
Otto-von-Guericke University Magdeburg



Faculty of Electrical Engineering and Information Technology
Institute of Medical Technology (IMT)

Master Thesis

Deep Learning and Deep Embedding Clustering Analysis for Laryngeal Cancer Classification on Contact Endoscopy - Narrow Band Images

Author:

ESAM SHARAF

Medical Systems Engineering

November 23, 2020

Examiners:

Prof. Dr. Michael Friebe

Faculty of Medicine
Otto-von-Guericke University
39120 Magdeburg, Germany

Prof. Dr. Christoph Hoeschen

Faculty of Electrical Engineering
and Information Technology
Otto-von-Guericke University
39106 Magdeburg, Germany

Advisors:

Ms. Nazila Esmaili, MSc.

INKA-Application Driven Research
Otto-von-Guericke University
39120 Magdeburg, Germany

Mr. Elmer Jeto Gomes Ataide, MSc.

Clinic for Radiology and Nuclear Medicine
Department of Nuclear Medicine
Otto-von-Guericke University
39120 Magdeburg, Germany



OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

FACULTY OF
ELECTRICAL ENGINEERING AND
INFORMATION TECHNOLOGY

Task Description of a Master Thesis

Esam M. Sharaf

Student ID: 218124

Topic

**Deep Learning and Deep Embedding for Clustering Analysis for
Laryngeal Cancer Classification on Contact Endoscopy - Narrow Band Images**

Task Description

Laryngeal cancer is one of the most common tumors of the respiratory tract. Changes in the organization and structure of the vocal fold's blood vessels are directly related to the development of benign and subsequent malignant laryngeal lesions. Contact Endoscopy (CE) is a minimally invasive procedure providing a real time in situ examination of superficial layer of laryngeal mucosa. CE is an essential optical imaging modality for visualizing vessels structure in details in the larynx. In order to obtain high-contrast visualization, CE tends to be adapted to Narrow Band Imaging (NBI). NBI acts by emitting filtered wavelengths for detecting abnormal changes in the mucosa precisely.

Laryngeal malignancies are difficult to be distinguished from benign ones as well as the related diagnosis outcomes may rely on subjective interpretation. Therefore, this leads to reduce the efficacy of conventional diagnosis and prognosis procedures in terms of accuracy and the time needed in the practice.

Advent machine learning (ML) models - supervised and unsupervised methods, have shown to expediting the classification of various cancer types, but more to outperform humans' capacity. Cascading such AI models with CE-NBI acquired images seems to have the potential to increase laryngeal cancer detection accuracy rates. Also, this combination allows an objective diagnosis of the cancerous lesions in the larynx automatically, by which compensates the current limitations mentioned earlier.

The aim of this work is to compare the performance of two different approaches on the classification of the larynx CE-NBI images into benign and malignant classes: supervised Deep Convolution Neural Network (DCNN) and unsupervised K-means Clustering. The work milestones are:

1. Literature review and understanding the problem.
2. Building of DCNN structure with transfer learning for benign/ malignant classification. Fine tuning network's parameter for the best possible results.
3. Extracting Auto-encoder features and feeding them into the K-means Clustering model for benign/ malignant classification for the best possible performance.
4. Comparing the output results from both approaches and discussing the whole work outcomes.
5. Full paper (conference/journal).

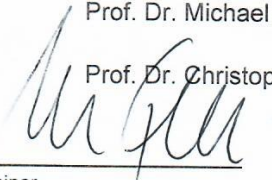
Magdeburg, 24/06/2020

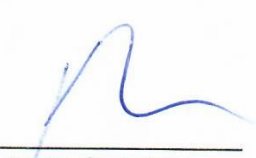
Start of thesis work: 15/06/2020

Date of submission: 02/11/2020

1st Examiner: Prof. Dr. Michael Friebe.

2nd Examiner: Prof. Dr. Christoph Hoeschen.


1st Examiner


Prof. Dr. rer. nat. Georg Rose
(Chairman of Examination Board)

Declaration of Academic Integrity

I hereby declare that I have written the present work myself and did not use any sources or tools other than the ones indicated.

Datum:

.....

(Signature)

Contents

List of Figures

List of Tables

Abstract

1 Introduction

2 Methods

2.1 The Dataset	6
2.2 ResNet50 DCNN as Binary Classifier	9
2.3 Deep Embedding Clustering for Binary Classification	14
2.3.1 Unsupervised Deep Embedding for Clustering Analysis (DEC)	15
2.3.2 Plain Variant of DEC	18
2.3.3 DEC based on Deep-Convolutional-Extracted Features	18

3 Experimental Evaluation

3.1 ResNet50 DCNN as Binary Classifier	20
3.1.1 Binary Classification for Dataset A	21
3.1.2 Binary Classification for Dataset C	26
3.1.3 Binary Classification for Augmented Dataset C	33
3.2 Deep Embedding Clustering for Binary Classification	38

4 Discussion

4.1 Contribution of Deep CNN	41
4.2 Contribution of Deep Embedding Clustering	43

5 Conclusions and Future Work

A Appendix

B Bibliography

List of Figures

2.1	CE-NBI images examples of different kinds of laryngeal cancer: (a) samples from Dataset A. Reinke's edema (left), Hyperkeratosis (middle), and Dysplasia severe (Right). (b) samples from Dataset B. Cyst (left), Papilloma (middle), and Dysplasia Severe (Right)	8
2.2	Key elements of ResNet's structure	10
2.3	The basic parts of Deep Convolutional Neural Network: feature extraction and the classifier	10
2.4	Unsupervised Deep Embedding for Clustering Analysis (DEC) scheme . .	15
2.5	Autoencoder structure of initialisation phase in DEC method	17
2.6	Structure of Deep Embedding Clustering model at the optimisation phase	18
2.7	The standard VGG16 network architecture	19
3.1	Model 1 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	21
3.2	Model 2 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	22
3.3	Model 3 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	23
3.4	Model 4 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	24
3.5	Model 5 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	27
3.6	Model 6 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	28
3.7	Model 7 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	29
3.8	Comparison of the accuracy track between Model 5 and Model 7. Subfigure numbering stands for the fold number. Yellow and blue curves represent the training and validation curve of Model 5, respectively. Black and green lines represent the training and validation curve of Model 7, respectively. .	30
3.9	Model 8 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	31

3.10	Model 5 5-folds cross validation results at the second run. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	32
3.11	5-folds cross validation results of model five on dataset C without being augmented. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	34
3.12	5-folds cross validation results of model 5 on augmented dataset C (horizontal and vertical flipping). Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	35
3.13	5-folds cross validation results of model 5 on augmented dataset C (horizontal and vertical flipping) with extended early stopping. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.	36
3.14	Confusion matrix of 903 test images by model five being trained on augmented dataset.	36
3.15	Examples of images incorrectly classified by model five. Reink's edema (left). Cyst (Middle). Severe dysplasia (right).	37
3.16	Mean Square Error during initialisation phase. (a) Unsupervised Deep Embedding Clustering Analysis(DEC). (b) Plain variant of DEC. (c) DEC based on deep-convolutional extracted features.	38
3.17	Examples of input images (left) and their reconstructed version (right). . .	39
3.18	Projection of the latent features of z-space by Principal Components Analysis (PCA). (a) DEC method. (b) Plain DEC. (c) DEC based on deep-convolutional extracted features. The upper projection based on ground-truth labels. The lower projection based on clustering prediction.	40
4.1	How Global Max Pooling Layer maps input feature map to a single value .	42
4.2	Examples of preprocessed images through skeletonization function. . . .	44
4.3	Samples from MNIST dataset	44
4.4	Clustering visualisation of plain variant of DEC method on MNIST images. Upper projection based on truth labels. Lower projection based on clustering prediction.	44
4.5	Clustering visualisation of DEC method based on deep-convolutional-extracted features of MNIST images. Upper projection based on truth labels. Lower projection based on clustering prediction.	45
A.1	Model summary of plain version of DEC model.	54

List of Tables

2.1	Dataset A subclasses.	6
2.2	Dataset B subclasses.	7
2.3	Comparison of clustering accuracy on four datasets. LDMGI: Local Discriminant Models and Global Integration. SEC: Spectral Embedded Clustering. DEC: Deep Embedding for Clustering Analysis	17
3.1	Detailed results of experiment number one of DCNN approach. Numbers are averages at the peak accuracy over 5-folds cross validation.	25
3.2	Detailed results of experiment number two of DCNN approach. Numbers are averages at peak accuracy over 5-folds cross validation.	32
3.3	Detailed results of experiment number three of DCNN approach. Numbers are averages at peak accuracy over 5-folds cross validation.	37
3.4	Comparing the accuracy achieved by the three experiment of deep embedding clustering approach with k-means as a reference.	40

Abstract

Laryngeal cancer has small survival rate at advance disease stages. Detection of this kind of cancer at late stage leads to consequences affect life quality of individuals who encounter it. Number of imaging tools have been used for histopathologic assessment the tumors in the larynx. Laryngeal white light endoscopy, narrow band imaging endoscope, and contact endoscopy are largely accepted for visualisation the region of the larynx in this matter. In spite of the technological advancements, the differentiation between malignant and benign lesions in the larynx is difficult in reality, irrespective to the experience level. Also, The subjectivity in laryngeal cancer diagnosis has been recorded, which reduces the confidence in the submission process by visual tools. Following these challenges, additional costs and time are required, especially that having high recognition rate of cancers calls for long training period such as laryngeal cancer. This works investigates using machine learning for classifying 2D images acquired by contact endoscopy-narrow band endoscope. Two methods are proposed, namely, deep convolutional neural networks and deep embedding clustering. Each represent a different machine learning approach. The first one represents a supervised learning method. The second one represents an unsupervised learning method. The deep convolutional network involves using a pretrained model by transfer learning technique, in order to decrease the time required for training and to increase the generalisability. Different models of different structure and hyperparamters are executed, on augmented and non-augmented data. 83.5%, 97.68% accuracy rates were achieved on augmented and non-augmented trials, respectively. With respect of the deep embedding clustering method, three variant implementations were examined. The first was about testing Deep Embedding Clustering Analysis (DEC) method on the dataset, which achieved clustering accuracy equals to 58.61%. The second trial was about plain structure of DEC with accelerating batch normalisation layers. It achieved 51.69% accuracy rate. The third implementation was a try to cluster the images based on their extracted features by deep convolutional network. The clustering process followed DEC method, with clustering accuracy equals 55.92%. The outcome of this work demonstrates that supervised learning proposed method surpasses deep embedding clustering. We believe that the deep convolutional neural network method can act as an assistant tool in submitting similar cases through standard medical procedures.

1

Introduction

Laryngeal Cancer is one of the most common tumors of the respiratory tract [1, 2]. Laryngeal cancer is one of few oncologic diseases in which the 5-year survival rate has decreased over the past 40 years, from 66% to 63%, although the overall incidence is declining [1]. Endoscopy is a widely used procedure for early laryngeal cancer detection, by using White Light Imaging (WLI) and Narrow Band Imaging (NBI) techniques interchangeably during the medical procedures [3].

NBI-equipped endoscope emits a filtered-band light (415 & 540nm) that interacts efficiently with laryngeal epithelium. Interestingly, NBI technology provides a unique insight by an enhanced-contrast visualization for laryngeal mucosa, specifically superficial mucosal capillaries. It allows characterising the irregularities of the vascular pattern in superficial neoplastic lesions in the mucosa. Therefore, it improves the improved interpretation of the color change through the morphologic analysis of the superficial vascular network, which is altered in tumoral angiogenesis [4]. NBI is able to detect small lesions of a size smaller than 1cm, which would be overlooked by white light imaging [5]. NBI imaging has a significant added value over the normal WLI alone [6–9], helping in laryngeal lesions submission as a result [10]. However, a limitation of NBI exhibits when keratin layer covering underlying mucosa, which led to false negative cases [6]. In such cases, a biopsy is needed by direct laryngoscopy under general anesthesia. Attaching a magnification function to NBI will improve the accuracy, the study suggested. Furthermore, study [11] pointed to pale depressed lesions in the gastric cancer as a limitation of Magnifying NBI(M-NBI), but not for white light imaging endoscopy, by which those lesions are detectable. This finding mentioned in study [12], where most misdiagnosed lesions had flat-type gross morphology. Another two randomised trials were conducted for assessing detection and miss rate of different types of endoscopes at screening colonoscopy including NBI, concluded that NBI imaging made no difference in missing rates, even with experienced endoscopists. [13, 14].

Although NBI imaging has shown to be beneficial in a big margin, it does not provide detailed description of the structure of the mucosa. Contact endoscopy (CE) is an optical technique that magnifies the target area in the mucosa. CE allows examining superficial layer of the mucosa closely at microscopic scale. By that, blood vessels and capillary loops in the mucosa are manifested. Angiogenesis, which means a growth of new blood vessels, is an essential process extends a long tumors growth course for the blood supply [15]. These new vessels' morphology can be differentiated at each stage during tumor development [16]. Therefore, CE imaging can be an appropriate tool for distinguishing between benign/malignant lesions, particularly in the mucosa. Seeking more reliable visualization in this matter, several studies indicated the added value of using contact endoscopy with NBI image enhancing technique, in great margin over conventional routines [17–19].

Despite the advantages of implementing previous procedures, either NBI videoscopes or CE-NBI laryngoscopy procedures, appropriate expertise is required; that even expert-level examiners are susceptible to false positive and false negative diagnoses [6]. By having a wide array of laryngeal lesions subclasses, within benign and malignant classes, as vascular structure for each type is subtle, endoscopists face a difficulty in confirming the disease stage [18, 20], by which adding the subjectivity during the diagnosis. In study [21], fairly high specificity rate (80%) achieved by a single investigator across learning process covered 134 vocal fold lesions NBI examinations. Also, the study indicated that initial examinations had sensitivity and specificity equal to 83.7%, 76.7% respectively. Similarly, [22] evaluated the learning process of neoplasia detection and found that corresponding sensitivity ranging between 62% and 90% irrespective of operator's experience. This goes in accordance with [18] emphasizing extensive learning is required for achieving acceptable results. Such accuracy rates could lead to missing positive cases at early stage, considering that the treatment at advanced stage of larynx cancer significantly reduces the survival rate [23]. As well relevant to this work's application, study [24] found that identification of particular vascular changes in the target area is partly subjective. Moderate level of agreement between less-experienced observers was concluded by the study. Therefore, current non-automated kind of assessments include unnecessary biopsies. In consequence, adding more to the financial burden beside the required in-house expert-level operators.

Developing a fully automated method for detecting abnormal changes in the larynx is a key for objective diagnosis as well as efficient treatment planning. In this regard, modern machine learning (ML) techniques have shown a big potential on field of head and neck oncology, by outstanding performance on both

related diagnoses and treatment planning challenges [25]. The striking learning capacity of modern ML methods has reviewed through many studies. Halicek et al [26] proposed Deep Convolutional Neural Network (DCNN) model for classifying patches of hyperspectral images of squamous-cell carcinoma tissues. The model achieved accuracy, sensitivity, specificity on holdout cross-validation method equal to 96.8%, 96.4%, 96.1% respectively. These results surpassed other ML methods results like k-Nearest Neighbors (kNN) and Support Vector Machine (SVM) that were tested equally in the study. Different deep learning-based applications for laryngeal cancer detection has achieved notable success [27, 28]. Deep learning based approaches offer self-learning and rapid pattern recognition, for which overtaking humans ability [29]. For example in [26], each of learning and validation phase required only 1.5h and 30s in order. Deep learning is one of the most attractive techniques amongst supervised learning methods in the present time. In similar applications, supervised learning achieved promising results. For example in [30], a real time SVM classifier for diagnosing colorectal lesions achieved 93.2% accuracy. The classification output upon NBI visualization is based on SIFT features, which may compromise the classification robustness, especially under different illuminations levels. [30, 31] strongly demonstrated that supervised learning is comparable to expert ability in lesions diagnosis, but more within much shorter time.

Unsupervised ML, differently from supervised learning, deals with label-free input data. The ability of unsupervised learning to act as stand-alone classifier is still latent, particularly for image classification. In study [32] which evaluated K-means algorithm on classifying 10 cases of ductal carcinoma in situ by semi-automated classifier. That after manual marking of the region of interest - where high dense cancer tissues for each input image, the algorithm successfully achieved the classification with sensitivity of 85.45%, and specificity of 94.64% with a true-negative rate (TNR) of 95.8%, and a false-positive rate (FPR) of 4.2%. The obtained results are limited because of the small size of the dataset. Unsupervised ML is often used to find a structure of given data based on the similarity of selected features. For example, a computer-aided system was proposed by Take-mura et al [33] for classifying NBI images of colorectal lesions. The system was designed for the differentiation between neoplastic and nonneoplastic lesions utilizing hierarchical k-means clustering of local features. The study's classifier achieved detection accuracy of 97.8%. Yet, hand crafting region of interest is a big challenge to it for having a clinical deployment. Study [34] represents another variant usage of k-means. In this study, k-means clustering method was adapted for initial data labeling acting as preliminary stage for U-net model. The work

intended for segmenting and classifying skin lesions images automatically, for which 0.8384 F1 score has achieved.

In the recent years, new algorithms have been proposed to enhance the clustering objective in terms of purity of the output clusters. For example, Unsupervised Deep Embedding Clustering Analysis (DEC)[35] and Deep Clustering with Convolutional [36] algorithms have shown positive impact on image classification tasks with state-of-art measures. These unsupervised algorithms involve adapting an Autoencoder at initial stage for extracting latent features from input images. This work is extended to examine the capacity of advent unsupervised machine learning algorithms, in particular to contact endoscopy-narrow band images.

Since biopsy procedure is remained as the golden standard for diagnosis the suspicious tissues in head and neck cancers [37, 38], which is taken by direct laryngoscopy under general anesthesia. Direct endoscopy could lead to many complications. laryngeal stenosis, stimulation the vagus nerve, highly innervated glottis by the laryngoscopy blade are few examples among many [39–41]. As well, the endoscope may have to be placed very close to the mucosal surface, which can lead to contact bleeding [42]. Therefore, reducing the unnecessary biopsies was a motivating factor in this work. In addition to that, introducing a machine learning based classifier that can bring robust objective performance was an important aspect, to compensate the expertise gap between endoscopists performance.

This work aims at classifying 2D laryngoscopy images into binary form thorough two machine learning methods, namely Deep Convolutional Neural Network (DCNN) and Unsupervised Deep Embedding Clustering (DEC), each one at a time. Following institutional approval, CE-NBI dataset was acquired at University Hospital Magdeburg for training, validating and testing the potential models at each of the intended approaches. Up to our knowledge, this work is first investigation of its kind that comprises DCNN and DEC algorithms on CE-NBI images.

The remained sections were structured as follows: Chapter 2 includes detailed description of proposed approaches and their corresponded implementations. Chapter 3 provides results of each contribution of this work. Chapter 4 highlights notes and observations regarding each approach, and includes general inferences upon the results. Chapter 5 concludes the results in addition to brief suggestions for future work.

2

Methods

This chapter describes the dataset this work relays on, followed an explanation to the chosen methods, in a detailed way. Each method represents different machine learning approach. First approach includes an implementation of ResNet50 [43] DCNN as binary classifier. The second approach is about examining the ability of Unsupervised Deep Embedding Clustering (DEC) algorithm in inferencing clusters from unlabeled examples. In addition, this chapter states the metrics on which assessing different models was concluded.

In doing so, Keras API of TensorFlow [44] framework was used for coding different models seeking rapid prototyping in reasonable time span.

The main objectives of this work:

1. Implementing ResNet50 DCNN with fine-tuning technique as a classifier for differentiating benign/malignant 2D images.
2. Implementing DEC model for clustering dataset images into two groups with proper visualizations, beside to DEC variant models.
3. Discussing the results of each method plus setting-up a fair comparison between the two approaches proposed.

2.1 The Dataset

Machine Learning is being described as data-driven approach. Generally, more data for fitting a model leads to better generalization. The dataset this work relied on was prepared at University Hospital Magdeburg. The dataset consists of two subsets of 2D RGB images, both were captured by the same equipment. The equipment used consists of an Evis Exera III video system with a xenon light source, plus an integrated NBI filter (Olympus Medical Systems, Hamburg, Germany), and a rigid 30-degree contact endoscope (Karl Storz, Tuttlingen, Germany). The two subsets will be referred as Dataset A and Dataset B, respectively. The naming given is on chronological-basis. The subsets details as the following:

1. Dataset A:

This dataset comprises **2736** images, divided into 15 classes listed in the next table:

Table 2.1: Dataset A subclasses.

Lesion	Histologic Type	Histologic Type Count	Lesion Count (Binary)
Benign	Amyloidosis	32	1649
	Cyst	215	
	Fibroma	2	
	Granuloma	28	
	Hyperkeratosis	82	
	Hyperplasia	75	
	Nodule	26	
	Papilloma	467	
	Polyp	239	
	Reinke's Edema	483	
Malignant	Dysplasia Mild	242	1087
	Dysplasia Moderate	76	
	Dysplasia Severe	110	
	Carcinoma in Situ	311	
	Squamous Cell Carcinoma	348	

2. Dataset B:

This dataset comprises **5445** images, divided into 12 classes listed in the next table:

Table 2.2: Dataset B subclasses.

Lesion	Histologic Type	Histologic Type Count	Lesion Count (Binary)
Benign	Amyloidosis	55	3664
	Cyst	207	
	Granuloma	45	
	Hyperkeratosis	210	
	Hyperplasia	21	
	Papilloma	699	
	Polyp	346	
	Reinke's Edema	2081	
Malignant	Dysplasia Mild	750	1781
	Dysplasia Severe	85	
	Carcinoma in Situ	106	
	Squamous Cell Carcinoma	840	

The total images of Dataset A and Dataset B combined equals to **8181** images. The compiled images will be referred as **Dataset C** throughout this work. Dataset C comprise **5313** and **2868** for class 0 and class 1, respectively.

Fig. 2.1 shows some samples from datasets A and B. Dark structures, relative to the surrounding, represents the vasculature pattern which is what matters in differentiating tumor class, being either benign (0) or malignant (1) for this task. Magnitude of the irregularity is a main criteria for determining which class the image belongs to. In fig. 2.1, images belong to benign class, reinke's edema ((a) left) and cyst ((b) left), have more regular vessel structures, such as cyst case where the vessel are almost straight. On the other side, images on the far left in the figure that belongs to dysplasia severe malignant class, contains vessels of higher irregularities than those are seen in images of benign class. As well, the target structures in some images looked pale (fig.2.1(a) left), this might be due to different magnification settings, or because of the depth a which the microvessels is located, the thing made it difficult to have a clear view.

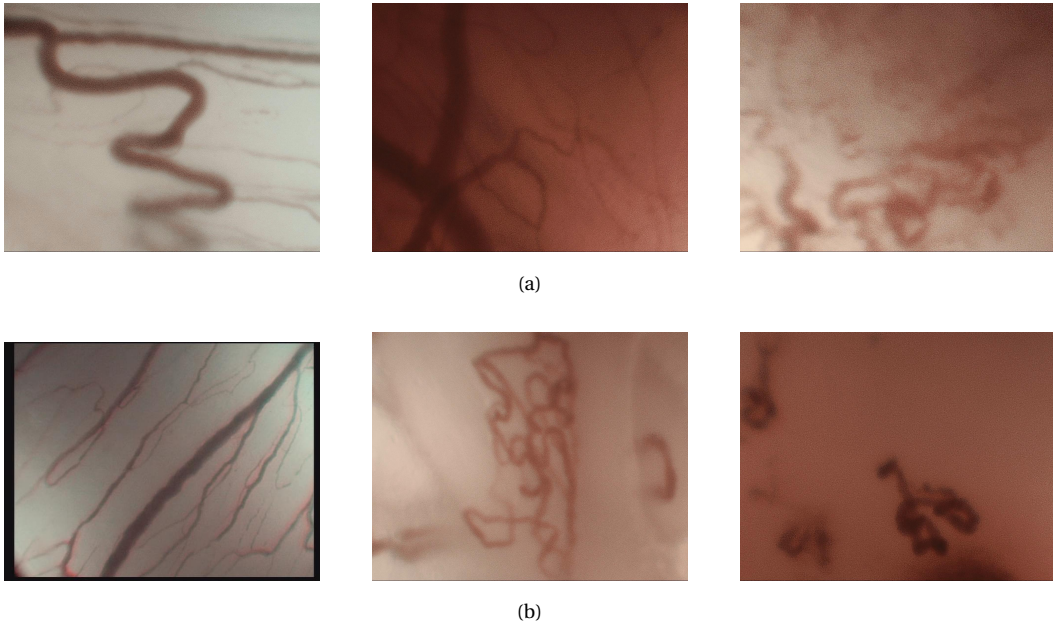


Figure 2.1: CE-NBI images examples of different kinds of laryngeal cancer: (a) samples from Dataset A. Reinke's edema (left), Hyperkeratosis (middle), and Dysplasia severe (Right). (b) samples from Dataset B. Cyst (left), Papilloma (middle), and Dysplasia Severe (Right) (University hospital Magdeburg).

2.2 ResNet50 DCNN as Binary Classifier

This section explains the method followed for implementing ResNet50 DCNN. Deep learning models tend to have very broad solution space (weights of the network). Therefore, transfer learning and fine-tuning techniques are adapted in order to expedite the time required for having optimal weights ultimately. ResNet50 is pre-trained model on ImageNet [45] database, which comprises a thousand different classes. None of them is close to the scope of this application.

Residual Networks (ResNets) consider as an example of very deep classic structures in computer vision literature. ResNet50, by name, has fifty layers. The basic component called "Convolutional Block" which ensembles a 2D convolutional layer, Batch Normalization plus a Rectified Linear Unit (ReLU) [46] layers. Another essential entity of Residual Networks is the "Identity Block" which forwards layer's output by skip connection technique. The way ResNets are designed is to solve the "vanishing gradient" problem in DCNN networks, by that the gradients are propagated all way from last layer to the input one for optimization. Batch Normalization is another interesting property of ResNets. It speeds up the convergence - and so reducing training epochs required-, it has a regularization effect during training phase as well. That is done by reducing the negative effect of internal covariance shift occurs between network layers [47]. Fig. 2.2 shows the key elements of ResNets networks.

ResNets convolutional networks has had a success in similar applications the in medical field. ResNet34 was evaluated to determine the class of laryngeal Stimulated Raman Scattering (SRS) images if the input is normal or neoplastic. ResNets showed rapid and automated recognition on the validation set, with accuracy approximate to 95.9% [48]. Also, study [49] used fine-tuned ResNet50 network for classifying multimodal images of breast issues, into three labels: normal, fat, and cancerous. Using leave-one-patient-out cross-validation, the model achieved mean sensitivity on the validation images equals 86.23%. In addition to that, [50] selected three popular pretrained models for fine-tuning, in order to multiclassify laryngoscopic frames into informative, blurry, containing saliva/specularities, and underexposed classes. Resnet50, Inception V2 and SqueezeNet fine-tuned models achieved macro-average AUC equivalent to 0.998, 0.989, 0.999 respectively. This near-perfect performance hints to the large potentials in convolutional neural networks pretrained models. For that, ResNet50 was considered a good candidate for this work's application. Listing A.1 in the appendix illustrates ResNet50 structure.

Typical deep convolutional neural network consists of two parts: feature extraction part and the classifier. Feature extraction part is basically for capturing

different features through out the image grid by series of filters. Different sized filters are applied through the convolution process at each convolution block of the network. The higher (deeper) on feature extraction part, the more abstract representation of the input. As shown in fig. 2.3, an image excites specific number of neurons (i.e. feature maps) in order to feed the features extracted in the classifier. The classifier, in its turn, converts the features to a vector for making the decision to which class the input belongs to. For that, the classifier usually contains fully connected layers and an output single-neuron layer.

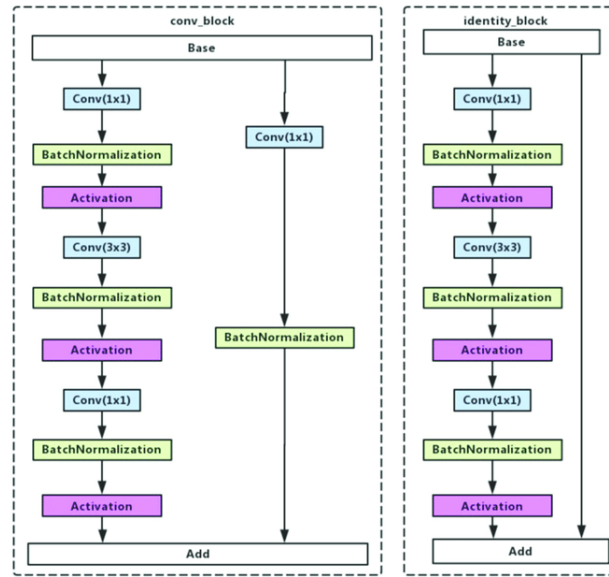


Figure 2.2: Key elements of ResNet's structure [51].

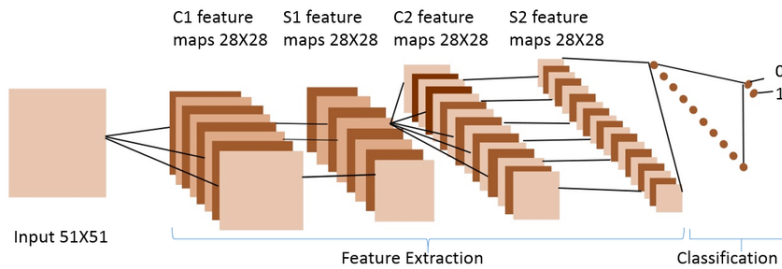


Figure 2.3: The basic parts of Deep Convolutional Neural Network: feature extraction and the classifier [52].

Transfer Learning is a popular technique for training neural networks, where a well-vetted structure had been trained on huge dataset before, therefore, its weights got adjusted optimally according to classes of initial dataset. Then, this

pretrained model can be used to leverage classifying other datasets that are much smaller in size. For this work, ResNet50 model was adapted in doing so. Having a small dataset, speeding-up model training, and seeking better generalization are the reasons to adopt Transfer Learning technique in the current case. In that, there are three ways available to treat the weights of the pretrained model along this work: freeze all the weights, or fine tune portion of them, or fine tune all of them by the available images. The first option is not efficient since the nature of images and their respective classes of this work are very different from those of original dataset (ImageNet). Choosing between the second and the third option was not easy, but since fine-tuning process implies insignificant change in updating the weights at lower layers according to [53], then fine-tuning all layers of the pretrained model was the proper option for implementing different DCNN models through out this task.

The overfitting phenomenon is a major concern in neural networks implementations, especially in case of large structures like ResNet50. Large capacity networks, in terms of number of parameters, are more susceptible to get overfit. The degree of such thing differs depending on number of classes to get recognised by the network. Considering the goal of this task is designing a binary classifier, part of the imported pretrained model was discarded by defining a cut-off layer. The cut-off layer is the last layer in feature extraction part of the network, where the classifier part begins. This layer tends to be where the activation occurs. To that, picking what is believed the right layer was done empirically through running many models with different layers count, by different cut-off layers. In other words, different capacity models were implemented for having sufficient amount of features in a trade off between the training stage success and generalization ability of the model on unseen images.

Selecting the cut-off layer is one hyperparameter among many hyperparameters to care about during developing deep convolutional model, recalling that the hyperparameter is a value has to be defined by the designer of the model before starting the training process. The Learning Rate, input image size, the optimizer, the loss function, performance metrics are typical examples. Some of these parameters are fixed a long this work' experiments, particularly input image size, the optimizer, and the loss function. The input size was either 224 X 224 pixels or 284 X 224 pixels. Both options match the input size requirement of ResNet50. The latter size was adopted to keep the ratio of original image size. For the optimizer, Stochastic Gradient Descent (SGD) was selected. Binary Cross Entropy loss function was used as this task is being a binary classification one.

In addition to the previous aspects, getting deterministic random image generation to the input layer of the model was taken into account for generating re-

producible implementation as well as standardising the results across different models. Also, a callback function was created for terminating model training by itself when model performance on the validation set has not been improving. This technique is called early stopping. This function enlists defining Patience value, where zero value of epochs makes the action of stopping happens instantly right after the performance on the the validation set starts declining. Otherwise, this function gives the model more time (epochs) for getting back with better results. The caveat here is that too much training (high Patience value) would lead to overfitting. In addition to early stopping callback function, two callbacks were added in order to track the performance on real-time fashion and to check network input visually. Another callback function was set up for saving the best model based on accuracy metric.

The current supervised learning approach comprises three experiments. The following experiments are chronologically ordered.

1. Experiment one for building a binary classification model for dataset (A) by applying 5-cross-validation technique for the evaluation. This initial experiment includes converts the classes of images into binary ones. shuffling the dataset images and following that the training process begins until early stopping function triggers. For the pre-processing. pixels values were normalised in range (0-1), only.
2. Experiment two for building a binary classification model for dataset (C). This experiment procedure is similar to one of experiment number one, adding to it a testing stage as a second method for evaluating the model on unseen images, beside the 5-cross-validation technique. This is done by taking out portion of the dataset, after shuffling its images. This portion acted as testing data, for which counts for 1636 images.
3. The third experiment was done after the midterm presentation. The training images had been augmented, for enlarging the size of the dataset. This was done to reduce any effect of underlying overfitting, if exists. Other aspects of this implementation are same as experiment number two.

Other aspects of DCNN models design (i.e hyperparamters) are different from model to another, so it was included in the experimental evaluation chapter, and was illustrated along with each model's results.

For this approach, two additional metrics, Sensitivity and Specificity, were added in addition to the Accuracy. They are calculated as per the following formulas:

$$Sensitivity = \frac{TruePositives}{TruePositives + FalseNegatives} \quad (2.1)$$

$$Specificity = \frac{TrueNegatives}{TrueNegatives + FalsePositives} \quad (2.2)$$

2.3 Deep Embedding Clustering for Binary Classification

The unique kind of images of the dataset was an urging factor for investigating more about Machine Learning methods, in particular the unsupervised learning approach. K-means is considered the most common unsupervised method, though it is known for owning compromised performance on high dimensional inputs such as images [54]. Therefore, K-means can be considered as a baseline for any performance scale. Among many methods, this work focused on implementing Unsupervised Deep Embedding for Clustering Analysis (DEC) algorithm, which has had a very good impact in the literature.

In the sense of DEC method, this approach has been extended to two more variant methods to DEC as explained later in this section. Overall, this approach falls in three experiments where all of them share same goal, that is clustering the input images in two clusters (binary). The experiments intended are:

1. Experiment one stands for implementing DEC method. Dataset C images had been resized 96 X 96 pixels 2D-RGB images at model input side.
2. The second experiment was about investigating similar yet simple crafted structure to DEC. This implementation was different about two points, specifically regarding the layer-wise technique plus having batch normalization layers in the structure. These points will be clarified later in this section.
3. The third experiment involved using features which were extracted by deep convolutional neural network. Thereafter, the features was fed in the DEC structure for clustering the collected features rather than images, the way it is done in the previous experiments.

2.3.1 Unsupervised Deep Embedding for Clustering Analysis (DEC)

This method represents joining a data driven approach with clustering objective. Therefor, its structure contains two parts, an Autoencoder which is considered a data driven model as well as a clustering layer which is, by name, for performing clustering process for its inputs based on the similarity. The inputs of the clustering layer can be raw pixels, or features in a feature space. Selecting the feature space is pivotal for clustering outcome. This method's implementation has two phases: Initialization phase and Optimization phase in order, as shown in fig. 2.4, where the upper drawing depicts the Initialization phase, and the lower one depicts the Optimization phase.

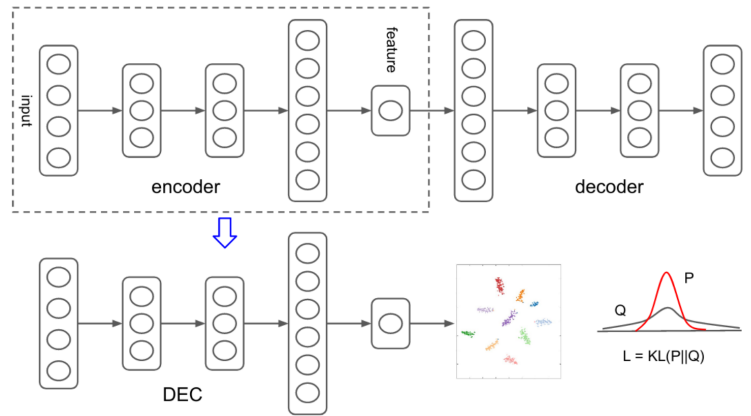


Figure 2.4: Unsupervised Deep Embedding for Clustering Analysis (DEC) scheme [35].

The implementation begins with the Initialization phase, which involves using a deep autoencoder. A typical multi-layer autoencoder consists of an encoder, a decoder and a bottleneck layer in the middle. The bottleneck is the feature space Z that holds latent features extracted through out the encoding process. On the other side, a reconstructed version of each input image is rendered across the decoder based on the learned features. By that, the autoencoder has a reverse symmetry on its sides. DEC for instance, has d -500-500-2000-10 fully connected layers for the encoder part, where d is the data-space dimension, 10 points to the size of feature space (inspired by [55]). The decoder has same characteristics in reversed layer arrangement. The stacked autoencoder network, being a form of neural network, is capable for non-linear mapping the input images into lower-dimensional feature space.

Additional distinct technique in the intializaton phase of this experiment, is the layer-wise training technique. That's done by initialising the stacked autoencoder layer by layer with each layer being a denoising autoencoder trained to re-

construct previous layer's output after random corruption, by Dropout [56] acting as noise source [57]. Layer-wise is beneficial for robust learning to the hidden layers in addition to reducing the risk of overfitting. After that, all the layers are concatenated to form stacked autoencoder for fine-tuning. Both layer-wise and fine-tuning executed for predefined number of iterations. Mean Squared Error (MSE) was the loss function in all stages of this phase.

After running the initialization phase for number of iterations, the optimization phase begins. In this phase, the clustering objective happens in the emerged feature space. This phase includes a clustering layer that computes a soft assignment between embedded representations and the cluster centroids ($n \times 2$ matrix, n is number of images, 2 is number of clusters). Student's t -distribution was used for computing the soft assignments. Furthermore, the Kullback–Leibler (KL) loss function was adapted for the divergence between the soft assignments (Q) and an auxiliary target distribution (P), As shown in down right corner of fig 2.4. The target distribution is based on high confidence prediction, in consequence, the clustering purity is getting improved across the iterations of this phase.

When running the DEC model, greedy layer-wise training initializes with random weights drawn from zero-mean Gaussian distribution with a standard deviation of 0.01. Each layer is pretrained for 50000 iterations with a dropout rate of 20%. Then, The entire deep autoencoder is further finetuned for 100000 iterations without dropout. For both layer-wise pretraining and end-to-end finetuning of the autoencoder the minibatch size is set to 256, starting stochastic gradient Descent (SGD) learning rate is set to 0.1, which is divided by 10 every 20000 iterations, and weight decay is set to 0 [35]. During this, the reconstruction error is getting minimized by MSE as loss function. At the end of the finetuning, the decoder part is discarded and the clustering layer is cascaded on the top, as shown on the lower drawing in figure 2.4. The resulted embedded values that the encoder holds are used for initializing clusters centroids. Afterwards, in iterative way, the clusters are getting refined from current high confidence predictions by KL divergence function with 0.01 learning rate. This process is repeated until a convergence criterion is met, that of a value equals to 0.1% enhancement threshold of the loss.

The proposed method in the published DEC paper achieved state-of-art results on popular dataset, as described in table 2.3. MNIST and STL-HOG are image datasets. REUTERS is a textual dataset. The summary of implemented model illustrated in fig. 2.5 and fig.2.6.

Table 2.3: Comparison of clustering accuracy on four datasets [35]. LDMGI: Local Discriminant Models and Global Integration. SEC: Spectral Embedded Clustering. DEC: Deep Embedding for Clustering Analysis.

Method	MNIST	STL-HOG	REUTERS-10k	REUTERS
k-means	53.49%	28.39%	52.42%	53.29%
LDMGI	84.09%	33.08%	43.84%	N/A
SEC	80.37%	30.75%	60.08%	N/A
DEC	84.30%	35.90%	72.17%	75.63%

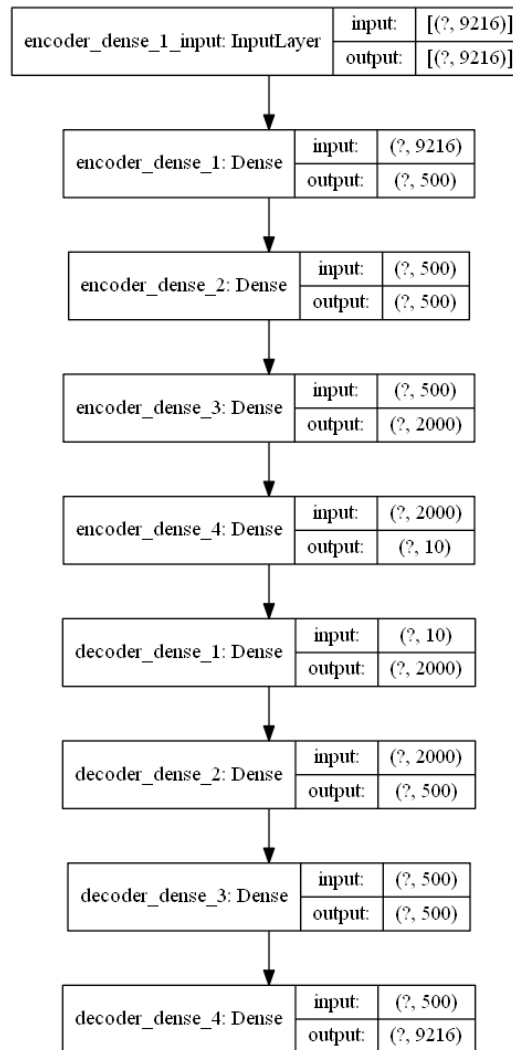


Figure 2.5: Autoencoder structure of initialisation phase in Deep embedding Clustering method.

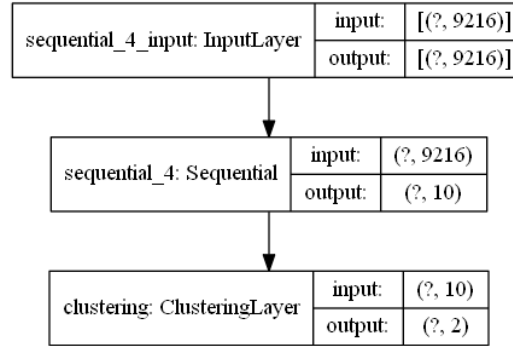


Figure 2.6: Structure of Deep Embedding Clustering model at the optimisation phase.

2.3.2 Plain Variant of DEC

In this experiment, a simple version of DEC algorithm had been coded. This model has same autoencoder dimensions, number of iterations, batch size, learning rate, and clustering layer. It is different from DEC of the first experiment about two things. First, a Batch Normalization layer had been inserted between each two layers. Second, autoencoder layer-wise training technique was removed from this experiment's implementation. As stated earlier in section 2.2, batch normalization accelerates model convergence. The model summary is illustrated in fig. A.1.

2.3.3 DEC based on Deep-Convolutional-Extracted Features

In this trial, DCNN model was introduced for feature extraction. In this way, each image was transformed to 4096 feature vector by DCNN network. VGG16 model [58] was chosen for that. This 16-layer CNN consists of 5 convolutional blocks of 3 X 3 filter size with stride 1. Also, All blocks have maxpool layer of 2 X 2 with stride equals to 2. VGG16 won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 competition. Figure 2.7 clarifies the structure of VGG16.

The autoencoder in this model did train on extracted features, by end-to-end instead of layer-wise training. Adam optimizer was picked with 0.001 learning rate. MSE loss function was used as the usual case in the previous experiments. All Other aspects of this implementation are fixed for matching the previous trials.

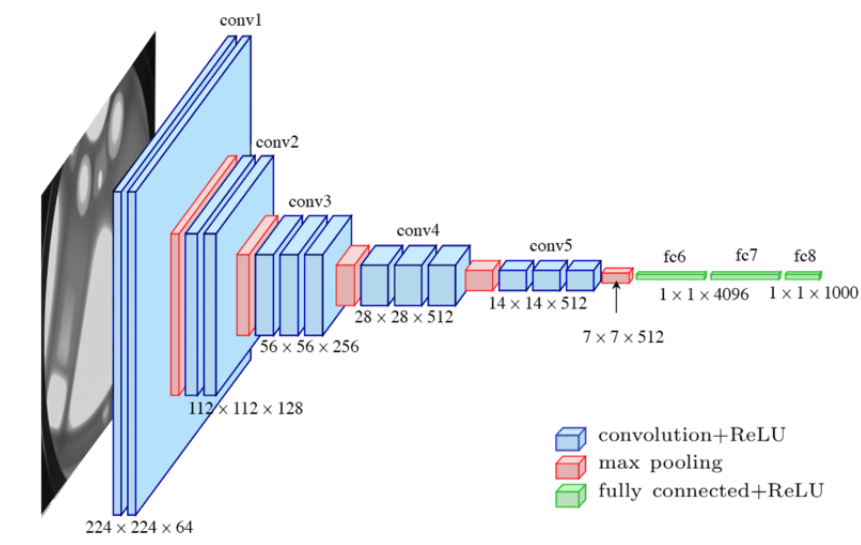


Figure 2.7: The standard VGG16 network architecture as proposed in [58]. Courtesy of [59].

Parameter optimisation process in the above methods happen through two loss functions. Mean Square Error (MSE) is the loss function of the neural network part, as follows:

$$MSE = \left(\frac{1}{n}\right) \sum_{i=1}^n (y_i - x_i)^2 \quad (2.3)$$

Having a low value of MSE loss is critical during the initialisation phase, as it indicates to the amount of error between the pixels of the input image (X) and their respective pixels in the reconstructed image (Y).

In order to refine the clusters in Z space, Kullback-Leibler (KL) divergence loss was defined between the soft assignments q_{ij} and auxiliary target distribution P_i , as follows:

$$KL(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (2.4)$$

3

Experimental Evaluation

In this chapter, results of each approach are illustrated. Each approach comprised experiments as described in Methods chapter. Each experiment had many models as well. Due to high number of models that had been run, this section includes the significant models and related implications to the performance on the classification task. This chapter follows the flow of Methods chapter, where first approach (i.e. DCNN approach) included three experiments. The second approach had three experiments too.

3.1 ResNet50 DCNN as Binary Classifier

Seeking a good DCNN model depends on several criteria. First, number of epochs has been required for fitting the data. By transfer learning, models tends to get trained much faster than models training from scratch. In addition to that, number of parameters in the DCNN network is another important feature in evaluating different models. Smaller number of parameters points to less of a chance of having an overfitting case. The third criteria is relevant to visualising the network performance, which is the asymptote between model's training and validation accuracy curves. That is if a gap between the two curves exists, it prompts to overfitting condition. We assessed each tested model through these three criteria, taking into account conventional metrics such as accuracy, sensitivity and specificity as usual.

For validating the obtained results, 5-cross-validation technique was adapted to all experiments. In the second and the third experiment, testing on unseen image was added, for checking the generalisation ability of the models.

3.1.1 Binary Classification for Dataset A

The following figure, fig. 3.1 depicts the best results obtained throughout the first experiment of the DCNN approach. For sake of easy tracking, this model labeled as Model 1. Here, the cutoff was done at layer "conv3_block4_2_relu" (Listing A.1 in the appendix). The classifier on the top of it has Global Max Pooling layer, Dropout layer and a single "Sigmoid" output neuron, layer-by-layer. The rate of Dropout is 20%. Other features of this model, the optimizer that had been used was Stochastic Gradient Descent (SGD) with 0.001 learning rate with decay rate equals to $1e-6$. Batch size was equal to 32 images. Early stopping equals 7 epochs. As shown in the figure, the model converged in short time of epochs, the gap between model's performance on training and validation data is very small which demonstrates that Model 1 is well-generalized. The average validation accuracy, sensitivity and specificity over the five folds was 95.85%, 93.52%, 97.49% respectively, as listed in table 3.1.

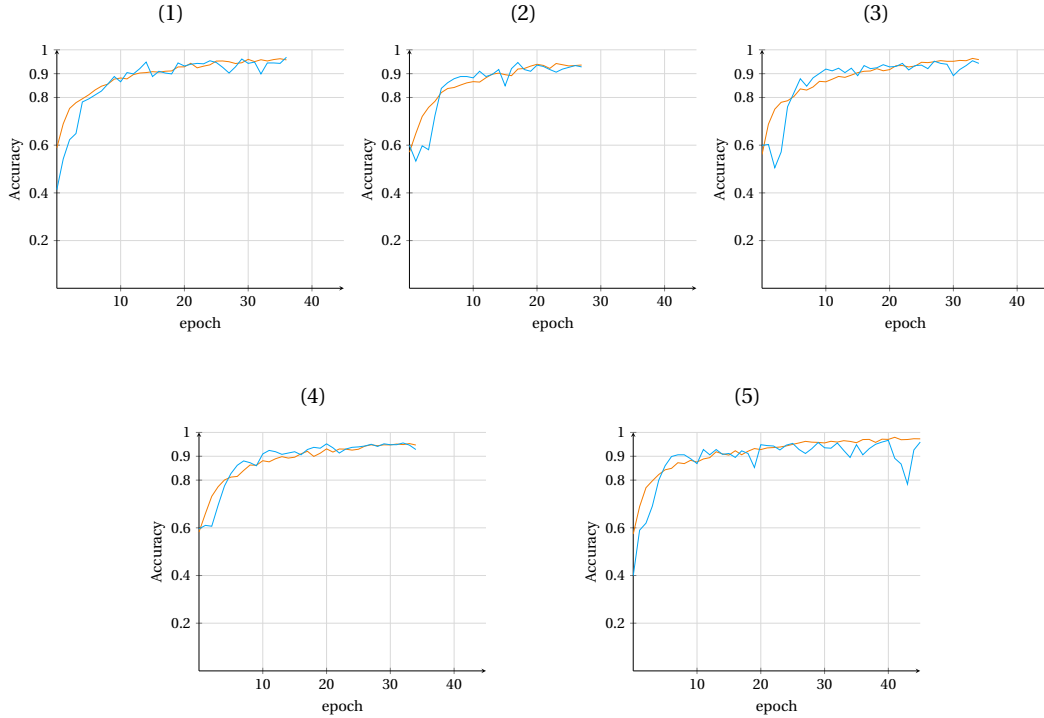


Figure 3.1: Model 1 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

The next model, Model 2, had a good performance on the validation phase. This model is different than Model 1 about including Dropout layer in the classifier part of the network. Other than that, the two implementations are identical. As appeared in fig. 3.2, the model was being successful in classifying validation images. Yet, There is a continuous gap between the accuracy curves along training and validation phases. This gap is a sign of overfitting. Actually, Model 1 was executed after Model 2 for getting the curves crossed at a point. As a result of using Dropout layer in Model 1, the model had better asymptote between training and validation curves in all folds comparing to what Model 2. Conversely, Model 2 had an average accuracy equals to 96.54% which slightly better than Model 1. That being mention, the primary concern in DCNN models is the generalization. As a result, the previous model was considered more favourable in that matter. Table 3.1 lists the other performance metrics of Model 2.

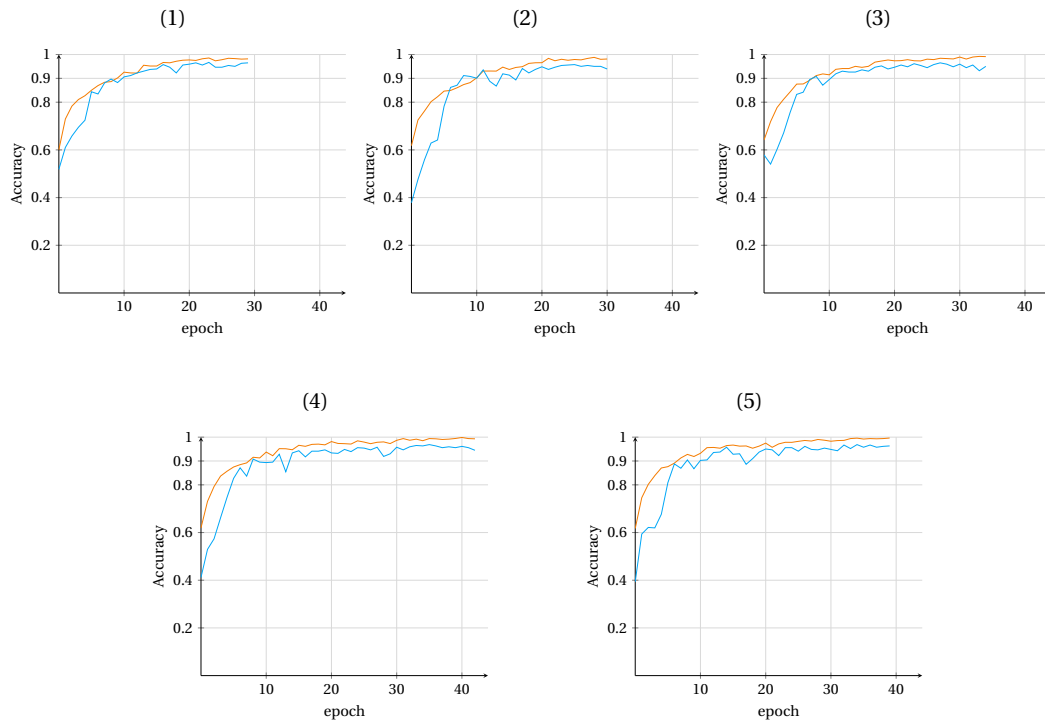


Figure 3.2: Model 2 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

The following case is a typical example about setting-up a wrong classifier and its negative effect on model's performance. This model, Model 3 in order, owns likewise network of the previous models, except one layer in the classifier. Global Max Pooling layer that has been the first layer of the classifier in the previous models, was replaced by Average Pooling layer.

This small modification in only one layer had a negative effect on the ability for classifying the images correctly. This model suffered from severe overfitting very early, shown in fig. 3.3. Obviously across the five folds, the model was learning the correct classes on the training images (the yellow lines), but it fails to hold with that at the validation. This created the gap between the curves repetitively over the 5-folds test. Furthermore, the model could not survive early stopping technique, and for that model runs were terminated by it, when no significant improvement on model's performance occurs during the validation process.

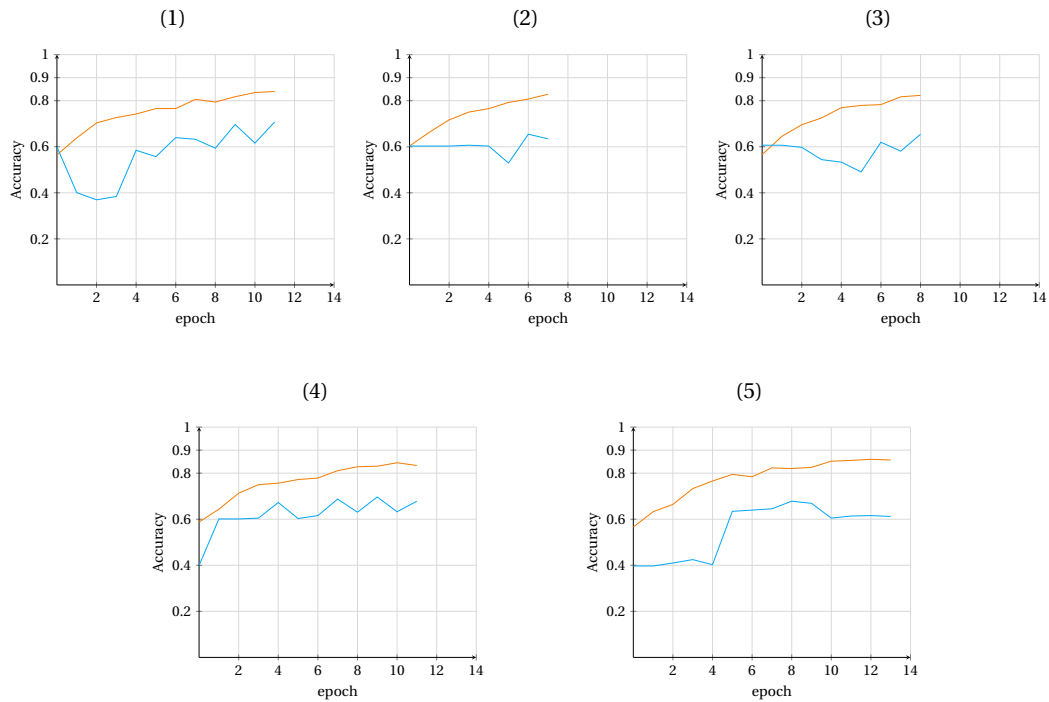


Figure 3.3: Model 3 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

Selecting the right cut-off layer thorough ResNet50 network is essential for getting sensible performance. For instance, the next model, Model 4, hit high metric values of registered accuracy, sensitivity and specificity which is equal to 94.3%, 92.03%, 95.88%, respectively. However, the blue lines in fig. 3.4 have fluctuations for this model, unlike those of model 1 and model 2 which were smooth and aligned.

To compare Model 1 and this model by network structure, at this model the cut-off is made at very shallow layer, layer "conv2_block1_out", while the cut-off made at layer "conv3_block4_2_relu" in Model 1 case, which is at a deeper level through ResNet50 network. Furthermore, the classifier of this model consists of Global Max Poling layer, two dense layers with Dropout rate equals to 40% in order, whereas model 1's classifier consists of one Global Max Pooling layer with 20% Dropout rate prior to the output neuron. For the remained features, the two models share same features in terms of optimizer, learning rate and so on.

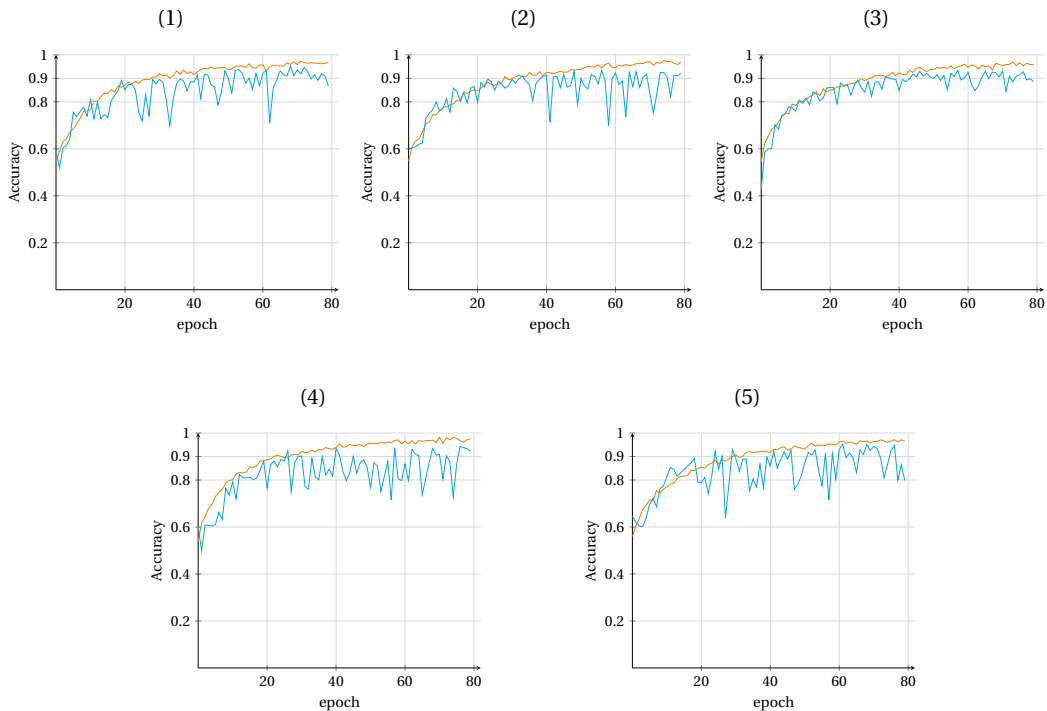


Figure 3.4: Model 4 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

Back to fig. 3.4 in which unsteady asymptotic pattern between yellow and blue curves is exhibited. This kind of pattern signals that the network is not capable on classifying images of the validation set. Because the cut-off was made at shallow layer, the properties of the extracted feature are not sufficient for having

confident model about the predicted classes. In other words, the prediction process is more of guessing than being a real prediction process. To that, this model's performance was not consistent during the validation phase, it was consistent on the training set, though.

Regarding the difference in classifier structure between Model 1 and Model 4, it is not plausible to be the reason. First, both classifiers include Global Max Pooling layer to be the first layer. The first layer in any classifier is critical since it draws the perception to the classifier about the extraction feature. Second, by many observations a long this work, dense layers slow down the convergence. As shown in this model where the run time spans two to three folds number of epochs that were required comparing to Model 1 or Model 2. If dense layers have any advantage in the current case, Model 1 and Model 2 could not have achieved very good results, as explained earlier.

Table 3.1: Detailed results of experiment number one of DCNN approach. Numbers are averages at the peak accuracy over 5-folds cross validation.

Model	Accuracy(%)	Sensitivity(%)	Specificity(%)	Loss
Model 1	95.85	93.52	97.49	0.14
Model 2	96.54	94.20	97.97	0.1
Model 3	67.83	39.55	86.03	0.75
Model 4	94.30	92.03	95.88	0.16

3.1.2 Binary Classification for Dataset C

This experiment was initiated after dataset B had been provided. Dataset B is two times bigger in size than dataset A. The purpose of this experiment is to find a model that can be successful on classifying dataset C, especially that Model 1 from the previous experiment was not that successful on dataset A as quite much as on dataset B. A couple of reasons could lead to this: either Model 1 is overfitting to dataset A, or dataset B includes images that require new features Model 1 could not obtain, simply as the dataset A is small (and smaller than dataset B). It is possible to be both reasons too.

For that, dataset A and B were merged and shuffled in order to take the advantage of having a bigger dataset in size at this data driven approach. Taking into account having an overfit case as explained in the previous paragraph, this experiment's endeavor was to reduce the number of model parameters by truncating Resnet50 network at shallower layer than before. Upon the observations of the first experiment, finding the right cut-off layer is the first step. Then setting-up a classifier on the top to be the second step.

In this experiment, 20% of dataset C images was taken out for model testing beside running the 5-folds cross validation on the remained 80% of total images. The testing phase is an important addition for checking model ability to generalize on unseen images.

Fig. 3.5 depicts classification results of Model 5. For this model, the cut-off layer is made at "conv2_block3_out" layer. About the classifier, it is made up of two layers, Global Max Pooling and single output neuron. Following the settings of the previous experiment, this network had Stochastic Gradient Descent(SGD) as optimizer with 0.001 learning rate, model training gets terminated by early stopping function, and Batch Size equals to 32 images. Binary cross entropy was the loss function.

Model 5 achieved accuracy, sensitivity, and specificity over the 5-folds cross validation equal to 97.95%, 96.77%, 98.61% , respectively. By looking on fig. 3.5, 90% accuracy rate achieved within the first 5 epochs, indicating that the convergence to the optimal point on solution space was done in short training time. In addition, training and validation curves are aligned in a good approximate at every fold. Moreover, Model 5 achieved 99.12% accuracy rate on the testing images. By that, Model 5 can be considered a good classifier on dataset C. In fact, it is the best among different models that are going to be explained afterwords.

The results of Model 5 have demonstrated that the delivered features from feature extraction part of the network into the classifier are sufficient. Otherwise,

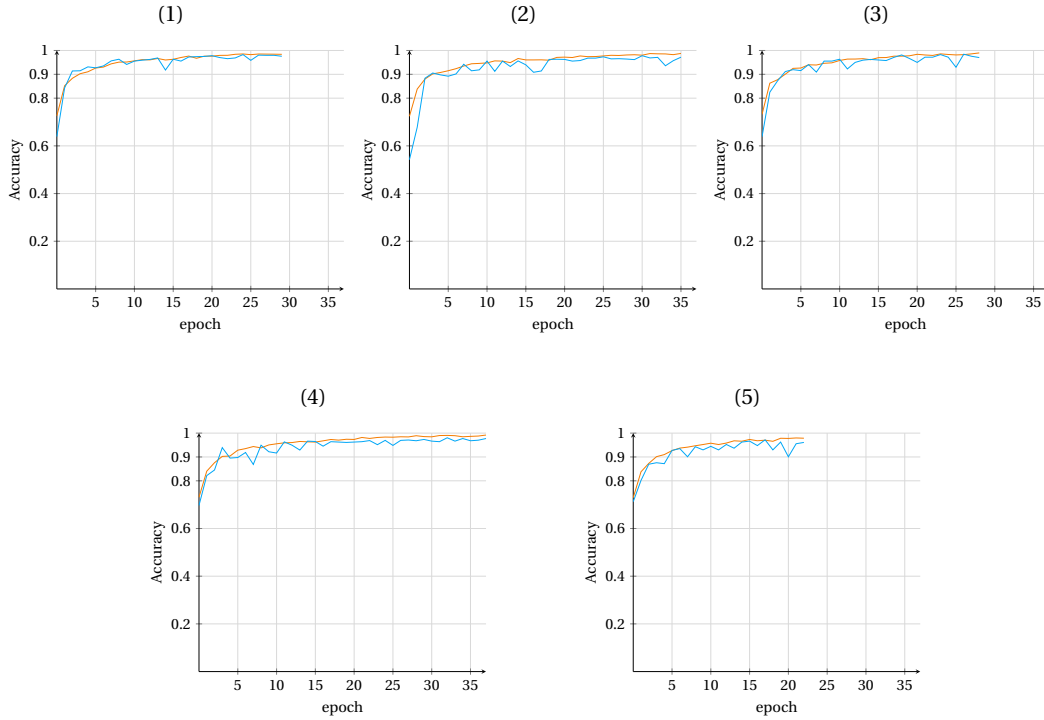


Figure 3.5: Model 5 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

blue curves would be rippling as the case of Model 4. Then the next question is: How much the classifier of Model 5 is well designed? The following models will illustrate that in addition to testing a common classifier on dataset C.

The next model, Model 6, with exact network structure of Model 5, but one dropout layer was created between Global Max Pooling and the output layer. The dropout value was equal to 10%. This layer, by name, will omit 10% of the collected features. Every other aspect of this model is identical to Model 5.

The dropping-out action resulted an underfitting model learning. When a CNN model is performing better on the validation phase than the training phase, this is strong sign of having underfitting case (i.e lack of features). Visually as seen in fig. 3.6, Model 6 is underfitting, specifically on the first three folds. That because the validation curves are having higher plateaus than the training curves. Additionally, the blue curves are not regular on epoch course, particularly in folds 1, 4 and 5.

In consequence of that, metrics achieved by this model had decreased significantly. For example, Model 6 accuracy is less than Model 5 by more than 3%, and same observation applies on other performance measurements (see Table 3.2). It is worth mentioning that the average minimal losses value has increased to 0.15

against 0.06 that of Model 5 over the five folds. About classification accuracy of the testing phase, this model achieved 95.88% comparing to 99.12% for Model 5. Upon that, Model 5 is more capable at the current application by all performance measures.

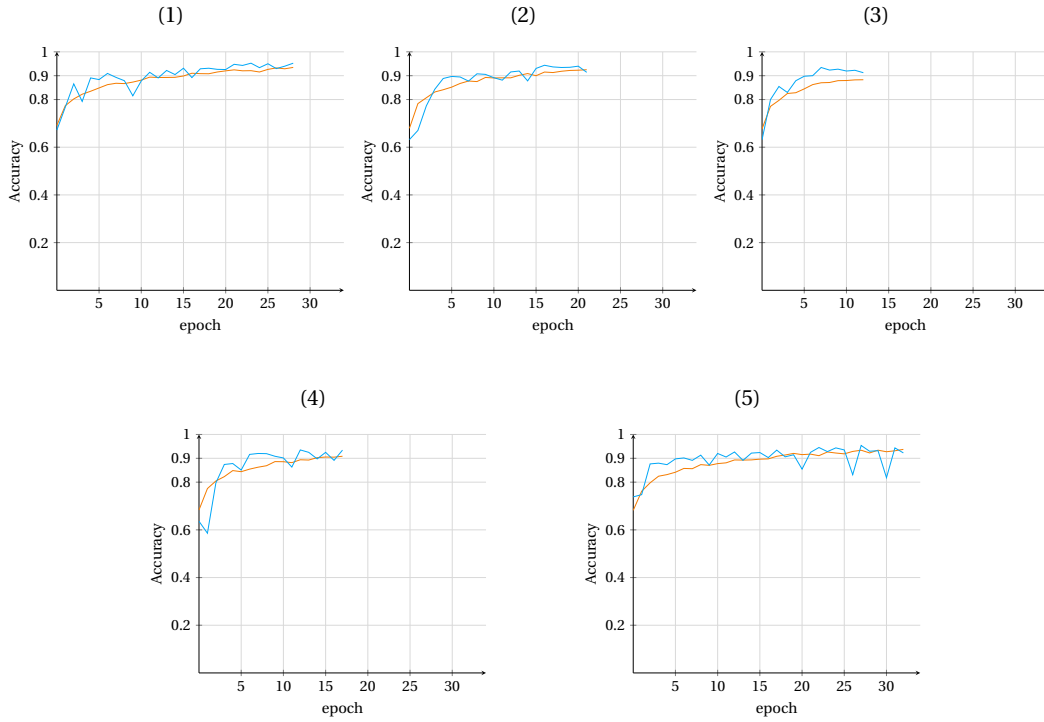


Figure 3.6: Model 6 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

ResNet50 pre-trained model is a very deep DCNN. The full structure of this model was intended for differentiating 1000 classes. ResNet50's feature extractor owns more than 23.5 million parameters. For the following example, along the comparison between Model 5 with different networks, Model 7 is a label for complete ResNet50 model. For comparing Model 5 and model 7 in a fair way, both models share same classifier structure, yet each one has a unique feature extractor. As a rule of thumb in DCNNs, parameter count is relevant to model's capacity to learn. The feature extractor of Model 7, as stated earlier, have millions of parameters (weights), while Model 5 has about 230 thousand parameters. To that theoretically, Model 7 is far more capable in classifying its input.

Surprisingly according to fig. 3.7, classification outcomes of Model 7 were not better than those of Model 5. For instance, Model 7 was getting learned in slower rate than what Model 5 did. This is can be assessed by the slope of a curve.

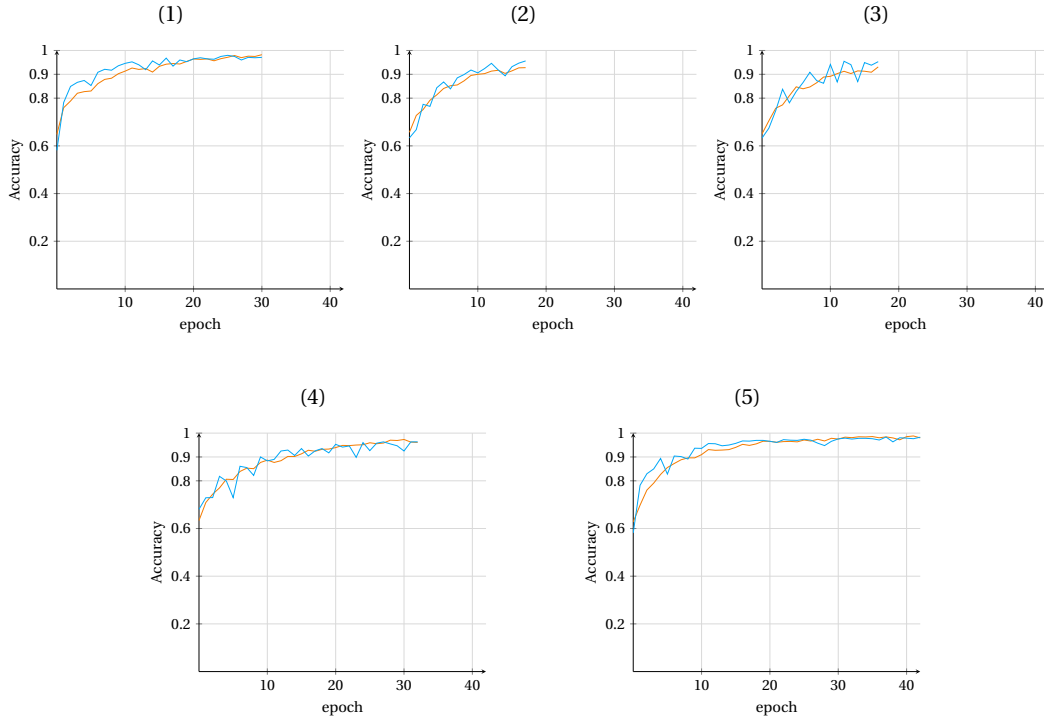


Figure 3.7: Model 7 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

Fig. 3.8 depicts the progression of Model 5 and Model 7 through the training and the validation phases at fold number one (left) and two (right). By looking at fold one's graph, epoch five in particular, the accuracy that was achieved by Model 5 was equal to 92.73% on validation data, while Model 7 had lower rate equals 85.31% for the same epoch. This goes on all folds. Also, the right side graph shows that model 7 got early-stopped at epoch 17. Early stopping function was triggered at loss value equals to 0.19 by this epoch. On the other side, Model 5 continued its training phase up to epoch number 35 with loss value equals 0.08.

To conclude this example, an implementation with the full ResNet50 structure does not bring outcomes in proportion with model's capacity. As shown earlier, it learns slowly, and more significantly, it increases the chance of having an overfitting model. Even visually, it does not progress as nicely as small portion of it does.

Next example is about investigating a classifier structure which is popular in DCNNs' implementations. This classifier involves flatten layer on the feature extractor, followed by two fully connected layers. For that, feature extractor part of Model 5 was attached with this common kind of classifier structure. So, Model 5 and Model 8 have exact feature extractor, but totally different classifiers.

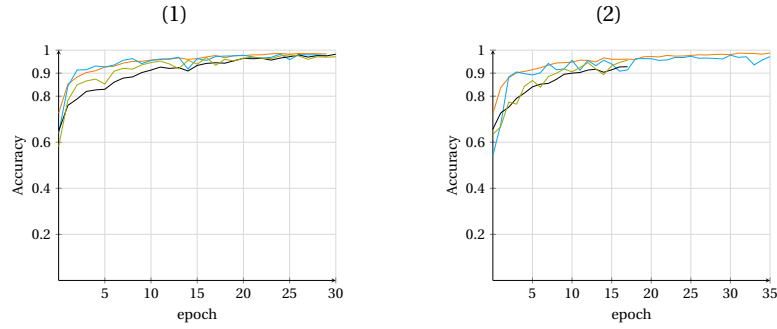


Figure 3.8: Comparison of the accuracy track between Model 5 and Model 7. Subfigure numbering stands for the fold number. Yellow and blue curves represent the training and validation curve of Model 5, respectively. Black and green lines represent the training and validation curve of Model 7, respectively.

The suggested structure of Model 8 did not adhere with transfer learning goals, which "nice asymptote" between the training and validation curves is one of them. As appeared in fig. 3.9, the training process of the model was terminated early without any indication of valid convergence, this has shown clearly on folds four and five. As well, the model is overfitting by the existing gap between training curve and validation curve. At parameters count, Model 8 includes more than 94 million parameters against about 230 thousand parameters for Model 5. The very high number of parameters of Model 8 explains why it began being overfitting since the first epoch on. This observation held true for all folds.

Expressing DCNN performance by numbers without visual revision could be misleading. Taking Model 8 as an example, 85.3% accuracy rate achieved, though it is acting badly on the validation images as explained in the previous paragraph, since this accuracy percentage is an average of the peak accuracies across the cross-validation.

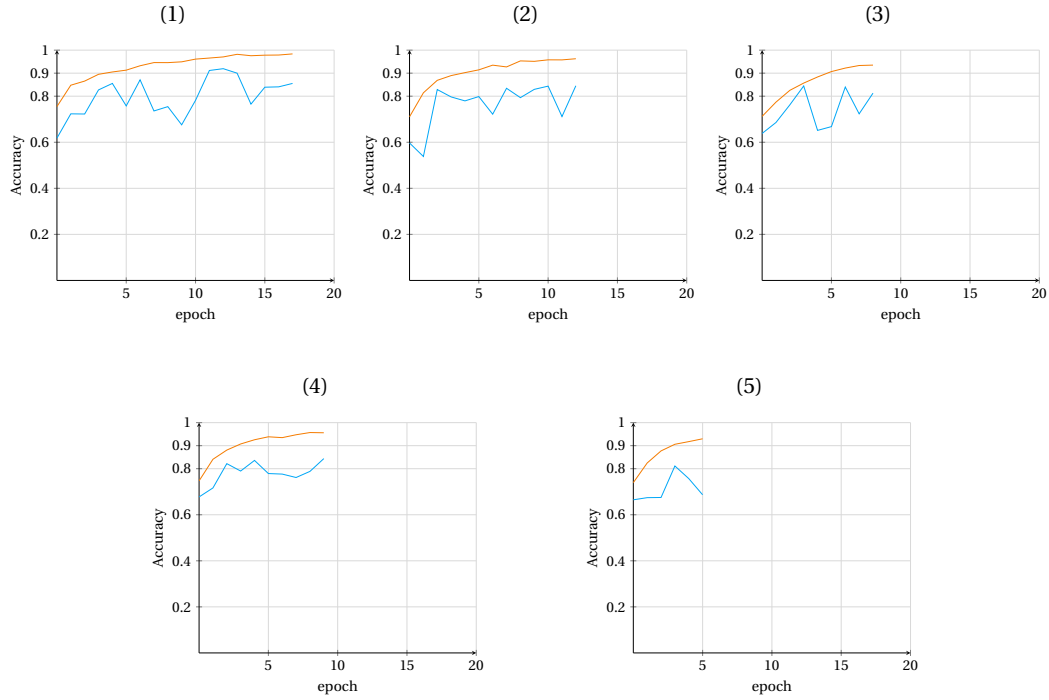


Figure 3.9: Model 8 5-folds cross validation results. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

The last evaluation at this experiment was a rerun of Model 5. This is in purpose of checking whether the model can have parallel performance to the first trial of it. For that purpose, different portion of images was excluded for testing. Other than that, the two runs are identical.

The 5-fold progress of this trial is illustrated in Fig. 3.10. Model's behavior in this case is corresponding to the first run. In both runs, the models have nice alignment between training and validation curves, and high plateau through the cross-validation stage. In numbers, The accuracy of first and second run was equal to 97.95% and 97.19%, respectively. About testing results, which is what matters in this case, the model achieved 99.12% on the first run, and 98.44% on the second one. Hence, model five was deemed to be a generalized DCNN model.

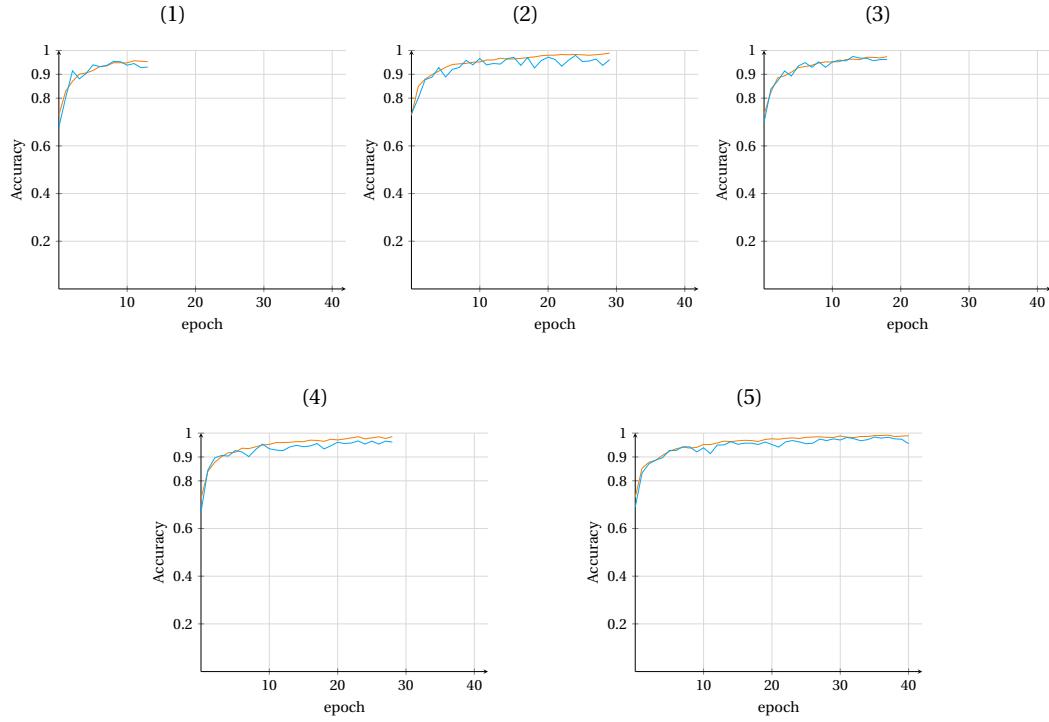


Figure 3.10: Model 5 5-folds cross validation results at the second run. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

Table 3.2: Detailed results of experiment number two of DCNN approach. Numbers are averages at peak accuracy over 5-folds cross validation.

Model	Accuracy(%)	Sensitivity(%)	Specificity(%)	Loss
Model 5	97.95	96.77	98.61	0.06
Model 6	94.39	91.44	95.96	0.15
Model 7	96.75	96.03	97.47	0.11
Model 8	85.30	58.82	94.89	0.45
Model 5 (2nd trial)	97.19	95.48	98.20	0.08

3.1.3 Binary Classification for Augmented Dataset C

Overfitting phenomenon is a major concern in neural networks. Overfit models cannot retain good results on unseen images. In this experiment, model five, the best so far, was put in more challenging situation by adding two things. First, data augmentation technique was included in the implementation. That means challenging the network by altering input images of training phase in different ways so that the network is forced to learn the necessary features. Data augmentation helps to reduce the chance of the model to get overfit. The second thing was, instead of taking out fixed portion of dataset C for testing - the way it was done in the second experiment, 930 images were listed from thirty one patients (at maximum 30 images each). Most of these images are of different views from the training set, for adding more realistic model evaluation.

The experiment comprised three runs, the first one as a reference without any kind of augmentation, the second and the third ones were done on augmented images. Two types of augmentation were applied: horizontal and vertical flipping to the images. By these trials, Model 5 was examined to report its tendency to fail if it was overfitting.

For the first (i.e reference) run in this experiment, Fig. 3.11 depicts that model five acted well on the new image arrangement, that the cross-validation results were similar to results of the previous run in the second experiment. For example, model five achieved 97.68% for this trial, which is a close percentage to the model accuracy in the last experiment (97.95%). About the accuracy of the testing phase, 92.92% accuracy was for this trial against 99.12% for the counterpart trial in experiment number two. The other metrics are listed in table 3.2 and table 3.3 of experiments two and three, respectively.

Overfittig is a term for describing a model when there is a deviation between training and validation results, represented as a gap between accuracy curves of training and validation phases. This time, as no gaps between accuracy curves according to fig. 3.11, the case is different. The deviation occurs between validation and testing accuracies. Number of reasons could be the cause to that. First, images of testing set have different distribution, or because the validation set is not representative to the testing dataset as a second reason, which leads to biased fine-tuned model to the validation set. The first reason is related when a DCNN model is being trained on one dataset, and validated (fine-tuned) on another but similar dataset, for which it has different data distribution. Our implementation does not match this case. On the side, the second reason is more likely to be the cause, since dataset C is relatively small in number of images.

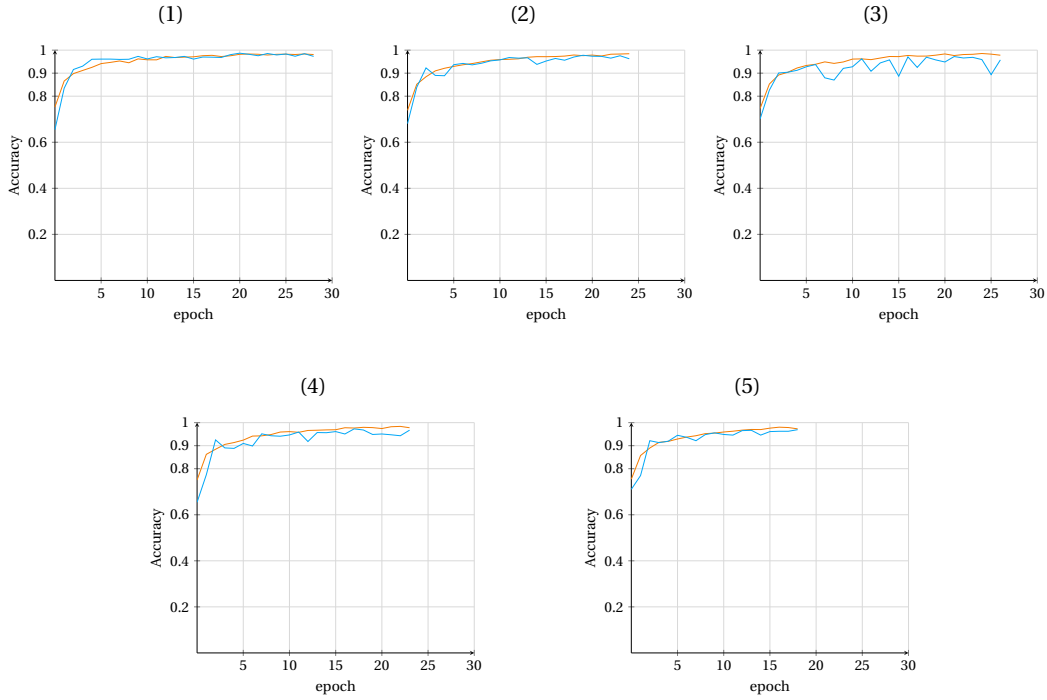


Figure 3.11: 5-folds cross validation results of model five on dataset C without being augmented. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

Even though model five showed no signs to be overfitting, further investigation was done on how much data augmentation can influence model training to recognize the correct classes. For that, data augmentation function was added to model five for making up horizontal and vertical versions to training images. It was noted that the pace of learning slowed down. Obviously at epoch five across the folds, the model could achieve 93.15% average training accuracy in the reference trial against 82.9% on augmented training images, as noticed in fig. 3.12. More significantly, testing accuracy went down from 92.92% to 83.5%. As well, the validation curve was a little more rocky than the one of the reference trial, yet it damps to align the training curve to be in nice asymptote, which is a sign that the model is away from being overfitting or underfitting.

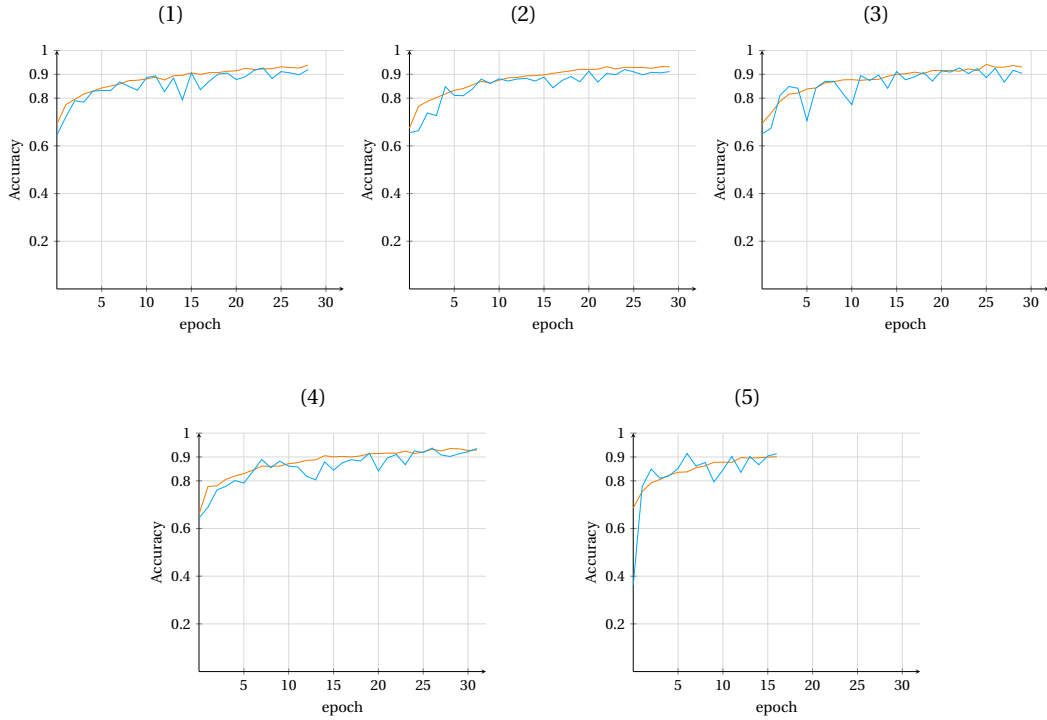


Figure 3.12: 5-folds cross validation results of model 5 on augmented dataset C (horizontal and vertical flipping). Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

Slowing down the training process and making the curves a little less steady in the previous run might have not allowed model five to proceed by the selected parameters of early stopping function. Next run was a try to compensate that by increasing the Patience value of the function to seven epochs rather than five. This in turn will wait for more epochs checking any improvement on the accuracy, through which higher rates could obtain.

By looking on fig. 3.13 about this run's results, it can be seen that model five performance has not changed significantly from the previous run. Average accuracies achieved were 93.33% and 84.08% on validation and testing images, respectively. Additional comments on this observation in the discussion section. The confusion matrix of model five with augmented input data is shown in fig. 3.14. Fig. 3.15 shows samples of images were classified incorrectly.

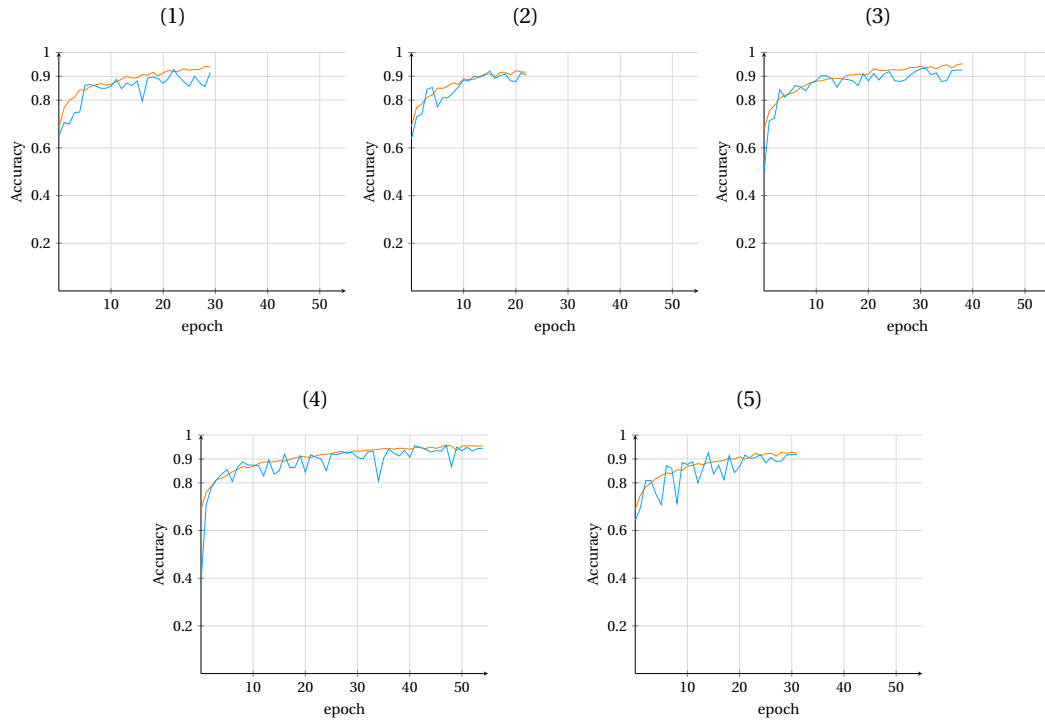


Figure 3.13: 5-folds cross validation results of model 5 on augmented dataset C (horizontal and vertical flipping) with extended early stopping. Subfigure numbering stands for the fold number. Yellow lines represent the training phase. Blue ones for the validation phase.

		Predicted class		Total
		0	1	
Truth class	0	503	38	541
	1	121	241	362
Total		624	279	

Figure 3.14: Confusion matrix of 903 test images by model five being trained on augmented dataset.

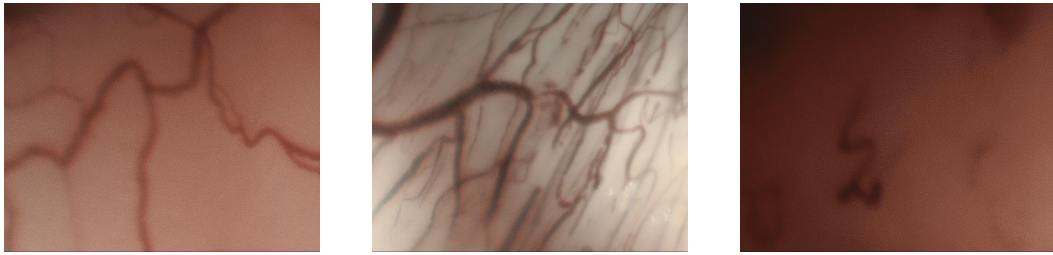


Figure 3.15: Examples of images incorrectly classified by model five. Reink's edema (left). Cyst (Middle). Severe dysplasia (right).

Table 3.3: Detailed results of experiment number three of DCNN approach. Numbers are averages at peak accuracy over 5-folds cross validation.

Model	Accuracy(%)	Sensitivity(%)	Specificity(%)	Loss
Model 5 (as ref.)	97.68	95.85	98.56	0.07
Model 5 w/ Augment.	92.52	88.88	96.00	0.20
Model 5 w/ Augment. (Extended Earl. Stop.)	93.33	88.26	94.89	0.18

3.2 Deep Embedding Clustering for Binary Classification

Three experiments were conducted for investigating the ability of recent deep embedding methods at classifying 2D images. As described earlier in Methods chapter, models' implementation share two phases, initialisation phase and optimisation phase. Results of each model are tracked by phase through this section.

Fig. 3.16 depicts Mean Square Error (MSE) along the initialisation phase of each experiment. Generally in the three experiments, autoencoder was able to reconstruct input images within a few number of epochs by a very small error. However, DEC method had a convergence started at high MSE value (0.24) relative to the case of plain variant DEC, at which the convergence started at lower value (0.0041).

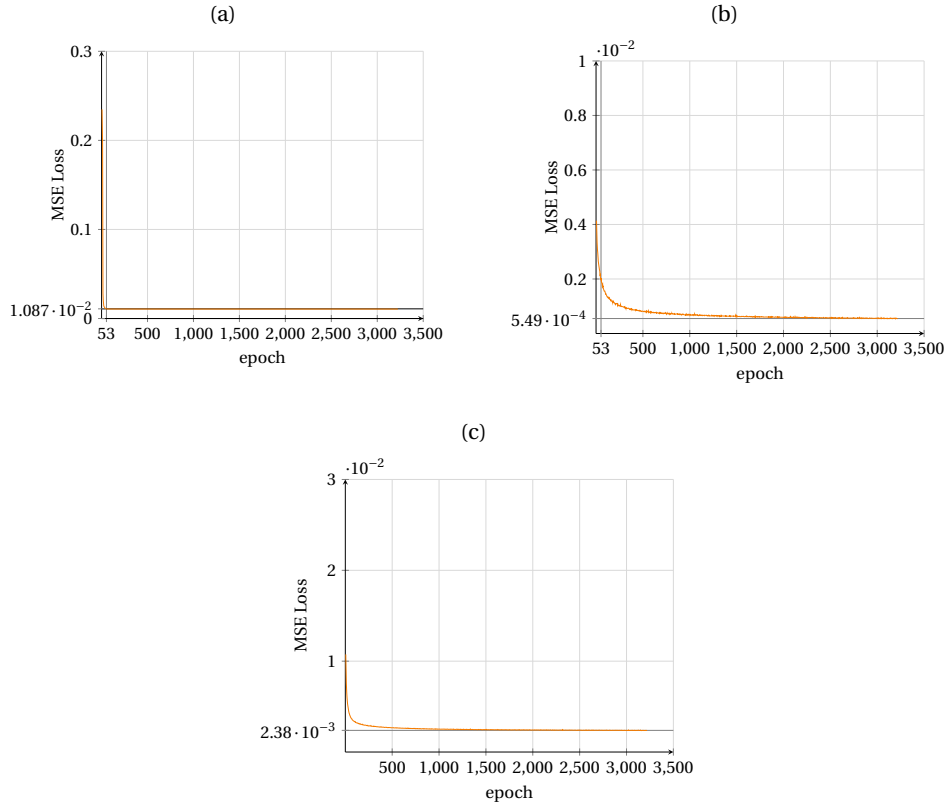


Figure 3.16: Mean Square Error during initialisation phase. (a) Unsupervised Deep Embedding Clustering Analysis(DEC). (b) Plain variant of DEC. (c) DEC based on deep-convolutional extracted features.

In addition, no significant improvement in image reconstruction after epoch 53 for DEC method contrary to the plain version of it, as marked on fig. 3.16 (graphs (a) and (b)). In case of the third experiment, input CNN-extracted fea-

tures were reconstructed successfully by looking at error value as seen in fig.3.16 (c). The number of epochs of the initial phase was prolonged in the second and the third experiment to meet experiment one's implementation in this respect. For example in the third experiment, only ten epochs were enough to have same clustering outcome to that of any longer training period. Fig. 3.17 includes examples of input images and autoencoder output after finishing the initialisation phase.

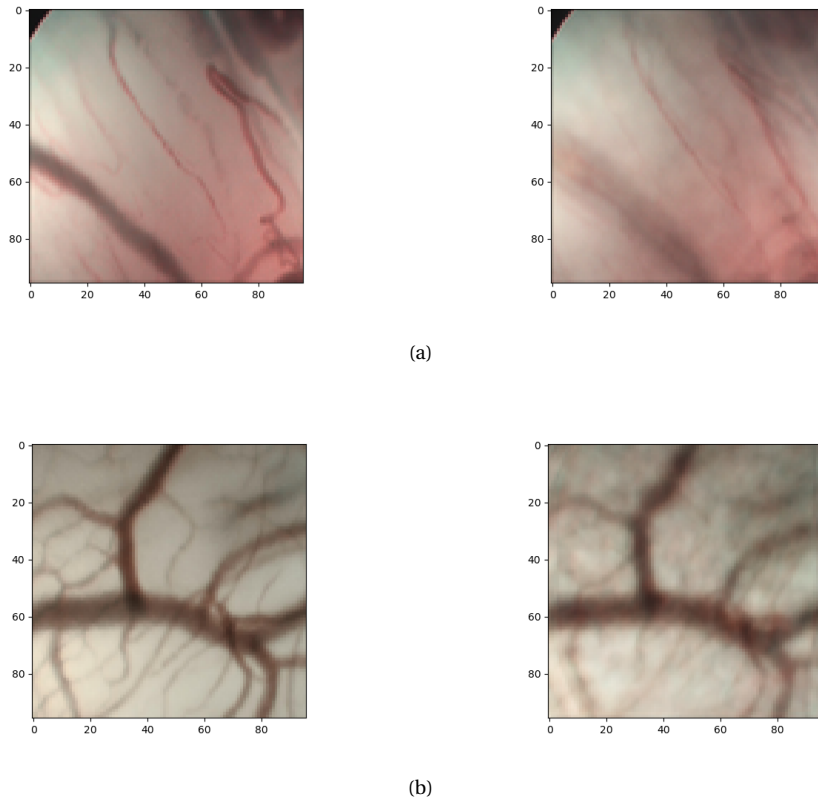


Figure 3.17: Examples of input images (left) and their reconstructed version (right).

Accuracy results for the experiments of this approach comparing to k-means method are listed in table 3.4. The accuracy of k-means method considers as a reference. Embedded features in Z-space are projected on 2D plain by principal components analysis (PCA) in fig. 3.18. The projection captured at the corresponded epoch to the accuracy listed in table 3.4. As seen, there is an overlap in classes distribution which led to low accuracy rates, however, the implemented models of this work showed improvement in the accuracy metric comparing to k-means method, which is a result of the iterative refining property in the deep embedding clustering models.

Table 3.4: Comparing the accuracy achieved by the three experiment of deep embedding clustering approach with k-means as a reference.

Metric	k-means	DEC	Plain DEC	DEC(DCNNN features)
Accuracy(%)	48.52	58.61	51.69	55.92

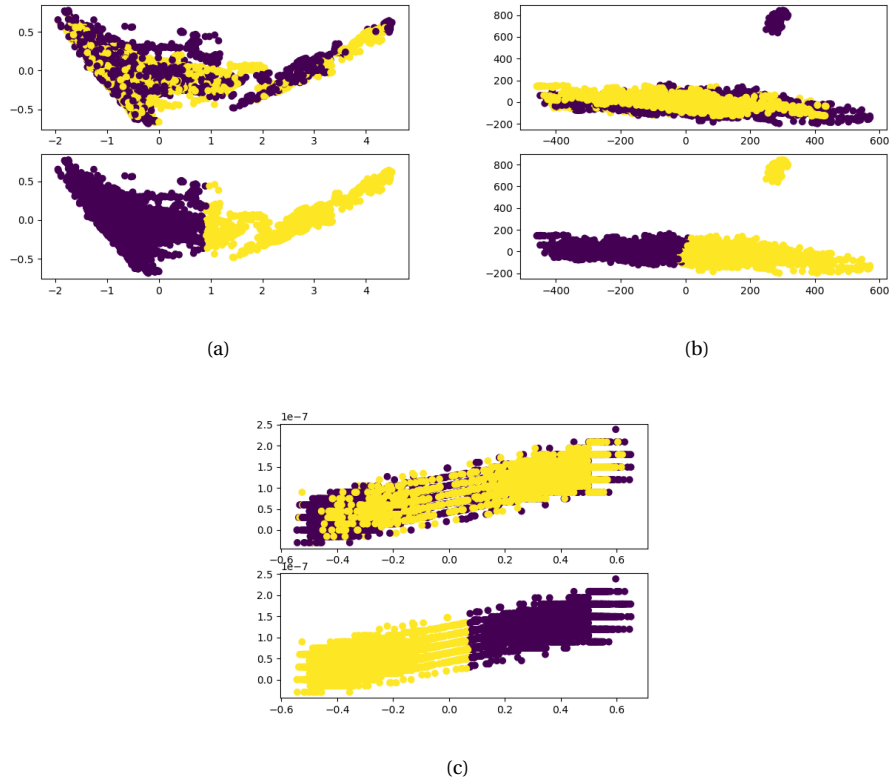


Figure 3.18: Projection of the latent features of z-space by Principal Components Analysis (PCA). (a) DEC method. (b) Plain DEC. (c) DEC based on deep-convolutional extracted features. The upper projection based on ground-truth labels. The lower projection based on clustering prediction.

4

Discussion

4.1 Contribution of Deep CNN

This part of the work was about finding a good DCNN model on extended criteria. Generalization and number of epochs were the top two criteria over different implementations. Considering these criteria, this work took the advantage of Transfer Learning technique for shaping different DCNN structures for investigation, to cut down the total parameters of the DCNN network. which highlighted notes and observations as follows.

Using "cutt-off" layer for limiting the feature extractor is not common in CNNs transfer learning investigations. This allows to reduce the number of model's parameters, and subsequently to reduce the risk of overfitting. Model five for example, was able to approach high results by small number of epochs comparing to model seven which could not have the equal performance qualities, in spite of the huge capacity of it. This improvement is attributed to the "cut-off" technique.

Another important highlight about creating a good classifier, that is Global Max Pooling layer type, which allowed reduced the parameter count strongly, and so the chance of overfitting. Model five performance was much better than model eight in the first experiment, as illustrated earlier. This type of layer transforms the feature maps in the last convolutional layer from $X \times Y \times Z$ dimension to $1 \times 1 \times Z$, where Z is the number of feature maps. Fig. 4.1 shows how Global Max Pooling layer treats a feature map. Setting-up the classifier with this layer on the top of the feature extractor captures important spatial information, which are curves and edges in our application.

Following that, using Global Max Pooling layer shows robust performance without fully connected layers and Dropouts. Fully connected layers are parameter-dense layers, and so they increase strongly the chance of getting a model overfit, hampering the model convergence additionally. This was investigated through model seven trial explained early on this work. Results of model five demon-

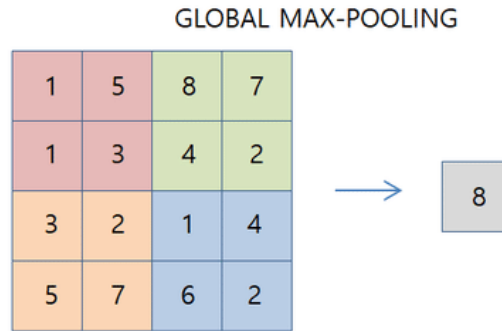


Figure 4.1: How Global Max Pooling Layer maps input feature map to a single value [60].

strates that two-layer classifier made up of Global Max Pooling and a single output neuron is adequate.

Data Augmentation is meant to increase the generalization ability of CNN model, so then it is relevant to training and validation phases, yet not testing phase. According to [61], fine-tuned CNN models do not benefit much from data augmentations technique comparing to models training from scratch. Since both transfer learning and data augmentation are for increasing the generalisability, adopting them in one implementation leads to neutralize the effectiveness of one of them, the third experiment as an example. In experiment number three, pace of training decreased as well as the plateau occurred at lower value, which did counteract model five performance in achieving high accuracy on validation images (4.35% accuracy drop). As a result to that, the model did achieve less favourable results at the testing phase (9.48% accuracy drop). As experiment three has more realistic features, the obtained results of it are more acceptable with current introductions of this work.

4.2 Contribution of Deep Embedding Clustering

Although performance of the implemented methods in this work yielded low accuracy in absolute measures, the different versions of DEC has improved clustering value in terms of accuracy metric. The results of this approach are in symmetric pattern with the published results in [35], specifically the corresponded results of STL dataset as both STL dataset and ours own same type of images. The reported improvement was 3.17-10.09% in our case.

The exceptional performance of DEC algorithm on MNIST dataset was the driven cause to bring this algorithm under the microscope. In additional trials, images of dataset C had been converted to grayscale kind of images by a skeletonisation function, as shown in fig. 4.2, similar to MNIST images (fig. 4.3). Nevertheless, that did not match the expectation about getting higher accuracy than the values reported in table 3.4. We believe that the high magnitude of similarity within each class of MNIST has a fundamental role in obtaining high accuracy value.

Adding batch normalisation layer to autoencoder's structure outperforms the other methods in respect to the reconstruction error, as previously illustrated in fig. 3.16 (b), without being advantageous on clustering accuracy. In order to check the integrity of the proposed plain version of DEC method, the method executed on MNIST and achieved very close results. Fig. 4.4 portrays the embedded features of MNIST dataset using PCA in coloured clusters. Further implementations are required to assess the benefit of the batch normalisation layer on autoencoder related methods.

About the third experiment, in which convolutional features were extracted for deep embedding clustering, the experiment was repeated on MNIST dataset, by which 86.31% accuracy achieved. Fig. 4.5 shows the feature space, in which the latent features of Z space are projected in 2D plain. As seen, these representations are grouped densely in ten clusters, where the perceived classes distribution is similar to ground-truth labels, which explains the good performance on MNIST.

Moreover, It was noticed that DEC methods are robust against changes in hyperparameters, contrary with DCNN models. DEC models had very approximate performance with different learning rates, types of optimizer, image sizes and even different dimensions of Z-space. Additionally, it was noted that long autoencoder training at the initialisation phase is redundant as long as no significant enhancement regarding quality of the reconstruction, as shown in fig.3.16.

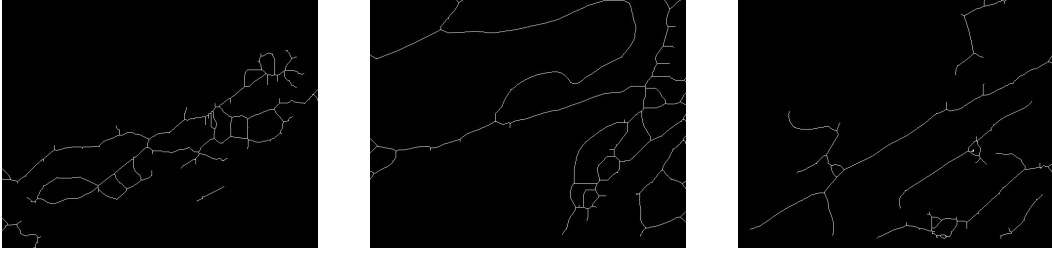


Figure 4.2: Examples of preprocessed images through skeletonization function.

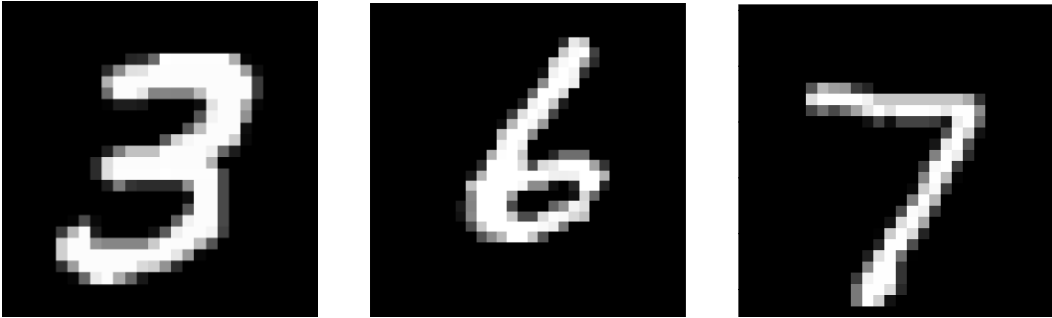


Figure 4.3: Samples from MNIST dataset [62].

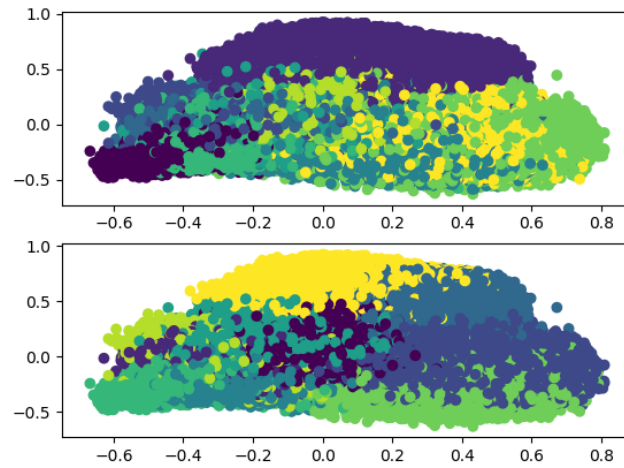


Figure 4.4: Clustering visualisation of plain variant of DEC method on MNIST images. Upper projection based on truth labels. Lower projection based on clustering prediction.

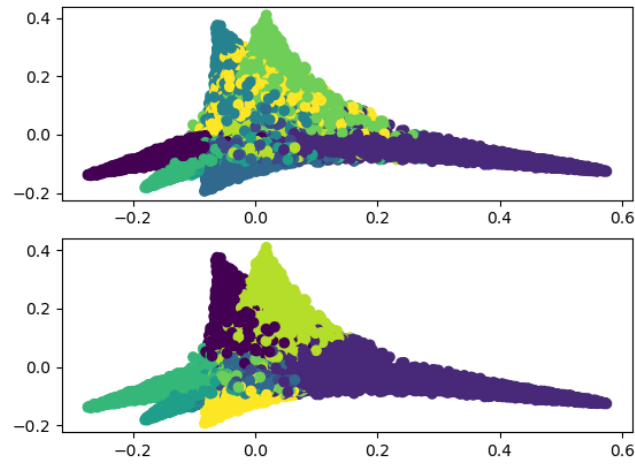


Figure 4.5: Clustering visualisation of DEC method based on deep-convolutional-extracted features of MNIST images. Upper projection based on truth labels. Lower projection based on clustering prediction.

5

Conclusions and Future Work

To conclude this work, the outcomes of deep neural network approach by fine-tuning technique have shown the potential for being a useful tool at diagnosing laryngeal cancer images into benign and malignant. Taking portions of ResNet50 was a valid path that achieved comparable performance to the whole fine-tuned network. Decreasing the total parameter count was the main advantage, in addition to short training time required for model training. As well, augmenting the images led to lower the learning curve of the model slightly, nevertheless, robust model performance was remained with competitive metrics to humans. Utilising part of the pretrained model requires less memory, which facilitates later deployment through portable devices in the medical domain.

The investigation of deep embedding clustering has shown enhancement on clustering accuracy compared with k-means algorithm, but still to be considered a poor classifier, as the extracted features were overlapping in the feature space. Adding the batch normalisation layers to plain variant DEC model accelerated the initialisation phase, and obtained more convenient loss values, yet no improvement witnessed on clustering accuracy. Since DEC method improves the present baseline, leveling up the baseline by using multiclassifiers will be considered in the future. Additionally, Seeking further characteristics in CE-NBI images could be helpful to overcome the curse of dimensionality.

Since the supervised approaches is far more capable than the unsupervised in case of CE-NBI images. In near future, we will design a DCNN model for regression predictions to laryngeal cancer subclasses. Furthermore, creating real-time identification system for laryngeal tumors based on deep learning will be helpful in shortening diagnosis time during the standard routines.



Appendix

Listing A.1: Structure of Resnet50 Pre-Trained Model (Keras API).

Model: "resnet50"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 150, 150, 3)	0	
conv1_pad (ZeroPadding2D)	(None, 156, 156, 3)	0	input_1[0][0]
conv1_conv (Conv2D)	(None, 75, 75, 64)	9472	conv1_pad[0][0]
conv1_bn (BatchNormalization)	(None, 75, 75, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 75, 75, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 77, 77, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 38, 38, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 38, 38, 64)	4160	pool1_pool[0][0]
conv2_block1_1_bn (BatchNormalization)	(None, 38, 38, 64)	256	conv2_block1_1_conv[0][0]
conv2_block1_1_relu (Activation)	(None, 38, 38, 64)	0	conv2_block1_1_bn[0][0]
conv2_block1_2_conv (Conv2D)	(None, 38, 38, 64)	36928	conv2_block1_1_relu[0][0]
conv2_block1_2_bn (BatchNormalization)	(None, 38, 38, 64)	256	conv2_block1_2_conv[0][0]
conv2_block1_2_relu (Activation)	(None, 38, 38, 64)	0	conv2_block1_2_bn[0][0]
conv2_block1_0_conv (Conv2D)	(None, 38, 38, 256)	16640	pool1_pool[0][0]
conv2_block1_3_conv (Conv2D)	(None, 38, 38, 256)	16640	conv2_block1_2_relu[0][0]
conv2_block1_0_bn (BatchNormalization)	(None, 38, 38, 256)	1024	conv2_block1_0_conv[0][0]
conv2_block1_3_bn (BatchNormalization)	(None, 38, 38, 256)	1024	conv2_block1_3_conv[0][0]
conv2_block1_add (Add)	(None, 38, 38, 256)	0	conv2_block1_0_bn[0][0] conv2_block1_3_bn[0][0]
conv2_block1_out (Activation)	(None, 38, 38, 256)	0	conv2_block1_add[0][0]
conv2_block2_1_conv (Conv2D)	(None, 38, 38, 64)	16448	conv2_block1_out[0][0]
conv2_block2_1_bn (BatchNormalization)	(None, 38, 38, 64)	256	conv2_block2_1_conv[0][0]

conv2_block2_1_relu (Activation)	(None, 38, 38, 64)	0	conv2_block2_1_bn[0][0]
conv2_block2_2_conv (Conv2D)	(None, 38, 38, 64)	36928	conv2_block2_1_relu[0][0]
conv2_block2_2_bn (BatchNormali	(None, 38, 38, 64)	256	conv2_block2_2_conv[0][0]
conv2_block2_2_relu (Activation)	(None, 38, 38, 64)	0	conv2_block2_2_bn[0][0]
conv2_block2_3_conv (Conv2D)	(None, 38, 38, 256)	16640	conv2_block2_2_relu[0][0]
conv2_block2_3_bn (BatchNormali	(None, 38, 38, 256)	1024	conv2_block2_3_conv[0][0]
conv2_block2_add (Add)	(None, 38, 38, 256)	0	conv2_block1_out[0][0] conv2_block2_3_bn[0][0]
conv2_block2_out (Activation)	(None, 38, 38, 256)	0	conv2_block2_add[0][0]
conv2_block3_1_conv (Conv2D)	(None, 38, 38, 64)	16448	conv2_block2_out[0][0]
conv2_block3_1_bn (BatchNormali	(None, 38, 38, 64)	256	conv2_block3_1_conv[0][0]
conv2_block3_1_relu (Activation)	(None, 38, 38, 64)	0	conv2_block3_1_bn[0][0]
conv2_block3_2_conv (Conv2D)	(None, 38, 38, 64)	36928	conv2_block3_1_relu[0][0]
conv2_block3_2_bn (BatchNormali	(None, 38, 38, 64)	256	conv2_block3_2_conv[0][0]
conv2_block3_2_relu (Activation)	(None, 38, 38, 64)	0	conv2_block3_2_bn[0][0]
conv2_block3_3_conv (Conv2D)	(None, 38, 38, 256)	16640	conv2_block3_2_relu[0][0]
conv2_block3_3_bn (BatchNormali	(None, 38, 38, 256)	1024	conv2_block3_3_conv[0][0]
conv2_block3_add (Add)	(None, 38, 38, 256)	0	conv2_block2_out[0][0] conv2_block3_3_bn[0][0]
conv2_block3_out (Activation)	(None, 38, 38, 256)	0	conv2_block3_add[0][0]
conv3_block1_1_conv (Conv2D)	(None, 19, 19, 128)	32896	conv2_block3_out[0][0]
conv3_block1_1_bn (BatchNormali	(None, 19, 19, 128)	512	conv3_block1_1_conv[0][0]
conv3_block1_1_relu (Activation)	(None, 19, 19, 128)	0	conv3_block1_1_bn[0][0]
conv3_block1_2_conv (Conv2D)	(None, 19, 19, 128)	147584	conv3_block1_1_relu[0][0]
conv3_block1_2_bn (BatchNormali	(None, 19, 19, 128)	512	conv3_block1_2_conv[0][0]
conv3_block1_2_relu (Activation)	(None, 19, 19, 128)	0	conv3_block1_2_bn[0][0]
conv3_block1_0_conv (Conv2D)	(None, 19, 19, 512)	131584	conv2_block3_out[0][0]
conv3_block1_3_conv (Conv2D)	(None, 19, 19, 512)	66048	conv3_block1_2_relu[0][0]
conv3_block1_0_bn (BatchNormali	(None, 19, 19, 512)	2048	conv3_block1_0_conv[0][0]
conv3_block1_3_bn (BatchNormali	(None, 19, 19, 512)	2048	conv3_block1_3_conv[0][0]
conv3_block1_add (Add)	(None, 19, 19, 512)	0	conv3_block1_0_bn[0][0] conv3_block1_3_bn[0][0]
conv3_block1_out (Activation)	(None, 19, 19, 512)	0	conv3_block1_add[0][0]
conv3_block2_1_conv (Conv2D)	(None, 19, 19, 128)	65664	conv3_block1_out[0][0]

conv3_block2_1_bn	(BatchNormali	(None, 19, 19, 128)	512	conv3_block2_1_conv[0][0]
conv3_block2_1_relu	(Activation	(None, 19, 19, 128)	0	conv3_block2_1_bn[0][0]
conv3_block2_2_conv	(Conv2D)	(None, 19, 19, 128)	147584	conv3_block2_1_relu[0][0]
conv3_block2_2_bn	(BatchNormali	(None, 19, 19, 128)	512	conv3_block2_2_conv[0][0]
conv3_block2_2_relu	(Activation	(None, 19, 19, 128)	0	conv3_block2_2_bn[0][0]
conv3_block2_3_conv	(Conv2D)	(None, 19, 19, 512)	66048	conv3_block2_2_relu[0][0]
conv3_block2_3_bn	(BatchNormali	(None, 19, 19, 512)	2048	conv3_block2_3_conv[0][0]
conv3_block2_add	(Add)	(None, 19, 19, 512)	0	conv3_block1_out[0][0] conv3_block2_3_bn[0][0]
conv3_block2_out	(Activation)	(None, 19, 19, 512)	0	conv3_block2_add[0][0]
conv3_block3_1_conv	(Conv2D)	(None, 19, 19, 128)	65664	conv3_block2_out[0][0]
conv3_block3_1_bn	(BatchNormali	(None, 19, 19, 128)	512	conv3_block3_1_conv[0][0]
conv3_block3_1_relu	(Activation	(None, 19, 19, 128)	0	conv3_block3_1_bn[0][0]
conv3_block3_2_conv	(Conv2D)	(None, 19, 19, 128)	147584	conv3_block3_1_relu[0][0]
conv3_block3_2_bn	(BatchNormali	(None, 19, 19, 128)	512	conv3_block3_2_conv[0][0]
conv3_block3_2_relu	(Activation	(None, 19, 19, 128)	0	conv3_block3_2_bn[0][0]
conv3_block3_3_conv	(Conv2D)	(None, 19, 19, 512)	66048	conv3_block3_2_relu[0][0]
conv3_block3_3_bn	(BatchNormali	(None, 19, 19, 512)	2048	conv3_block3_3_conv[0][0]
conv3_block3_add	(Add)	(None, 19, 19, 512)	0	conv3_block2_out[0][0] conv3_block3_3_bn[0][0]
conv3_block3_out	(Activation)	(None, 19, 19, 512)	0	conv3_block3_add[0][0]
conv3_block4_1_conv	(Conv2D)	(None, 19, 19, 128)	65664	conv3_block3_out[0][0]
conv3_block4_1_bn	(BatchNormali	(None, 19, 19, 128)	512	conv3_block4_1_conv[0][0]
conv3_block4_1_relu	(Activation	(None, 19, 19, 128)	0	conv3_block4_1_bn[0][0]
conv3_block4_2_conv	(Conv2D)	(None, 19, 19, 128)	147584	conv3_block4_1_relu[0][0]
conv3_block4_2_bn	(BatchNormali	(None, 19, 19, 128)	512	conv3_block4_2_conv[0][0]
conv3_block4_2_relu	(Activation	(None, 19, 19, 128)	0	conv3_block4_2_bn[0][0]
conv3_block4_3_conv	(Conv2D)	(None, 19, 19, 512)	66048	conv3_block4_2_relu[0][0]
conv3_block4_3_bn	(BatchNormali	(None, 19, 19, 512)	2048	conv3_block4_3_conv[0][0]
conv3_block4_add	(Add)	(None, 19, 19, 512)	0	conv3_block3_out[0][0] conv3_block4_3_bn[0][0]
conv3_block4_out	(Activation)	(None, 19, 19, 512)	0	conv3_block4_add[0][0]
conv4_block1_1_conv	(Conv2D)	(None, 10, 10, 256)	131328	conv3_block4_out[0][0]

conv4_block1_1_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block1_1_conv[0][0]
conv4_block1_1_relu (Activation	(None, 10, 10, 256)	0	conv4_block1_1_bn[0][0]
conv4_block1_2_conv (Conv2D)	(None, 10, 10, 256)	590080	conv4_block1_1_relu[0][0]
conv4_block1_2_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block1_2_conv[0][0]
conv4_block1_2_relu (Activation	(None, 10, 10, 256)	0	conv4_block1_2_bn[0][0]
conv4_block1_0_conv (Conv2D)	(None, 10, 10, 1024)	525312	conv3_block4_out[0][0]
conv4_block1_3_conv (Conv2D)	(None, 10, 10, 1024)	263168	conv4_block1_2_relu[0][0]
conv4_block1_0_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block1_0_conv[0][0]
conv4_block1_3_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block1_3_conv[0][0]
conv4_block1_add (Add)	(None, 10, 10, 1024)	0	conv4_block1_0_bn[0][0] conv4_block1_3_bn[0][0]
conv4_block1_out (Activation)	(None, 10, 10, 1024)	0	conv4_block1_add[0][0]
conv4_block2_1_conv (Conv2D)	(None, 10, 10, 256)	262400	conv4_block1_out[0][0]
conv4_block2_1_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block2_1_conv[0][0]
conv4_block2_1_relu (Activation	(None, 10, 10, 256)	0	conv4_block2_1_bn[0][0]
conv4_block2_2_conv (Conv2D)	(None, 10, 10, 256)	590080	conv4_block2_1_relu[0][0]
conv4_block2_2_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block2_2_conv[0][0]
conv4_block2_2_relu (Activation	(None, 10, 10, 256)	0	conv4_block2_2_bn[0][0]
conv4_block2_3_conv (Conv2D)	(None, 10, 10, 1024)	263168	conv4_block2_2_relu[0][0]
conv4_block2_3_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block2_3_conv[0][0]
conv4_block2_add (Add)	(None, 10, 10, 1024)	0	conv4_block1_out[0][0] conv4_block2_3_bn[0][0]
conv4_block2_out (Activation)	(None, 10, 10, 1024)	0	conv4_block2_add[0][0]
conv4_block3_1_conv (Conv2D)	(None, 10, 10, 256)	262400	conv4_block2_out[0][0]
conv4_block3_1_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block3_1_conv[0][0]
conv4_block3_1_relu (Activation	(None, 10, 10, 256)	0	conv4_block3_1_bn[0][0]
conv4_block3_2_conv (Conv2D)	(None, 10, 10, 256)	590080	conv4_block3_1_relu[0][0]
conv4_block3_2_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block3_2_conv[0][0]
conv4_block3_2_relu (Activation	(None, 10, 10, 256)	0	conv4_block3_2_bn[0][0]
conv4_block3_3_conv (Conv2D)	(None, 10, 10, 1024)	263168	conv4_block3_2_relu[0][0]
conv4_block3_3_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block3_3_conv[0][0]
conv4_block3_add (Add)	(None, 10, 10, 1024)	0	conv4_block2_out[0][0] conv4_block3_3_bn[0][0]
conv4_block3_out (Activation)	(None, 10, 10, 1024)	0	conv4_block3_add[0][0]

conv4_block4_1_conv (Conv2D)	(None, 10, 10, 256)	262400	conv4_block3_out[0][0]
conv4_block4_1_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block4_1_conv[0][0]
conv4_block4_1_relu (Activation	(None, 10, 10, 256)	0	conv4_block4_1_bn[0][0]
conv4_block4_2_conv (Conv2D)	(None, 10, 10, 256)	590080	conv4_block4_1_relu[0][0]
conv4_block4_2_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block4_2_conv[0][0]
conv4_block4_2_relu (Activation	(None, 10, 10, 256)	0	conv4_block4_2_bn[0][0]
conv4_block4_3_conv (Conv2D)	(None, 10, 10, 1024)	263168	conv4_block4_2_relu[0][0]
conv4_block4_3_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block4_3_conv[0][0]
conv4_block4_add (Add)	(None, 10, 10, 1024)	0	conv4_block3_out[0][0] conv4_block4_3_bn[0][0]
conv4_block4_out (Activation)	(None, 10, 10, 1024)	0	conv4_block4_add[0][0]
conv4_block5_1_conv (Conv2D)	(None, 10, 10, 256)	262400	conv4_block4_out[0][0]
conv4_block5_1_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block5_1_conv[0][0]
conv4_block5_1_relu (Activation	(None, 10, 10, 256)	0	conv4_block5_1_bn[0][0]
conv4_block5_2_conv (Conv2D)	(None, 10, 10, 256)	590080	conv4_block5_1_relu[0][0]
conv4_block5_2_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block5_2_conv[0][0]
conv4_block5_2_relu (Activation	(None, 10, 10, 256)	0	conv4_block5_2_bn[0][0]
conv4_block5_3_conv (Conv2D)	(None, 10, 10, 1024)	263168	conv4_block5_2_relu[0][0]
conv4_block5_3_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block5_3_conv[0][0]
conv4_block5_add (Add)	(None, 10, 10, 1024)	0	conv4_block4_out[0][0] conv4_block5_3_bn[0][0]
conv4_block5_out (Activation)	(None, 10, 10, 1024)	0	conv4_block5_add[0][0]
conv4_block6_1_conv (Conv2D)	(None, 10, 10, 256)	262400	conv4_block5_out[0][0]
conv4_block6_1_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block6_1_conv[0][0]
conv4_block6_1_relu (Activation	(None, 10, 10, 256)	0	conv4_block6_1_bn[0][0]
conv4_block6_2_conv (Conv2D)	(None, 10, 10, 256)	590080	conv4_block6_1_relu[0][0]
conv4_block6_2_bn (BatchNormali	(None, 10, 10, 256)	1024	conv4_block6_2_conv[0][0]
conv4_block6_2_relu (Activation	(None, 10, 10, 256)	0	conv4_block6_2_bn[0][0]
conv4_block6_3_conv (Conv2D)	(None, 10, 10, 1024)	263168	conv4_block6_2_relu[0][0]
conv4_block6_3_bn (BatchNormali	(None, 10, 10, 1024)	4096	conv4_block6_3_conv[0][0]
conv4_block6_add (Add)	(None, 10, 10, 1024)	0	conv4_block5_out[0][0] conv4_block6_3_bn[0][0]
conv4_block6_out (Activation)	(None, 10, 10, 1024)	0	conv4_block6_add[0][0]

conv5_block1_1_conv (Conv2D)	(None, 5, 5, 512)	524800	conv4_block6_out[0][0]
conv5_block1_1_bn (BatchNormali	(None, 5, 5, 512)	2048	conv5_block1_1_conv[0][0]
conv5_block1_1_relu (Activation	(None, 5, 5, 512)	0	conv5_block1_1_bn[0][0]
conv5_block1_2_conv (Conv2D)	(None, 5, 5, 512)	2359808	conv5_block1_1_relu[0][0]
conv5_block1_2_bn (BatchNormali	(None, 5, 5, 512)	2048	conv5_block1_2_conv[0][0]
conv5_block1_2_relu (Activation	(None, 5, 5, 512)	0	conv5_block1_2_bn[0][0]
conv5_block1_0_conv (Conv2D)	(None, 5, 5, 2048)	2099200	conv4_block6_out[0][0]
conv5_block1_3_conv (Conv2D)	(None, 5, 5, 2048)	1050624	conv5_block1_2_relu[0][0]
conv5_block1_0_bn (BatchNormali	(None, 5, 5, 2048)	8192	conv5_block1_0_conv[0][0]
conv5_block1_3_bn (BatchNormali	(None, 5, 5, 2048)	8192	conv5_block1_3_conv[0][0]
conv5_block1_add (Add)	(None, 5, 5, 2048)	0	conv5_block1_0_bn[0][0] conv5_block1_3_bn[0][0]
conv5_block1_out (Activation)	(None, 5, 5, 2048)	0	conv5_block1_add[0][0]
conv5_block2_1_conv (Conv2D)	(None, 5, 5, 512)	1049088	conv5_block1_out[0][0]
conv5_block2_1_bn (BatchNormali	(None, 5, 5, 512)	2048	conv5_block2_1_conv[0][0]
conv5_block2_1_relu (Activation	(None, 5, 5, 512)	0	conv5_block2_1_bn[0][0]
conv5_block2_2_conv (Conv2D)	(None, 5, 5, 512)	2359808	conv5_block2_1_relu[0][0]
conv5_block2_2_bn (BatchNormali	(None, 5, 5, 512)	2048	conv5_block2_2_conv[0][0]
conv5_block2_2_relu (Activation	(None, 5, 5, 512)	0	conv5_block2_2_bn[0][0]
conv5_block2_3_conv (Conv2D)	(None, 5, 5, 2048)	1050624	conv5_block2_2_relu[0][0]
conv5_block2_3_bn (BatchNormali	(None, 5, 5, 2048)	8192	conv5_block2_3_conv[0][0]
conv5_block2_add (Add)	(None, 5, 5, 2048)	0	conv5_block1_out[0][0] conv5_block2_3_bn[0][0]
conv5_block2_out (Activation)	(None, 5, 5, 2048)	0	conv5_block2_add[0][0]
conv5_block3_1_conv (Conv2D)	(None, 5, 5, 512)	1049088	conv5_block2_out[0][0]
conv5_block3_1_bn (BatchNormali	(None, 5, 5, 512)	2048	conv5_block3_1_conv[0][0]
conv5_block3_1_relu (Activation	(None, 5, 5, 512)	0	conv5_block3_1_bn[0][0]
conv5_block3_2_conv (Conv2D)	(None, 5, 5, 512)	2359808	conv5_block3_1_relu[0][0]
conv5_block3_2_bn (BatchNormali	(None, 5, 5, 512)	2048	conv5_block3_2_conv[0][0]
conv5_block3_2_relu (Activation	(None, 5, 5, 512)	0	conv5_block3_2_bn[0][0]
conv5_block3_3_conv (Conv2D)	(None, 5, 5, 2048)	1050624	conv5_block3_2_relu[0][0]
conv5_block3_3_bn (BatchNormali	(None, 5, 5, 2048)	8192	conv5_block3_3_conv[0][0]
conv5_block3_add (Add)	(None, 5, 5, 2048)	0	conv5_block2_out[0][0] conv5_block3_3_bn[0][0]

conv5_block3_out (Activation)	(None, 5, 5, 2048)	0	conv5_block3_add[0][0]
-------------------------------	--------------------	---	------------------------

Total params: 23,587,712
Trainable params: 0
Non-trainable params: 23,587,712

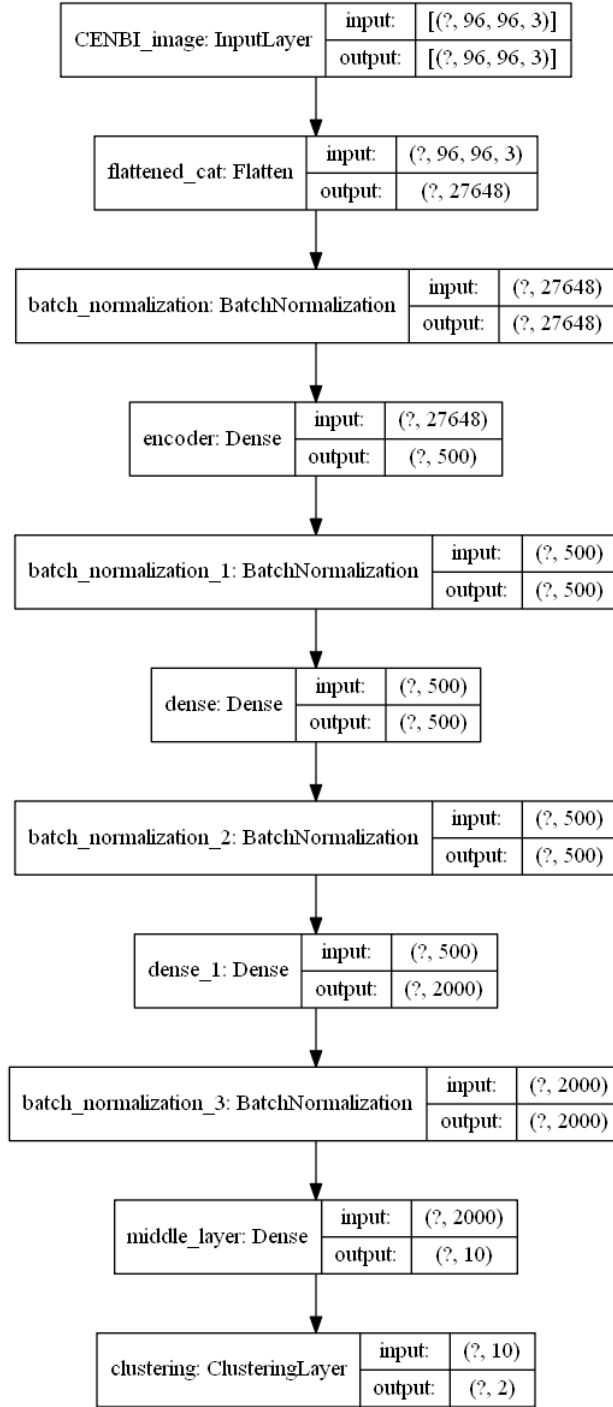


Figure A.1: Model summary of plain version of DEC model.



Bibliography

- [1] Conor E Steuer, Mark El-Deiry, Jason R Parks, Kristin A Higgins, and Nabil F Saba. An update on larynx cancer. *CA: A Cancer Journal for Clinicians*, 67(1): 31–50, 2017. ISSN 1542-4863. doi: 10.3322/caac.21386.
- [2] Giovanni Franchin, Emanuela Vaccher, Dorian Politi, Emilio Minatel, Carlo Gobitti, Renato Talamini, Simon Spazzapan, Maria Savignano, Mauro Trovò, and Luigi Barzan. Organ preservation in locally advanced head and neck cancer of the larynx using induction chemotherapy followed by improved radiation schemes. *European archives of oto-rhino-laryngology : official journal of the European Federation of Oto-Rhino-Laryngological Societies (EUFOS) : affiliated with the German Society for Oto-Rhino-Laryngology - Head and Neck Surgery*, 266:719–26, 10 2008. doi: 10.1007/s00405-008-0798-2.
- [3] Toru Ugumori, Manabu Muto, Ryuichi Hayashi, Tomomasa Hayashi, and Seiji Kishimoto. Prospective study of early detection of pharyngeal superficial carcinoma with the narrowband imaging laryngoscope. *Head & Neck*, 31(2):189–194, 2009. doi: 10.1002/hed.20943. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/hed.20943>.
- [4] J. Rey, K. Kuznetsov, and R. Lambert. Narrow band imaging: a wide field of possibilities. *Saudi journal of gastroenterology : official journal of the Saudi Gastroenterology Association*, 13 1:1–10, 2007.
- [5] D. K. Yeung, A. Vlantis, Eddy W. Y. Wong, M. Tong, and J. Chan. A meta-analysis of narrow-band imaging for the diagnosis of primary nasopharyngeal carcinoma. *F1000Research*, 7, 2018.
- [6] Akihito Watanabe, Masanobu Taniguchi, Hitoshi Tsujie, Masao Hosokawa, Masahiro Fujita, and Shigeyuki Sasaki. The value of narrow band imaging for early detection of laryngeal cancer. *European archives of oto-rhino-laryngology : official journal of the European Federation of Oto-Rhino-*

- Laryngological Societies (EUFOS) : affiliated with the German Society for Otorhino-Laryngology - Head and Neck Surgery*, 266:1017–23, 12 2008. doi: 10.1007/s00405-008-0835-1.
- [7] Marcel Kraft, Karolos Fostiropoulos, Nicolas Gürtler, André Arnoux, Nikolaos Davaris, and Christoph Arens. Value of narrow band imaging in the early diagnosis of laryngeal cancer. *Head & Neck*, 38(1):15–20. doi: 10.1002/hed.23838. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/hed.23838>.
 - [8] Giulia Bertino, Salvatore Cacciola, Wladir Bastos Fernandes Jr, Carolina Muniz Fernandes, Antonio Occhini, Carmine Tinelli, and Marco Benazzo. Effectiveness of narrow band imaging in the detection of premalignant and malignant lesions of the larynx: Validation of a new endoscopic clinical classification. *Head & Neck*, 37(2):215–222, 2015. doi: 10.1002/hed.23582. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/hed.23582>.
 - [9] Camile S. Farah, Andrew J. Dalley, Phan Nguyen, Martin Batstone, Farzaneh Kordbacheh, Joanna Perry-Keene, and David Fielding. Improved surgical margin definition by narrow band imaging for resection of oral squamous cell carcinoma: A prospective gene expression profiling study. *Head & Neck*, 38(6):832–839, 2016. doi: 10.1002/hed.23989. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/hed.23989>.
 - [10] K Kuznetsov, R Lambert, and jean-francois Rey. Narrow-band imaging: Potential and limitations. *Endoscopy*, 38:76–81, 02 2006. doi: 10.1055/s-2005-921114.
 - [11] K. Yao, H. Doyama, T. Gotoda, H. Ishikawa, T. Nagahama, C. Yokoi, I. Oda, H. Machida, K. Uchita, and M. Tabuchi. Diagnostic performance and limitations of magnifying narrow-band imaging in screening endoscopy of early gastric cancer: a prospective multicenter feasibility study. *Gastric Cancer*, 17: 669–679, 2013.
 - [12] J. Kim, K. S. Hong, and H. Jung. A randomized controlled clinical study comparing the diagnostic accuracy of the histologic prediction for colorectal polyps depending on the use of either magnified or nonmagnified narrow band imaging. *Clinical Endoscopy*, 48:528 – 533, 2015.
 - [13] S. J. Chung, D. Kim, J. H. Song, H. Y. Kang, G. E. Chung, Jeongmin Choi, Y. S. Kim, M. Park, and J. Kim. Comparison of detection and miss rates of narrow band imaging, flexible spectral imaging chromoendoscopy and white light

- at screening colonoscopy: a randomised controlled back-to-back study. *Gut*, 63:785 – 791, 2013.
- [14] A. Adler, J. Aschenbeck, Timur Yenerim, M. Mayr, A. Aminalai, R. Drossel, A. Schroeder, M. Scheel, B. Wiedenmann, and T. Rösch. Narrow-band versus white-light high definition television endoscopic imaging for screening colonoscopy: a prospective randomized trial. *Gastroenterology*, 136 2:410–6.e1; quiz 715, 2009.
- [15] Cinzia Benazzi, Ahmad N. Al-Dissi, Cheuk Him Chau, William D Figg, Giuseppe Sarli, Joana-Tavares de Oliveira, and Fátima Gärtner. Angiogenesis in spontaneous tumors and implications for comparative tumor biology. In *TheScientificWorldJournal*, 2014.
- [16] Jaakko Laitakari, Veera Näyhä, and Frej Stenbäck. Size, shape, structure, and direction of angiogenesis in laryngeal tumour development. *Journal of clinical pathology*, 57 4:394–401, 2004.
- [17] Mineo Iwatate, Taro Ikumoto, Yasushi Sano, Fabian Emura, and Takahiro Fujimori. Diagnosis of neoplastic and non-neoplastic lesions and prediction of submucosal invasion of early cancer during colonoscopy. 2011.
- [18] Filippo Carta, Sara Sionis, Daniela Cocco, Clara Gerosa, Caterina Ferreli, and Roberto Puxeddu. Enhanced contact endoscopy for the assessment of the neoangiogenetic changes in precancerous and cancerous lesions of the oral cavity and oropharynx. *Archives of Oto-Rhino-Laryngology*, 273, 07 2015. doi: 10.1007/s00405-015-3698-2.
- [19] Filippo Carta, Sara Sionis, Daniela Cocco, Clara Gerosa, Caterina Ferreli, and Roberto Puxeddu. Enhanced contact endoscopy for the assessment of the neoangiogenetic changes in precancerous and cancerous lesions of the oral cavity and oropharynx. *European Archives of Oto-Rhino-Laryngology*, 273: 1895–1903, 2015.
- [20] Nazila Esmaeili, Alfredo Illanes, Axel Boese, Nikolaos Davaris, Christoph Arens, and Michael Friebe. Novel automated vessel pattern characterization of larynx contact endoscopic video images. *International Journal of Computer Assisted Radiology and Surgery*, 14, 07 2019. doi: 10.1007/s11548-019-02034-9.
- [21] Michał Żurek, Anna Rzepakowska, Ewa Osuch-Wójcikiewicz, and Kazimierz Niemczyk. Learning curve for endoscopic evaluation of vocal folds lesions with narrow band imaging. *Brazilian Journal of Otorhinolaryngology*, 85

- (6):753 – 759, 2019. ISSN 1808-8694. doi: <https://doi.org/10.1016/j.bjorl.2018.07.003>. URL <http://www.sciencedirect.com/science/article/pii/S1808869418301356>.
- [22] Francisco Baldaque-Silva, Margarida Marques, Nuno Lunet, Gonalo Themudo, Kenichi Goda, Ervin Toth, Jos  Soares, Pedro Bastos, Rosa Ramalho, Pedro Pereira, Nuno Marques, Miguel Coimbra, Michael Vieth, Mario Dinis-Ribeiro, Guilherme Macedo, Lars Lundell, and Hanns-Ulrich Marschall. Endoscopic assessment and grading of barrett’s esophagus using magnification endoscopy and narrow band imaging: Impact of structured learning and experience on the accuracy of the amsterdam classification system. *Scandinavian Journal of Gastroenterology*, 48(2):160–167, 2013. doi: 10.3109/00365521.2012.746392. URL <https://doi.org/10.3109/00365521.2012.746392>.
- [23] Manabu Muto, Mari Nakane, Chikatoshi Katada, Yasushi Sano, Atsushi Ohtsu, Hiroyasu Esumi, Satoshi Ebihara, and Shigeaki Yoshida. Squamous cell carcinoma in situ at oropharyngeal and hypopharyngeal mucosal sites. *Cancer*, 101(6):1375–1381, 2004. doi: 10.1002/cncr.20482. URL <https://acsjournals.onlinelibrary.wiley.com/doi/abs/10.1002/cncr.20482>.
- [24] Nikolaos Davaris, Anke Lux, Nazila Esmaeili, Alfredo Illanes, Axel Boese, Michael Friebe, and Christoph Arens. Evaluation of vascular patterns using contact endoscopy and narrow-band imaging (ce-nbi) for the diagnosis of vocal fold malignancy. *Cancers*, 12, 2020.
- [25] Matthew G. Crowson, Jonathan Ranisau, Antoine Eskander, Aaron Babier, Bin Xu, Russel R. Kahmke, Joseph M. Chen, and Timothy C. Y. Chan. A contemporary review of machine learning in otolaryngology–head and neck surgery. *The Laryngoscope*, 130(1):45–51, 2020. doi: 10.1002/lary.27850. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/lary.27850>.
- [26] Martin Halicek, Guolan Lu, James V. Little, Xu Wang, Mihir Patel, Christopher C. Griffith, Mark W. El-Deiry, Amy Y. Chen, and Baowei Fei. Deep convolutional neural networks for classifying head and neck cancer using hyperspectral imaging. *Journal of Biomedical Optics*, 22(6):1 – 4, 2017. doi: 10.1117/1.JBO.22.6.060503. URL <https://doi.org/10.1117/1.JBO.22.6.060503>.
- [27] Lili Zhang, Yongzheng Wu, Bin Zheng, Lizhong Su, Yuan Chen, Shuang Ma, Qinqin Hu, Xiang Zou, Lie Yao, Yinlong Yang, Liang Chen, Ying Mao, Yan Chen, and Minbiao Ji. Rapid histology of laryngeal squamous cell carcinoma

- with deep-learning based stimulated raman scattering microscopy. *Theranostics*, 9:2541–2554, 2019. doi: 10.7150/thno.32655. URL <http://www.thno.org/v09p2541.htm>.
- [28] Marc Aubreville, Christian Knipfer, Nicolai Oetter, Christian Jaremenko, Erik Rodner, Joachim Denzler, Christopher Bohr, Helmut Neumann, Florian Stelzle, and Andreas Maier. Automatic classification of cancerous tissue in laserendomicroscopy images of the oral cavity using deep learning. *Scientific Reports*, 7, 2017. doi: 10.1038/s41598-017-12320-8.
- [29] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, Dec 2015. doi: 10.1109/ICCV.2015.123.
- [30] Yoko Kominami, Shigeto Yoshida, Shinji Tanaka, Yoji Sanomura, Tsubasa Hirakawa, Bisser Raytchev, Toru Tamaki, Tetsusi Koide, Kazufumi Kaneda, and Kazuaki Chayama. Computer-aided diagnosis of colorectal polyp histology by using a real-time image recognition system and narrow-band imaging magnifying colonoscopy. *Gastrointestinal Endoscopy*, 83(3):643 – 649, 2016. ISSN 0016-5107. doi: <https://doi.org/10.1016/j.gie.2015.08.004>. URL <http://www.sciencedirect.com/science/article/pii/S0016510715027388>.
- [31] Masashi Misawa, shin-ei Kudo, Yuichi Mori, Kenichi Takeda, Yasuharu Maeda, Shinichi Kataoka, Hiroki Nakamura, Toyoki Kudo, Kunihiko Wakamura, Takemasa Hayashi, Atsushi Katagiri, Toshiyuki Baba, Fumio Ishida, Haruhiro Inoue, Yukitaka Nimura, Masahiro Oda, and Kensaku Mori. Accuracy of computer-aided diagnosis based on narrow-band imaging endocytoscopy for diagnosing colorectal lesions: comparison with experts. *International Journal of Computer Assisted Radiology and Surgery*, 12, 02 2017. doi: 10.1007/s11548-017-1542-4.
- [32] Yasser Khouj, Jeremy Dawson, James Coad, and Linda Vona-Davis. Hyperspectral imaging and k-means classification for histologic evaluation of ductal carcinoma in situ. *Frontiers in Oncology*, 8:17, 2018. ISSN 2234-943X. doi: 10.3389/fonc.2018.00017. URL <https://www.frontiersin.org/article/10.3389/fonc.2018.00017>.
- [33] Yoshito Takemura, Shigeto Yoshida, Shinji Tanaka, Rie Kawase, Keiichi Onji, Shiro Oka, Toru Tamaki, Bisser Raytchev, Kazufumi Kaneda, Masaharu Yoshihara, and Kazuaki Chayama. Computer-aided system for predicting the histology of colorectal tumors by using narrow-band imaging

- magnifying colonoscopy (with video). *Gastrointestinal Endoscopy*, 75(1): 179 – 185, 2012. ISSN 0016-5107. doi: <https://doi.org/10.1016/j.gie.2011.08.051>. URL <http://www.sciencedirect.com/science/article/pii/S0016510711021687>.
- [34] Anandhanarayanan Kamalakannan, Shiva Shankar Ganesan, and Govindaraj Rajamanickam. Self-learning ai framework for skin lesion image segmentation and classification. *International Journal of Computer Science and Information Technology*, 11(6):29–38, Dec 2019. ISSN 0975-4660. doi: 10.5121/ijcsit.2019.11604. URL <http://dx.doi.org/10.5121/ijcsit.2019.11604>.
- [35] Junyuan Xie, Ross B. Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. *ArXiv*, abs/1511.06335, 2016.
- [36] Xifeng Guo, Xinwang Liu, En Zhu, and Jianping Yin. Deep clustering with convolutional autoencoders. In *ICONIP*, 2017.
- [37] B Swinson, Waseem Jerjes, M El-Maaytah, P Norris, and C Hopper. Optical techniques in diagnosis of head and neck malignancy. *Oral oncology*, 42: 221–8, 04 2006. doi: 10.1016/j.oraloncology.2005.05.001.
- [38] Awadhesh Mishra, Ajith Nilakantan, Rakesh Datta, Kavita Sahai, Singh SP, and Ashwani Sethi. Contact endoscopy - a promising tool for evaluation of laryngeal mucosal lesions. *Journal of Laryngology and Voice*, 2:53–9, 01 2012. doi: 10.4103/2230-9748.106978.
- [39] J. Cooper. Tracheal injuries complicating prolonged intubation and tracheostomy. *Thoracic surgery clinics*, 28 2:139–144, 2018.
- [40] E. Caruana, F. Duchateau, C. Cornaglia, Marie-Laure Devaud, and R. Pirracchio. Tracheal intubation related complications in the prehospital setting. *Emergency Medicine Journal*, 32:882 – 887, 2015.
- [41] Y. Fujii, H. Tanaka, and H. Toyooka. Retracted article: Circulatory responses to laryngeal mask airway insertion or tracheal intubation in normotensive and hypertensive patients. *Canadian Journal of Anaesthesia*, 42:32–36, 1995.
- [42] Yi hui Wen, Xiao lin Zhu, W. Lei, Yu hui Zeng, Y. Sun, and W. Wen. Narrow-band imaging: a novel screening tool for early nasopharyngeal carcinoma. *Archives of otolaryngology–head neck surgery*, 138 2:183–8, 2012.
- [43] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

- [44] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL <https://www.tensorflow.org/>. Software available from tensorflow.org.
- [45] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [46] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.
- [47] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv*, abs/1502.03167, 2015.
- [48] L. Zhang, Y. Wu, B. Zheng, Lizhong Su, Y. Chen, S. Ma, Qinqin Hu, X. Zou, L. Yao, Y. Yang, L. Chen, Ying Mao, Yan Chen, and Minbiao Ji. Rapid histology of laryngeal squamous cell carcinoma with deep-learning based stimulated raman scattering microscopy. *Theranostics*, 9:2541 – 2554, 2019.
- [49] Nairveen Ali, Elsie Quansah, Katarina Köhler, T. Meyer, M. Schmitt, J. Popp, A. Niendorf, and Thomas Bocklitz. Automatic label-free detection of breast cancer using nonlinear multimodal imaging and the convolutional neural network resnet50. 2019.
- [50] Adrian Galdran, P. Costa, and A. Campilho. Real-time informative laryngoscopic frame classification with pre-trained convolutional neural networks. *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 87–90, 2019.
- [51] Qingge Ji, Jie Huang, Wenjie He, and Yankui Sun. Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images. *Algorithms*, 12:51, 02 2019. doi: 10.3390/a12030051.
- [52] Mina Khoshdeli, Richard Cong, and Bahram Parvin. Detection of nuclei in H&E stained sections using convolutional neural networks. 02 2017.

- [53] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio. Transfusion: Understanding transfer learning for medical imaging. In *NeurIPS*, 2019.
- [54] M. Steinbach, Levent Ertoz, and V. Kumar. The challenges of clustering high dimensional data. 2004.
- [55] L. V. D. Maaten. Learning a parametric embedding by preserving local structure. In *AISTATS*, 2009.
- [56] N. Srivastava, Geoffrey E. Hinton, A. Krizhevsky, Ilya Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958, 2014.
- [57] P. Vincent, H. Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010.
- [58] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.
- [59] Max Ferguson, Ronay ak, Yung-Tsun Lee, and Kincho Law. Automatic localization of casting defects with convolutional neural networks. pages 1726–1735, 12 2017. doi: 10.1109/BigData.2017.8258115.
- [60] Xi Ouyang, Pan Zhou, Cheng Hua Li, and L. Liu. Sentiment analysis using convolutional neural network. *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pages 2359–2364, 2015.
- [61] P. Pawara, E. Okafor, Lambert Schomaker, and M. Wiering. Data augmentation for plant classification. In *ACIVS*, 2017.
- [62] Y. LeCun, L. Bottou, Yoshua Bengio, and P. Haffner. Gradient-based learning applied to document recognition. 1998.