Technical University of Košice
Faculty of Electrical Engineering and Informatics

Deep convolutional neural network for detection of pathological speech

Master thesis

2020 Bc. Lukáš Vavrek

Technical University of Košice
Faculty of Electrical Engineering and Informatics

Deep convolutional neural network for detection of pathological speech

Master thesis

Study programme:	Informatics
Field of study:	Computer Science
Training center:	Department of Computers and Informatics (DCI)
Supervisor:	doc. Ing. Peter Drotár PhD.

Košice 2020 **Bc. Lukáš Vavrek**

Thesis title:	Deep convolutional neural network for detection of pathological speech
Department:	Department of Computers and Informatics, Technical University of Košice
Author:	Bc. Lukáš Vavrek
Supervisor:	doc. Ing. Peter Drotár PhD.
Date:	4. 5. 2020
Keywords:	pathological voice detection, machine learning, deep learning, transfer learning

Abstract: In this thesis, we are researching the usage of deep neural networks for the detection of pathological speech. We analyze existing solutions and published research papers and study them. We use the gained knowledge in the process of designing our solutions based on state-of-the-art convolutional neural networks. These existing pretrained convolutional neural networks are initially trained and used for the image classification task. Using a transfer learning technique, we reuse them for the task of detection of pathological speech. We develop each individual proposed solution in order to verify their function. Results are also analyzed and compared together and with existing published results. Our best-achieved result represents an improvement of 2,5%, compared to the latest results, while using a smaller dataset. We achieved an accuracy of pathological speech detection of 82%.

Názov práce:	Využitie hlbokých neurónových sietí pre rozpoznanie patologickej reči
Pracovisko:	Katedra počítačov a informatiky, Technická univerzita v Košiciach
Autor:	Bc. Lukáš Vavrek
Školiteľ:	doc. Ing. Peter Drotár PhD.
Dátum:	4. 5. 2020
Kľúčové slová:	detekcia patologického hlasu, strojové učenie, hlboké učenie, učenie prenosom

Abstrakt:

Táto práca sa venuje skúmaniu využitia hlbokých neurónových sietí pre rozponanie patologickej reči. Analyzuje existujúce riešenia a publikované výskumné články a rozoberá ich. Skúma použité metódy, dáta a dosiahnuté výsledky. Nadobudnuté vedomosti využíva pri návrhu vlastných riešení, ktoré používajú najmodernejšie riešenia v podobe konvolučných neurónových sietí. Tieto predtrénované existujúce konvolučné neurónové siete, pôvodne určené na klasifikáciu obrázkov, využíva prostredníctvom techniky učenia prenosom a adaptuje ich na použitie pre rozponávanie patologickej reči. Jednotlivé navrhnuté riešenia boli implementované a overené a ich výsledky analyzované medzi sebou a vrámci porovnania s publikovanými článkami. Najlepší dosiahnutý výsledok predstavuje zlepšenie o 2,5% v porovnaní s najnovšími publikovanými výsledkami, za použitia menšieho množstva dát a dosahuje presnosť detekcie patologickej reči na úrovni 82%.

57706

TECHNICAL UNIVERSITY OF KOŠICE

FACULTY OF ELECTRICAL ENGINEERING AND INFORMATICS

Department of Computers and Informatics

DIPLOMA THESIS ASSIGNMENT

Field of study:	**Computer Science**
Study programme:	**Informatics**

Thesis title:

Deep convolutional neural network for detection of pathological speech

Využitie hlbokých neurónových sietí pre rozpoznanie patologickej reči

Student:	**Bc. Lukáš Vavrek**
Supervisor:	**doc. Ing. Peter Drotár, PhD.**
Supervising department:	**Department of Computers and Informatics**
Consultant:	
Consultant`s affiliation:	

Thesis preparation instructions:

The goals of the diploma thesis are:
- provide the state-of-the art of the approaches for pathological speech detection,
- propose and implement neural network able to identify pathological speech,
- perform experiments to validate the proposed approach.

Language of the thesis:	English
Thesis submission deadline:	04.05.2020
Assigned on:	31.10.2019

TECHNICKÁ UNIVERZITA V KOŠICIACH FAKULTA ELEKTROTECHNIKY A INFORMATIKY -02-

12. [illegible]

prof. Ing. Liberios Vokorokos, PhD.

Dean of the Faculty

Declaration

I confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Košice, 4.5.2020 . . . .

Signature

Acknowledgment

I want to thank my supervisor for his time, mentoring, and knowledge shared while working on my master thesis. His guidance, ideas, and general help were essential and helped me learn a lot about the field of machine learning and grow professionally.

I also want to thank my family. My parents, for their love and continual support over the years. My wife, for her patience with me when I had to focus and put hours into work. At last, my biggest thank you belong to my daughter, Alica, for her support, smiles, and love.

Contents

List of Figures

List of Tables

Introduction

Artificial intelligence is everywhere. We may or may not notice it, but it helps us perform our day-to-day tasks easier and better. It is implemented in our mobile phones, computers and perhaps in the infrastructure of a city we live in. With the rise of new powerful computer hardware, it became the number one toolbox for starting a tech-startup. With an unconventional approach, it may help us figure out problems that we had no idea how to solve them, sometimes simply because we are unable to describe them to the computer in a general form.

When solving a problem using machine learning, we tackle it from a completely different perspective. Instead of describing the computer precisely what to do and what to look for - which could be extremely hard and complicated - we let the computer learn by itself. We feed it with a massive amount of data, examples of inputs and outputs, and make use of powerful machine learning algorithms. [1] Deep learning, a more specialized field of machine learning, has proven to be a powerful tool for tackling some of the hardest problems and now is one of the fundamental parts behind modern AI development.

With the use of deep learning, we can explore several different areas. Some problems, like speech to text conversion, language translation, and image recognition, are being researched heavily, which led to significant progress and real-world applications. Others, like the health segment, are still mostly in a state of research.

We became interested in health because of several different reasons. Being able to make some impact and contribute to something that could potentially increase the quality of life for people is the most important one. Because the health segment is vast and offers lots of different problems to tackle, it is necessary to have many people doing the research, and many problems are still not very well solved.

[1] This is only one way of approaching a problem, called supervised learning.

In this thesis, we decided to focus on pathological voice research. It is an interesting problem, as it allows us to work directly with a patient's audio samples. Historically, there are methods with excellent precision but require an understanding of data. In order to work, they extract some metadata from the original record, for example, harmonic-to-the-noise ratio (HRN) or Mel-frequency cepstral coefficient (MFCC). A new modern way of solving the same problem we will try to explore is using deep neural network and let the computer extract everything, what contains any value for calculating the final result.

Our main goal is to build upon, replicate, and possibly improve the current research in a field of pathological voice detection. We use available resources to understand the challenges, approaches, and achieved results and use this knowledge to form a new potential way of solving the same problems. After we design a new approach, the next step will be to design and develop a neural network, so it is possible to test the idea. Achieved results from each idea will then be evaluated and compared to published papers.

1 Pathological voice detection overview

The detection of pathological voice disorders is a complicated process. It requires sophisticated equipment and a trained doctor to perform the diagnosis. People with potential disease have to attend a unique diagnostic process at their local clinic.

The classic process of detecting the pathological voice disorders used today is expensive and time-consuming. Even with the required cost and time effort, the outcome is highly dependent on the doctor's skill and experience.

This hard dependency on medical specialists creates a demand that they always have to be a reasonable amount of skilled specialists for a particular region. Otherwise, the waiting times would be too high, and the costs would increase even more. Most probably, in that situation, not everyone would be able to attend a diagnosis.

Since the cost of classical diagnostic methods used today is already high, not everyone can afford to attend the examination process. Because of the cost involved, there might be cases left without a medical diagnosis. Those patients would lack proper treatment, therefore, putting their health and maybe even life to the risk.

When it comes to voice disorders, it is crucial to get a proper diagnosis at the early stages, as they might be an indicator of more severe problems. One of those problems could potentially be larynx cancer.

Health issues caused by pathological disorders are severe, but they are only one part of the patient's complications. Voice disorders affect the larynx and its ability to produce sound. Health issues may result in irregular vibrations of the vocal folds [1]. Problems with vocal folds may impact the ability of an individual to speak, which could have several consequences. One of them is most likely a reduced quality of social life. People often tend to feel ashamed of their imperfections and might have a problem starting a conversation with strangers that do not have any knowledge about their health issues. This separation from society might eventually result in depression or other psychological difficulties.

Based on the paragraph above, that presented complications and problems of the pathological patients with the voice diseases, there is an inevitable and necessary space for improvement in diagnosing the pathological voice diseases. We want to decrease the time of the diagnostic process, reduce the overall costs, and increase the percentage of diagnosed patients with proper treatment. The general idea is to eliminate all the cons of traditional methods.

There has been an effort to create affordable, noninvasive methods that makes use of available technology to perform the diagnosis successfully. This approach also eliminates potential human error, making it reliable and reducing the need for trained experts in the field. Doctors would have a better tool at their disposal that assists them and enable them to dedicate more time and effort to each patient. Patients might also use the new approach for self-examination.

With the rise of the machine learning field, researchers from all over the world come up with new approaches and methods very frequently. The central part of the research focuses on improving existing methods and making computers be able to solve problems that cannot be solved or are incredibly hard to solve, using traditional algorithmization.

Another part of the research is more focused on making use of invented methods in real life, to solve real-world problems. One of the fields that are profiting from the artificial intelligence (AI) boom is the health domain.

When it comes to pathological voice diagnosis, several methods were already proposed and are under the research. We will take a look at them in the next section.

1.1 Using Mel-Cepstrum Vectors with SVM

In Pathological Voice Classification Using Mel-Cepstrum Vectors and Support Vector Machine research paper [2] (PVC-1), researchers are presenting a supervised classification model trained to classify pathological defects. They developed their model using a dataset of voice samples, being able to classify the pathological defects into three commonly identified vocal disorders: Neoplasm, Vocal Palsy, and

Phonotrauma. The dataset used in their research consists of 50 healthy individuals and 150 subjects with a pathological voice disorder. They obtained the voice samples for their experiments from a voice clinic in a tertiary hospital, Far Eastern Memorial Hospital (FEMH).

They utilized the Mel-frequency cepstral coefficients, called MFCC and its temporal derivatives called MFCC deltas. Mentioned features were extracted directly from the raw audio signals. The SVM classifier uses the extracted features as inputs. The number of computed MFCC coefficients was 15. The hyperparameters of the SVM classifier were tuned using the SHAC algorithm [3]. The primary purpose of a given model, presented in the paper, is to create a baseline for which physicians can diagnose vocal disorders accurately and have a model that other researchers can compare their results.

"Mel-frequency Cepstrum is a representation of a sound signal, based on the linear cosine transform of a log power spectrum on a nonlinear mel scale frequency" [4]. In a mel-frequency cepstrum, the frequency bands are uniformly spaced on the mel scale. The MFCC deltas are computed as temporal derivatives of the MFCC features. Computing MFCC and MFCC deltas serve as the extraction of dynamic features of speech [5].

They used a set of Random Forest models trained on the training data to compute the importance of the 45 dimension input vector. Only features above some threshold (sampled from SHAC algorithm) were used for the training, as inputs for Kernel Support Vector Machine with the Gaussian Radial Basis Function kernel. Figure 1.1 summarizes the overall model and its components.

Because of the dataset size, they have decided to use the 5-fold cross-validation process, instead of having a distinct test set.

The achieved specificity in classifying the binary output, whether the sample is healthy or pathological, was 0.7823. When classifying a concrete disease, the achieved accuracy was 59%.

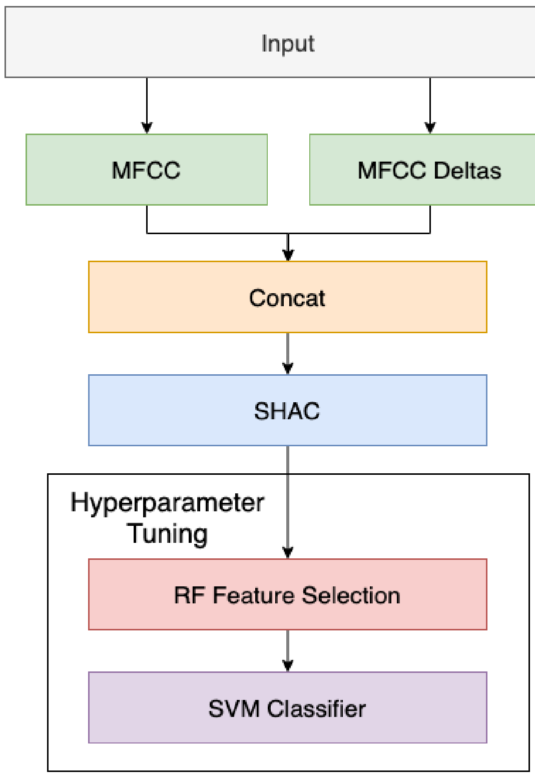


Figure 1.1: Proposed model pipeline, adopted from [2]

1.2 Classification Using Acoustic Features

In this next paper called Pathological Voice Classification Based on a Single Vowel's Acoustic Features [6] (PVC-2), authors describe the approach of classifying pathological voice from a healthy voice, based upon 30 acoustic features derived from a single sound of vowel /a/.

The idea behind their research is to examine an acoustic analysis and find out which factors are affecting the human voice production mechanism. Using this process can lead to the noninvasive diagnosis of diseases. Also, they are investigating the influence of the acoustic features on the accuracy of voice disorder detection when reducing dimensions by the PCA algorithm.

In their research, they used a dataset collected by the Massachusetts Eye and Ear Infirmary (MEEI). It includes several hundreds of voice recordings, belonging to either healthy or pathological individuals. For this particular paper, authors decided to use recordings of a single vowel tone /a/.

For experiments, they have used data from 216 subjects, from which 177 subjects were pathological, and 39 were healthy subjects. Each data point, from a given subject, consists of 30 acoustic features, which are defined by the multiple dimension voice program (MDVP). Examples of used features are Amplitude Perturbation Quotient (%), Degree of Subharmonic components (%), Standard Deviation of the Fundamental Frequency (Hz), and Average Pitch Period (msec).

To be able to classify pathological and healthy samples, the authors designed a two-step process. First, they have made use of Principal Component Analysis (PCA) algorithm to reduce the dimensionality of used data samples from the original 30 to a specified value. (In the original paper, they present a table with results for multiple dimensions, from 1 to 30.)

"The PCA transform is used to reduce dimensionality by rotating the dataset in a way such that the rotated features are statistically uncorrelated" [7]. Data from the PCA, with reduced dimensionality, are then used with the Least-squares support vector machine (LS-SVM) classifier with the RBF kernel function.

To verify the model's performance, they have used 5-fold cross-validation. The K-fold validation helped to acquire the most accurate results, as the data set was too small to have a separate test data set.

The achieved classification accuracy was 98,1%, with sensitivity and specificity 0.925 and 0.994, respectively.

The achieved results are excellent compared to other papers. However, there has been an evidence [8] and criticism against using MEEI dataset for binary pathological voice classification. As the authors point out, the pathological and healthy voice recordings were recorded in two different environments. Thus, it is hard to distinguish whether the classifier is correctly picking the voice features, or the environment difference is essential for the classification outputs.

1.3 Using Fusion of Scores for Pathology Detection

Another paper Voice Pathology Detection on the Saarbrucken Voice Database with Calibration and Fusion of Scores Using MultiFocal Toolkit [9] (PVC-3) uses the Gaussian mixture model-based classifier on multiple datasets.

For their experiments, they decided to use and compare two different datasets. The first one is the Massachusetts Eye and Ear Infirmary (MEEI) database. They have used 226 recordings, corresponding to the vowel /ah/ sustained. From the picked subset, 173 files were representing pathological patients, and 53 recordings belong to healthy subjects. For a second dataset, they have chosen the Saarbrucken Voice Database (SVD), recorded by the Institute of Phonetics of Saarland University. They found this dataset interesting as at the time of writing their paper, SVD was a new and unexplored database. The SVD database contains voice recordings from more than 2000 persons, with multiple data points for each subject. For their experiments, they have decided to use only files with sustained vowels and subjects that are older than 18 years. In total, the subset they use contains 1970 recording sessions, from which 1320 were belonging to pathological patients, and 650 recordings come from healthy subjects.

From the used voice recordings, they have extracted multiple features. Acoustic features characterize the frequency content of the signal and are represented by the MFCC family of parameters. Noise-related features focus on analyzing how much noise does the signal contains. They are represented by the Harmonic-to-Noise Ratio (HNR), Normalized Noise Energy (NNE), and Glottal-to-Noise Excitation Ratio (GNE).

The extracted features from the audio signals are then used to train a generative GMM model [10] for each predicting class. The model is a generalization of the Gaussian model, and it helps to generate much more complicated likelihood functions.

For training the GMM model, they have used the expectation-maximization (EM) algorithm [11], starting with K random Gaussians and executing 10 algorithm iterations. After that, they calculated the log-likelihood ratio between likelihoods for pathological and regular classes for each test file. The decision of the final predicted class utilized the calculated log-likelihood ratio.

For calibration of calibration-sensitive metrics, they have used MultiFocal Toolkit developed in Matlab and designed for calibrating and fusing scores of a language recognition tasks [12]. The authors decided that based on the interest in hard decisions made by the classifier. Examples of calibration-sensitive metrics are correct classification rate (CCR), error rate (ER), sensitivity (S), and specificity (E). An additional metrics authors considered necessary were detection cost function (DCF), also called empirical Bayes risk and its minimum value for the selected operating point (minDCF). Authors used the MultiFocal Toolkit for score calibration and to fuse scorings coming from different recognizers to obtain a better recognizer. They also describe that the idea behind calibration is that scores are converted in such a way that the Bayes decision threshold can be used for making the best possible decision.

Executed experiments using MEEI database were using 30 folds of data, following a process from the [13]. For each test fold, the remaining 29 folds were used for training. They have trained GMMs with three components, but instead of following [11], they have decided to group the same speakers within the same fold. It was done as a prevention from the model recognizing the already seen subject. For the uncalibrated model, they achieved the correct classification rate (CCR) of 92,3%. With calibration in the process, the CCR increased to 94,8%.

Experiments executed using the SVD dataset were using 12 subsets to validate the impact of each vowel and all intonations. 30-fold strategy and three components in the GMM stayed the same as to match the previous experiments. On the other hand, the grouping of the same subject into the same subset is not guaranteed. The possibility of having the same subject in test and validation sets could potentially mean that the model might learn to recognize the speaker and do the classification based on that information instead of strictly relying on provided data. This fact can impact the results, making the overall performance results more optimistic than they are.

When comparing the performance for each vowel, they have found that the recognition rate is slightly better for /a/ vowel, comparing to other vowels. The best achieved CCR was for the /a/ vowel, natural intonation with the result of 67%. Next, they have found that the partial fusion for each vowel, with all four intonations, helped with the classification task. The best results were also achieved for /a/ vowel, with the CCR being 71,8%.

At last, the final global fusion with all vowels and intonations were verified and tested. They have found a significant increase in performance. The achieved CCR results were 79,4%.

1.4 Pathology Detection Using Deep Learning

A scientific paper titled A Deep Learning Method for Pathological Voice Detection using Convolutional Deep Belief Network [1] (PVC-4), describes an approach, where authors used a combination of Convolutional Deep Belief Network and

Convolutional Neural Network (CNN) to classify the pathological and normal healthy voices.

For their experiments, they have chosen to use the Saarbruecken Voice Database (SVD) dataset. During their research, they have developed a deep learning method to discriminate pathological voice and healthy voice automatically. This approach was not yet widespread, and not enough research was focused on it, as DNNs ofter require to learn from a massive amount of data.

They have decided to use sustained vowel /a/ samples at a neutral intonation pitch. The pathologies to represent the pathological samples were limited to organic dysphonia, caused by structural changes in the vocal cord. Diseases that were included are laryngitis, leukoplakia, Reinke's edema, recurrent laryngeal nerve paralysis, vocal fold carcinoma, and vocal fold polyps. The used subset consisted of 482 healthy subjects and 482 pathological patients. The data were divided into a training set and testing set, containing 75% and 25%, respectively.

During the preprocessing stage, at first, they resampled the original recordings to 25kHz. This was done to reduce the amount of data in the feature map, which should eventually speed up the training process. After resampling, they utilized the Short-time Fourier transform (STFT) process to transform the time-domain signal into a spectral-domain signal. For that, each file was divided into 10ms Hamming window segments, with a 50% overlap. At last, they reshaped the spectrogram to 60x155 points to remove part of the data without information.

The classifier then uses the preprocessed data as an input. The classifier is a Convolutional Neural Network (CNN), and it consists of 10 hidden layers. To reduce the resolution in convolutional processing layers and to reduce the computational complexity, they utilized max-pooling layers. After convolutional layers, they used a Dense, fully-connected layer, to apply the final classification. To avoid overfitting problems and to improve generalization, they applied the L2-regularization.

To increase the robustness of a trained network, they decided to use generative models. Generative models should help to improve the overall deep learning network performance on small datasets and to eliminate, or at least reduce, the over-fitting problems. The typical generative model used is the Convolutional Restricted Boltzmann Machine (CRBM). Stacks of CRBM constitutes a Convolutional Deep Belief Network (CDBN). After pre-training the weights in each layer of the classifier, they apply the back-propagation algorithm for fine-tuning the weights to achieve better classification results. The overall network diagram is represented in figure 1.2.

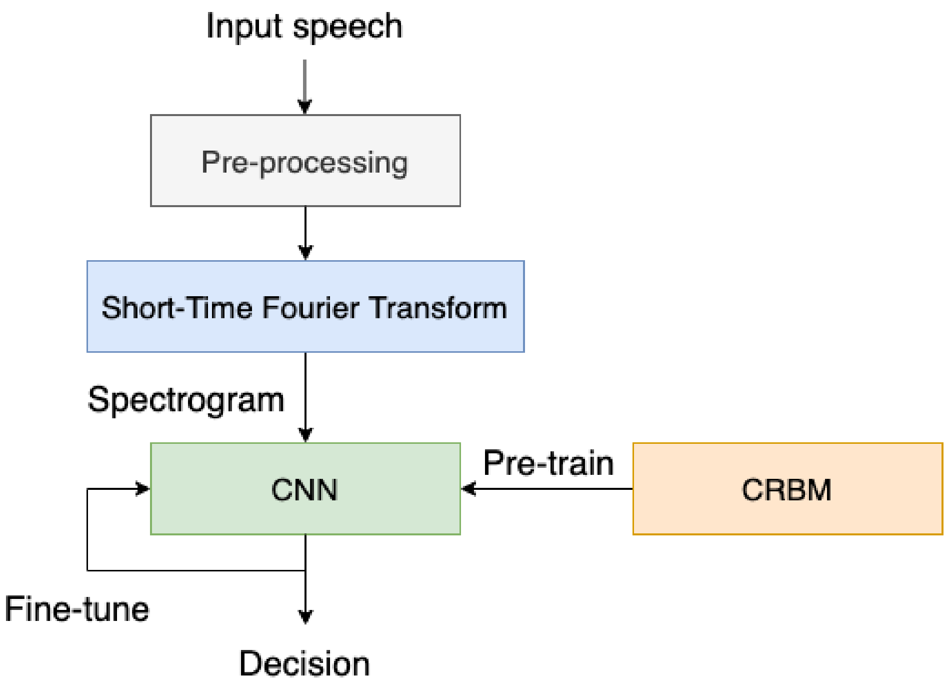


Figure 1.2: Block diagram of proposed pathological voice detection system, adopted from [1]

The achieved accuracy of a pure CNN network was 66% for the validation dataset and 77% for the testing dataset. On the other hand, when using a combination of CNN and CDBN networks, the accuracy of the testing dataset decreased to 71%, while the accuracy on the validation dataset increased to 68%. A decrease of the accuracy on the test dataset indicates better generalization and increased robustness, which decreased the overall network's accuracy on a previously unseen data.

1.5 Summary

In this chapter, we have studied several different methods used for pathological voice detection problem. Each method was different and presented a unique approach compared to the others.

We have seen that the Saarbruecken Voice Database (SVD) presents a challenge for the detection of pathological speech, compared to other datasets. The achieved results using SVD were around 75%, with the best accuracy being 79,5%.

From the analyzed papers, we can extract some useful findings for our future work. As [9] presented, using the bigger dataset (i.e., multiple vowels) and a fusion of multiple models increases the overall performance. [1], on the other hand, showed that the Convolutional Neural Networks (CNNs) could be used for the detection of pathological speech. The usage of CNNs opens up new possibilities for research.

A table 1.1 summarizes results achieved by the mentioned papers.

	Dataset	**Approach used**	**Results**
PVC-1	FEMH	MFCC features, SHMAC + SVM	TNR - 0,807
PVC-2	MEEI	Acoustic features + PCA, LS-SVM	ACC - 98,1%
PVC-3	MEEI, SVD	Acoustic and noise related features, GMM	MEEI ACC - 94,8% SVD ACC - 79,4%
PVC-4	SVD	STFT features, CDBN + CNN	ACC - 77%

Table 1.1: Summary of methods and results presented by analyzed papers

2 Fundamentals of neural networks

Artificial intelligence (AI) is a form of intelligence possessed by the machines.

Z. Voulgaris and Y. E. Bulut, define in their book [14] the AI as "a set of algorithms that make use of information – mainly in the form of data – to make decisions and carry out tasks, much like a human would." Based on the definition, we may say that AI is every single computer program that contains logic to solve a task that is usually performed by humans. Rules or instructions to perform a specific job might be defined (hardcoded) by the programmer writing the program in the form of computer code (symbolic AI), or they might be learned from examples, using some sophisticated algorithms. The most well-known cases that everyone can imagine could be computer programs playing chess. More modern examples include chatbots, advanced bot implementations in games, or applications that create video recommendations based on the viewer's watchlist history.

The connection between artificial intelligence and humans is so deep that Alan Turing came up with the test [15] of a machine's ability to demonstrate intelligent behavior indistinguishable from the human.

Machine learning (ML) is a subset of artificial intelligence. It uses several different algorithms and statistical models to learn how to perform the given task, without explicitly knowing the exact instructions. The actual knowledge is extracted from the data provided during the training process. Multiple different types of machine learning algorithms differ in their approach to solving the problem. For example, most used supervised learning is a process of learning how to map the input data to the expected results.

On the other hand, clustering or dimensionality reduction algorithms are from the group of unsupervised algorithms. They analyze the data and output of the computed results. They have proven to be very useful, for example, during the exploratory data analysis process.

According to the F. Chollet [16], "machine learning opened the door to a new programming paradigm, where humans input data as well as the answers expected from the data, and the program learns the rules."

Neural networks (NN) are a specific subset of the machine learning field. In general, J. J. Hopfield [17] defines neural networks as "networks or circuits of neurons composed of artificial neurons or nodes." To improve the clarity, we differentiate between biological neural networks made up of biological neurons and artificial neural networks that we build and use to solve problems.

2.1 Building blocks of neural networks

The fundamental block of neural networks is the artificial neuron. Artificial neurons are mathematical functions that receive one or more inputs and calculate a sum to produce a single output. The calculation is often referred to as a linear combination. Neurons are a generalization of perceptrons, introduced by F. Rosenblatt [18]. During the computation, each input is multiplied with a belonging weight. Weights can be changed, and they represent the neuron's ability to be adapted by modifying weight values. Internal diagram of artificial neuron is displayed in diagram 2.1.

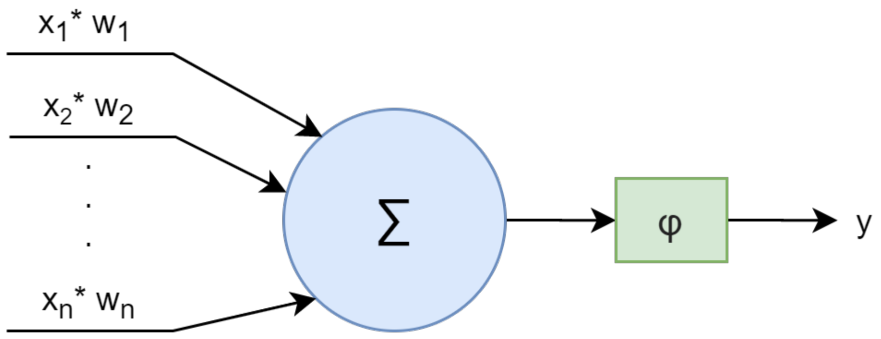


Figure 2.1: Artificial neuron

The function of the artificial neuron can be mathematically described as:

$$y = \phi\left(\sum_{i=1}^{n} x_i * w_i\right)$$

The activation function adds a non-linearity to the computation of the result y. The most popular activation functions are rectified linear unit (RELU) and tangens hyperbolicus (tanh), which graphs are illustrated in diagram 2.2.

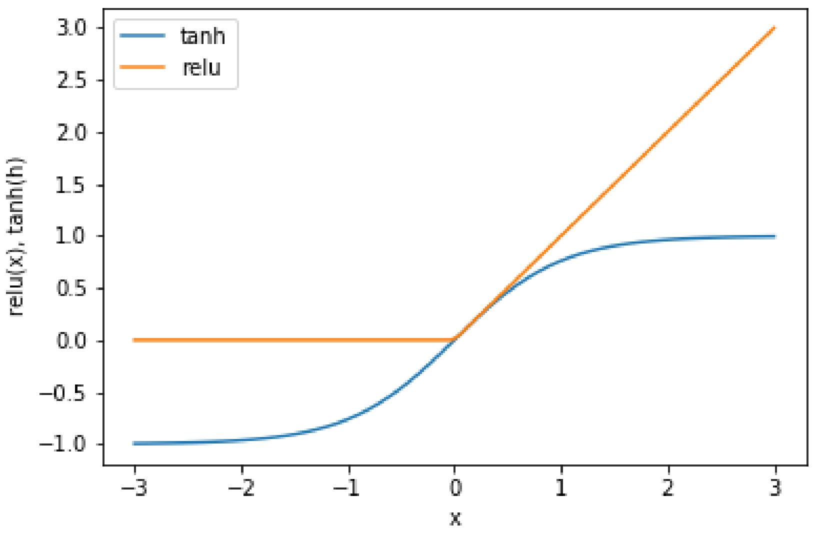


Figure 2.2: RELU and tanh activation functions

The neural network combines neurons to form a structure (hence the name network), that can adapt and model sophisticated data scenarios. It consists of an input layer, one or more hidden layers, and one output layer. The network structure build-up from the input layer with four inputs, a single hidden layer with seven nodes and an output layer with two outputs, is illustrated in diagram 2.3.

There are two groups of artificial neural networks, feed-forward, and recurrent neural networks. Feedforward neural networks, also called as multilayer perceptrons (MLP), combine neurons in such a way that neurons in any layers are only connected to neurons in the next layer, which means that they cannot form any cycle. Recurrent neural networks, on the other hand, do contain cycles in their internal structure, and their architecture can be much more complicated. Examples of recurrent neural networks are Long short-term memory (LSTM) or Gated recurrent units (GRU).

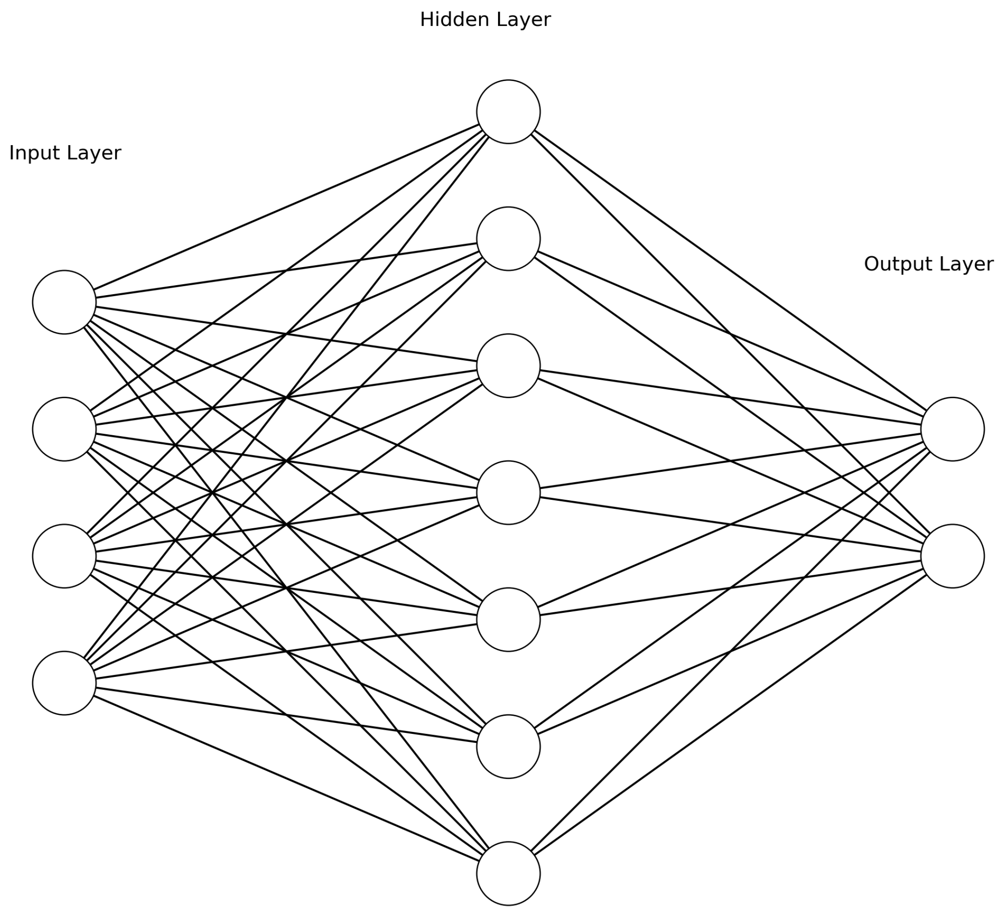


Figure 2.3: Simple neural network with 1 hidden layer

2.2 How do neural networks learn?

As F. Chollet defined in his book [16], "A machine-learning model transforms its input data into meaningful outputs, a process that is "learned" from exposure to known examples of inputs and outputs." However, how exactly is this learning process achieved?

The algorithm learns by continually adjusting the weights of corresponding neuron connections within the neural network based on a feedback signal. It is done repeatedly by exposing the network to training data over and over, which is a process called training loop. A single exposure to training data is called a batch. After each batch, the training algorithm computes the achieved error rate, defined to precisely fit the solving task. The error rate is a difference between predicted values and actual values, and it is computed using loss function. The goal of the learning process is to learn how to map the network's inputs to correct targets by minimizing the loss between predicted results and actual results. Because of that, the correct definition of the error metric is crucial for achieving wanted progress.

Once the model knows the error rate, weights have to be adjusted to reduce the error and provide better predictions. Adjusting weights is based on a feature, that all operations in the neural network are differentiable, which allows us to compute the gradient (a derivative of a tensor operation) of the loss with regard to network coefficients.

In an ideal world, we can find the best weight values by solving

$$gradient(f)(W) = 0$$

function [16]. Solving this equation is, however impossible in real-world scenarios as modern networks have too many parameters.

In all modern implementations, adjusting weights is achieved using the combination of backpropagation [19] and gradient descent algorithms. The backpropagation algorithm computes the gradient of the loss function, one layer at the time starting from the end, with respect to all weights using the chain rule of calculus. Gradient descent algorithm is an algorithm for finding a local minimum of a differentiable function. In practice, more sophisticated versions of stochastic gradient descent algorithm are used, that adjust the weights not only by computed gradient but also based on previous adjustments. Taking into account previous adjustments maintains momentum and helps to avoid the local minimum trap problem.

2.3 Deep learning

Deep learning is a subfield of the machine learning field. It focuses on building neural networks with multiple hidden networks, using successive layers of data representation. Modern deep learning models usually contain between tens and hundreds of hidden layers. It matches an idea of how we, humans, usually make use of the information by combining abstract knowledge to form concrete knowledge. Earlier layers are supposed to extract an abstract, general knowledge and which should be reused by the following layer, gradually until we have a final concrete prediction. The count of layers in a deep learning model is referred to as the depth of a model.

An example of a deep neural network is illustrated in figure 2.4.

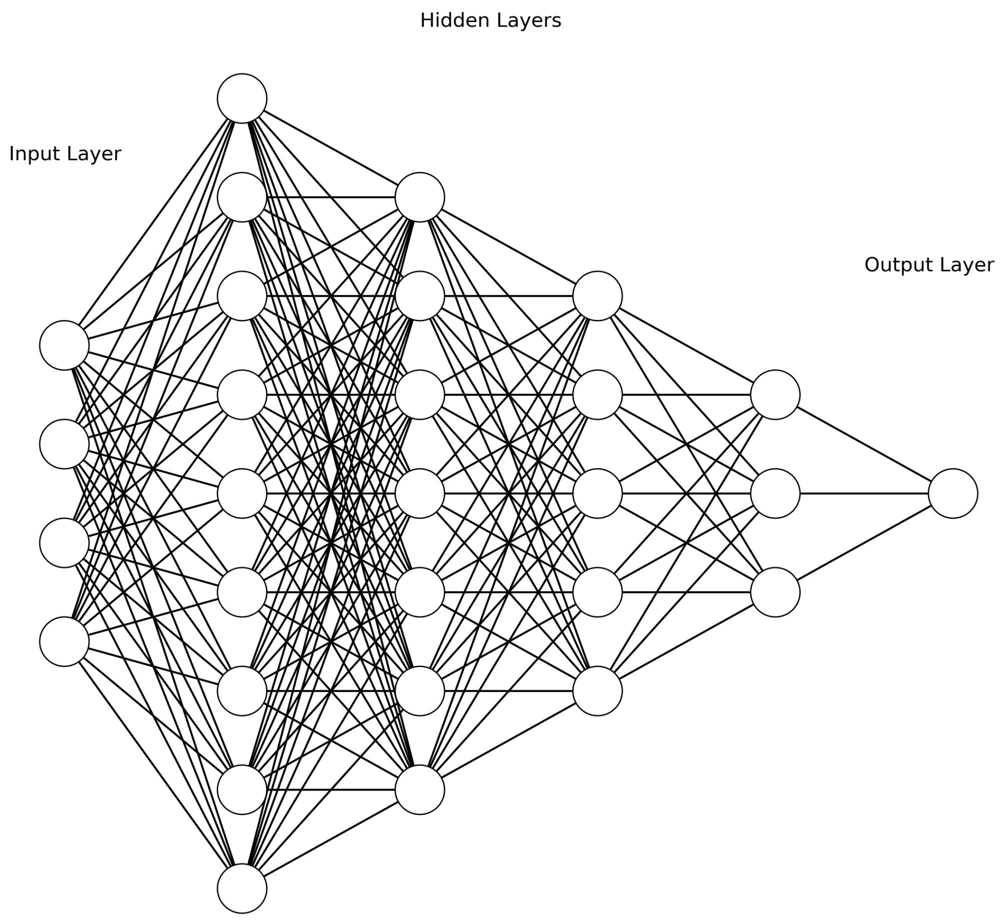


Figure 2.4: Deep neural network with 4 hidden layers

Deep learning networks became very popular in the past few years. They completely replaced all classical approaches for tasks like computer vision and speech recognition. The performance they provide is unmatched for many types of problems. Despite the performance, they also eliminated feature engineering, a task that is crucial yet often complicated or tricky. By design, they do not require massive data preprocessing as they can learn the essential features right from the "raw" data, therefore automate this process entirely. This advantage leads to more flexible, simplified machine learning pipelines.

Deep learning models gain their advantage from the sophisticated yet straightforward learning process. They are not only stacked layers. Instead, they are multiple layers that are used and adjusted together to form a robust model. As F. Chollet stated [16], "there are two essential characteristics of how deep learning learns from data: the incremental, layer-by-layer way in which increasingly complex representations are developed, and the fact that these intermediate incremental representations are learned jointly, each layer being updated to follow both the representational needs of the layer above and the needs of the layer below."

2.3.1 Convolutional neural networks

Convolutional neural networks (CNN) are a go-to algorithm for all modern computer vision problems. They are a specific version of deep neural networks. Convolutional neural networks were first used in 1990 by the group of researchers around Yann LeCun when they used the network for handwritten digit recognition problem [20].

CNNs make use of hierarchical patterns presented within the data. Each layer is building upon more abstract patterns extracted by the previous layer, moving towards a final prediction at the end. In the last layer, CNN can identify the presence of a complicated pattern and use this information to form a final prediction. Convolutional neural networks proved to be useful for tasks, including image recognition, image classification, and medical image analysis.

The name, convolutional neural network, is based upon a usage of a mathematical operation, called convolution, which is highly used in image processing algorithms. Convolution operation translates an input feature map to an output feature map. An input of the first convolution is an image with three dimensions: height, width, and depth, representing the number of channels in the image. RGB images have three channels, whereas monochromatic images have only one. A depth of the output of a convolution operation is dependent on a number of used filters.

The filter is a core block of a convolution. It is a matrix (for example, of size 5x5x3 for an image with three channels), and it encodes a specific aspect of the input data. During the forward pass of the convolution layer, the filter convolves (slides) across the width and the height of the input feature map. For each input at a position of a filter and filter itself, the dot product is computed. An example of the computation is visualized in figure 2.5.

For each convolution layer, there are multiple filters learned, and their count is a parameter of a given layer. The goal of the convolution layer is to learn the right

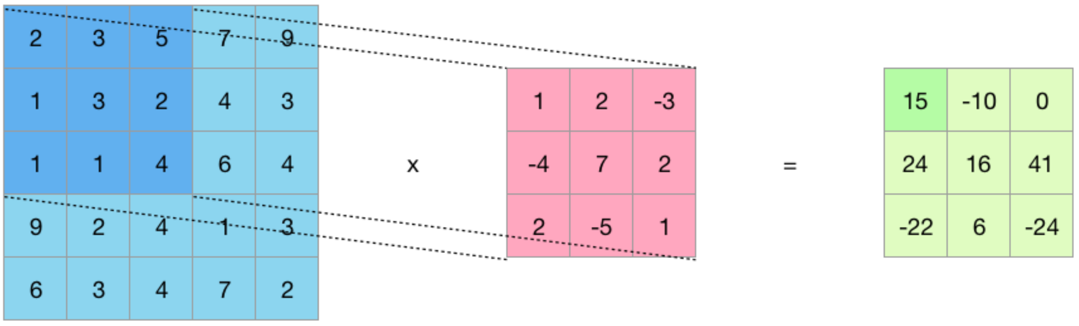


Figure 2.5: Convolution operation

filters that provide the best output for the next layer. Examples of encoded information of a filter might be detecting a circle, detecting the presence of a human face, or detecting the presence of an elephant's ear. Filters are exploiting a feature called local connectivity, which uses the idea that it is more practical to connect neurons to a local region of the input volume, rather than trying to process the input as a whole.

Another common technique used within convolutional neural networks is pooling operation. The pooling operation is represented by a single two-dimensional matrix, a filter, which divides the input feature map into the small parts of a size of a filter and operates on them, as presented in figure 2.6. Pooling operates independently on each depth slice of the input feature map.

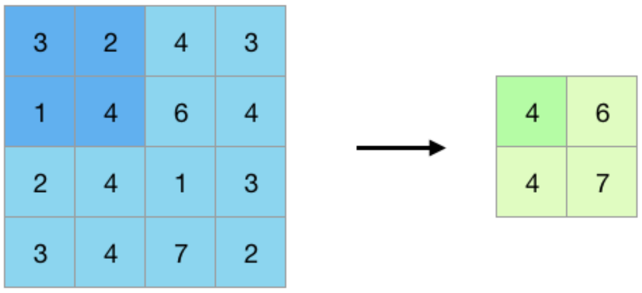


Figure 2.6: Max pooling operation with 2x2 filter

The most common types of pooling used are average pooling and max pooling. A max-pooling filter of size 2x2 takes four values as an input and outputs a single value, the maximum of input values, therefore, downsizing the input it operates on. Because of the downsizing characteristic, the pooling operation is usually inserted in-between successive convolutional layers. It reduces the size of the data representation, passed between convolutional layers. Reducing the size of passed information is useful for reducing the number of parameters in a network, which directly helps to prevent the overfitting problem.

The most significant advantage of CNN architecture, compare to regular neural networks, is their ability to scale-up to more significant inputs, which is essential when dealing with images.

Convolutional neural networks are very popular for their versatility and performance and are being studded heavily [21]. Currently, there are many well-known architectures of CNNs. One of the most popular ones are VGG16 [22], ResNet [23], DenseNet [24] and NasNet [25].

2.4 Transfer learning

Transfer learning is a technique, used in machine learning than allows us to reuse acquired knowledge gained during learning to solve one problem and apply it to a different, somehow related problem. An example of reusing the knowledge might be using a network trained for animal species recognition to be also reused for humans.

The process of reusing network to another problem is usually referred to as a transfer between domains. Domains, in this case, represent the scope of a particular problem, either the source (original) problem or the destination (new) problem. In order to make this transfer as smooth and effective as possible, we go through the process of adaptation. The more related are the source and the destination problems, the less adaptation (if any) is needed.

There are multiple ways we can do the transfer learning, using deep learning networks. The simplest and most straightforward process uses the existing pre-trained neural network with weights as a base. In order to be able to extract more abstract information, we remove to last layers (usually dense, fully connected layers, that are responsible for final classification) and attach our own, custom classifier on top of the base network. The individual layers of the base network could be either frozen, meaning that weights will stay the same, or not, allowing them to adapt better to a destination problem.

The transfer learning technique proved itself to be useful in several different problems. It helps to reduce the overall training time, build more robust networks, and train powerful networks with a small dataset for destination problem.

There are multiple ways of doing transfer learning within a speech processing field. In general, all techniques are based on an idea, that the features learned by deep neural network models are abstract at low layers, therefore language/gender/person independent. A cross-lingual and multilingual transfer uses patterns shared across languages to deliver better models and improve statistical strength in multilingual conditions [26]. A speaker adaptation process focuses on adapting a general model to a specific speaker. A speaker adaptation process might be beneficial for personalized experiences, such as for speech understanding in voice assistants. Another interesting approach, called a model transfer, exploits the knowledge of the so-called teacher model and uses it to guide the training of the child model.

2.5 Evaluation metrics for binary prediction

When experimenting with machine learning models, we need to have the ability to compare the performance of the two models. There are several different ways of doing that. The most two used ones are accuracy and loss.

The accuracy metric is used to measure the algorithm's performance in an interpretable way by measuring how accurate the model's predictions are compared to the referential values. The loss, on the other hand, measures how well the model is doing. It is computed as a sum of errors made and is used to optimize a trained machine learning algorithm.

Often, we are dealing with the binary prediction models, when the answer is in the form of boolean. Yes or no, black or white, cat or dog, or in a case of pathological voice detection, healthy or pathological. For binary predictions, the very popular method of representing results and evaluating model performance is a confusion matrix.

A confusion matrix is a two by two matrix, as shown in the table XY. The organization of the confusion matrix allows us to see the model's performance and be able to sense how does it behave.

Rows are representing the predicted classes, i.e., the output of our model. Columns, on the other hand, represent the actual classes taken from our reference data.

There are many derivations we can calculate from the confusion matrix. However, the most commonly used ones are accuracy (ACC), sensitivity (true positive rate, TPR) and specificity (true negative rate - TNR). They are calculated as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$TPR = \frac{TP}{TP + FN}$$

$$TNR = \frac{TN}{TN + FP}$$

Using accuracy, sensitivity, and specificity, we can tune our final model according to our needs. Sensitivity, for example, is beneficial when we need to identify all positive samples, while it is crucial for avoiding false negatives [7]. An example of such a task would be classifying cancer when it is safer to classify as sick event healthy patients, rather than not classifying patients with the disease [7].

3 Processing audio recordings

The way we store audio-recording affects some essential characteristics and features. We can store audio in lossless formats like FLAC (Free Lossless Audio Codec), which compresses raw audio-data without losing any information and quality or use MP3 format with the lossy but storage-efficient compression algorithm.

When working with audio-data as source material for some follow-up analysis and examination, we want to work with raw audio data captured directly by our microphone. In raw data, we can search for patterns, characteristics, and anomalies, that will give us a better understanding of recorded audio.

The most commonly used audio-format for that purpose is WAV (Waveform Audio File Format, sometimes called wave files). Wave files store raw uncompressed audio using the linear pulse-code modulation format (LPCM). Originally designed and developed by Microsoft and IBM, wave files became de facto standard in the industry. Because of their popularity, they are very well supported by audio editing software designed for amateurs and professionals. They are also lots of open source libraries for all popular programming languages that equip developers with tooling to interact with wave files directly. Based on the reasons above, wave files are prevalent among audiophiles, musicians, and researchers working with audio data.

3.1 Visualizing audio data

There are multiple ways of visualizing audio data, but the most common are oscillograms and spectrograms. Options on varieties of those two methods are infinite, and the only limitation is our imagination.

Oscillograms present waveforms and amplitude of the audio signal over time.

Figure 3.1 shows an example of the /a/ vowel sample. With just basic understanding, we can see that it is a continuous signal of sound and that the amplitude is decreasing over time as the patient is losing their breath.

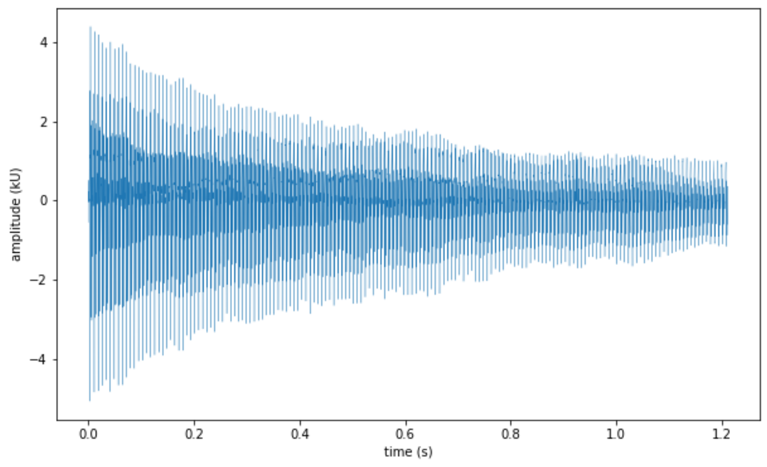


Figure 3.1: Oscillogram of the pathological sample of /a/ vowel

On the other hand, figure 3.2 contains the spoken phrase "Guten Morgen". Compared to the previous oscillogram example, we see that the signal is choppy and looks completely different. That is caused by the patient going through each letter of the phrase. Further analysis would reveal some interesting information like letter recognition, subject's gender classification, and more.

Oscillograms can help us understand audio data we are working with from one side. Having another perspective is always a good idea, especially when doing research. Sometimes, we do not care about amplitudes, but frequencies in a given sample.

Spectrograms help us visualize the frequency of the sound over time. Spectrograms preserve the amplitude (the energy presented in each frequency) and represent it as the intensity of the color. A brighter color represents more energy.

Figures 3.3 and 3.4 contain spectrograms of previously mentioned audio samples. As you can see, a lot more data is present in them as the information is encoded in three dimensions.

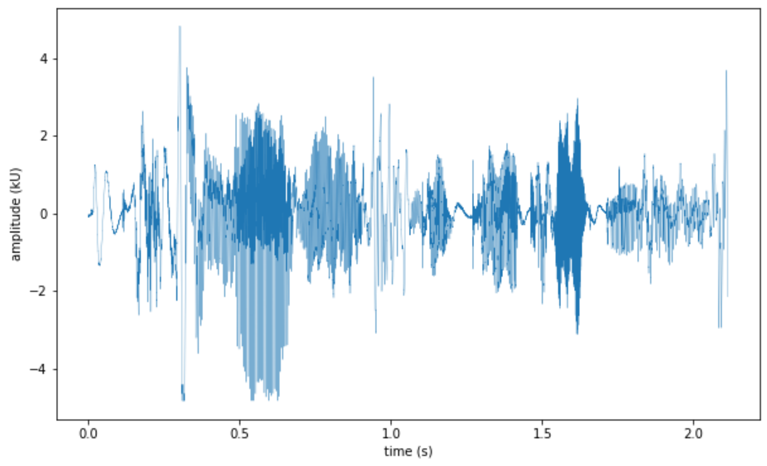


Figure 3.2: Oscillogram of the healthy sample of "Guten Morgen" phrase

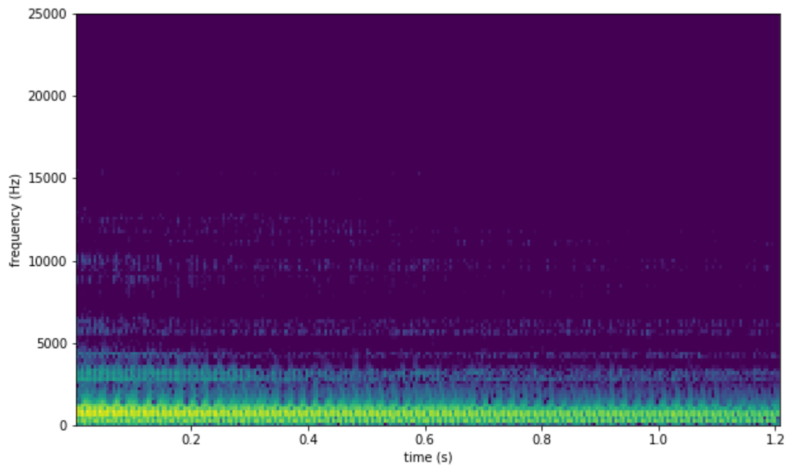


Figure 3.3: Spectrogram of the pathological sample of /a/ vowel

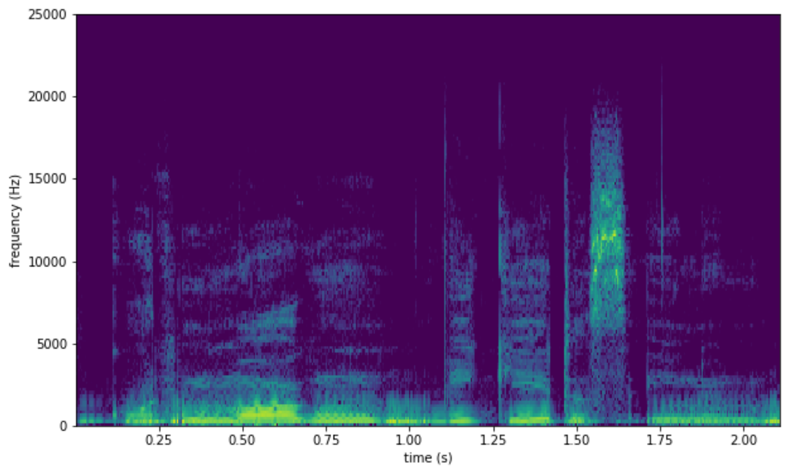


Figure 3.4: Spectrogram of the healthy sample of "Guten Morgen" phrase

3.2 Acoustic analysis

Audio files contain information about the recorded audio signal. When working with speech or voice recordings, we are interested in multiple voice specific features that we can extract from a recorded signal. The process of extracting features out of the voice signal is called acoustic analysis. The main advantage of performing acoustic analysis is that it is a non-invasive tool and provides objective diagnosis of a voice signal [27]. Based on the nature of extracted features, they can belong to either the acoustic features group or noise-related features group.

The measures done as a part of acoustic analysis, when proven reliable and reproducible, provide a means of following changes in the voice over time or between subjects [27]. There are multiple acoustic measurements, that can be computed for any voice signal and are a foundation of acoustics. The most commonly used are Fundamental Frequency, Jitter, Intensity, Shimmer, and Signal-to-Noise Ration.

"Fundamental Frequency is an acoustic measure that directly reflects the rate of vocal fold vibration" [27]. It varies between the population, and it is influenced by the age and sex of a speaker [28]. "Vocal Fundamental Frequency is reflective of the biomechanical characteristics of the vocal folds as they interact with glottal airflow" [27]. Speakers may also be able to adjust their Fundamental Frequency based on the situation. The ability to do it "gives information regarding the mechanical adequacy of the laryngeal structures, and about the precision of laryngeal control" [29].

"Jitter or frequency perturbation is defined as small, cycle-to-cycle changes of the period that occur during phonation which is not accounted for by voluntary changes in frequency" [27]. Jitter is a useful feature that can be indirectly used in pathological voice classification as healthy voices should have little jitter. In contrast, voices with the presence of hoarseness or breathiness that can be an indication of voice pathology are expected to have higher degrees of jitter [28].

"Intensity is defined as power per unit area. Vocal intensity is dependent on the interaction of subglottal pressure, biomechanics, and aerodynamics at the level of the vocal fold as well as the status of the vocal tract" [27] [30].

"Shimmer measures small, cycle-to-cycle changes of amplitude which occur during phonation and quantify short-term amplitude instability" [27].

"Signal-to-Noise Ratio is the ratio of the total energy of a voice signal to the energy of the aperiodic component of the voice signal" [27] [29].

3.2.1 Acoustic features

Acoustic features characterize the frequency content of a signal [9]. They are represented by the MFCC family of parameters and are widely used for speech processing and voice analysis tasks. They are extracted using a frequency analysis process that analyzes the audio signal based on the human perception of the sound. For pathological voice classification tasks, those features are important as they are based on the fact that an experienced medical specialist can detect a presence of a voice disorder by listening to the signal or hearing out the patient [31].

3.2.2 Noise related features

Noise related features are, as the name suggests, focused on measuring the signal quality or the quantity of presented noise [9]. The utilization of noise-related features for pathology classification tasks is based on the assumption that the pathological voice signals contain in general more noise compared to the healthy recordings. The most popular noise-related features are Harmonic-to-Noise Ratio (HNR), Normalized Noise Energy (NNE), and Glottal-to-Noise Excitation Ratio (GNE).

Harmonic-to-Noise Ratio objectively measures the perceptual feeling of hoarseness presented in a voice [32] [9]. Said, it can also be defined as a measurement of voice pureness [33]. It is based on calculating the ratio of the energy of the harmonics related to the noise energy present in the voice [27].

"Normalized Noise Energy is a measurement of the noise present in the voice respect to the total energy" [9] [34]. It has proven to be useful for detecting glottic cancer, recurrent nerve paralysis, and vocal cord disease nodules [34].

"Glottal-to-Noise Excitation Ratio parameter compares the amount of signal due to vocal folds vibration with the amount of signal due to noise produced by air turbulences produced during phonation" [9] [35]. "GNE is based on the correlation between Hilbert envelopes of different frequency channels extracted from the inverse filtering of the speech signal" [35]. GNE can often be an indication of a breathiness presence in the voice signal.

4 Detection of pathological speech

When designing a solution, it is fundamental to build on top of the existing research. The latest machine learning trends focus on designing multi-layer deep learning networks, with minimized data preprocessing algorithms. The general idea is to let the network decide what is essential for it and what data pieces contain the real value.

We started with the idea that we can visualize voice recordings and represent them as images. Therefore, we have an opportunity to treat them not only as audio files but also as pictures.

When it comes to pictures, the latest machine learning research is focusing on this direction. Image recognition and object detection, just to name a few, are all used as a fundamental problem for ideas like self-driving cars.

When working with images, the latest machine learning trends focus on the development and usage of Convolutional Neural Networks [36]. They showed an ability to learn abstract things and patterns and use that information for detecting more complicated objects within the given picture. Most of state of the art modern neural networks are using Convolutional Neural Networks inside their internal architecture.

We have decided to reuse the power of CNNs and build our classifier on top of them. The technique we used is called transfer learning. It is a process that allows us to build sophisticated networks with limited data and resources.

4.1 Data

In the following experiments, we used the Saarbrucken Voice Database[37] as our source dataset. It is a collection of voice recordings from more than 2000 persons, with several different samples for each subject. They are recordings of /i/, /a/ and

/u/ vowels in ordinary, lower, higher and rising-falling pitch and "Guten Morgen, wie geht es Ihnen?" sentence. It can be accessed by a web interface, which provides users with an ability to query the database for needed data. For example, we can filter by the subject's age, gender or pathologies, and disorders.

In our experiments, we have used both a complete and a reduced dataset. The complete dataset consists of many pathologies and subjects. It is useful for experimental work and analysis. The reduced dataset consists of pathologies that are all organic dysphonia, which is caused by structural changes in the vocal cord. The pathologies considered as organic dysphonia are laryngitis, leukoplakia, Reinke's edema, recurrent laryngeal nerve paralysis, vocal fold carcinoma, vocal fold polyps. All described experiments are using reduced dataset as we are mostly interested in diseases caused by injuries to the vocal folds, or near them. This decision limits the data, which reduces the overall training time and simplifies the overall task, as the neural network model can focus its understanding to a reduced pathologies list. By using reduced datasets, we are also able to compare ourselves with the papers which use the same subset, for example, [1]. We choose an equal number of healthy subjects randomly to balance the dataset.

The complete dataset consists of 677 healthy subjects and 1354 pathological subjects. The reduced dataset consists of 506 pathological subjects.

Before each test, we filter out samples that are very short in length compared to the others. We also tried to fill the void with some default values, but we did not see any improvement. To maintain a robust dataset, we also balance the data in both groups. The balancing process is done by randomly removing subjects from the group that contains more samples. This process ensures that the network will not be biased towards a larger group. On the other hand, we have to sacrifice potentially valuable data.

During all of our experiments, we split data into three parts. Training data are 60%, validation data are 20%, and test data are 20%. We use stratified splits that will ensure that the proportion of values in produced groups will stay consistent according to provided data.

4.2 Methods/tools used

To verify the ideas presented in published research papers and newly designed approaches, we write computer programs. Within those programs, we visualize the data, do a data preprocessing, model neural networks, train them, and evaluate them using achieved results.

We write all of our computer programs in the Python programming language. Python [38] is a general-purpose, high-level programming language. Together with R, it has become a standard programming language used by the machine learning community, as it allows researchers and programmers to implement ideas clearly and understandably. Because of its popularity, there are several different machine learning libraries that we make use of to our advantage.

For data analysis and traditional machine learning models, we use Scikit-learn (sklearn) [39]. We also make use of Keras [40]. Keras is a high-level neural network library that enables us to build neural network models quickly and easily, making it an excellent tool for fast prototyping ideas. As a machine learning, tensor-based, low-level "backend" library used by Keras, we decided to go with TensorFlow [41].

Most computations during the data preprocessing or data analysis stage are using the NumPy [42] library. For audio analysis and spectrograms conversion, we utilized LibROSA [43] package. All visualizations are created using the Matplotlib [44] library.

For the development and execution of our written programs, we use Jupyter Notebooks. [45]. Jupyter Notebook is a top-rated, website based development tool that provides the ability to write clear, descriptive programs with built-in documentation and visualization.

All experiments were run on the computer with the AMD Ryzen Threadripper 1900X 8-Core CPU, 64 GB of RAM, and two NVIDIA GeForce RTX 2080 graphics cards.

4.3 CNN single vowel approach

In our initial approach, we have decided to try a straightforward and simple solution. As a base network for transfer learning, we've decided to use the VGG16 [22] network which internal architecture is presented in figure 4.1. The network was pre-trained on the ImageNet [46] dataset, and its original purpose is to classify objects from pictures.

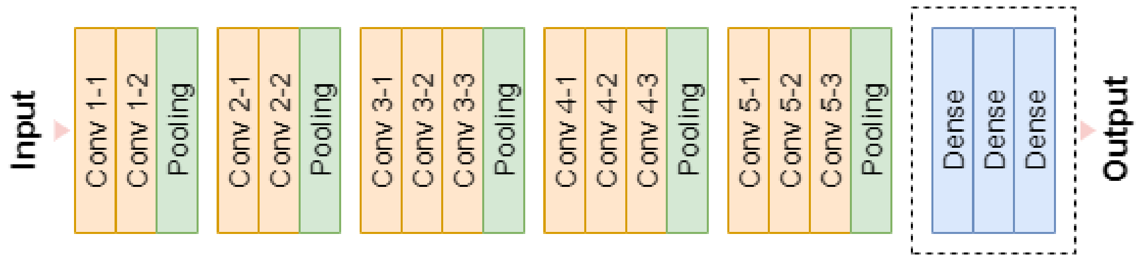


Figure 4.1: VGG16 internal architecture

During the process of transfer learning, we take the network and freeze all its layers, so they do not adjust to our new data. It is also a performance improvement as we do not need to add its internal layers into backpropagation calculation.

The VGG16 neural network expects input in the form of raw image pixel data in RGB form. That means that it expects three channels: red, green, and blue. The data that we get, when we convert our wav audio file sample to the spectrogram form, have only one channel. To ensure compatibility, we have decided to duplicate our spectrogram three times, once for each expected channel. This action should not have any impact on the VGG16's data processing ability, but it will allow us to reuse the robust, already pretrained network.

Since we are not interested in predicting objects classifications from pictures, we remove dense layers from the base network. By removing dense layers, we can access more "raw" data, which allows us to build our classifier on top of them.

The classifier we will put on top of the CGG16 network will be trained on our data so that it can learn and achieve the desired predictions on the target problem. Its internal structure needs to be designed during a process of trial and error, as there is no "proper" way to do it. What works for one problem may not necessarily work for others.

At least we can reuse the knowledge from the machine learning community. We have decided to use dense layers and experiment with the count of hidden layers and neurons in them. We also tried some generalization techniques to observe their impact on the network's performance.

We have started with a basic dense layer-based classifier with only one hidden layer and adjusted it based on the results and observed behavior.

We have found two classifiers architectures whose results were promising.

4.3.1 Setting a base with single layer classifier

Starting with a single layer classifier allowed us to set a base result for all future experiments. Because the network contained only a small portion of trainable parameters, it did not have a problem with overfitting, and we were able to observe, whether this approach can generalize to provided data.

Diagram 4.2 represents an architecture with one hidden layer of 32 neurons. The input for it is an output from the VGG16 network, which we first flattened, so it is only one-dimensional. Between the hidden layer and an output layer is a dropout layer with a 0.5 coefficient, which means that the dropout layer will set 50% of each information detail to zero during each training iteration. We found that it improves the network's ability to learn and generalize significantly. Without it, the network had trouble making sense of the data, and its results were highly affected by the initial data distribution, which is, in our case, always random. We can say that it was not able to learn and perform an action it should.

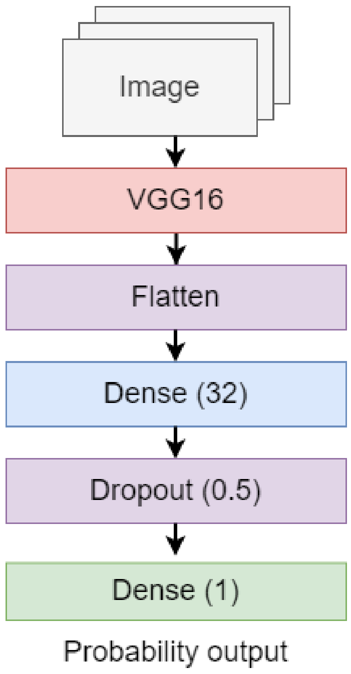
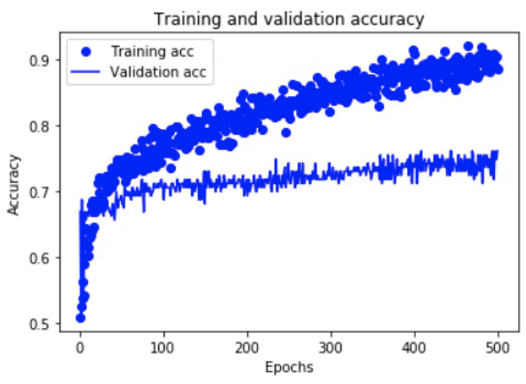


Figure 4.2: Classifier with 1 hidden layer

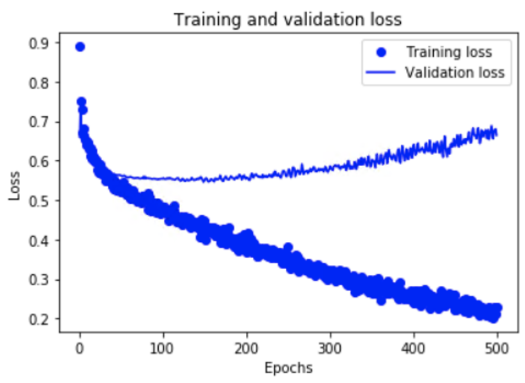
From the diagram 4.3a that represents the training and validation accuracy for epochs, we can see that we run our experiment over the 500 epochs. The network was able to learn, and it achieved 70% accuracy on the validation data, approximately around epoch number 70. From that point, its improvement was very subtle, and it increased by about 5%.

Training accuracy is improving with each epoch, as we intentionally tried to see the correlation between those two datasets (training and validation).

From the training and validation loss diagram 4.3b, we see almost the same trend with one exception. It clearly shows that we intentionally overfitted the network on the training data, with an increasing number of epochs. With the networks starting to overfit, the actual performance dropped as the loss on the validation data began to increase. Based on the graph, we can say that the network performed at its peak after around 100 training epochs.



(a) Training and validation accuracy



(b) Training and validation loss

Figure 4.3: Single layer classifier, training metrics

After we finished with the training process, we proceed to validate network performance. We used a test dataset that contains the data only from the subjects that the network never saw. From the confusion matrix diagram 4.4, we see that the network learned how to recognize pathological and healthy samples. The network achieved an accuracy of 74,23%.

4.3.2 Experimenting with network size

As we already had a base result, we wanted to know whether the network's performance can be improved by adding hidden layers and increasing the number of neurons in them. We found that in general, the network achieved comparable results. The accuracy ranged between 73,6% and 76,4%, although we found one

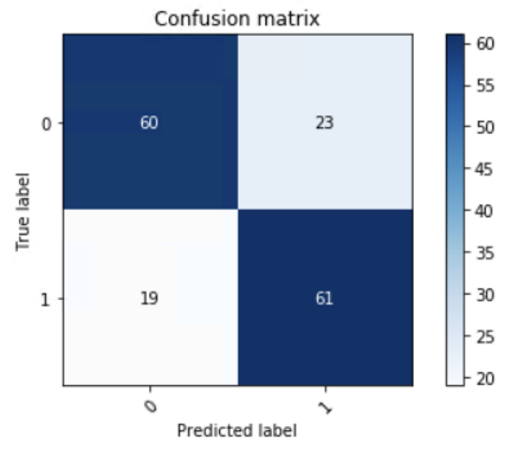


Figure 4.4: Single layer classifier, confusion matrix

exception, which is described below. However, we were unable to test networks with the five and more hidden layers within the classifier due to memory allocation issues.

The network that its diagram is shown in figure 4.5, extends the initial architecture with one more layer with 16 hidden neurons. This layer was placed directly between the first hidden layer and a dropout layer.

As we can see from the diagram 4.6a, we have run our experiment over a 100 training epochs. From the training and validation accuracy graph, the network was a bit unstable during the first 50 epochs. However, it managed to stabilize itself, and its performance ended up around 72% for validation data.

Training and validation loss diagram 4.6b shows that the loss decreased with more epochs, and it reached about 0.57.

When validating the network's performance on the test dataset, we saw that the accuracy increased to 79,14%. From the confusion matrix presented in diagram 4.7 we see, that the network was able to generalized well and achieved better results on new data.

Increasing the network size further by adding more layers with more neurons in them slowly degraded the network's overall performance. We have seen that the network with lots of trainable parameters was very prone to overfitting on training data. Increasing the dropout did not help, as with less information, the network was unable to learn anything.

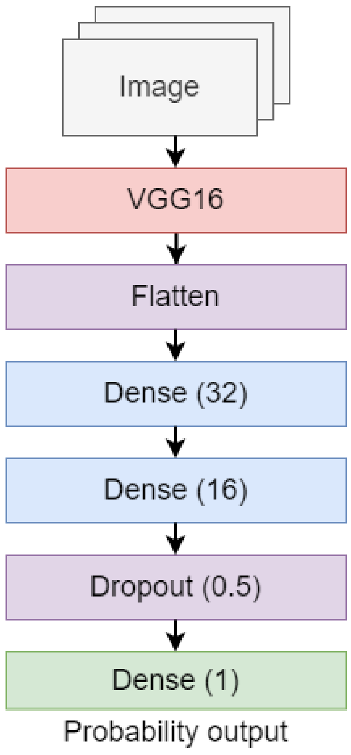
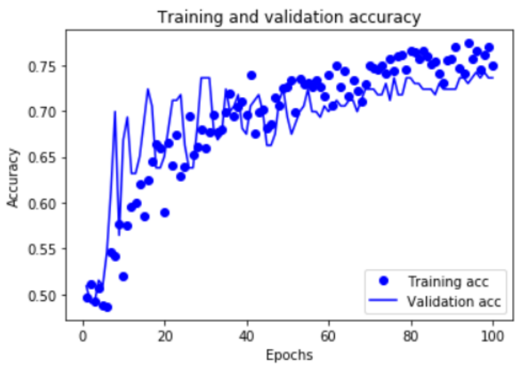
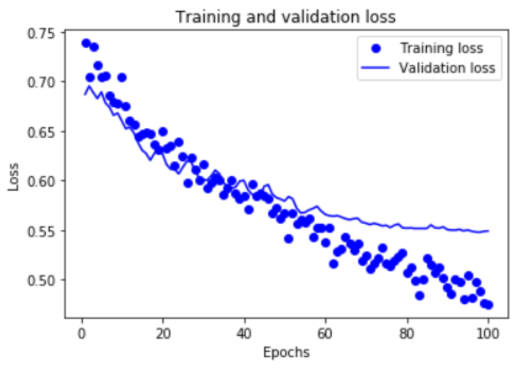


Figure 4.5: Classifier with 2 hidden layers



(a) Training and validation accuracy



(b) Training and validation loss

Figure 4.6: Enhanced classifier with two layers, training metrics

A table 4.1 contains a summary of results of executed CNN single vowel experiments.

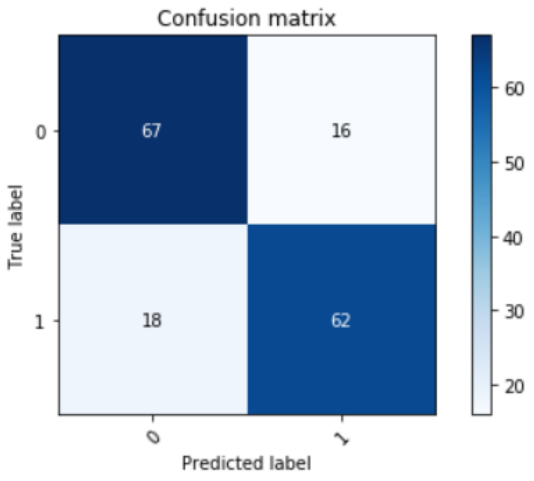


Figure 4.7: Enhanced classifier with two layers, confusion matrix

	Accuracy	**Specificity**	**Sensitivity**
Single layer classifier	74,23%	0,763	0,723
Enhanced classifier with two layers	79,14%	0,775	0,807

Table 4.1: Table of results of executed CNN single vowel experiments

4.4 Multi-input model with one CNN per input

During our initial experiments, we have proven that the method we use is, in fact, suitable for a given problem. Using transfer learning, we were able to reuse a pre-trained VGG16 convolutional neural network and adapt it to our destination problem, the classification of pathological voice. We achieved the adaptation to the new problem by building a custom dense classifier learning from VGG's output. As a result, it output the probability of pathological and healthy classes.

The designed approach used in the first experiment had its limitations. The biggest one was the minimal usage of the dataset. It only worked with one data input at a time. Each subject in the Saarbruecken Voice Database has recorded three audio files with different vowels and one sentence. The initial approach did not utilize the potential of a given dataset.

For our next approach, we wanted to design an experiment that will extend our initial effort by utilizing more data for each subject. Ideally, the network will expect three inputs for each subject, corresponding to three recorded vowels. In theory, increased data diversity should help the network understand the more abstract features, rather than limiting its knowledge to single vowel samples. A network with multiple inputs should be able to utilize the given data better, learn more from them, improve the generalization, and as a result, achieve better accuracy.

The network designed for this experiment expects three inputs, one for each vowel, from a given subject. All inputs are in the form of spectrograms, converted from audio recordings, as it was in the previous experiment. Each input is then processed individually by three separate VGG16 networks. The output of all networks is then combined using the concatenation layer. After all, outputs are combined, the flatten layer flattens the produced structure.

We have started with a network with one dense layer, but we shortly discovered that it was complicated to adjust. When we tried a hidden layer with a smaller number of neurons, it was not able to learn at all. On the other hand, a hidden layer with more neurons was very prone to overfit the training data, failing to generalize correctly.

Therefore, we expanded our network with the second dense hidden layer and re-ran our experiments. This time, it was much simpler to achieve comparable results.

The first network that was successful in learning contained two hidden layers with 32 and 16 neurons. It's internal architecture diagram is displayed in figure 4.8. As the diagram describes, it contains three VGG16 networks. The concatenation layer then combines the output of each VGG16 network preparing the data to be processed by the custom classifier.

As we can see from the training metrics (figure 4.9), the validation accuracy and validation loss were relatively stable during the whole training process, except for some local spikes. After about 30 epochs, the network began to overfit, which is better visible from the loss diagram, but it can be observer also in the accuracy diagram.

From the confusion diagram (figure 4.9), the network very prone to choosing the pathological class, represented by the 0. This slight bias towards the one class is a result of insufficient generalization. Besides the mentioned problem, the network achieved an accuracy of 74,8%.

During our first step, we managed to find a network architecture that was able

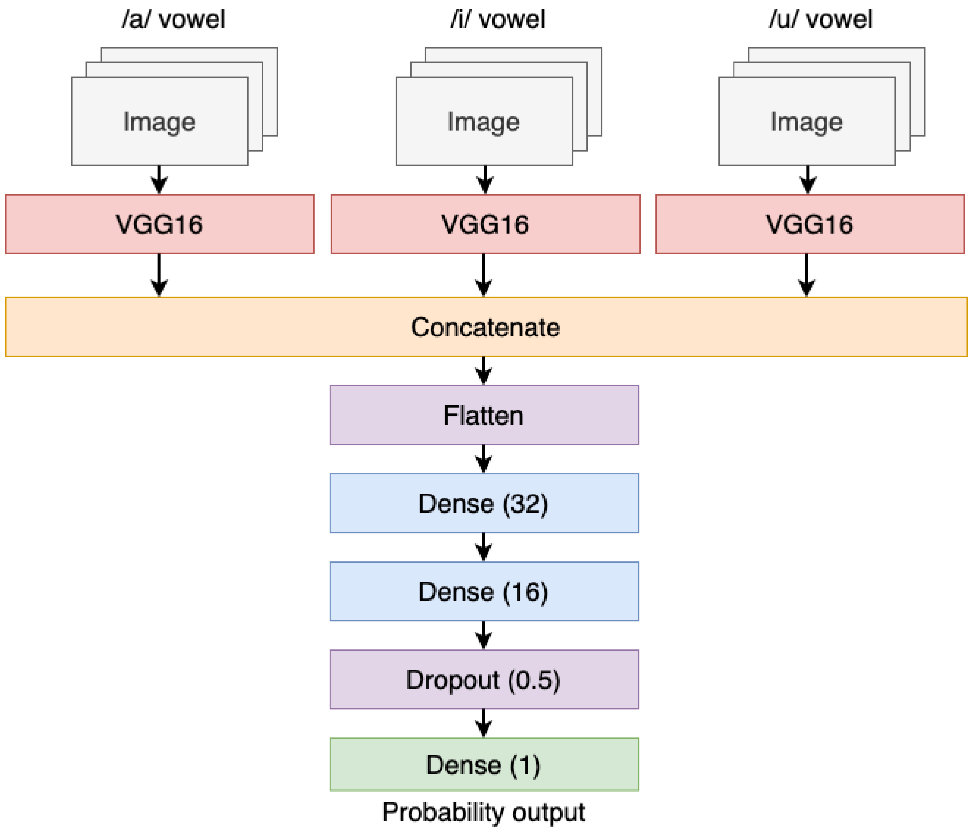
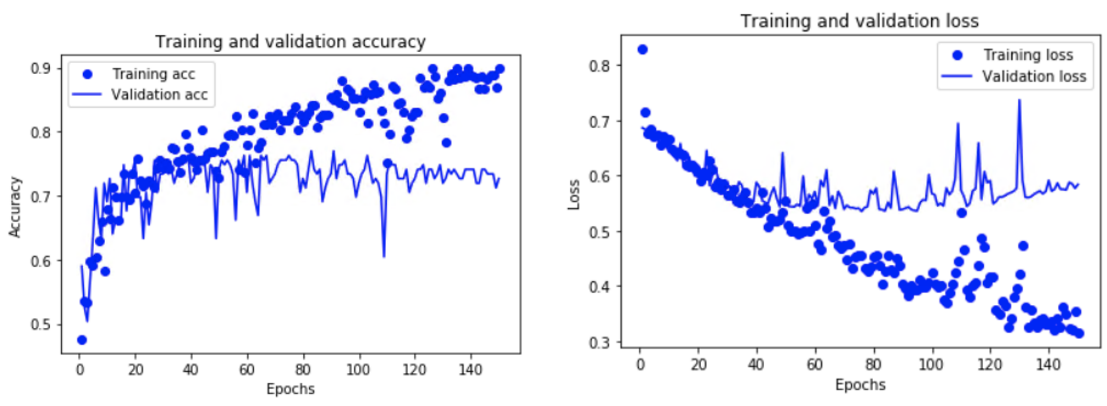


Figure 4.8: Classifier with 2 hidden layers with 32 and 16 neurons



(a) Training and validation accuracy (b) Training and validation loss

Figure 4.9: Multi-input, two dense layers with 32 and 16 neurons - training metrics

to learn for a given task. It achieved comparable results, but it had trouble generalizing, which resulted in a bias towards the one prediction class.

In order to allow the network to learn better, we have experimented with the

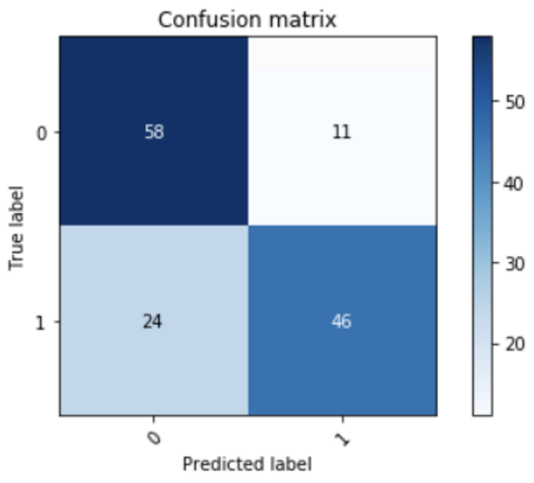


Figure 4.10: Multi-input, two dense layers with 32 and 16 neurons - confusion matrix

size of hidden layers. We had seen an improvement in accuracy when we increased the number of hidden neurons in the first dense layer of the classifier to the 64. The other components of the network's architecture remained the same, which is represented by the graph located in figure 4.11.

From the training metrics diagrams, located in figure 4.12, we can see that we trained our networks for 150 epochs. We intentionally let the network to overfit on the training data so that we can observe the dependency between the training and the validation curves. From the training and validation accuracy diagram (figure 4.12a), we can see that the network reached its validation accuracy in about 30 epochs and it stayed on about 68% for the rest of the epochs. From the validation loss diagram (figure 4.12b), it is more clear that the validation loss started to decrease from about 80 epochs. Until the end of the training process, the network began to overfit on the training data heavily.

From the confusion matrix (figure 4.13), we can see that the network was able to learn and generalize to the given data. Its predictions are balanced, although there is a slight bias towards class 0, which represents the pathological class. The network achieved an accuracy of 76,3%, which represents the 1,5% increase from the previous network setup.

A table 4.2 contains a summary of results of executed multi-input model experiments.

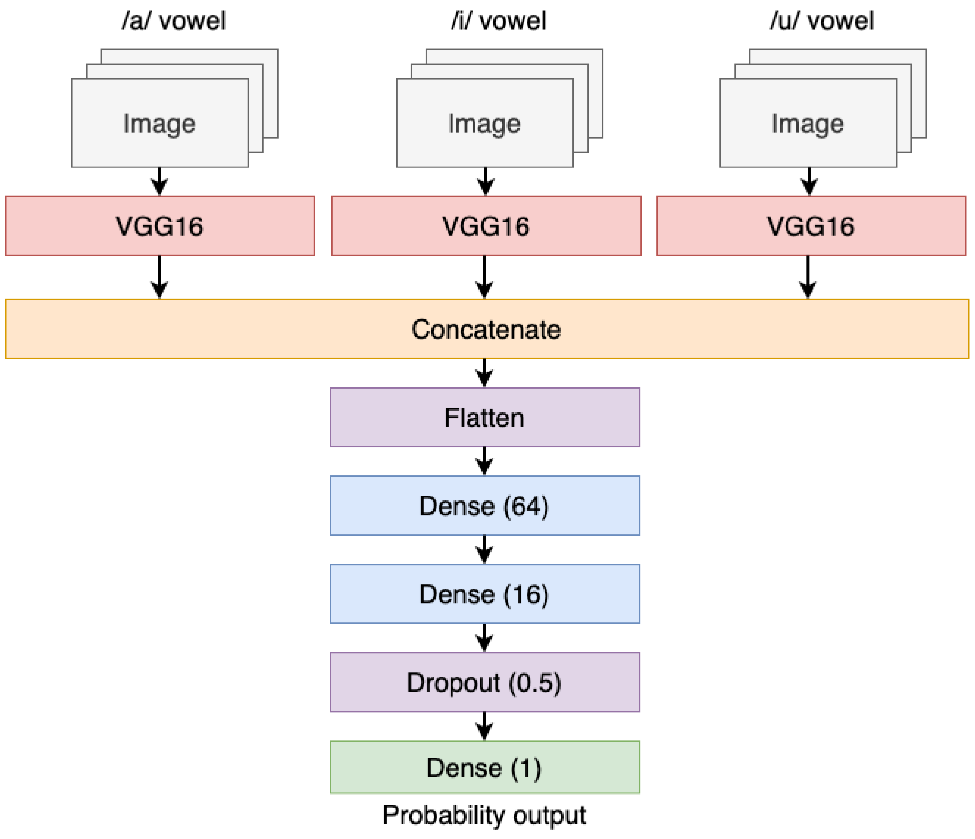
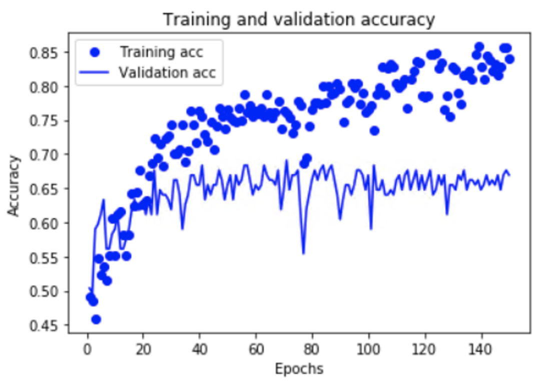
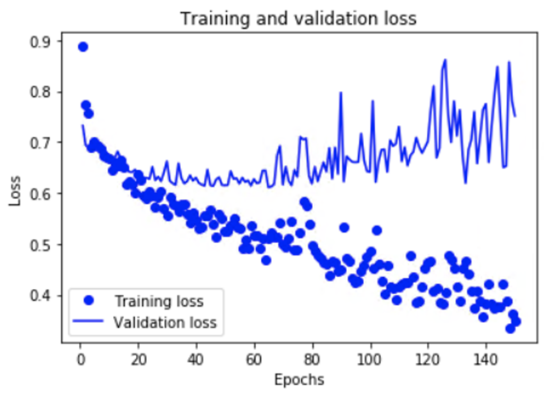


Figure 4.11: Classifier with 2 hidden layers with 64 and 16 neurons



(a) Training and validation accuracy



(b) Training and validation loss

Figure 4.12: Multi-input, two dense layers with 64 and 16 neurons - training metrics

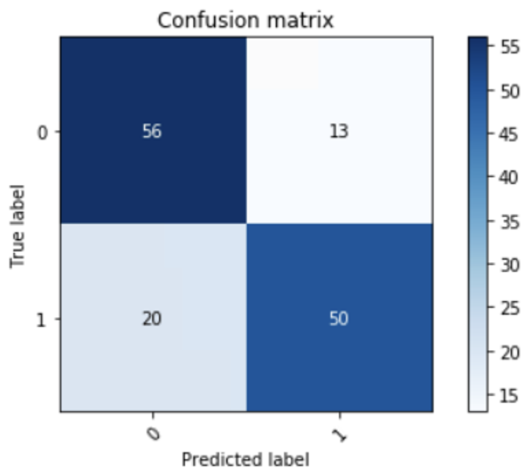


Figure 4.13: Multi-input, two dense layers with 64 and 16 neurons - confusion matrix

	Accuracy	**Specificity**	**Sensitivity**
Multi-input model, two dense layers with 32 and 16 neurons	74,82%	0,657	0,841
Multi-input model, two dense layers with 64 and 16 neurons	76,26%	0,714	0,812

Table 4.2: Table of results of executed multi-input model experiments

4.5 Encoding multiple inputs into image channels

From the experiments that we already executed, we saw that the network could learn, understand, and classify, whether given samples contain a pathological or healthy voice.

The initial approach, with the most straightforward setup, achieved the best results so far. The average accuracy that networks are consistently reaching is approximately around 73%. No matter what we tried, the accuracy only hardly went above 75%, with one exception described in section 4.3.2.

In this another approach, we have designed a solution that is the combination of two previous methods. We wanted to verify whether the smaller network could improve its accuracy by working with more data.

In our first experiment, described in section 4.3, we used a dataset that only contained single vowel recordings. To ensure compatibility with input that is expected by the VGG16 network, we duplicated monochromatic spectrograms three times, to simulate RGB input.

In the current experiment, we decided to use all three recorded vowels, as we already did in section 4.5. During a data preprocessing stage, we group all inputs for a given subject to create a single data input. This step is crucial as later, during a data split process, each subject with all its data will be classified into training, validation, or test group. All inputs are converted to spectrograms as it was in all previous experiments.

The critical element that differentiates this new approach from the previous one is the usage of the original simple network with only one VGG16 network accepting the input. Instead of duplicating the convolutional networks, we use a fact that they accept the input image in the form of RGB colors. Instead of duplicating the same monochromatic spectrogram taken from a single vowel recording, we provide different spectrograms, each representing a particular vowel for a given subject.

The network we have started with contains one dense layer and is described in diagram 4.14.

From the diagram 4.15, we can see that the training and validation curves follow each other very closely. Similiar training and validation curves mean that the results achieved during the training process were very stable and consistent. We

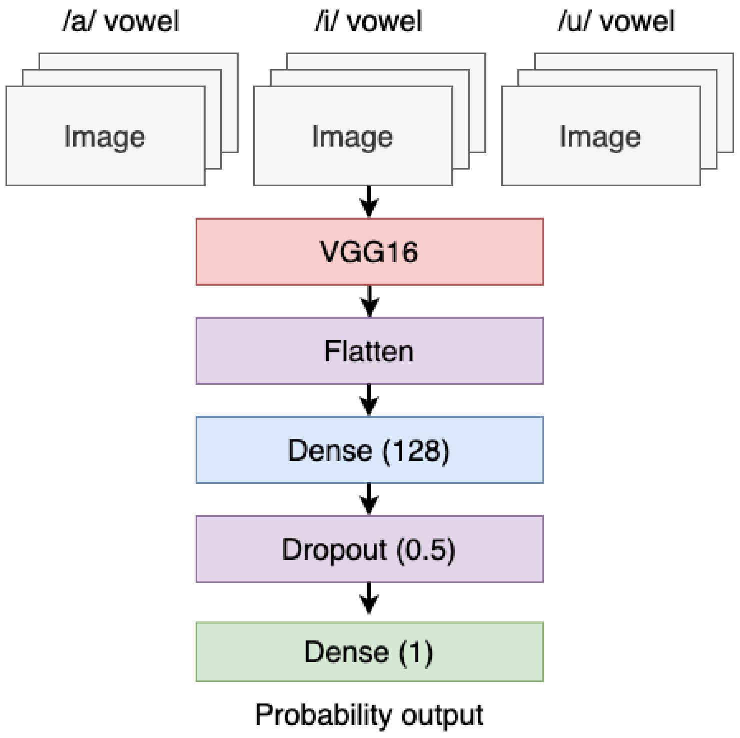
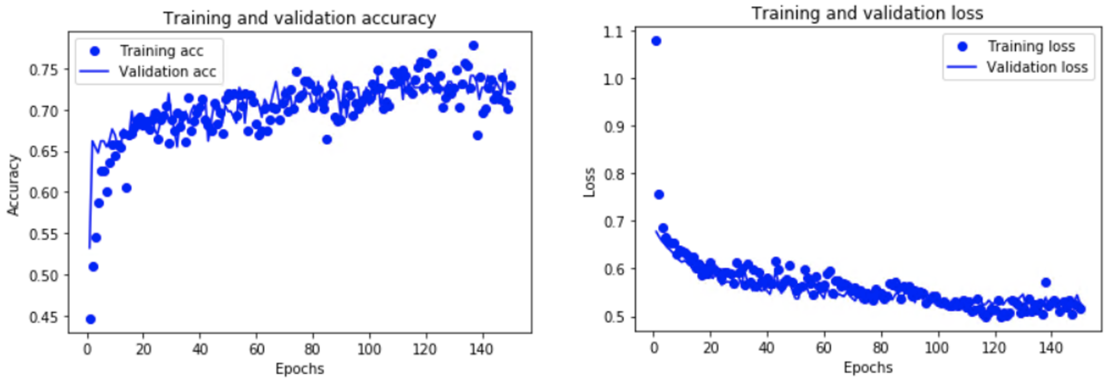


Figure 4.14: Classifier with 1 hidden layers with 128 neurons

can assume that the network was able to learn.



(a) Training and validation accuracy (b) Training and validation loss

Figure 4.15: Encoded multiple inputs, one dense layer with 128 neurons - training metrics

The confusion matrix diagram represented in figure 4.16 indicates the network accuracy on test data, from previously unseen subjects. We see that the network achieve accuracy of 72,67% and that it correctly learned to recognize both the pathological and healthy subjects.

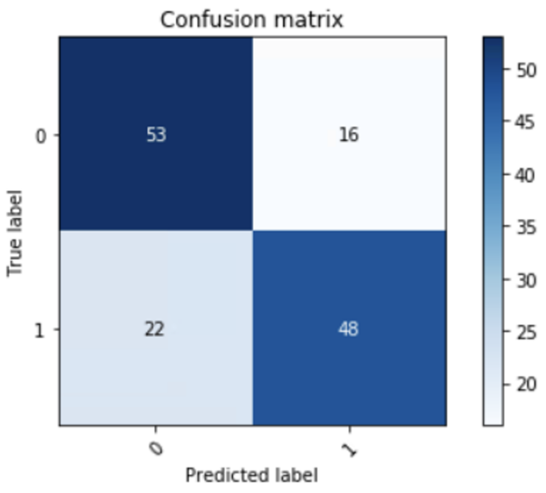


Figure 4.16: Encoded multiple inputs, one dense layer with 128 neurons - confusion matrix

After we executed our experiments, that have achieved comparable results, we experimented with the classifier structure, as we already described in section 4.3. We found that the single dense layer network was more consistent than the networks with multiple layers, and in general, it was able to achieve a good balance between overfitting and generalization. The network described in diagram 4.17 improved the results. Its hidden layer only contains 32 neurons, which reduced the change of overfitting to the training and validation data. To compensate for the reduced size, we lowered the dropout rate of 45%.

From the training metrics in figure 4.18, we see that the network slightly overfitted on the training data, but the validation metrics were consistent. Consistency is an indication that it achieved some improved generalization.

A good generalization can also be observed from the confusion matrix in figure 4.19. The network achieved an accuracy of 75%, increasing the previous result by more than 2%.

We also see, that the network is slightly biased towards the label 0, which represents the pathological class. In real-life usage, this might be considered a benefit as more people, that cannot be categorized will more likely be marked as unhealthy. It will be able to attend a diagnostic process executed by a medical specialist. Having a model that tends to choose the pathological class in a state of uncertainty will minimize the percentage of patients without proper treatment.

When experimenting with multiple layers, the results we achieved were sim-

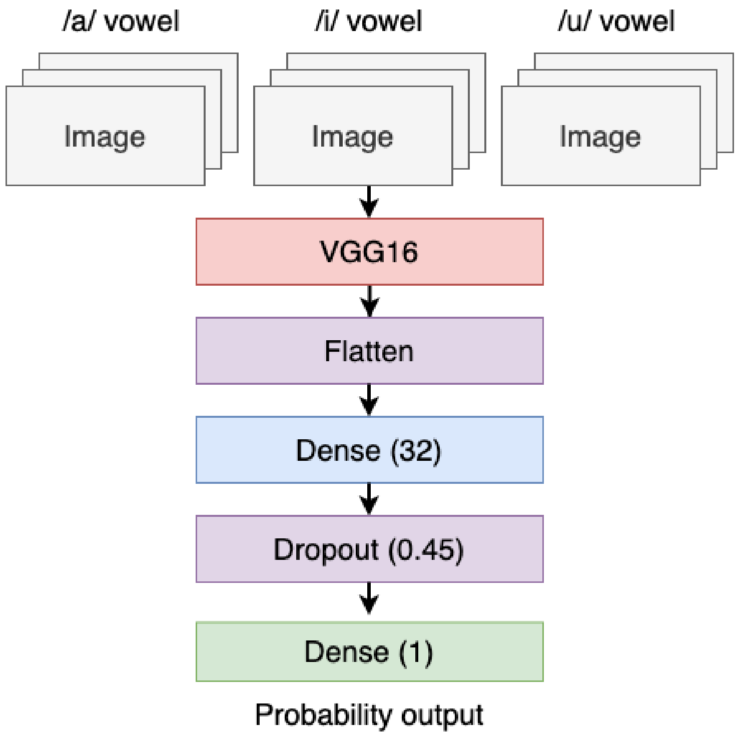
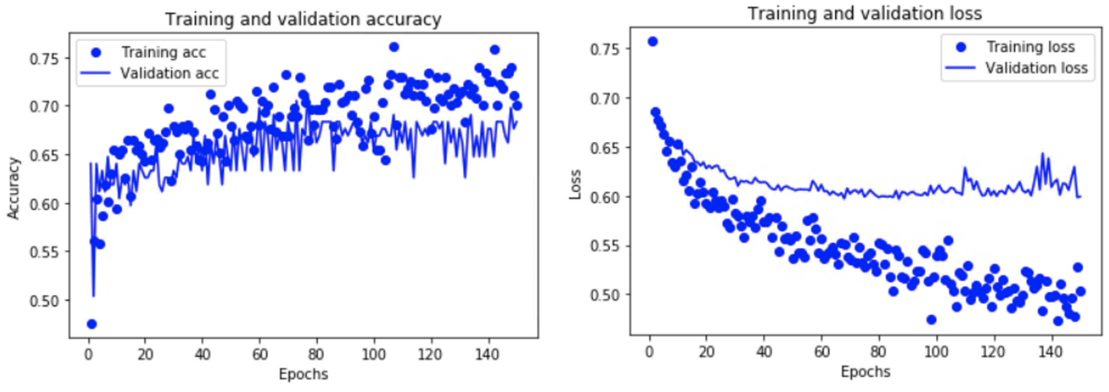


Figure 4.17: Classifier with 2 hidden layers



(a) Training and validation accuracy (b) Training and validation loss

Figure 4.18: Encoded multiple inputs, one dense layer with 32 neurons - training metrics

ilar compared to single-layer classifiers. The best accuracy of 73.38% on test data achieved a network with two dense layers, with 32 and 16 hidden neurons, respectively. Dropout rates of 0.2 and 0.5 were located immediately after each dense layer.

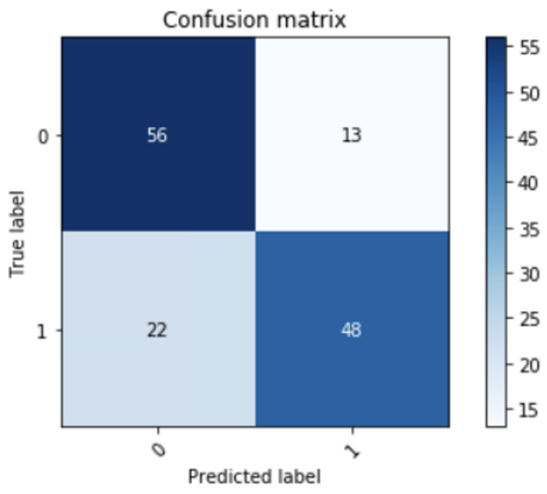


Figure 4.19: Encoded multiple inputs, one dense layer with 32 neurons - confusion matrix

A table 4.3 contains a summary of results of executed experiments with encoded multiple inputs.

	Accuracy	**Specificity**	**Sensitivity**
Encoded multiple inputs, single dense layer with 128 neurons	72,66%	0,686	0,768
Encoded multiple inputs, single dense layer with 32 neurons	74,82%	0,686	0,812
Encoded multiple inputs, two dense layers with 32 and 16 neurons	73,38%	0,829	0,638

Table 4.3: Table of results of executed experiments with encoded multiple inputs

4.6 Fine tuning model with multiple inputs encoded into image channels

In our previous experiment, we used a simple one-input neural network with all three vowels packed into the single input of the RGB spectrogram image. With a better performant network, we achieved an accuracy of 75%. In this next experiment, we were interested in the question, whether we can improve those results by fine-tuning the base VGG16 CNN network.

In order to only verify the fine-tuning process impact, we adopted the exact same network structure, as it is described in section 4.5 in diagram 4.17.

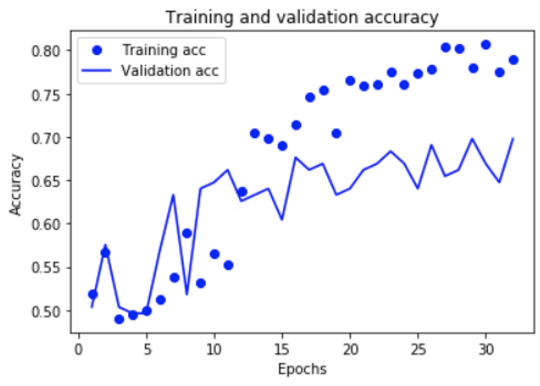
The VGG16 network, used as a base layer, is trained to classify objects of more than 1000 types from the source images. We have already proven that the network can be reused to our destination task. However, the adaptation might be hard, as the source and destination tasks and datasets are entirely different. The idea behind the fine-tuning is to enable the base network to adapt to the destination task by enabling layers from the end of the CNN network to be adjusted during a training process. We achieve this fine-tuning by setting the $trainable$ parameter of layers to be $true$. Under the hood, this unfreezes the layers which mean, that their weights will be included during a weight update procedure as a part of backpropagation and gradient descent algorithms. Fine-tuning the last layers enables the model to better adapt to a destination problem, which, in our case, is the detection of pathological speech.

First, we enabled only the last layer to be trainable, then two, up until the five last layers. During the process, we observed the results and network behavior. In general, we noticed an improved network's ability to learn from the training data, up until the point where it was almost impossible not to completely overfit the network in the first few epochs.

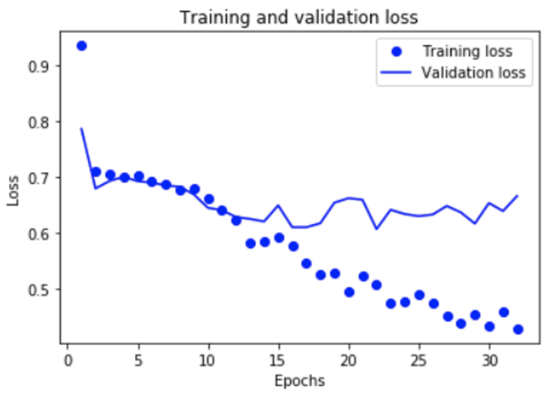
We achieved the best results with the network, where the last three layers of the CNN base network were enabled to be adjusted. As we can see from the training metrics in figure 4.20, the network aggressively started to overfit on the training data from about epoch 13.

We were also interested in achieved generalization on a separate test dataset. The network's accuracy was 76,98%, which is approximately a 2% increase from the original network.

A table 4.4 contains a summary of results of executed experiment of fine tuned model with encoded multiple inputs.



(a) Training and validation accuracy



(b) Training and validation loss

Figure 4.20: Fine tuned model - training metrics

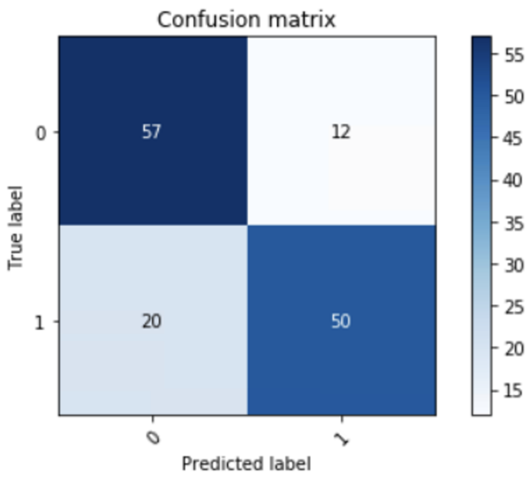


Figure 4.21: Fine tuned model - confusion matrix

	Accuracy	**Specificity**	**Sensitivity**
Fine tuned model	76,98%	0,714	0,826

Table 4.4: Table of results of executed experiment of fine tuned model with encoded multiple inputs

4.7 Comparison of different pre-trained CNN models

During all of our previous experiments, we have used VGG16 as a feature extractor for the pathological voice classification problem. We have seen that the network was able to adapt and can be reused during transfer learning to help solve our destination problem. In this new experiment, we are interested in comparing other pre-trained convolutional neural networks as testing, whether they are suitable (or easy to adapt) as feature extractors to our problem. We have decided to test the multi-data approach, described in section 4.5. It is very suitable as it utilized all three vowels as a single input for a particular subject.

Besides the VGG16, we've chosen to examine the ResNet50 [23], DenseNet121 [24] and NasMobileNet [25]. The classifiers on top of networks were set and tuned individually for each base network, to maximize the achieved results. Data preprocessing and evaluation methods used were the same during all experiments. From all of the chosen networks, we have decided to use their smallest versions with the least amount of parameters. This decision reduces the time and complexity of combined models. All pre-trained weights were learned from the ImageNet dataset.

The VGG16 network served us as a baseline with the achieved accuracy of 75%, on test data.

4.7.1 ResNet

We started with the ResNet50 [23] network, which was designed and developed at Microsoft Research. As the name hints, it is based on the concept of Residual Learning and Residual Network. A residual network, in comparison to its plain counterpart, contains a shortcut connection. Having shortcut connections was proven to be a good step forward to have deeper neural networks as it directly fights against the vanishing gradient problem.

Compared to VGG16, the ResNet50 contains 5.4 times less trainable parameters.

During a training process, we had a hard time trying to adapt the network to our destination task. We were experimenting with several different classifiers on top of the ResNet, but we were not able to come up with any good results. In all experiments, the network was not able to learn and adapt. The network always has chosen a "side" and always predicted a single class.

After closer examination, we saw that the ResNet's output data, used as an input for a classifier, contained a considerable percentage of zeros. This information lack in data is probably an indication that the network is tied to the destination problem very closely. In order to use ResNet, it might be needed to allow fine-tuning of some layers or simply train the network from scratch directly on the pathological voice detection problem.

4.7.2 DenseNet

The next network we've decided to try was DenseNet121 [24]. Dense Convolutional Network that was designed by the authors of a [24] paper connects each layer to every other layer in a feed-forward way. This idea builds upon research that shows that convolutional neural networks benefit from shorter connections between layers, close to the input and close to the output.

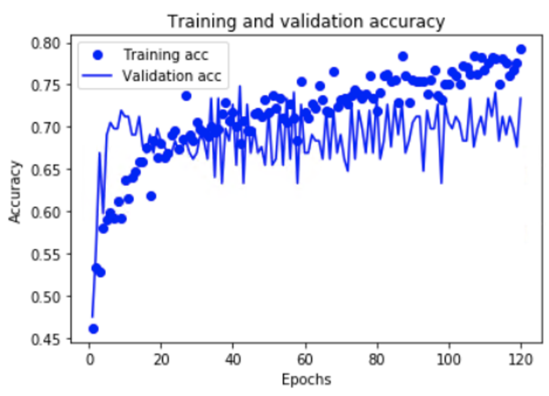
From a size perspective, the DenseNet121 is more than five times deeper than the VGG16, due to its nature. Other the other hand, it contains approximately 17 times less trainable parameters.

From the training metrics (fig. 4.22), we can observe similar behavior compared to our previous experiments. Network's accuracy and loss curves suggest that the network was able to learn to differentiate between pathological and healthy samples. Based on the small difference between training and validation loss values, we can expect similar performance on unseen data, due to generalization.

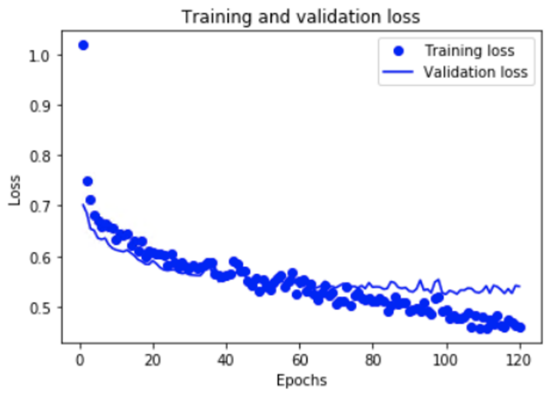
On the test set, the network achieved 71,22% accuracy. Compared to VGG16, the accuracy is worse by more than 3%. Although, with proper fine-tuning and more time spent, these results could probably be improved. According to achieved results, it is safe to say that the network can be utilized for similar tasks.

4.7.3 NASNet

Our last approach was to verify the NASMobileNet [25] network. NASNet networks represent an attractive technical solution, different from other networks. Their model architectures are learned directly from the dataset of interest. Although the full potential of this network would be utilized when modeling the



(a) Training and validation accuracy



(b) Training and validation loss

Figure 4.22: DenseNet network multi-data approach - training metrics

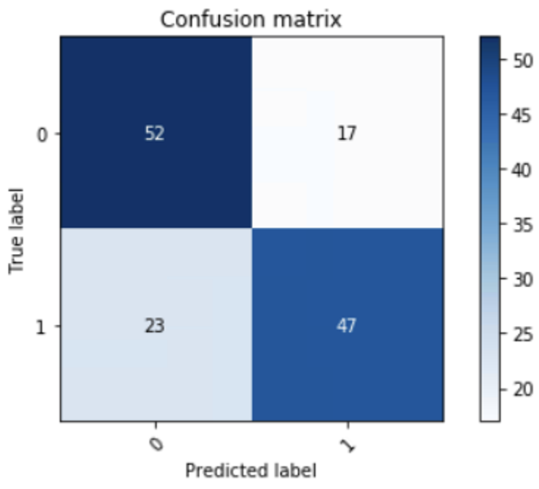
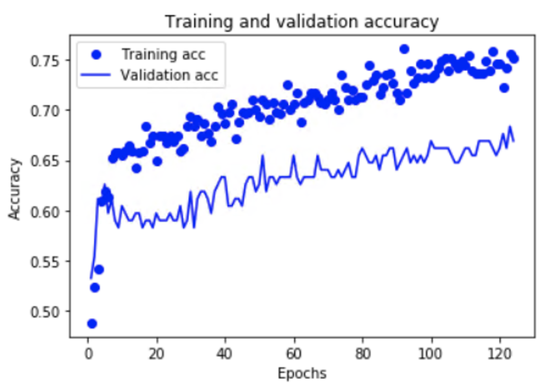


Figure 4.23: DenseNet network multi-data approach - confusion matrix

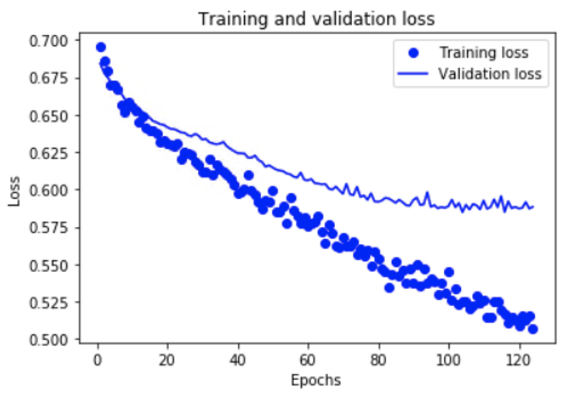
NASNet network's architecture directly from our dataset, we used a transfer learning approach, so we are consistent with previous experiments.

NASMobileNet network is the smallest network from the list of networks we have tried with just 5,326,716 trainable parameters.

As we can see from the training metrics (fig. 4.24), the network's learning rate was stable, and the accuracy and loss were improving up until the 120th epoch. Based on the distances between training and validation accuracy and loss, we can say that it was challenging for the network to find the right balance adapting to training and validation data, which might be a good indication of a proper generalization.



(a) Training and validation accuracy



(b) Training and validation loss

Figure 4.24: NASNet network multi-data approach - training metrics

Based on the confusion matrix (fig. 4.25), we see that the network was able to learn to recognize healthy and pathological samples. The achieved performance was 72,66%, which is about a 1,5% increase compared to the DenseNet experiment.

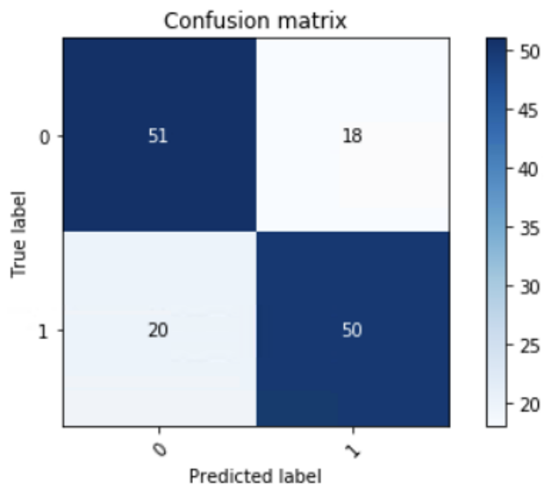


Figure 4.25: NASNet network multi-data approach - confusion matrix

Our experiments with NASNet indicates that it is the right candidate for further research as it shows great potential with adapting to pathological voice detection task.

A table 4.5 contains a summary of results of executed experiments of different pre-trained CNN networks.

	Accuracy	**Specificity**	**Sensitivity**
DenseNet	71,22%	0,671	0,754
NASNet	72,66%	0,714	0,739

Table 4.5: Table of results of executed experiments of different pre-trained CNN networks

4.8 Model ensembles

So far, during our previous experiments, we have examined several different methods and multiple models. The best accuracy so far was 79,14% achieved by our simplest model shown, that the single vowel subset contains enough information. We also tried to reuse other vowel subsets, with a multi-data experiment achieving 75% and 76,98% accuracy for fine-tuned model. It showed a possibility of using a bigger data subset, however, not in a single model. Having to learn to understand all vowels at once is a challenging task for a network.

In this next experiment, we wanted to combine the advantage of using a bigger data subset with all vowels in it and using multiple simpler models. In order to achieve that, we train three models separately on a particular vowel subset, and then, for final prediction, we combine the partial answers to form a final prediction.

The model ensemble is composed of the same networks used within the 4.3.2. Each network is trained separately on a different data subset. First is trained on an /a/ vowel, second on an /u/ vowel and third on an /i/ vowel.

We use the same training, validation, and test sets for all models. This step is crucial in order to avoid mixing data and classifying already seen subjects.

For combining partial answers, we used a weighted average method. Based on each model evaluation, we assign each model a prediction weight, calculated based on the achieved validation loss.

$$w_i = \frac{(1 - loss_i)}{\sum_{j=1}^{n}(1 - loss_j)}$$

The final prediction p_e is calculated as a weighted sum of probabilities predicted by each model $p(i)$.

$$p_e = \sum_{i=1}^{n} w_i * p\left(i\right)$$

From the confusion matrix, presented in figure 4.26, we can see that the model achieved an accuracy of 82,01%, which improves the best result, described in section 4.3, by 2%.

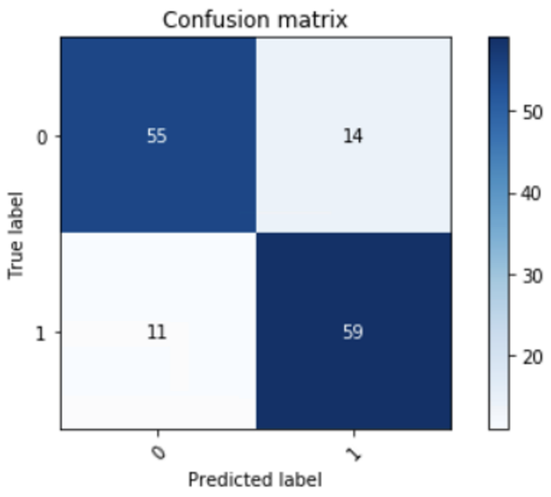


Figure 4.26: Model ensembles - confusion matrix

Our results confirm the work presented in [9] paper, proving that, for Saarbruecken Voice Database, increasing the subset improves the accuracy. Compared to paper's final result, where they used the fusion of 12 models (three vowels, natural, low, high, and low-high-low intonation), therefore four times more data, we achieved more than 2,5% increase in accuracy (compared to claimed 79,4%).

A table 4.6 contains a summary of results of executed experiments of model ensembles.

	Accuracy	**Specificity**	**Sensitivity**
Model ensembles	82,01%	0,843	0,797

Table 4.6: Table of results of executed experiments of model ensembles

4.9 Summary

In this thesis, we designed multiple experiments to perform a pathological voice detection task. Using the transfer learning technique, we reused state-of-the-art image recognition deep neural networks, like VGG16, to extract information-rich features from input images for our custom classifier. All experiments were executed using the reduced sub-set of the Saarbruecken Voice Database, containing only organic dysphonia diseases.

A table 4.7 contains an overview of results of all executed experiments.

	Accuracy	**Specificity**	**Sensitivity**
VGG16 single vowel	79,14%	0,775	0,807
VGG16 multi-input	76,26%	0,714	0,812
VGG16 multi-data	74,82%	0,686	0,812
VGG16 fine-tuned multi-data	76,98%	0,714	0,826
DenseNet multi-data	71,22%	0,671	0,754
NASNet multi-data	72,66%	0,714	0,739
VGG16 model ensembles	82,01%	0,843	0,797

Table 4.7: Table of results of executed experiments

By using state-of-the-art convolutional neural networks with minimum preprocessing, we confirmed that deep neural networks are self-sufficient and can adapt to incoming data.

In comparison with [9], when they extracted acoustic and noise-related features during a preprocessing phase, we achieved a better accuracy by 2,5% while using four times fewer data.

Similar to their research, we have also seen that utilizing multiple smaller models as a form of model ensemble and using a bigger dataset (more vowels, multiple voice modulations) significantly increases the overall network performance and results.

Comparing our results with [1], which uses the same preprocessing method and also utilizes CNNS, although they only use /a/ vowel samples, we achieved a better accuracy by 5%, or 14% compared to an experiment that includes CDBN. Those improvements represent a high potential of using pre-trained CNNS with a transfer learning technique.

The next steps, based on results presented in this thesis can include: extending the model ensemble experiment to make use of all available data from SVD, executing more in-depth experiments that include networks mentioned in section 4.7, study possible improvements of networks with multiple inputs, compared to model ensembles, and execute experiments to see whether the pre-trained state-of-the-art speech-to-text networks have a potential to be adapted using transfer learning to a problem of detecting the pathological speech.

5 Conclusion

In this thesis, we have explored the research field of detecting pathological voice disorders. The classical examination is an expensive and challenging process that required a trained medical professional. This fact directly affects the percentage of the population that is lucky enough to get diagnosed and underwent proper treatment. As it is not an ideal situation, researches started to investigate possibilities to create an automatic, non-invasive, cheap, and simple solution that will work with audio recordings. For this purpose, multiple datasets were created with samples from healthy and pathological subjects. One of them being Saarbruecken Voice Database, a collection of a voice recording of more than 2000 subjects.

Initial approaches used sophisticated feature extractors and relied on heavy preprocessing in order to serve the polished, information-rich data to the classifier. The acoustic analysis process serves the purpose, extracting multiple acoustic and noise-related features.

With the latest popularity and achievements of machine learning, more and more approaches started to make use of deep learning techniques. One of the most significant advantages of deep learning is that it requires minimum preprocessing and its ability to learn to understand complicated data relations. On the other hand, deep learning requires a massive amount of data.

In this thesis, we analyzed the existing research in the field of pathological voice detection. We come up with a method of reusing existing state-of-the-art convolutional neural networks, that are typically used for image classification problems, using transfer learning process. With transfer learning, we were able to re-purpose the existing pre-trained CNN networks as a feature extractors for our custom classifiers.

We designed several experiments using different base networks, different subsets, and multiple different network architectures in order to validate our assumptions and verify them according to achieved results.

With our experiments, we have shown that the pre-trained image classification CNN networks can be re-purposed using transfer learning to pathological voice detection task. Networks we built during our experiments achieved results that exceed the current reported results we found in published research.

During our work, we found out that the simpler networks with single vowels as their inputs achieved better, more stable results compared to the more complicated networks with multiple inputs. In order to improve achieved results, we tested the possibility of using more extensive data subsets for each subject. We observed that the simple model ensembles composed of one network per vowel subset performed significantly better compared to bigger networks that dealt with the data complexity internally.

In this thesis, we designed and tested a new approach to detecting the presence of pathology in the voice recording. We improved the existing results and confirmed that the transfer learning approach is a viable technique that is worth researching further for this field.

References

[1] Huiyi Wu et al. "A Deep Learning Method for Pathological Voice Detection Using Convolutional Deep Belief Networks". In: *Interspeech 2018*. Interspeech 2018. ISCA, Sept. 2, 2018, pp. 446–450. DOI: `10.21437/Interspeech.2018-1351`. URL: `http://www.isca-speech.org/archive/Interspeech_2018/abstracts/1351.html` (visited on 07/10/2019).

[2] M. Pishgar et al. "Pathological Voice Classification Using Mel-Cepstrum Vectors and Support Vector Machine". In: *2018 IEEE International Conference on Big Data (Big Data)*. Dec. 2018, pp. 5267–5271. DOI: `10.1109/BigData.2018.8622208`.

[3] Manoj Kumar et al. "Parallel Architecture and Hyperparameter Search via Successive Halving and Classification". In: *arXiv e-prints*, arXiv:1805.10255 (May 2018), arXiv:1805.10255. arXiv: `1805.10255 [cs.CV]`.

[4] Homayoon Beigi. *Fundamentals of speaker recognition*. New York: Springer Science+Business Media, LLC, 2011. ISBN: 978-0-387-77591-3.

[5] Sunder Krishnan, Mathew Magimai-Doss, and Chandra Sekhar Seelamantula. "A Savitzky-Golay Filtering Perspective of Dynamic Feature Computation". In: *Signal Processing Letters, IEEE* 20 (Mar. 2013), pp. 281–284. DOI: `10.1109/LSP.2013.2244593`.

[6] C. Peng et al. "Pathological Voice Classification Based on a Single Vowel's Acoustic Features". In: *7th IEEE International Conference on Computer and Information Technology (CIT 2007)*. Oct. 2007, pp. 1106–1110. DOI: `10.1109/CIT.2007.126`.

[7] Andreas ller. *Introduction to machine learning with Python : a guide for data scientists*. Sebastopol, CA: O'Reilly Media, Inc, 2017. ISBN: 978-1-449-36941-5.

[8] Ghulam Muhammad et al. "Voice pathology detection using interlaced derivative pattern on glottal source excitation". In: *Biomedical Signal Processing and Control* 31 (Jan. 2017). DOI: 10.1016/j.bspc.2016.08.002.

[9] David Martínez et al. "Voice Pathology Detection on the Saarbrücken Voice Database with Calibration and Fusion of Scores Using MultiFocal Toolkit". In: *Advances in Speech and Language Technologies for Iberian Languages*. Ed. by Doroteo Torre Toledano et al. Vol. 328. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 99–109. ISBN: 978-3-642-35291-1 978-3-642-35292-8. DOI: 10.1007/978-3-642-35292-8_11. URL: http://link.springer.com/10.1007/978-3-642-35292-8_11 (visited on 01/22/2020).

[10] D. A. Reynolds and R. C. Rose. "Robust text-independent speaker identification using Gaussian mixture speaker models". In: *IEEE Transactions on Speech and Audio Processing* 3.1 (1995), pp. 72–83.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (1977), pp. 1–38. ISSN: 00359246. URL: http://www.jstor.org/stable/2984875.

[12] N. Brümmer. *FoCal Multi-class: Toolkit for Evaluation, Fusion and Cal- ibration of Multi-class Recognition Scores - Tutorial and User Manual*. 2007. URL: https://sites.google.com/site/nikobrummer/focalmulticlass (visited on 03/25/2020).

[13] N. Sáenz Lechán. "Contribuciones Metodológicas para la Evaluación Objetiva de Patologías Laríngeas a partir del Ánalisis Acústico de la Voz en Diferentes Escenarios de Producción". PhD thesis. 2010.

[14] Zacharias Voulgaris. *AI for data science : artificial intelligence frameworks and functionality for deep learning, optimization, and beyond*. Basking Ridge, New Jersey: Technics Publications, 2018. ISBN: 978-1634624091.

[15] INTELLIGENCE BY AM TURING. "Computing machinery and intelligence-AM Turing". In: *Mind* 59.236 (1950), p. 433.

[16] François Chollet. *Deep learning with Python*. Shelter Island, NY: Manning Publications Co, 2018. ISBN: 978-1-61729-443-3.

[17] John Hopfield. "Neural Networks and Physical Systems with Emergent Collective Computational Abilities". In: *Proceedings of the National Academy of Sciences of the United States of America* 79 (May 1982), pp. 2554–8. DOI: `10.1073/pnas.79.8.2554`.

[18] F. Rosenblatt. "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". In: *Psychological Review* (1958), pp. 65–386.

[19] David E Rumelhart et al. "Backpropagation: The basic theory". In: *Backpropagation: Theory, architectures and applications* (1995), pp. 1–34.

[20] Yann LeCun et al. "Handwritten Digit Recognition with a Back-Propagation Network". In: *Advances in Neural Information Processing Systems 2*. Ed. by D. S. Touretzky. Morgan-Kaufmann, 1990, pp. 396–404. URL: `http://papers.nips.cc/paper/293-handwritten-digit-recognition-with-a-back-propagation-network.pdf`.

[21] Jiuxiang Gu et al. "Recent Advances in Convolutional Neural Networks". In: *arXiv e-prints*, arXiv:1512.07108 (Dec. 2015), arXiv:1512.07108. arXiv: `1512.07108 [cs.CV]`.

[22] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *arXiv e-prints*, arXiv:1409.1556 (Sept. 2014), arXiv:1409.1556. arXiv: `1409.1556 [cs.CV]`.

[23] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *arXiv e-prints*, arXiv:1512.03385 (Dec. 2015), arXiv:1512.03385. arXiv: `1512.03385 [cs.CV]`.

[24] Gao Huang et al. "Densely Connected Convolutional Networks". In: *arXiv e-prints*, arXiv:1608.06993 (Aug. 2016), arXiv:1608.06993. arXiv: `1608.06993 [cs.CV]`.

[25] Barret Zoph et al. "Learning Transferable Architectures for Scalable Image Recognition". In: *arXiv e-prints*, arXiv:1707.07012 (July 2017), arXiv:1707.07012. arXiv: `1707.07012 [cs.CV]`.

[26] D. Wang and T. F. Zheng. "Transfer learning for speech and language processing". In: *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. 2015, pp. 1225–1237.

[27] PG Pop, E Lupu, and M Roman. "Pathological voice assessment". In: *1st International Conference on Advancements of Medicine and Health Care through Technology, MediTech2007*. 2007.

[28] R. J. Baken. *Clinical measurement of speech and voice*. San Diego: Singular Thomson Learning, 2000. ISBN: 978-1565938694.

[29] Raymond Kent. *The acoustic analysis of speech*. Australia United States: Singular/Thomson Learning, 2002. ISBN: 978-0769301129.

[30] Ingo Titze. *Principles of voice production*. Englewood Cliffs, N.J: Prentice Hall, 1994. ISBN: 978-0137178933.

[31] Julian D. Arias-Londoño et al. "On combining information from modulation spectra and mel-frequency cepstral coefficients for automatic detection of pathological voices". In: *Logopedics, phoniatrics, vocology* 36 (Nov. 2010), pp. 60–9. DOI: 10.3109/14015439.2010.528788.

[32] Eiji Yumoto, W Gould, and Thomas Baer. "Harmonics-to-noise ratio as an index of the degree of hoarseness". In: *The Journal of the Acoustical Society of America* 71 (July 1982), pp. 1544–9.

[33] Guus de Krom. "A Cepstrum-Based Technique for Determining a Harmonics-to-Noise Ratio in Speech Signals". In: *Journal of Speech, Language, and Hearing Research* 36.2 (1993), pp. 254–266. DOI: 10.1044/jshr.3602.254. eprint: https://pubs.asha.org/doi/pdf/10.1044/jshr.3602.254. URL: https://pubs.asha.org/doi/abs/10.1044/jshr.3602.254.

[34] Hideki Kasuya et al. "Normalized noise energy as an acoustic measure to evaluate pathologic voice". In: *The Journal of the Acoustical Society of America* 80 (Dec. 1986), pp. 1329–34. DOI: 10.1121/1.394384.

[35] Dirk Michaelis, Tino Gramss, and Hans Werner Strube. "Glottal-to-noise excitation ratio–a new measure for describing pathological voices". In: *Acta Acustica united with Acustica* 83.4 (1997), pp. 700–706.

[36] A. Elhassouny and F. Smarandache. "Trends in deep convolutional neural Networks architectures: a review". In: *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*. July 2019, pp. 1–8. DOI: 10.1109/ICCSRE.2019.8807741.

[37] W. J. Barry M. Pützer. *Saarbruecken Voice Database*. URL: http://www.stimmdatenbank.coli.uni-saarland.de (visited on 02/12/2020).

[38] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[39] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[40] François Chollet et al. *Keras*. https://github.com/fchollet/keras. 2015.

[41] Martin Abadi et al. "Tensorflow: A system for large-scale machine learning". In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 2016, pp. 265–283.

[42] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[43] Brian McFee et al. "librosa: Audio and Music Signal Analysis in Python". In: 2015.

[44] John D Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in science & engineering* 9.3 (2007), pp. 90–95.

[45] Thomas Kluyver et al. "Jupyter Notebooks – a publishing format for reproducible computational workflows". In: *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. Ed. by F. Loizides and B. Schmidt. IOS Press. 2016, pp. 87–90.

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

List of attachements

Attachement A CD - master thesis in PDF and LaTeX source codes, source codes of executed experiments in Python, as Jupyter Notebook documents