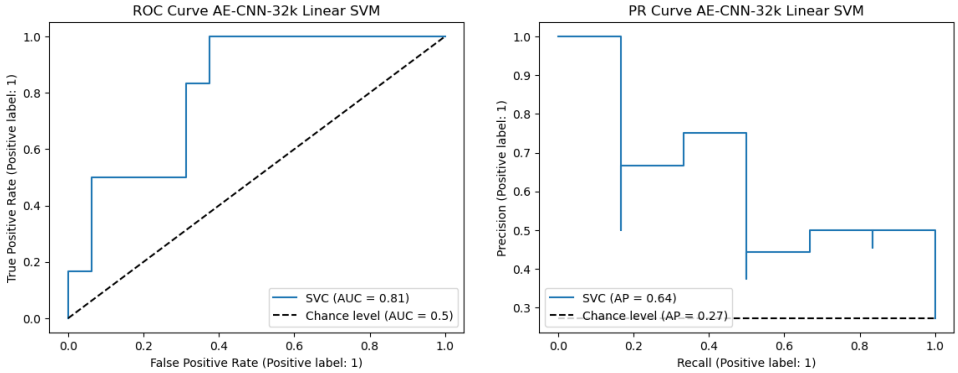
Table 5.8: Non-perfect *training set* results of classifiers trained on the latent variables from the CNN-32k encoder.



(a) Test set ROC curve for the logistic regression classifer.

(b) Test set PR curve for the logistic regression classifier.

Figure 5.6: The ROC and PR curves for the logistic regression classifier with the 32k latent variable input.



(a) Test set ROC curve.

(b) Test set PR curve.

Figure 5.7: The test set ROC and PR curves of the Linear SVM with the 32k latent variable input.

5.3 Transfer Learning

The hyperparameter tuning of the ViT-Base model resulted in the chosen hyperparameters in Table B.4. Training and validation loss per epoch during fine-tuning of the ViT-Base model with 400×500 sized input images is shown in Figure 5.8. We compare two model training schemes; one trained for 20 epochs, and the other employed the early stop procedure. The early stop procedure interrupted the training at the 19th epoch. Numerical test results for both schemes are shown in table Table 5.9 for early stop, and in Table 5.10

for fixed 20 epochs. The test set ROC and PR curves for the model trained for 20 epochs are shown in Figure 5.10a and Figure 5.10b, respectively.

The same metrics also exist for the version with 640×640 sized input images, with the training and validation losses per epoch shown in Figure 5.9. The test metrics for the model state saved from the early stopping procedure are shown in Table 5.9, where the training was interrupted at the 5th epoch. Results from the model version that trained for 20 epochs are shown in Table 5.10.

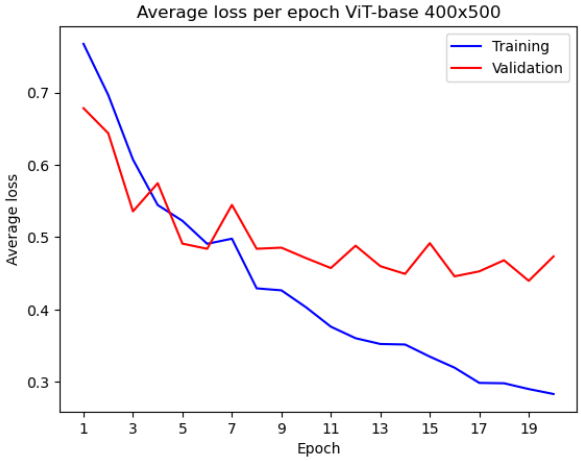


Figure 5.8: Average loss per epoch during fine-tuning of ViT-Base with 400x500 sized input images.

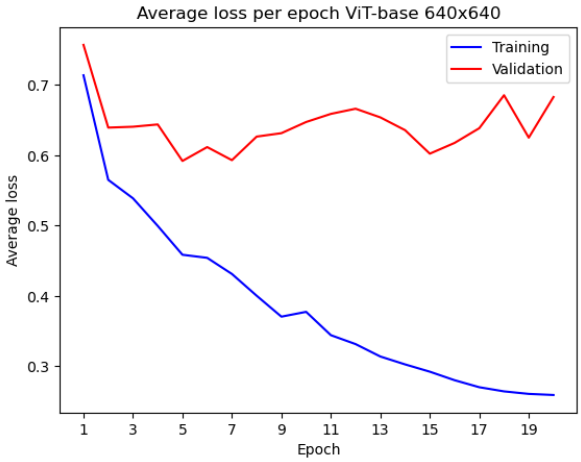


Figure 5.9: Average loss per epoch during fine-tuning of ViT-Base with 640x640 sized input images.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
ViT-Base 400x500 input w/ early stop	**0.64**	0.25	0.17	0.20	**0.60**	**0.36**
ViT-Base 640x640 input w/ early stop	0.59	**0.29**	**0.33**	**0.31**	0.45	0.29

Table 5.9: *Test set* results after fine-tuning ViT-Base with different image input sizes, employing early stopping of training.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
ViT-Base 400x500 input 20 epochs	**0.68**	**0.33**	**0.17**	**0.22**	**0.61**	**0.38**
ViT-Base 640x640 input 20 epochs	0.59	0	0	0	0.45	0.29

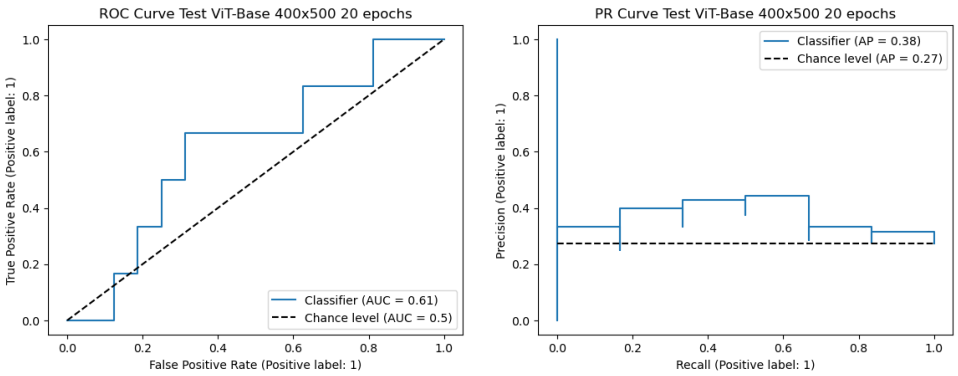
Table 5.10: *Test set* results after fine-tuning ViT-Base for 20 epochs with different image input sizes.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
ViT-Base 400x500 input w/ early stop	**0.95**	**0.88**	**0.88**	**0.88**	**0.99**	**0.97**
ViT-Base 640x640 input w/ early stop	0.83	0.59	0.76	0.67	0.91	0.80

Table 5.11: *Training set* results after fine-tuning ViT-Base with different image input sizes, employing early stopping of training.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
ViT-Base 400x500 input 20 epochs	0.95	**1.0**	0.76	0.87	0.99	0.97
ViT-Base 640x640 input 20 epochs	**0.96**	**1.0**	**0.83**	**0.90**	**1.0**	**0.99**

Table 5.12: *Training set* results after fine-tuning ViT-Base with different image input sizes, trained for 20 epochs.



(a) Test set ROC curve.

(b) Test set PR curve.

Figure 5.10: The test set ROC and PR curves of the fine-tuned ViT-Base with 400x500 sized input images, trained for 20 epochs.

5.4 Test Metrics Summary

The metrics of the best-performing classifiers from each approach are summarized in Table 5.13. The table allows for a straightforward comparison of the achieved metrics.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Baseline MLP	0.73	0.50	0.33	0.40	0.54	0.48
CNN-8k Poly SVM	0.77	**1.0**	0.17	0.29	0.74	**0.72**
CNN-32k Linear SVM	**0.82**	0.75	**0.50**	**0.60**	**0.81**	0.64
ViT-Base 640x640 input w/ early stop	0.59	0.29	0.33	0.31	0.45	0.29

Table 5.13: Overview of the highest *test set* metrics from each approach.

5.5 Average Metrics

This section presents the models' average performance on 20 different training and testing splits of the dataset. The models have the same hyperparameters found through the grid searches, see Appendix B for details. The metrics in this section are presented as $mean \pm std$.

The average metrics achieved by the baseline classifiers on the test set are shown in Table 5.14. Table 5.15 presents the corresponding training set results. The average test set results from the classifiers taking the latent variables from the CNN-8k and CNN-32k encoders are shown in Table 5.16 and Table 5.17, respectively. Table 5.18 displays the average train set metrics of the fine-tuned ViT-Base classifiers.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	**0.75**± 0.058	0	0	0	0.44 ± 0.14	0.32 ± 0.10
Linear SVM	**0.75**± 0.058	0	0	0	0.43 ± 0.16	0.35 ± 0.14
Poly SVM	0.71± 0.073	0.36± 0.28	**0.25**± 0.19	0.28± 0.20	**0.55** ± 0.14	**0.41** ± 0.17
RBF SVM	0.74± 0.071	**0.48**± 0.40	0.18± 0.16	0.25± 0.20	0.51 ± 0.14	0.40 ± 0.16
Random Forest	0.72± 0.073	0.13± 0.31	0.040± 0.083	0.058± 0.12	0.49 ± 0.17	0.35 ± 0.13
MLP	0.72± 0.091	0.42± 0.35	0.24± 0.17	**0.29**± 0.21	0.54 ± 0.15	**0.41** ± 0.16

Table 5.14: Baseline classifiers and their mean *test* metrics on 20 different train-test (80% - 20%) splits.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	0.78± 0.013	0.95± 0.22	0.075± 0.028	0.14± 0.050	0.73 ± 0.038	0.58 ± 0.056
Linear SVM	0.79± 0.018	0.95± 0.22	0.086± 0.049	0.15± 0.078	0.65 ± 0.056	0.54 ± 0.088
Poly SVM	**0.94**± 0.013	**1.0** ± 0	**0.73**± 0.056	**0.85**± 0.037	**0.98** ± 0.018	**0.98** ± 0.016
RBF SVM	0.88± 0.020	**1.0** ± 0	0.48± 0.087	0.65± 0.078	0.94 ± 0.029	0.94 ± 0.027
Random Forest	0.83± 0.022	0.95± 0.10	0.28± 0.086	0.42± 0.11	0.88 ± 0.032	0.76 ± 0.053
MLP	0.92± 0.029	0.94± 0.033	0.70± 0.13	0.80± 0.094	**0.98** ± 0.015	0.94 ± 0.036

Table 5.15: Baseline classifiers and their mean *training* metrics on 20 different train-test splits.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	0.65± 0.096	**0.30**± 0.18	**0.28**± 0.20	**0.27**± 0.16	0.55 ± 0.14	0.39 ± 0.12
Linear SVM	0.67± 0.11	0.23± 0.25	0.20± 0.22	0.19± 0.18	0.55 ± 0.14	0.38 ± 0.12
Poly SVM	0.67± 0.063	0.24± 0.22	0.22± 0.20	0.22± 0.21	**0.56** ± 0.13	0.39 ± 0.14
RBF SVM	**0.71**± 0.085	0.26± 0.36	0.14± 0.16	0.17± 0.19	0.55 ± 0.16	**0.42** ± 0.17
Random Forest	0.70± 0.088	0.18± 0.32	0.056± 0.088	0.084± 0.13	0.49 ± 0.073	0.28 ± 0.075
1-layer MLP	0.68± 0.089	0.25± 0.27	0.18± 0.18	0.20± 0.20	0.53 ± 0.14	0.38 ± 0.15
2-layered MLP	0.66± 0.096	0.24± 0.23	0.18± 0.16	0.19± 0.17	0.53 ± 0.13	0.37 ± 0.13

Table 5.16: Autoencoder CNN-8k classifiers and their mean *test* metrics on 20 different train-test (80% - 20%) splits.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
Logistic Regression	0.69± 0.099	**0.35**± 0.25	0.26± 0.19	**0.29**± 0.19	0.62 ± 0.15	0.47 ± 0.15
Linear SVM	0.70± 0.10	**0.35**± 0.27	**0.27**± 0.21	**0.29**± 0.21	**0.63** ± 0.16	**0.48** ± 0.17
Poly SVM	0.73± 0.074	0.19± 0.33	0.09± 0.15	0.11± 0.17	0.62 ± 0.15	0.46 ± 0.15
RBF SVM	**0.74**± 0.067	0.27± 0.37	0.092± 0.12	0.13± 0.16	0.61 ± 0.13	0.45 ± 0.13
Random Forest	0.61± 0.12	0.27± 0.18	0.28± 0.19	0.26± 0.17	0.50 ± 0.12	0.28 ± 0.075
1-layer MLP	0.69± 0.087	0.22± 0.22	0.20± 0.23	0.19± 0.18	0.57 ± 0.15	0.42 ± 0.14
2-layered MLP	0.73± 0.069	0.26± 0.30	0.16± 0.21	0.18± 0.21	**0.63** ± 0.12	0.43 ± 0.14

Table 5.17: Autoencoder CNN-32k classifiers and their mean *test* metrics on 20 different train-test (80% - 20%) splits.

Classifier	Acc.	Prec.	Recall	F1	AUROC	AUPR
ViT-Base 400x500 20 epochs	**0.68**± 0.079	0.29± 0.23	0.21± 0.13	0.23± 0.14	**0.61** ± 0.10	**0.44** ± 0.11
ViT-Base 400x500 early stop	0.66± 0.11	**0.33**± 0.24	**0.28**± 0.20	**0.27**± 0.16	0.51 ± 0.13	0.37 ± 0.13
ViT-Base 640x640 20 epochs	0.65± 0.086	0.25± 0.27	0.22± 0.21	0.22± 0.22	0.59 ± 0.14	0.39 ± 0.15
ViT-Base 640x640 early stop	0.67± 0.081	0.22± 0.27	0.17± 0.21	0.17± 0.18	0.54 ± 0.16	0.37 ± 0.17

Table 5.18: Fine-tuned ViT-Base classifiers and their mean *test* metrics on 20 different train-test splits.

Chapter 6

Discussion

The first section of this chapter presents an analysis of the results and metrics in Chapter 5. The following section discusses the shortcomings of the implementation and possible improvements.

6.1 Analysis of results

This section will first individually review each approach to the classification task and its metrics. Towards the end of this section, the results of the different approaches will be compared with each other, in order to discuss what methods are suitable for this problem. Any tendencies found across all approaches will also be discussed here.

6.1.1 Baseline

From Table 5.1, we see three baseline classifiers that have an F1-score of 0 on the test set: the logistic regression classifier, Linear SVM, and the Random Forest classifier. These classifiers also have an AUROC score below or close to 0.5, meaning that they behave as a random classifier or worse. These classifiers' accuracy score of 73% comes from only predicting samples to belong to the negative class, as using Equation 3.1,

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{Total number of predictions}} = \frac{16 + 0}{22} = 0.727.$$

We also see that the training set F1-score of these three classifiers in Table 5.2 are relatively low. This may indicate that these models are underfitting, *i.e.,* not finding the underlying patterns, perhaps due to low model complexity.

The highest achieved F1 score of the baseline classifiers was 0.4 and was obtained both by the SVM with a 4th-degree polynomial kernel, and the 1-layered MLP with 11 nodes. The Poly SVM obtained a slightly higher AUROC score of the two, with both scoring higher than 0.5. However, in Figure 5.1a and Figure 5.2a we see that both classifiers perform worse than random at certain points. Both obtain a higher AUPR than random, with the MLP yielding the highest baseline AUPR score of 0.48. From Figure 5.1b and Figure 5.2b we see a trade-off between precision and recall, where a lower recall comes with a higher precision, and vice-versa. While both classifiers perform around equal for higher recall scores, the MLP obtains higher precision scores at lower recall scores. Finally, the average metrics presented in Table 5.14 show that the baseline F1 score winners are consistent with what was found with the single train-test split. However, the mean of the F1 scores was noticeably lower.

Observing the results in Section 5.1.1, we see that the performance did not improve when training MLPs with the oversampled dataset. The model with 11 nodes performed slightly worse when trained with the oversampled training set compared to the standard training set, see Table 5.1 and Table 5.3. The MLP with hidden layer size (12,4), with hyperparameters found through grid search with the oversampled dataset, performed more poorly. Deducing from Table 5.3 and Table 5.4, it is probable that the models are overfitting on the training data. The classifiers perform perfectly on the training data, while having poor metrics on the test data.

6.1.2 Feature Learning

For the CNN-8k classifier metrics in Table 5.6, an overall poor performance on the F1-score is observed. The highest achieved was an F1 score of 0.29, by the SVM with a polynomial kernel of degree 2. This classifier also performed the highest in terms of AUROC and AUPR, when compared to the other CNN-8k classifiers. Its ROC and PR curves in Figure 5.4a and Figure 5.4b show that it performs better than random.

As for the CNN-32k classifier metrics in Table 5.7, the SVM with a linear kernel had overall the highest scores on the test set, obtaining an F1 score of 0.6. The logistic regression classifier also performed relatively well, with an F1 score of 0.55. The other classifiers failed to predict a single TP.

Observing the classification results from both autoencoder implementations, we notice that simpler models outperformed more complex ones. This could be attributed to the large dimensionality of the input data. The CNN-

8k encoder produces a latent space of 8192 dimensions, while the CNN-32k encoder output comprises 32768 dimensions. Consequently, the input to the classification models encompasses a significant feature space. The more complex models, capable of capturing intricate patterns, appear to overfit on this high-dimensional data, emphasizing noise rather than underlying patterns. Additionally, the higher dimensionality contributes to data sparsity, causing data points to become more spread out. In conjunction with a small training set, this exacerbates the risk of poor generalization for the complex models.

6.1.3 Transfer Learning

The loss plots from fine-tuning the models are shown in Figure 5.8 and Figure 5.9. Across both figures, the validation loss is noticeably noisy, suggesting potential issues with the representativeness or size of the validation dataset. Furthermore, there exist notable gaps between the validation and training loss curves, indicating instances of overfitting within the models. This overfitting tendency could stem from the complexity of the models coupled with the limited number of training samples.

Within Figure 5.8, a downward trend is observable in the validation loss, whereas the model with 640×640 input dimensions depicted in Figure 5.9 fails to converge. This discrepancy might be attributed to the introduction of more noise due to the larger input size of the 640×640 model. It may also be due to a mismatch of hyperparameters, as only the 400×500 model was considered when deciding the optimization algorithm and learning rate.

We can compare the classification metrics resulting from the two different input dimensions using Table 5.9 and Table 5.10. The model with 640×640 input dimensions achieved the highest F1 score after being trained for 20 epochs. However, when observing at the AUROC score, the model with 400×500 input dimensions performs better, both with early stopping and after 20 epochs. Additionally, the model with 640×640 input dimensions appears to be overfitting, as evidenced by the metrics on the training set in Table 5.12. This observation aligns with the analysis of the loss plot in Figure 5.9.

The model trained with 400×500 input dimensions for 20 epochs exhibited the best performance in terms of the test metrics. Its ROC curve is depicted in Figure 5.10a and the PR curve in Figure 5.10b. In the ROC curve, we observe instances where its performance falls below random. Although the PR curve remains above the chance level line, it maintains proximity to it.

6.1.4 Comparison of approaches

In this section, we make a comparative analysis of the different approaches to the classification task. Starting with test metrics performance, we find that the simpler CNN-32k classifiers, especially the Linear SVM, performed the best. They achieved the highest F1 score among all methods and also had high AUROC and AUPR scores. Inspecting all of the F1 scores, this method was the only one that outperformed the baseline metrics.

Concerning the baseline and autoencoder results, the RFC consistently underperformed, suggesting it might not be the best choice for this particular task. Optimizing the RFC hyperparameters through grid searches for both CNN-8k and CNN-32k configurations led to a surprising outcome where the optimal setting ended up being a single-tree random forest, see Table B.2 and Table B.3. This finding supports the idea that simpler models tend to generalize better when dealing with large feature spaces. Interestingly, despite the RFC's lackluster performance in a single test set, as noted in Table 5.7, it managed to achieve an average F1 score of 0.26 across 20 different test sets, as shown in Table 5.17.

Comparing the transfer learning approach to the baseline implementation, it is noted that the baseline F1 score was not beaten. With the background that feature learning methods are often employed for image classification tasks with small datasets, more was expected of the method. This outcome highlights the need for further research to explore the applicability and optimization of transfer learning techniques in similar contexts.

The metrics of each approach can also be compared to the performance of a so-called dummy classifier. For instance, with respect to the F1 score, the best dummy classifier would only predict positives. This results in a recall of 1 and a precision equal to the marginal probability of the positive class in the test set, a. For this test set, as explained in Table 4.2, $a = \frac{6}{22}$. Thus, the dummy classifier would obtain an F1 score of,

$$\mathrm{F1}_{\text{dummy}} = 2\frac{a \cdot 1}{a+1}\bigg|_{a=\frac{3}{11}} = \frac{3}{7} \approx 0.429.$$

Only two classifiers beat this F1 score: the CNN-32k logistic regression classifier and the CNN-32k Linear SVM. However, from the average results, it is observed that no classifier gets a mean F1 score above 0.429.

Finally, consistencies are found when analyzing the average metrics in Section 5.5. Firstly, the mean metrics winners are mostly aligned with what was found for the single test set. However, the average AUROC score is close

to 0.50 for many implementations, and the highest mean F1 is 0.29. Also, the standard deviation is considerably high for many metrics. This may indicate a difficulty for the classifiers to learn the patterns in the dataset. However, it should be noted the chosen hyperparameters for that experiment are fine-tuned to the single train-test split. The high standard deviation also indicates that the choice of test and training set has a impact on the final results, causing variations in the metrics.

6.2 Method shortcomings and improvements

This section reflects on what could have been done differently in the project implementation. Project limitations that cannot be controlled are not regarded in this section.

6.2.1 Implementation choices

In evaluating the chosen methodologies and implementation details, particularly the approach to hyperparameter tuning, several areas for improvement emerge. Specifically, the use of grid search for tuning both the baseline model and feature learning processes introduces constraints that could affect the overall model optimization process. One disadvantage with grid search is the chosen hyperparameter space being fixed. This rigidity means that the grid search may miss the optimal set of hyperparameters if they lie between the grid points. Also, all hyperparameters are not equally relevant, but the procedure puts equal computational effort for them all. Furthermore, relevant hyperparameters could have been missed with the choice of hyperparameter space, as not all hyperparameters of a model were considered. An alternative strategy, such as random search, offers several advantages. Notably, it provides a more flexible and efficient exploration of the hyperparameter space, often achieving faster convergence and improved effectiveness by focusing on a broader and more randomly selected set of hyperparameters [1].

Furthering the discussion on hyperparameter tuning, improvements could have also been beneficial for the transfer learning implementation. The approach to tuning the ViT-Base models was both minimal and manual, focusing largely on adjusting the learning rate in increments and evaluating the model based on loss plots alongside training and validation metrics. This method has notable drawbacks. Relying on manual selection from noisy loss plots can lead to suboptimal choices, potentially introducing selection bias. Moreover, the validation set comprised only 9 samples,

which raises concerns about the representativeness of the validation metrics for overall model performance. The process could be improved by using automatic hyperparameter optimization algorithms. Additionally, considering regularization and including it in hyperparameter tuning could be beneficial. In [26], regularization is not used during fine-tuning, but it could be worth investigating.

Lastly, relying on average metrics for model evaluation poses a potential risk, particularly due to the methodological approach of repeatedly splitting the dataset into numerous training and testing sets for model training and evaluation. This repeated division and usage can be seen as a data leakage. Essentially, the test data from one split might end up in the training set of another, which, although not directly influencing the training within the same cycle, complicates the integrity of the evaluation. This is particularly relevant since the final model settings were determined using a grid search that depended on a specific split. Despite this, the use of average metrics is maintained to explore how different training sets affect the model and to assess the performance of the chosen hyperparameters across various scenarios.

6.2.2 Project scope

With a broader perspective on the project scope, it is essential to evaluate the overall strategies employed to address the research question. Specifically, the project aimed to conduct a broad comparison of various ML methods. However, this ambition faced challenges due to time constraints, limiting the depth of analysis for each technique. Consequently, it became challenging to draw conclusions about the suitability of the different techniques for the task. For example, the results from the transfer learning classification were less than optimal. Many reasons could be behind this, such as the ViT architecture or the choice of pre-trained model to be fine-tuned not being appropriate for this task, or because of the limited hyperparameter tuning. It might have been more reasonable for the project to have a greater focus on transfer learning techniques, which are frequently discussed in the literature. Such a focus would have allowed a more detailed comparison of different pre-trained models. Regarding the use of the ViT architecture, while its application in dental image classification is novel and intriguing, opting for more established architectures such as CNNs might have offered a solid foundation for future explorations into alternative architectures, including ViTs.

Another aspect worth discussing is the selection of the classification task. In the early phases of the project, there was an interest in finding cases of

complicated canine impaction. With this as background and considering the available data, the classification task of CBCT Positive or Negative was chosen. However, the suitability of this task can be questioned due to the limited number of data points. A CBCT scan may be conducted for various reasons, and with a small dataset, it might be challenging for the model to accurately learn to the underlying patterns. An alternative task, such as classifying the position of the canine, could have been considered from the start. Even so, the practical value of this alternative classification might still be debatable.

Chapter 7

Conclusions

This chapter brings together the main points of the project, and highlights the most important findings. The limitations of the project will be stated, and suggestions for future work will be provided.

7.1 Conclusions

The goal of the project was to compare different ML techniques for a classification task concerning the dental condition of impacted canines. By first establishing baseline models utilizing measured features derived from PANs, feature learning and transfer learning approaches could be compared to these baseline metrics. The original problem asks whether these more intricate DL techniques can achieve better results than the baseline. From the results in Chapter 5 and the discussion in Chapter 6, it is found that one of the feature learning methods obtained better results than the baseline on the testing set. This approach entailed training a CNN autoencoder taking 256×256 input PAN images and with an encoder output of 32 thousand latent variables. Using these 32k latent variables together with a Linear SVM gave the best classification results on the test set.

Overall, the performance of the different models was of varying quality, with most lacking predictive power. Several of the classifiers show signs of overfitting. It is however not possible to deem a specific approach as inappropriate for the task. There can be several underlying factors behind the performance, such as non-optimal hyperparameters, or lack of training data paired with a complicated classification task. Therefore, the techniques should be further investigated.

7.2 Limitations

The primary limitation of this project was the small size of the dataset, which consisted of only 109 samples. For the baseline, the dataset included 10 measured features, expanded to 25 features through one-hot encoding. While the ratio of features to samples was manageable for this setup, 109 samples were found insufficient for DL techniques which utilized images. The small sample size is particularly challenging for more complex models, as they are prone to overfitting by adapting too closely to the noise within the training data.

Another limiting factor of the project was the constrained timeline. With more time it would be possible for an in-depth investigation of some approaches. More comprehensive testing could yield additional results, offering greater insights into which models are most suitable for the classification task.

7.3 Future work

In the following sections, ideas for future work are suggested.

7.3.1 Larger dataset

If additional ethically approved data becomes available, it would be beneficial to further explore the performance of different ML models for the task at hand. An increased dataset would enhance the reliability and generalizability of the findings.

In this project, limited experimentation was carried out with random data upsampling. This was examined only for the baseline MLP, and this preliminary testing found that it intensified overfitting. Future work could extend this approach to the image dataset utilized in the DL approaches. Employing techniques for achieving a balanced training set, such as the random undersampling method referenced in [9], might be similarly beneficial.

Another method for acquiring a larger dataset is to utilize the extra 50 PANs in the dataset that were not used. These were not utilized in this implementation as they did not have corresponding measured features, thus the baseline could still only use the 109 data points. There was a desire to have the DL approaches be directly comparable, with the same training and test sets as the baseline. It would be interesting to study whether the 50

extra PANs can help with the overfitting tendencies that some models had. However, there's uncertainty about the usefulness of these images, as they have noticeable differences from the initial 109 samples. Some, if not most, of these 50 PANs depict a state during treatment, where patients often wear braces. This may introduce more noise to the dataset.

7.3.2 Investigating other pre-trained models

Since fine-tuning of pre-trained models is emerging in the area of dental image analysis, it would be worthwhile to further explore transfer learning approaches. This thesis focused on a single ViT model, and extending the research to assess the broader applicability of ViTs could be insightful. Additionally, studying the use of various CNN-based pre-trained models could also be beneficial. This would enable a discussion of what architecture is appropriate for the CBCT classification task.

7.3.3 Feature importance

In ML projects, feature selection and preprocessing play crucial roles in the implementation process. Overall in this thesis, limited work was done with regard to feature selection. For future work, it would be beneficial to identify which features are most important for the models. Even among the 10 measured features for the baseline, some features showed a correlation with each other. This may also be valuable for impacted canines research, as an analysis may conclude that some features are more important for the classification task than others.

7.3.4 Model interpretability

Lastly, model interpretability is a critical aspect of medical ML applications. Users need to understand why a model makes specific predictions. Exploring methods to enhance the interpretability of ML models in healthcare could be a valuable direction for future research. This would not only improve trust in these systems but also provide crucial insights into their decision-making processes.

7.4 Reflections

In conclusion, the thesis has studied the application of various machine learning methods for predicting the need of CBCT for patients with impacted canines. Some insights about applicable methods have been provided to lay the ground for future research. A system able to assist practitioners with case complexity evaluation would be helpful for treatment planning. Ultimately, it could positively contribute to the treatment outcome and well-being of the patient.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, ser. Adaptive computation and machine learning. Cambridge, MA: MIT Press, 2016. ISBN 9780262035613 [Pages 1, 10, 11, 26, and 53.]

[2] J. Singaram, S. S. Iyengar, and A. M. Madni, *Introduction*. Cham: Springer Nature Switzerland, 2024, pp. 1–10. ISBN 978-3-031-39244-3. [Online]. Available: https://doi.org/10.1007/978-3-031-39244-3_1 [Page 1.]

[3] N. McLaren. (2023) Canine tooth. [Online]. Available: https://www.kenhub.com/en/library/anatomy/canine-tooth [Page 2.]

[4] R. Manne, C. Gandikota, S. R. Juvvadi, H. R. M. Rama, and S. Anche, "Impacted canines: Etiology, diagnosis, and orthodontic management," *Journal of pharmacy & bioallied science*, vol. 4, no. Suppl 2, pp. S234–S238, 2012. [Pages 2, 7, 8, and 9.]

[5] M. Lövgren, O. Dahl, P. Uribe, M. Ransjö, and A. Westerlund, "Prevalence of impacted maxillary canines-an epidemiological study in a region with systematically implemented interceptive treatment," *European journal of orthodontics*, vol. 41, no. 5, pp. 454–459, 2019. [Page 2.]

[6] G. Varghese, "Management of impacted canines," in *Oral and Maxillofacial Surgery for the Clinician*, 2021, pp. 329–347. ISBN 9789811513466 [Pages 2 and 7.]

[7] I. Różyło-Kalinowska, "Panoramic radiography in dentistry," *Clinical dentistry reviewed*, vol. 5, no. 1, 2021. [Pages 2 and 8.]

[8] R. Ren, H. Luo, C. Su, Y. Yao, and W. Liao, "Machine learning in dental, oral and craniofacial imaging: A review of recent progress," *PeerJ (San Francisco, CA)*, vol. 9, pp. e11 451–e11 451, 2021. [Page 2.]

[9] M. Aljabri, S. S. Aljameel, N. Min-Allah, J. Alhuthayfi, L. Alghamdi, N. Alduhailan, R. Alfehaid, R. Alqarawi, M. Alhareky, S. Y. Shahin, and W. Al Turki, "Canine impaction classification from panoramic dental radiographic images using deep learning models," *Informatics in medicine unlocked*, vol. 30, pp. 100 918–, 2022. [Pages 2, 13, 14, and 58.]

[10] S. Minhas, T.-H. Wu, D.-G. Kim, S. Chen, Y.-C. Wu, and C.-C. Ko, "Artificial intelligence for 3d reconstruction from 2d panoramic x-rays to assess maxillary impacted canines," *Diagnostics (Basel)*, vol. 14, no. 2, pp. 196–, 2024. [Page 2.]

[11] A. Swaity, B. M. Elgarba, N. Morgan, S. Ali, S. Shujaat, E. Borsci, I. Chilvarquer, and R. Jacobs, "Deep learning driven segmentation of maxillary impacted canine on cone beam computed tomography images," *Scientific reports*, vol. 14, no. 1, pp. 369–369, 2024. [Page 2.]

[12] Y. Brorsson and J. Naoumova, "Delayed diagnosis of displaced and impacted canines - a prospective longitudinal study," *Acta odontologica Scandinavica*, vol. 78, no. 3, pp. 165–172, 2020. [Pages 7 and 8.]

[13] M. Raes, M. Cadenas De Llano-Pérula, A. Alqerban, A. Laenen, A. Verdonck, and G. Willems, "Prediction of maxillary canine impaction based on panoramic radiographs," *Clinical and experimental dental research*, vol. 6, no. 1, pp. 44–50, 2020. [Pages 7 and 14.]

[14] K. Grisar, J. Fransen, M. Smeets, T. Hoppenreijs, H. Ghaeminia, C. Politis, and R. Jacobs, "Surgically assisted orthodontic alignment of impacted maxillary canines: A retrospective analysis of functional and esthetic outcomes and risk factors for failure," *American journal of orthodontics and dentofacial orthopedics*, vol. 159, no. 6, pp. e461–e471, 2021. [Page 7.]

[15] A. Alqerban, R. Jacobs, P. Lambrechts, G. Loozen, and G. Willems, "Root resorption of the maxillary lateral incisor caused by impacted canine: a literature review," *Clinical oral investigations*, vol. 13, no. 3, pp. 247–255, 2009. [Pages 7 and 8.]

[16] S. Patel and N. Saberi, "The ins and outs of root resorption," *British dental journal*, vol. 224, no. 9, pp. 691–699, 2018. [Page 7.]

[17] A. Alqerban, R. Jacobs, S. Fieuws, and G. Willems, "Comparison of two cone beam computed tomographic systems versus panoramic imaging

for localization of impacted maxillary canines and detection of root resorption," *European journal of orthodontics*, vol. 33, no. 1, pp. 93–102, 2011. [Pages 8 and 9.]

[18] Y. H. Jung, H. Liang, B. W. Benson, D. J. Flint, and B. H. Cho, "The assessment of impacted maxillary canine position with panoramic radiography and cone beam ct," *Dento-maxillo-facial radiology*, vol. 41, no. 5, pp. 356–360, 2012. [Page 8.]

[19] Z. S. T. Jebeli, M. Rafiei, and A. Torkzadeh, "Accuracy of panoramic radiography in assessing the labio-palatal position of maxillary impacted canines and root resorption of the adjacent tooth," *International journal of scientific research in dental and medical sciences (Online)*, vol. 2, no. 4, pp. 121–125, 2020. [Page 8.]

[20] M. Kumar, M. Shanavas, A. Sidappa, and M. Kiran, "Cone beam computed tomography - know its secrets," *Journal of international oral health*, vol. 7, no. 2, pp. 64–68, 2015. [Page 8.]

[21] J. K. Mah and S. Alexandroni, "Cone-beam computed tomography in the management of impacted canines," *Seminars in orthodontics*, vol. 16, no. 3, pp. 199–204, 2010. [Page 9.]

[22] A. Alqerban, R. Jacobs, S. Fieuws, and G. Willems, "Predictors of root resorption associated with maxillary canine impaction in panoramic images," *European journal of orthodontics*, vol. 38, no. 3, pp. 292–299, 2016. [Pages 9 and 14.]

[23] K. P. Murphy, "Introduction," in *Machine Learning: A probabilistic perspective*. Cambridge, MA, USA: MIT Press, 2012. ISBN 9780262018029 [Pages 9 and 10.]

[24] M. de Bruijne, "Machine learning approaches in medical image analysis: From detection to diagnosis," *Medical image analysis*, vol. 33, pp. 94–97, 2016. [Pages 9 and 10.]

[25] F. G. Bersimis and I. Varlamis, "Chapter 2 - use of health-related indices and classification methods in medical data," in *Classification Techniques for Medical Image Analysis and Computer Aided Diagnosis*, ser. Advances in ubiquitous sensing applications for healthcare, N. Dey, Ed. Academic Press, 2019, pp. 31–66. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128180044000029 [Page 10.]

[26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2020. [Pages 11, 20, 31, and 54.]

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Page 11.]

[28] J. Li, J. Chen, Y. Tang, C. Wang, B. A. Landman, and S. K. Zhou, "Transforming medical imaging with transformers? a comparative review of key properties, current progresses, and future perspectives," *Medical image analysis*, vol. 85, pp. 102 762–102 762, 2023. [Pages 11 and 12.]

[29] H. Zhao, Z. Lai, H. Leung, and X. Zhang, *A Gentle Introduction to Feature Learning*. Cham: Springer International Publishing, 2020, pp. 1–12. ISBN 978-3-030-40794-0. [Online]. Available: https://doi.org/10.1007/978-3-030-40794-0_1 [Page 12.]

[30] ——, *Neural-Network-Based Feature Learning: Auto-Encoder*. Cham: Springer International Publishing, 2020, pp. 195–217. ISBN 978-3-030-40794-0. [Online]. Available: https://doi.org/10.1007/978-3-030-40794-0_10 [Page 12.]

[31] H. E. Kim, A. Cosa-Linan, N. Santhanam, M. Jannesari, M. E. Maros, and T. Ganslandt, "Transfer learning for medical image classification: a literature review," *BMC medical imaging*, vol. 22, no. 1, pp. 69–69, 2022. [Page 13.]

[32] S. Joo, W. Jung, and S. E. Oh, "Variational autoencoder-based estimation of chronological age and changes in morphological features of teeth," *Scientific reports*, vol. 13, no. 1, pp. 704–704, 2023. [Pages 13 and 30.]

[33] A. Alqerban, A.-S. Storms, M. Voet, S. Fieuws, and G. Willems, "Early prediction of maxillary canine impaction," *Dento-maxillo-facial radiology*, vol. 45, no. 3, pp. 20 150 232–20 150 232, 2016. [Page 14.]

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of*

Machine Learning Research, vol. 12, pp. 2825–2830, 2011. [Pages 21, 67, 69, 70, and 85.]

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf [Page 21.]

[36] R. Wightman, "Pytorch image models," https://github.com/rwightman/pytorch-image-models, 2019. [Page 21.]

[37] D. Mason, "pydicom," https://https://github.com/pydicom/pydicom, 2020. [Page 21.]

[38] A. Kulkarni, D. Chong, and F. A. Batarseh, "Foundations of data imbalance and solutions for a data democracy," in *Data Democracy: At the Nexus of Artificial Intelligence, Software Development, and Knowledge Engineering*, 2020, pp. 83–106. ISBN 0128183667 [Pages 23, 24, and 25.]

[39] D. M. W. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," *arXiv (Cornell University)*, 2020. [Page 24.]

[40] J. Cook and V. Ramadas, "When to consult precision-recall curves," *The Stata journal*, vol. 20, no. 1, pp. 131–148, 2020. [Page 25.]

[41] M. E. Celik, "Deep learning based detection tool for impacted mandibular third molar teeth," *Diagnostics (Basel)*, vol. 12, no. 4, pp. 942–, 2022. [Page 29.]

Appendix A

Grid search parameter spaces

This appendix provides the parameter spaces used for each grid search. The baseline and feature learning implementations had the same grid search parameter spaces for the logistic regression, SVM classifiers.

A.1 Logistic Regression

Table A.1 shows the logistic regression hyperparameter values explored by the grid searches. According to the Scikit-learn [34] documentation, the parameter 'C' is a regularization coefficient, where larger values correspond to less regularization. The 'penalty' parameter determines what type of regularization term is used. In this implementation of logistic regression, only L2 regularization, or Ridge Regression, is used. The 'class_weight' parameter is used to weigh the loss of a sample depending on what class the sample belongs to. Lastly, the 'solver' parameter decides what optimization algorithm to use.

Hyperparameter	**Values**
'C'	0.1, 0.5, 1, 5, 10, 50, 100, 1000
'class_weight'	{0: 0.7, 1: 0.3}, {0: 0.75, 1: 0.25}, {0: 0.8, 1: 0.2}, {0: 0.85, 1: 0.15}, {0: 0.9, 1: 0.1}, {0: 0.95, 1: 0.05}
'penalty'	'l2'
'solver'	'lbfgs','liblinear'

Table A.1: Grid search parameter space for the baseline, CNN-8k and CNN-32k logistic regression classifier.

All other hyperparameters are set to their default values, except for 'max_iter', which is increased to 1000.

A.2 Support Vector Machine

The SVM grid search was divided into three parts, one for each kernel: linear, polynomial, and RBF. Table A.2 shows the parameter space for the linear SVM. Similarly to the logistic regression model, the 'C' parameter corresponds to an inverse regularization strength. In Table A.3, the parameter space for the polynomial SVM is shown. Polynomials of different degrees are examined in the grid search, and the 'gamma' parameter determines how the kernel coefficient is calculated. Table A.4 displays the parameter space of the RBF SVM.

Hyperparameter	**Values**
'C'	0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100
'class_weight'	{0: 0.7, 1: 0.3}, {0: 0.75, 1: 0.25}, {0: 0.8, 1: 0.2}, {0: 0.85, 1: 0.15}, {0: 0.9, 1: 0.1}, {0: 0.95, 1: 0.05}
'kernel'	'linear'

Table A.2: Grid search parameter space for the baseline, CNN-8k and CNN-32k SVM linear classifier.

Hyperparameter	**Values**
'C'	0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 125, 150
'class_weight'	{0: 0.7, 1: 0.3}, {0: 0.75, 1: 0.25}, {0: 0.8, 1: 0.2}, {0: 0.85, 1: 0.15}, {0: 0.9, 1: 0.1}, {0: 0.95, 1: 0.05}
'kernel'	'poly'
'degree'	2, 3, 4, 5
'gamma'	'scale', 'auto'

Table A.3: Grid search parameter space for the baseline, CNN-8k and CNN-32k SVM polynomial classifier.

Hyperparameter	Values
'C'	0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 125, 150
'class_weight'	{0: 0.7, 1: 0.3}, {0: 0.75, 1: 0.25}, {0: 0.8, 1: 0.2}, {0: 0.85, 1: 0.15}, {0: 0.9, 1: 0.1}, {0: 0.95, 1: 0.05}
'kernel'	'rbf'
'gamma'	'scale', 'auto'

Table A.4: Grid search parameter space for the baseline, CNN-8k and CNN-32k SVM RBF classifier.

A.3 Random Forest

Table A.5 shows the grid search values for the baseline RFC. The CNN-8k and CNN-32k RFC had a slightly increased parameter space, with 6 and 7 added as values to 'n_estimators' and 'max_depth'. All other hyperparameters were set to the default Scikit-learn [34] value, except for the randomization parameter, being set to the same value as the dataset split seed for repeatability. The 'n_estimators' parameter controls the number of trees in the random forest, and the 'max_depth' determines the maximum depth of each tree. The 'min_samples_split' sets the minimum number of samples that are needed to split a node in a tree. The hyperparameters available with the Scikit-learn [34], but not detailed in the following tabels, are set to their default values.

Hyperparameter	Values
'class_weight'	{0: 0.7, 1: 0.3}, {0: 0.75, 1: 0.25}, {0: 0.8, 1: 0.2}, {0: 0.85, 1: 0.15}, {0: 0.9, 1: 0.1}, {0: 0.95, 1: 0.05}
'n_estimators'	1, 2, 3, 4, 5
'max_depth'	2, 3, 4, 5
'min_samples_split'	2, 3, 4

Table A.5: Grid search parameter space for the baseline random forest classifier.

A.4 MLP

This section provides the parameter spaces used for the MLP grid searches. The baseline, CNN-8k, and CNN-32k MLP grid searches had different parameter spaces. This section is therefore divided into subsections.

The 'hidden_layer_sizes' and 'alpha' parameters were important for all grid searches. The parameter 'hidden_layer_sizes' determines the number of nodes in each layer in the MLP, and 'alpha' is a regularization coefficient that determines the strength of the L2 regularization [34].

A.4.1 Baseline

Table A.6 shows the parameter space used for the baseline MLP grid search. Due to the baseline having fewer input features, the parameter space could be relatively large compared to the corresponding feature learning parameter spaces. In addition to the 'hidden_layer_sizes' and 'alpha' parameters, the grid search looks at various optimization functions with the 'solver' parameter, and different learning rate update schemes with 'learning_rate'. The same parameter space was used when training a MLP with the oversampled training set.

Hyperparameter	**Values**
'hidden_layer_sizes'	(5,), (6,), (7,), (8,), (9,), (10,), (11,), (12,), (10, 3), (10, 4), (12,4), (12,5), (14, 4)
'alpha'	$5\cdot10^{-5}$, $1\cdot10^{-4}$, $5\cdot10^{-4}$, $1\cdot10^{-3}$, $5\cdot10^{-3}$, 0.01, 0.05, 0.1, 0.5
'activation'	'relu'
'solver'	'sgd', 'adam', 'lbfgs'
'learning_rate'	'constant', 'adaptive'

Table A.6: Grid search parameter space for the baseline MLP classifier. The same parameter space was used for the MLP trained with the oversampled training set.

A.4.2 CNN-8k and CNN-32k

The CNN-8k and CNN-32k latent variables were used to train two MLPs each; a 1-layer and 2-layer version. Table A.7 and Table A.8 shows the parameter spaces for the CNN-8k MLPs. Table A.9 and Table A.10 shows the parameter spaces for the CNN-32k MLPs

Hyperparameter	**Values**
'hidden_layer_sizes'	(25,), (50,), (100,)
'alpha'	0.25, 0.5, 1
'activation'	'relu'
'solver'	'adam'
'learning_rate'	'constant'

Table A.7: Grid search parameter space for the CNN-8k 1-layer MLP classifier.

Hyperparameter	**Values**
'hidden_layer_sizes'	(12, 4), (30, 10), (100, 20)
'alpha'	0.001, 0.01, 0.1, 1
'activation'	'relu'
'solver'	'adam'
'learning_rate'	'constant'

Table A.8: Grid search parameter space for the CNN-8k 2-layer MLP classifier.

Hyperparameter	**Values**
'hidden_layer_sizes'	(200,), (250,)
'alpha'	10, 50, 55
'activation'	'relu'
'solver'	'adam'
'learning_rate'	'constant'

Table A.9: Grid search parameter space for the CNN-32k 1-layer MLP classifier.

Hyperparameter	**Values**
'hidden_layer_sizes'	(12, 4), (18, 4), (20, 6), (30, 10), (100, 20)
'alpha'	$5 \cdot 10^{-6}$, $1 \cdot 10^{-5}$, $1 \cdot 10^{-4}$
'activation'	'relu'
'solver'	'adam'
'learning_rate'	'constant'

Table A.10: Grid search parameter space for the CNN-32k 2-layer MLP classifier.

Appendix B

Hyperparameters

Table B.1, Table B.2, and Table B.3 provide the chosen hyperparameters of the final models for the baseline and feature learning implementations, which were decided through grid searches. The grid searches were conducted with Stratified Kfold (K = 3) of the training set, and using the F1 metric for scoring. Table B.4 provides the hyperparameters of the fine-tuned ViT-Base model, where manual hyperparameter-tuning was performed with the choice of optimizer and learning rate.

Classifier	**Hyperparameters**
Logistic Regression	'C': 0.5, 'class_weight': {0: 0.7, 1: 0.3}, 'penalty': 'l2', 'solver': 'liblinear'
Linear SVM	'C': 50, 'class_weight': {0: 0.85, 1: 0.15}, 'kernel': 'linear'
Poly SVM	'C': 125, 'class_weight': {0: 0.75, 1: 0.25}, 'degree': 4, 'gamma': 'scale', 'kernel': 'poly'
RBF SVM	'C': 50, 'class_weight': {0: 0.9, 1: 0.1}, 'gamma': 'scale', 'kernel': 'rbf'
Random Forest	'class_weight': {0: 0.75, 1: 0.25}, 'max_depth': 5, 'min_samples_split': 4, 'n_estimators': 5
MLP	'activation': 'relu', 'alpha': 0.5, 'hidden_layer_sizes': (11,), 'learning_rate': 'constant', 'solver': 'adam'

Table B.1: Baseline classifiers and their hyperparameters found through grid search using a specific random dataset split. The grid searches were conducted with Stratified Kfold (K = 3), and using the F1 metric for scoring.

Classifier	**Hyperparameters**
Logistic Regression	'C': 100, 'class_weight' : {0: 0.75, 1: 0.25}, 'penalty': 'l2','solver' : 'lbfgs'
Linear SVM	'C': 0.1, 'class_weight': {0: 0.7, 1: 0.3}, 'kernel': 'linear'
Poly SVM	'C': 125, 'class_weight': {0: 0.7, 1: 0.3}, 'degree': 2, 'gamma': 'scale', 'kernel': 'poly'
RBF SVM	'C': 50, 'class_weight': {0: 0.9, 1: 0.1}, 'gamma': 'scale', 'kernel': 'rbf'
Random Forest	'class_weight': {0: 0.75, 1: 0.25}, 'max_depth': 2, 'min_samples_split': 2, 'n_estimators': 1
1 layer MLP	'activation': 'relu', 'alpha': 1, 'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'adam'
2-layered MLP	'activation': 'relu', 'alpha': 0.001, 'hidden_layer_sizes': (12,4), 'learning_rate': 'constant', 'solver': 'adam'

Table B.2: Feature Learning classifiers using the CNN-8k encoder, and their hyperparameters found through grid search using a specific random dataset split.

Classifier	Hyperparameters
Logistic Regression	'C': 1000, 'class_weight' : {0: 0.7, 1: 0.3}, 'penalty': 'l2','solver' : 'lbfgs'
Linear SVM	'C':1, 'class_weight': {0: 0.7, 1: 0.3}, 'kernel': 'linear'
Poly SVM	'C': 1, 'class_weight': {0: 0.7, 1: 0.3}, 'degree': 2, 'gamma': 'auto', 'kernel': 'poly'
RBF SVM	'C': 10, 'class_weight': {0: 0.75, 1: 0.25}, 'gamma': 'scale', 'kernel': 'rbf'
Random Forest	'class_weight': {0: 0.75, 1: 0.25}, 'max_depth': 6, 'min_samples_split': 3, 'n_estimators': 1
1 layer MLP	'activation': 'relu', 'alpha': 50, 'hidden_layer_sizes': (200,), 'learning_rate': 'constant', 'solver': 'adam'
2-layered MLP	'activation': 'relu', 'alpha': 0.0001, 'hidden_layer_sizes': (20,6), 'learning_rate': 'constant', 'solver': 'adam'

Table B.3: Feature Learning classifiers using the CNN-32k encoder, and their hyperparameters found through grid search using a specific random dataset split.

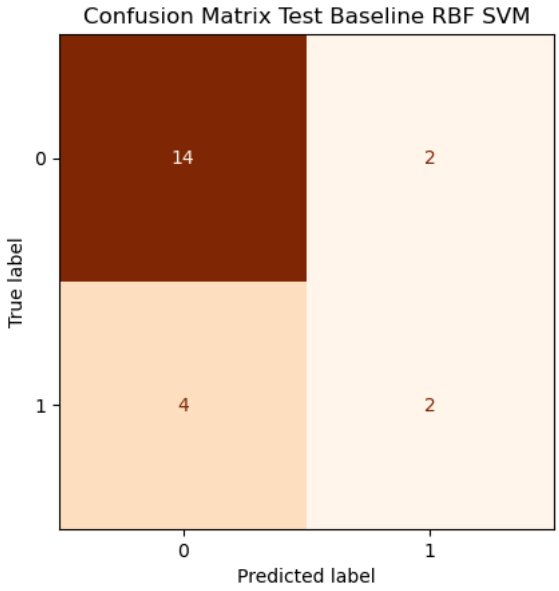
Classifier	Hyperparameters
Fine-tuned ViT-Base	optimizer: 'adam', learning rate: 0.0005, 'betas': (0.9, 0.999), class weights: {0: 0.7, 1: 0.3}, loss function: 'Cross entropy loss'

Table B.4: The chosen hyperparameters for the fine-tuned ViT-Base used in the project.

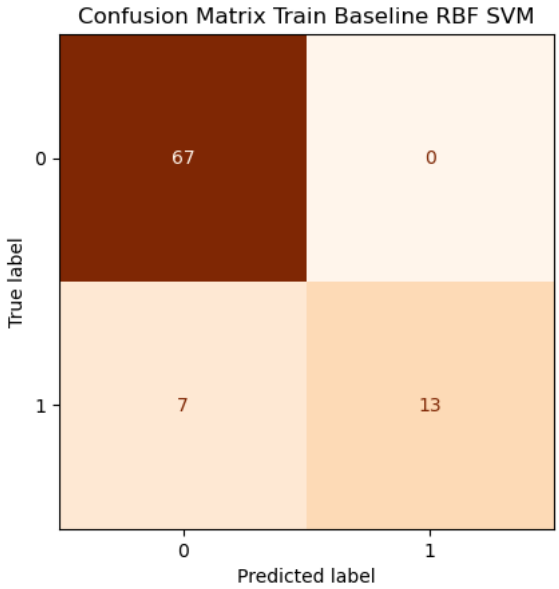
Appendix C

Confusion Matricies

In the figures of this appendix, the True/Predicted label ”1” represents CBCT Positive.

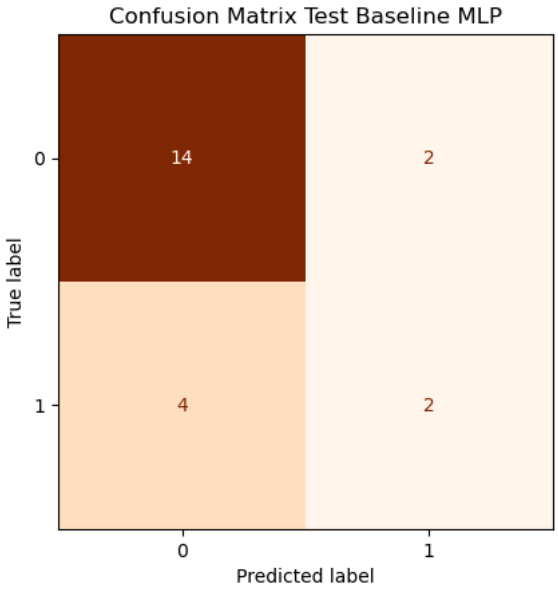


(a) Test set confusion matrix of the baseline RBF SVM.

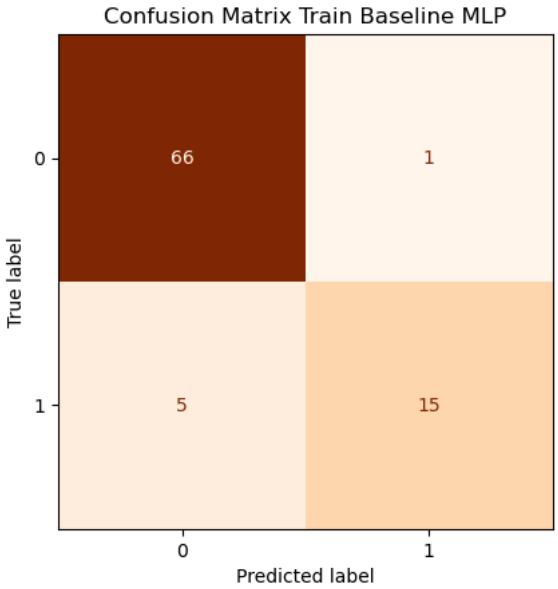


(b) Train set confusion matrix of the baseline RBF SVM.

Figure C.1: The confusion matrices of the baseline RBF SVM classifier.

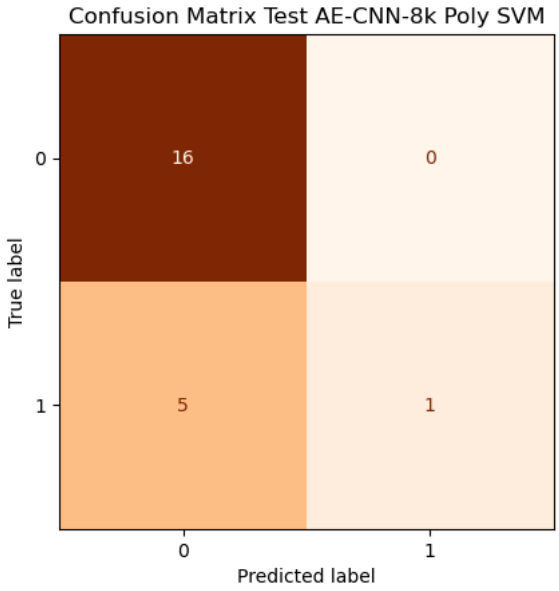


(a) Test set confusion matrix of the baseline MLP.

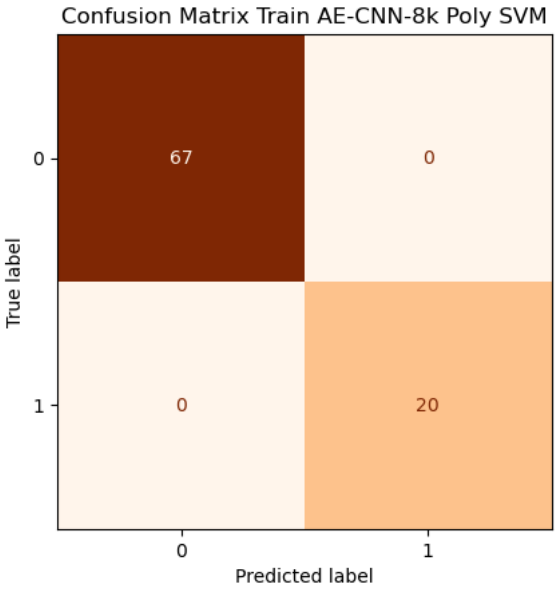


(b) Train set confusion matrix of the baseline MLP.

Figure C.2: The confusion matrices of the baseline MLP with layer_sizes = (11,).

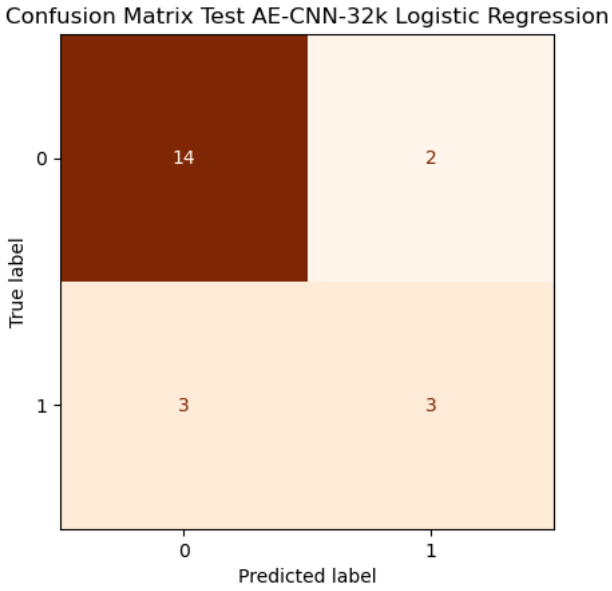


(a) Test set confusion matrix of SVM classifier

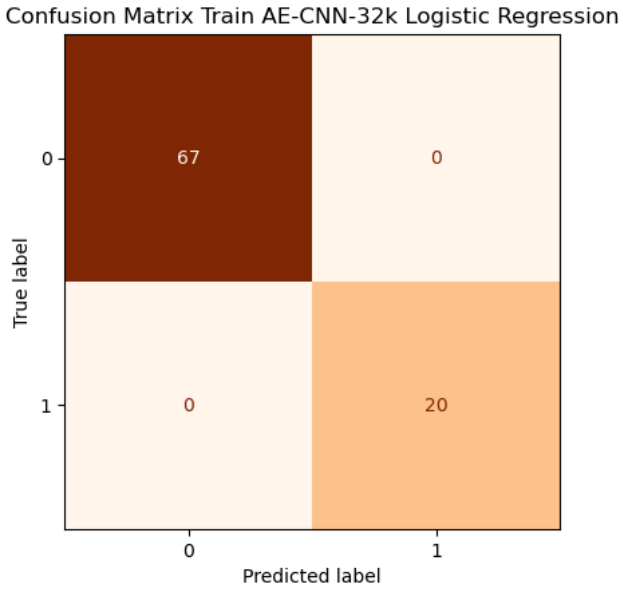


(b) Train set confusion matrix of SVM classifier.

Figure C.3: The confusion matricies for the SVM with a polynomial kernel of degree 2, taking the 8k latent variables as input.

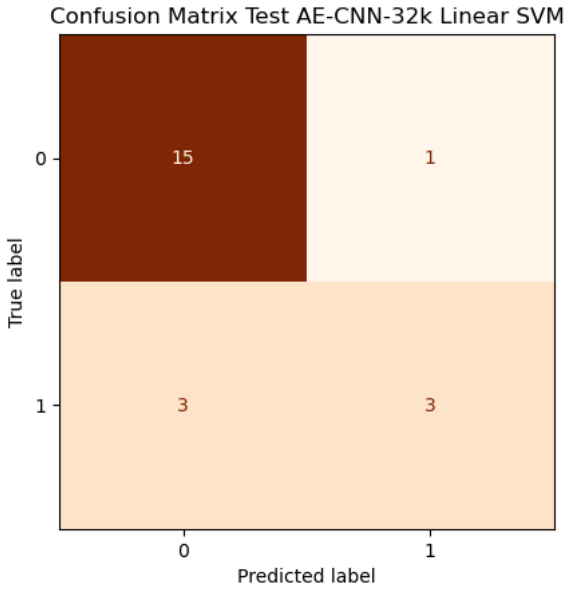


(a) Test set confusion matrix of the CNN-32k logistic regression classifier

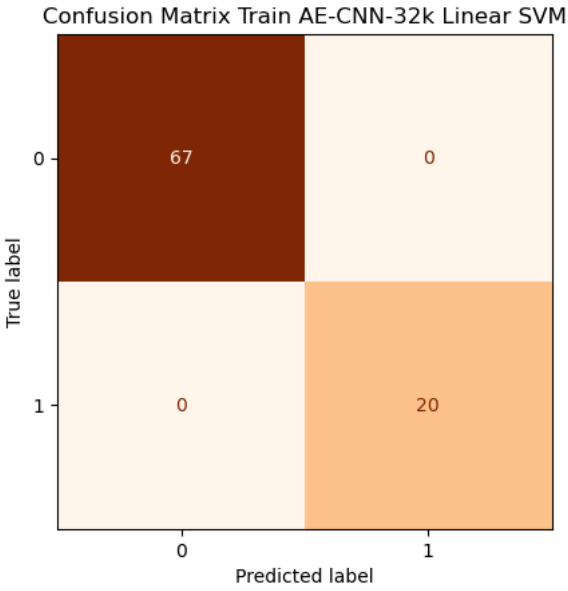


(b) Train set confusion matrix of the CNN-32k logistic regression classifier.

Figure C.4: The confusion matricies for the logistic regression classifier taking 32k latent variables as input.

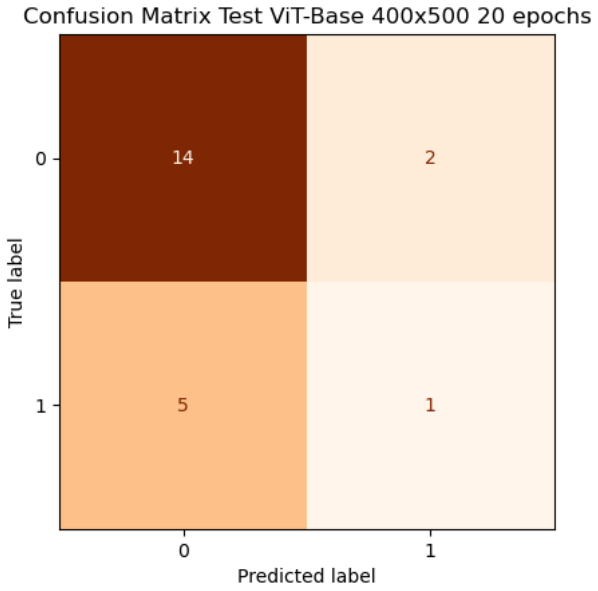


(a) Test set confusion matrix of the CNN-32k Linear SVM.

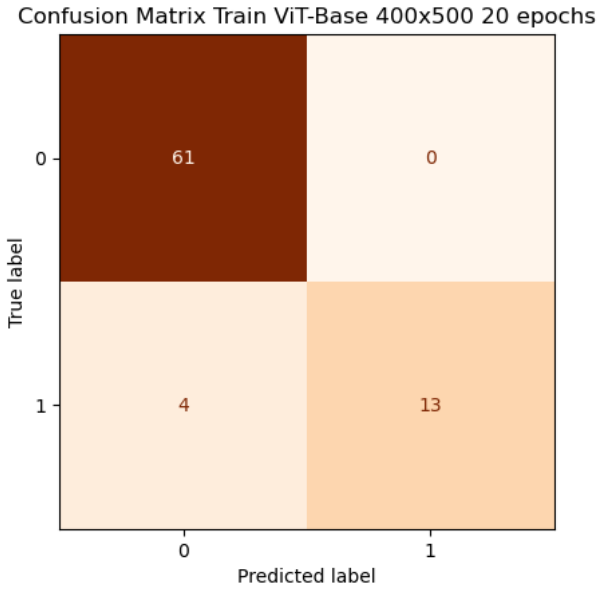


(b) Train set confusion matrix of the CNN-32k SVM.

Figure C.5: The confusion matricies for the Linear SVM taking 32k latent variables as input.

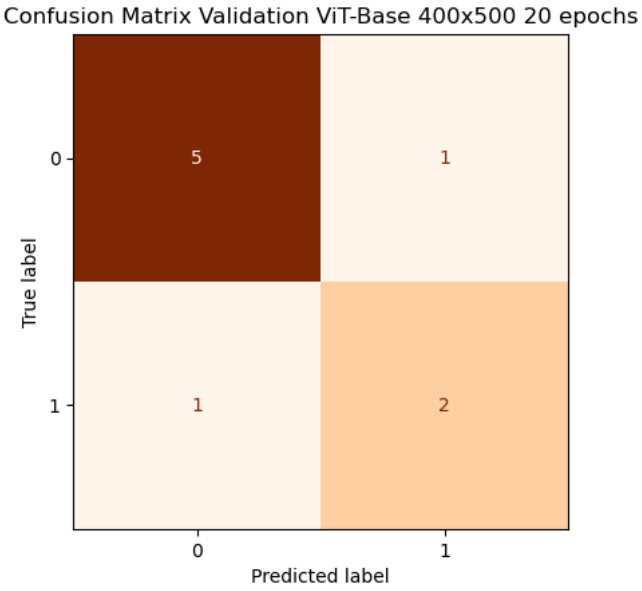


(a) Test set confusion matrix of the 20 epoch 400x500 ViT-Base



(b) Train set confusion matrix of the 20 epoch 400x500 ViT-Base

Figure C.6: The confusion matricies of the 400x500 input ViT-Base trained for 20 epochs.



(c) Validation set confusion matrix of the 20 epoch 400x500 ViT-Base.

Figure C.6: The confusion matricies of the 400x500 input ViT-Base trained for 20 epochs.

Appendix D

Supporting results

Figure D.1 displays the calibration curves for the baseline models. Calibration curves are constructed by letting the classifier predict probabilities on the test set. The predicted probabilities are binned, where 10 bins were used to produce Figure D.1. The fraction of actual true instances is determined for each bin, and is plotted versus the predicted probabilities of each bin [34]. The calibration curves offer some insights into the confidence of the model's predictions. Figure D.1 shows that the baseline models tend to be both overconfident and underconfident in their predictions, which implies that they are poorly calibrated. This could be due to underfitting and overfitting, or due to a lack of training data.

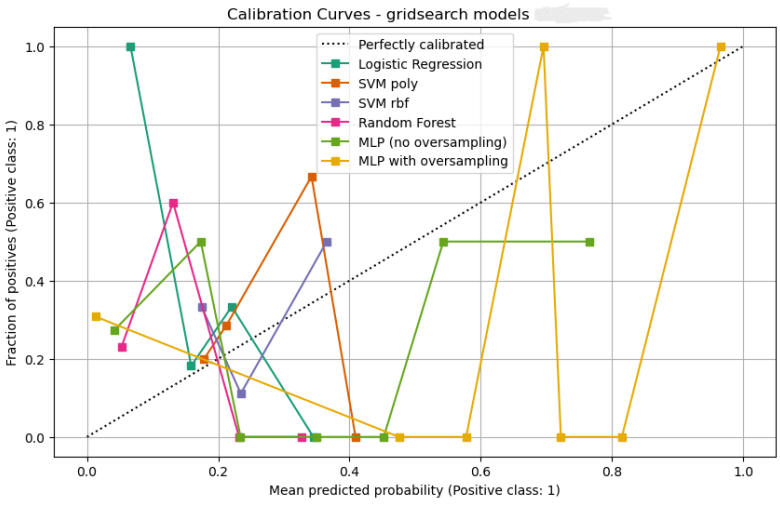


Figure D.1: Calibration curves of the baseline models.

TRITA-EECS-EX- 2023:0000

€€€€ For DIVA €€€€

{
”Author1”: { ”Last name”: ”Briffa”,
”First name”: ”Laura”,
”Local User Id”: ”u100001”,
”E-mail”: ”briffa@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
}
},
”Cycle”: ”2”,
”Course code”: ”EA260X”,
”Credits”: ”30.0”,
”Degree1”: {”Educational program”: ”Degree Programme in Electrical Engineering”
,”programcode”: ”CELTE”
,”Degree”: ”Degree of Master of Science in Engineering”
,”subjectArea”: ”Information and Network Engineering”
},
”Title”: {
”Main title”: ”Identifying Complications of Impacted Canines in Panoramic Radiographs Using Deep Learning Techniques”,
”Subtitle”: ”An investigation of methods for a small-sized dataset”,
”Language”: ”eng” },
”Alternative title”: {
”Main title”: ”Djupinlärning med ortopantomogram för identifiering av komplikationer från retinerade hörntänder”,
”Subtitle”: ”En undersökning av lämpliga metoder för en liten datamängd”,
”Language”: “swe”
},
”Supervisor1”: { ”Last name”: ”Honoré”,
”First name”: ”Antoine”,
”Local User Id”: ”u100003”,
”E-mail”: ”honore@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Department of Intelligent Systems” }
},
”Supervisor2”: { ”Last name”: ”Borsci”,
”First name”: ”Elena”,
”E-mail”: ”elena.borsci@ki.se”,
”Other organisation”: ”Karolinska Institutet, Department of Dental Medicine”
},
”Examiner1”: { ”Last name”: ”Chatterjee”,
”First name”: ”Saikat”,
”Local User Id”: ”u100003”,
”E-mail”: ”sach@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Department of Intelligent Systems” }
},
”National Subject Categories”: ”10299, 20299”,
”Other information”: {”Year”: ”2024”, ”Number of pages”: ”1,87”},
”Copyrightleft”: ”copyright”,
”Series”: { ”Title of series”: ”TRITA-EECS-EX” , ”No. in series”: ”2023:0000” },
”Opponents”: { ”Name”: ”Pia Appelquist”},
”Presentation”: { ”Date”: ”2024-04-26 10:00”
,”Language”:”eng”
,”Room”: ”via Zoom https://kth-se.zoom.us/j/61975476518”
,”Address”: ”Malvinas Väg 10”
,”City”: ”Stockholm” },
”Number of lang instances”: ”2”,
”Abstract[eng]”: €€€€

Maxillary canine impaction occurs when the canine tooth in the upper jaw fails to emerge as anticipated.
Addressing impacted canines presents significant challenges, including potential complications arising from delayed treatment.
While 2-dimensional panoramic radiographs are commonly employed for initial diagnostics of canine impaction, this modality has inherent limitations.
For more complex cases, cone beam computed tomography (CBCT) provides detailed 3-dimensional imaging. Compared to the panoramic radiograph, CBCT offers a higher diagnostic accuracy for detection of root resorption and localization of the canine.
However, the use of CBCT is associated with significantly higher radiation exposure to the patient. This thesis aims to explore the application of machine learning techniques to predict the need for CBCT evaluation for patients with impacted canines. Utilizing a dataset of 109 panoramic radiographs, baseline metrics are established based on measured features derived from the radiographs. These baseline results are compared to deep learning image classification techniques. The study examines a feature learning approach employing convolutional autoencoders and investigates a transfer learning strategy using a pre-trained Vision Transformer classifier.
The comparative study showed that the best baseline classifier achieved an F1 score of 0.40 and an AUROC of 0.56.

In the deep learning component of the study, two autoencoder models and two instances of fine-tuning a pre-trained Vision Transformer were investigated. Among these, one autoencoder implementation outperformed the baseline, achieving an F1 score of 0.60 and an AUROC of 0.81. This particular model employed an encoder that reduced the image input space by half and was paired with a linear Support Vector Machine for classification.
The findings suggest potential in these methods, yet underscore the need for a larger dataset to more thoroughly assess their suitability for the task.

€€€€,
"Keywords[eng]": €€€€
Impacted maxillary canines, Panoramic radiography, Autoencoders, Convolutional Neural Networks, Vision Transformers, Deep learning.
€€€€,
"Abstract[swe]": €€€€

Retinerade hörntänder uppkommer främst hos barn, och innebär att den permanenta hörntanden inte trängt fram när den ska. Behandlingen av retinerade hörntänder är invecklad, där en försenad behandling kan innebära komplikationer. Panoramaröntgen, en sorts 2D-bildtagning, är vanligt förekommande för diagnosering av retinerade hörntänder. Denna modalitet har däremot begränsningar som kan försvåra diagnostiken. I mer komplicerade fall kan datortomografi av tänder (CBCT) användas för att erhålla en 3D-volymbild. Jämfört med panoramaröntgen kan CBCT erbjuda en högre diagnostisk noggrannhet för rotresorption och lokalisering av hörntanden. Däremot avger CBCT en betydligt högre strålningsdos till patienten, och dess användning ska vara väl motiverad. Denna avhandling utforskar möjligheten att använda maskininlärningsmetoder för att förutse om en patient med retinerade hörntänder behöver CBCT-utvärdering givet en panoramaröntgen. Med en datamängd bestående av 109 panoramaröntgenbilder, skapas först baslinje-modeller tränade på numeriska värden uppmätta från dessa panoramabilder. Dessa baslinje-modeller jämförs sedan med djupinlärningmetoder som utnyttjat metoder inom bildklassificering. Närmare bestämt utforskar denna avhandling en metod som involverar autokodare uppbyggda av faltningsnät, samt en metod som innefattar finjustering av en tidigare tränad visionstransformator. Den jämförande studien visar att den bästa baslinje-modellen uppnår ett F1-tal på 0.40 och ett AUROC tal på 0.56. I djupinlärningsdelen av studien undersöktes två instanser av autokodare, och två instanser av finjustering av en visionstransformator. Bland dessa kunde en implementation av en autokodare uppnå högre resultat än basnivån, med ett F1-tal på 0.60 och en AUROC på 0.81. Denna implementation hade en kodare som halverade inmatningsstorleken, som parades ihop med en linjär stödvektormaskin för klassificering.
Resultaten indikerar en potential i de undersökta metoderna, men understryker behovet av en större datamängd för att noggrant utvärdera deras lämplighet för uppgiften.

€€€€,
"Keywords[swe]": €€€€
Retinerade hörntänder, Panoramaradiografi, Autokodare, Faltningsnätverk, Visionstransformator, Djupinlärning. €€€€,
}

acronyms.tex

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: t
%%% End:
% The following command is used with glossaries-extra
\setabbreviationstyle[acronym]{long-short}
% The form of the entries in this file is \newacronym{label}{acronym}{phrase}
%                                     or \newacronym[options]{label}{acronym}{phrase}
% see "User Manual for glossaries.sty" for the  details about the options, one example is shown below
% note the specification of the long form plural in the line below
\newacronym[longplural={Debugging Information Entities}]{DIE}{DIE}{Debugging Information Entity}
%
% The following example also uses options
\newacronym[shortplural={OSes}, firstplural={operating systems (OSes)}]{OS}{OS}{operating system}

% note the use of a non-breaking dash in long text for the following acronym
\newacronym{IQL}{IQL}{Independent -QLearning}

% example of putting in a trademark on first expansion
\newacronym[first={NVIDIA OpenSHMEM Library (NVSHMEM\texttrademark)}]{NVSHMEM}{NVSHMEM}{NVIDIA OpenSHMEM Library}

\newacronym{KTH}{KTH}{KTH Royal Institute of Technology}

\newacronym{LAN}{LAN}{Local Area Network}
\newacronym{VM}{VM}{virtual machine}
% note the use of a non-breaking dash in the following acronym
\newacronym{WiFi}{-WiFi}{Wireless Fidelity}

\newacronym{WLAN}{WLAN}{Wireless Local Area Network}
\newacronym{UN}{UN}{United Nations}
\newacronym{SDG}{SDG}{Sustainable Development Goal}

\newacronym{AI}{AI}{artificial intelligence}

\newacronym{ML}{ML}{machine learning}

\newacronym{DL}{DL}{deep learning}

\newacronym{CNN}{CNN}{convolutional neural network}

\newacronym{PAN}{PAN}{panoramic radiograph}

\newacronym{CBCT}{CBCT}{cone beam computed tomography}


\newacronym{RELU}{ReLU}{rectified linear unit}

\newacronym{MLP}{MLP}{multilayer perceptron}

\newacronym{AE}{AE}{autoencoder}

\newacronym{SVM}{SVM}{support vector machine}
\newacronym{RFC}{RFC}{random forest classifier}

\newacronym{ViT}{ViT}{Vision Transformer}

\newacronym{TP}{TP}{True Positive}
\newacronym{TN}{TN}{True Negative}
\newacronym{FP}{FP}{False Positive}
\newacronym{FN}{FN}{False Negative}

\newacronym{AUROC}{AUROC}{area under receiver operator characteristic curve}
\newacronym{ROC}{ROC}{receiver operator characteristic}
\newacronym{PR}{PR}{precision-recall}
\newacronym{AUPR}{AUPR}{area under precision-recall curve}
```