Bachelor's thesis

NTNU
Norwegian University of Science and Technology

Morten Stavik Eggen
Thomas Christ Huru

Using machine learning for advertisement detection in podcasts

Bachelor's thesis in Computer Science
January 2022

NTNU
Norwegian University of
Science and Technology

Morten Stavik Eggen
Thomas Christ Huru

Using machine learning for advertisement detection in podcasts

Bachelor's thesis in Computer Science
January 2022

Norwegian University of Science and Technology

NTNU
Kunnskap for en bedre verden

Abstract

This bachelor is written by two computer science students at NTNU, spring 2022. The task is issued and supervised by Donn Morrison, associate professor at NTNU. The task states that the aim is to make an artificial intelligence that can detect and annotate podcast audio files automatically and dynamically such that advertisements can be easily skipped or cropped from the file. We hypothesised that the sound data that exists in an advertisement would be different enough to distinguish without a special focus on the speech. This hypothesis turned out to be correct in our test cases except from when testing with new languages.

This project was chosen because we had already started reading about it during our machine learning course the semester earlier. It also seemed like an interesting project that solves a daily struggle that we have met, being part of a solution to such a problem sounded attractive. Machine learning is also a quickly developing field which is exiting to be part of and to participate in moving the field forward. Another reason is that we knew data-processing would be integral in the work, and learning to prepare our own dataset sounded like an useful experience if we want to do machine learning beyond what we learned in class.

The project has been an important experience, we have learnt to develop new tools to solve problems that doesn't have clear solutions yet. To do new research using the scientific method that we have learnt about for so many years and to feel the rush of getting results.

Lastly we would like to give a huge thanks to our guide Donn, for all his counseling and his advice. We would not have finished as we have without his fantastic insights in multiple points of the task.

Contents

List of Figures

List of Tables

1 Introduction

Machine learning is an exiting developing field of study, can we help move this field along a path we consider useful for us and hopefully others by figuring how we can use it to detect advertisements in podcasts. As consumers, advertisements are a break in the flow of the entertainment we enjoy. The loudness of the advertisements may be different from the content you are listing to, and when listening to someone telling a story in a podcast, it will break the flow if it suddenly starts playing music from an advert. It can be inappropriate for its audience. Many web-based advertisements are sexual and try to adhere to an adult audience. The audience ought to choose how they want to interact with ads. On another note, sound classification is an exciting field of science, and it is used to distinguish genres in music and find what animal is making a noise. But it is not currently used to analyze podcasts in this way, which seems like a natural fit for further research.

1.1 Research question

1. Can machine learning be used to effectively detect advertisements in podcasts?
2. Are MFCC features sufficient to classify between advertisement and podcast?
3. Can a model that has trained on one language detect advertisements in another language?
4. Is there enough available data training a machine learning model based on podcasts and advertisements?

The goal for this project is to develop an AI model that is able to find advertisements in podcast files. Specifically we chose to test a CNN model on sound data, without focusing in on specific parts of the sound like speech. Our hypothesis is that the advertisements is different enough from the other content of sound data that the model can find something unique to differentiate them. A way to easily gather viable data for podcasts with and without advertisements was found, and the motivation was to gather this data set and use it to train one or more models on classifying advertisements.

1.2 Thesis structure

1. **Chapter 1 Introduction**: Gives a short introduction to why this thesis was chosen, what questions the thesis answers and what goal was set.
2. **Chapter 2 Theory and Related work**: Explains the theory behind what is the technologies and math necessary to understand the results presented in the rapport. Mainly how machine learning works and how to convert sound to useful data and why this data is more useful.
3. **Chapter 3 Methodology**: Delves into what work methods was used, our plan for development. It also explains how we gathered data, what criteria we set when adding it to our dataset, how the data was processed and how we created a machine learning model to use it.
4. **Chapter 4 Results**: Presents the data from training, as well as the amount of data gathered.
5. **Chapter 5 Discussion**: presents thoughts and discussion about the results and the process. This includes what went well and our sources of error, what we would do different.

6. **Chapter 6 Conclusions and further work**: are conclusions based on the discussion, and how this work could be continued.

1.3 Acronyms

- **AI**: Artificial Intelligence
- **MFCC**: Mel Frequency Cepstral Coefficients
- **CNN**: Convolutional Neural Network
- **DNN**: Deep Neural Network
- **ANN**: Artifical Neural Network
- **FFT**: Fast Fourier Transform

2 Theory and related work

This chapter summarizes technology that is relevant to this project and the discussion. In addition other research where this technology has been applied and researched is reviewed. Technology that has been read about and was considered for use in this project is also summarized and used in section 5 and section 6.

2.1 Sound data processing technologies

2.1.1 Sound conversion

Fast Fourier Transform (FFT) is a technique that reduces an audio signal from multiple audio frames to its frequency domain. This highlights differences in audio by being able to note differences in the audio sources over a short period of time. Figure 1 shows how the axis of an audio signal is transformed through a FFT.

FFT is an optimized version of the discrete Fourier transform (DFT), which heavily reduces the necessary computation time from $O(n^2)$ to $O(nlog(n))$, which is a significant difference in computational time for preprocessing, especially so with larger datasets that is often the case with audio.

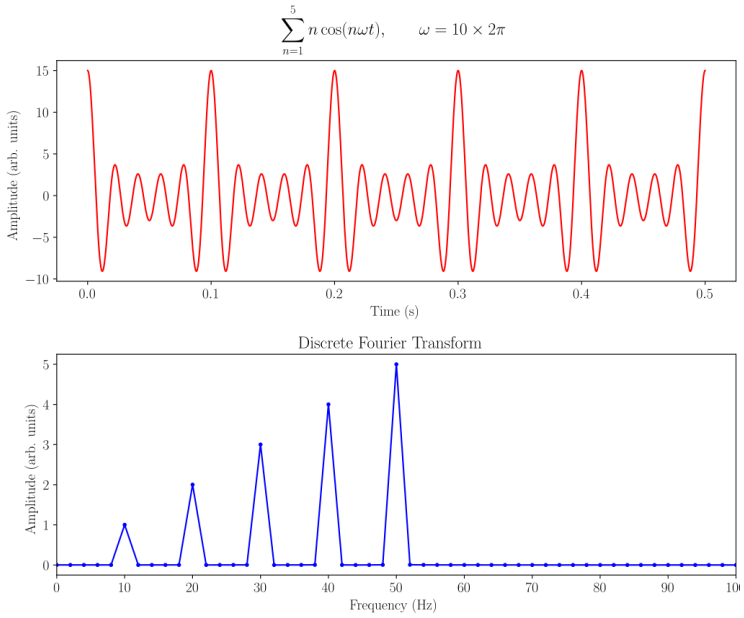


Figure 1: Fast fourier transform, By AkanoToE - Own work, CC BY-SA 4.0 [13]

Mel spectrograms reintroduces the time dimension through doing FFT over small intervals on the whole audio signals, and then storing the results over time, such as displayed in the figure 2. Convert the raw audio into a spectogram allows opens up the possibility for image based machine learning, treating the spectogram as the input for the model.

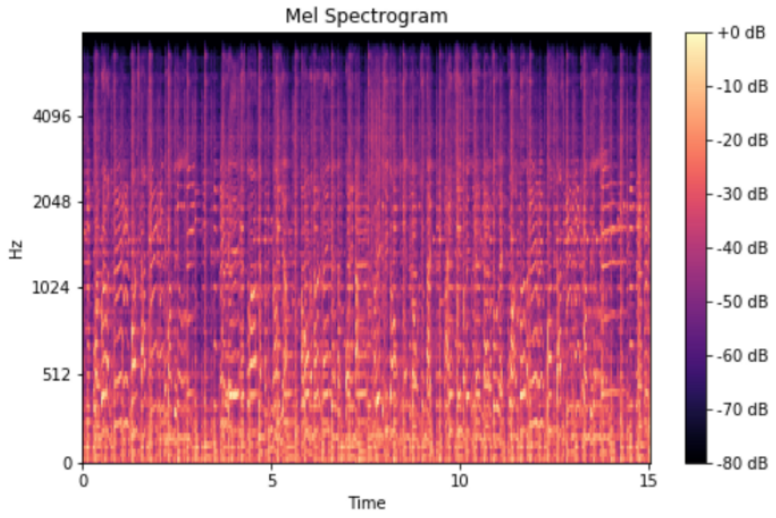


Figure 2: Example mel spectogram [36]

It also converts the audio signal to the mel scale. Mel scale is short for melody scale [43] and which mimics the way a human will be better at detecting differences in sound at different frequency ranges. This means the computer will detect differences in frequency at approximately the same significance that humans would on different frequency ranges. This means differences in frequency when comparing speech would be considered more significant. The mel scale is shown in figure 3

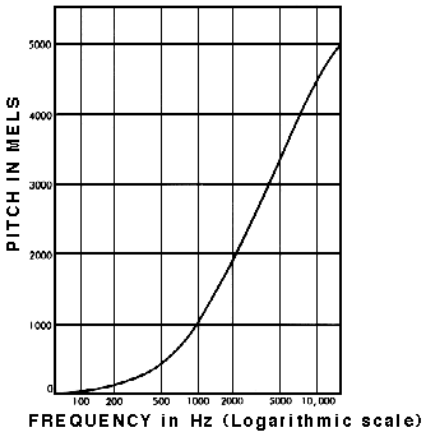


Figure 3: The mel scale [43]

One common method to convert from hertz into mel is:

$m = 2595log_{10}(1 + f/700)$

Where m is mel and f is the frequency in hertz

The mel scale is a scale designed to be a scale of subjectively equal pitch distances measured in an equal number of mels. That is equal size intervals in mels should be equally indistinguishable. [34] This means that sound differences that sound significant to humans. For instance, the difference in pitch should remain equally significant to a computer. Using the mel scale or another form of normalization of the frequencies is necessary so that the model does not prioritize differences in frequencies that are large but insignificant to humans.

There are alternatives to using the mel-scale, and they should be considered depending on the task. The meld scale is designed to fit sounds based on human hearing, and thus sometimes perform poorly or loses information when used on sounds that

lay on frequencies humans are poor at differing, or just on audio samples containing lots of background noise. Yi Ren Leng et al. [20] found that the exponential scale often outperformed the mel scale even when used on speech audio. This is due to the exponential scale filtering out sounds on the lower end of the spectrum effectively. Less random background noise means less non-relevant information for the classifier to latch on to. The Exponential scale is given by:

$g = h(f) = \frac{1}{10}e^{f*log(10)}$

Filtering out background noise means that it is much more effective as SNR (signal to noise ratio) increases. The exponential scale is also clearly better at performing well with non-speech sounds, proving it a better scale for general use.

The mel-scale has been criticized for experimental flaws by a student of the mel-scales innovator. In a mailing list the student; Donald D. Greenwood posted:

”I would ask, why use the Mel scale now, since it appears to be biased? If anyone wants a Mel scale they should do it over, controlling carefully for order bias and using plenty of subjects – more than in the past – and using both musicians and non-musicians to search for any differences in performance that may be governed by musician/non-musician differences or subject differences generally.” [1]

2.1.2 MFCC

Audio machine learning has been a long heavily changed subject over the years. To this end. multiple approaches to converting audio over to 2d images has been used. Geoffrey Hinton et al. [15] covered the recently explored field of deep neural networks for audio machine learning, finding it an improvement over previously used Hidden Markov Models and Gaussian mixture models comapring results and analytically covering the technologies. In addition, Abdel-rahman Mohamed et al. [27] had success with deep neural networks for image classification, experimenting with different amount of hidden layers. Further, based on Hendrik Purwins et al. [35] research on spectograms and deep learning, Dejan Ćirićet et al. [10] heavily explores different spectograms for audio machine learning, concluding with the viability of multiple spectogram types.

Y.M.D Chatarnga et av [8] researched automatic music genre classification. An approach using MFCC into feature selection is used, and testing accuracy using boosting techniques, especially adaBoost which increases accuracy for classifiers. In addition to adaboost, Chathuranga and Jayaratne use a SVM classifier or support vector matrix classifier is used in combination with other classifiers. Feature selection is heavily described and discussed, showing good results with rigid testing for the different approaches used. The feature extraction technique used is described in figure 4

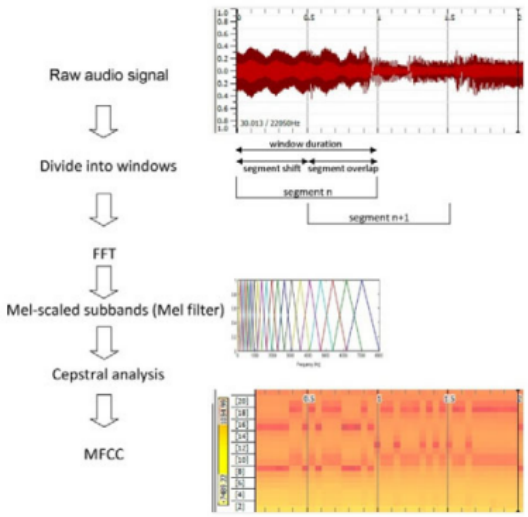


Figure 4: Sound classifying technique [8]

Similarily, Feng Rong proposes a SVM based classifier for classifying sounds. Feng Rong uses MFCCs as one feature in the classifier.

Based on previous work on convolutional networks based on fixed wavelet by Mallat et al. [23] [2] Monika Dorfler et al. [11] explores feature-network pairs in CNN's.] Based on Anden and Mallats research on approximating mel-spectograms, found that it could be approximated using an adaptive filter bank.

MFCCs have multiple implementations. Zheng Fang et al. [47] implements MFCC by using the average over the different filters rather than the sum:

$\Theta(M_k) = \sum_M P(M - M_k)\psi(M), k = 1, ...K$

Zheng Fang et al. also tests a rectangular filter bank shape based on the research done by, Hermansky as illustrated in figure 5 (c) but ultimately finds that the chosen shape does not make much of a difference.

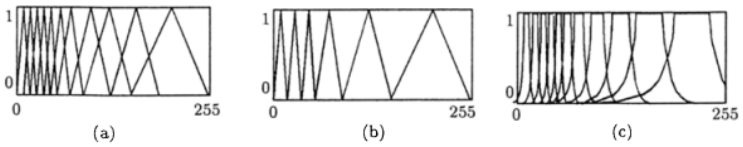


Figure 5: Filterbanks [47]

Zheng Fang et al. also finds that for high-quality environments a combination of MFCC's with frame energy performs well, while in lower-quality environments using auto-regressive analysis coefficients in addition performs best.

2.1.3 Advertisement detection

Shervin Minaee et al. use a deep learning approach to detect advertisements in videos using both audio and video data achieving an accuracy of around 70% when only using the audio data. [25]

Shashidhar.G.Koolagudi et al. prove that reaching an accuracy of 87% is possible on radio advertisements. Their research is based upon Srinivasan et al. *"Towards robust features for classifying audio in the cuevideo system"* in Proceedings of the seventh ACM international conference on Multimedia (Part 1). ACM, 1999, pp. 393-400.[42]

who created a model that differs between music and speech. They based their model on this because the quality of work related to speech detection when compared to advertisements. [17]

Ravi P. Ramachandran et al. do comprehensive research on speaker recognition and classify two features defining for doing machine learning for speaker recognition. the difference in speaking styles (accent included), (2) the difference in vocal tract shapes and vocal cords. Tumisho Billson Mokgonyane et al. test several classifying models [FIG NUM HERE] and finds MLP (Multilayer perceptron) to perform the best and get relatively poor results using KNN.

2.1.4 Mel Frequency Cepstral Coefficients

Mel Frequency Cepstral Coefficients or MFCCs are features that closely correlate with speaker pitch period. Historically, MFCCs were designed to be used with speech recognition. It's very effective at recognizing echoes in sound and overtones in voices. In more recent times, MFCCs has been used for other machine learning problems as well and are an effective algorithm for feature extraction. MFCCs are found by first doing FFT on the audio signal, and then putting the data through a mel-spaced filter bank. This is a set of triangular filters that are non-zero for a small portion of the spectrum, the filterbank of these filters can be seen in figure 6 Then the log of the resulting spectrum applied before a final FFT is done. The resulting output is the Mel Frequency Cepstral Coefficients of the original signal.

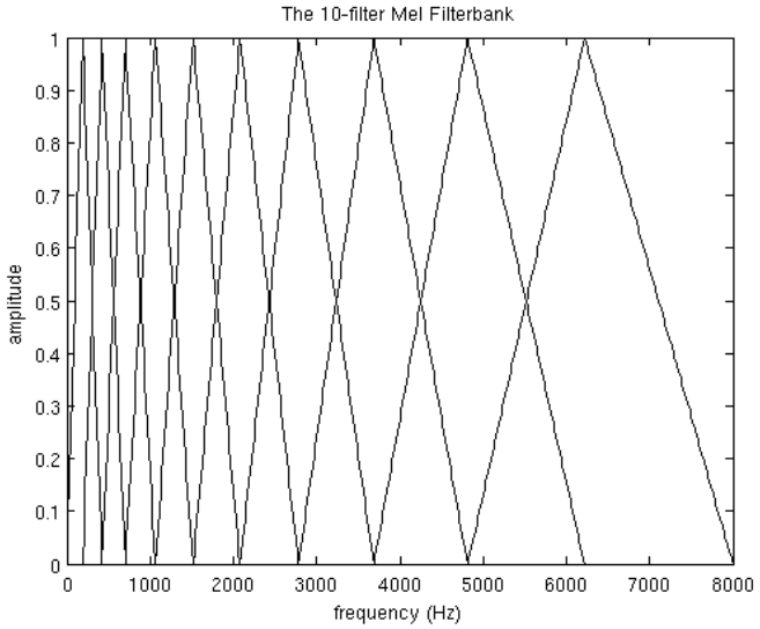


Figure 6: Mel filterbank [24]

MFCC's perform very poorly when a high amount of noise is present, and works best for clean audio.

2.2 Machine learning

2.2.1 Hyperparameters

Hyperparameters referes to the values used to define the training of the model. Examples of variables refered to as hyperparameters:

- Learning rate, learning rate is the rate of change the model does to the weights of the neurons in during training.
- Number of hidden layers, how many layers does a set of data traverse before the model makes a prediction.
- Epochs, number of times the model with train with a dataset.
- Batch, the dataset used for training the model.

These are set before training and cannot be changed while training.

2.2.2 CNN models versus DNN for audio classification

Shawn Hersey et al. tested how well CNNs worked for classifying a dataset of audio data from youtube. When compared to a fully connected DNN, the other CNNs they used gave more correct classifications on the sound data but also spent a much longer time training. The DNN gave an Area-Under-Curve (AUC) of 0.851 while the worst performing CNN model AlexNet gave 0.894 and the best ResNet-50 gave 0.926. However ResNet spent 10 times the time to reach that score.

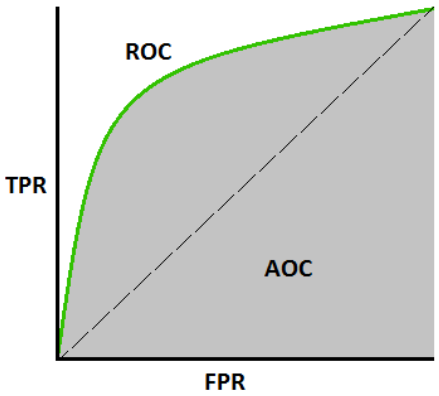


Figure 7: AUC - ROC curve [14]

As the name suggests, AUC describes the area under the green line in figure 4, which is how often the model does a correct positive assertion about the category. TPR means true positive rate and FPR means false positive rate. As AUC apporaches a value of 1, the model is more correct. Also testing with a ResNet-50 model they used different dataset sizes, concluding that the difference between using 70 million videos versus 700k was minimal, but 70k and lower was significantly worse which was chalked up to overfitting. [14]

2.2.3 Comparison and Analysis of SampleCNN Architectures for Audio Classification

When preparing audio data for a CNN, the majority first converts the audio data to an image called a mel-spectrogram, and then reads the pixels to convert them to

readable data for the model. This can require a lot of curating in the conversions. An alternative to this is reading the raw waveforms. [16]

Tripathi et al. [44] propose a Convolutional Neural Network using Multitask Learning and based on speech features trained under the joint supervision of softmax loss and center loss. Their network contains fewer parameters compared to current popular alternatives and performs 5.3 percent better. Based on this Abdul Qayyum et al. [2] proposes a simple model based on CNN with MFCC signal processing. Abdul Qayyum et al. [2] proves that their CNN model using categorical cross-entropy outperforms alternative models when using MFCC for signal processing. Similarly, Saraswathi et al. [39]¨uses a CNN with Long short-term memory (LSTM). The data is processed waveform and MFCC is done via Librosa. Their model achieves good results on the RAVDESS dataset.

Becker et al. [4] researched creating an interpretable model for artificial neural networks. They experimented with both spectrogram and raw waveform approaches. The spectrogram approach was based on Krizhevsky et als ALexNet [18] deep neural network model with 5 hidden layers and 60 million parameters except Becker et al. [4] removed the normalization layers. They formed a hypothesis on the features selected on the network by performing Layer wise relevance propagation (LRP), finding that a relatively small fraction of the data was relevant for the output of the model.

2.2.4 K-fold Cross-Validation

K-folding is a technique for cross-validation in machine learning that lets us iterate through the data we use for the model so that it all at some point is used as testing data for a model. Then we take the average accuracy of the end scores. The way this works is:

1. Choose a value for k, making the model run that many times.
2. Shuffle the dataset
3. Split it into k groups, each group is one unique set of testing data.
4. Fit the model, all but one group is training data, the last one is for testing.
5. Discard the model, but keep the evaluation of it.
6. Repeat the previous two steps until all groups has been used as training data.
7. Get the average evaluation from the k runs of training that is done.

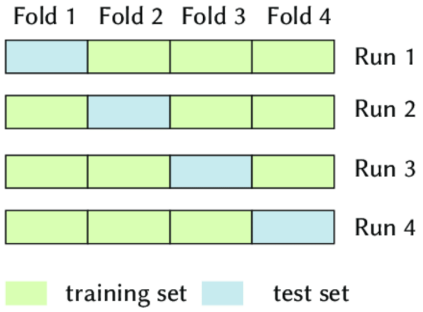


Figure 8: Example k-fold where k=4 [12]

2.2.5 Machine Learning

Machine learning is the development of technology that attempts to emulate Human Intelligent learning. The goal of machine learning is to have a computer perform a specific task with good results and without the use of direct instructions. As stated Arthur Samuel in his seminal work defined machine learning as:

"a field of study that gives computers the ability to learn without being explicitly programmed" [26] Machine learning aims to have the computer be able to recognize patterns and recognize its own successes on given tasks and improve upon them. One common feature of tasks solved using machine learning is that they have a high level of complexity that leaves them difficult to solve using traditional programming. Tasks such as recognizing an image can have a high degree of variation and require the algorithm to have a high level of adaptability to perform well. Another purpose of machine learning is that it should be adaptable. Writing a complicated algorithm to recognize images is a grueling task, and should some change be required of the system, such as a newer design for a traffic light, much of the previous work goes to waste. Machine learning solves this by the model itself being adaptable to changes.

A machine learning model refers to the expression or algorithm that encompasses and defines the input, output, and learning of a machine learning program. There are multiple kinds of models designed and theorized, and they can be combined for different results.

2.2.6 Training data, validation data, and test data

Training data refers to the data y given as input to the model. This is the data the model has available when performing training, and is the build of the data. Since the training data has such importance to the model, having a high variance in the data will provide for a more robust trained model.

Validation data is data that the model has available for testing while training. The validation data is used for the model to calculate its accuracy and tune its parameters to better fit the data. The validation data must be held separate from the test data, such that it should not achieve artificially high accuracy and overfit the data.

Testing data is similar to validation data but is only used on testing the accuracy of the model after training, and mainly exists for observers to judge the model.

2.2.7 Perceptrons

One of the first kinds of neurons created for artificial neural networks is perceptrons. Perceptrons are a kind of neuron that takes multiple inputs $x_1, x_2 ... x_n$ and produces an output based on the input, as shown in figure 9. In perceptrons, the output of the neuron is only equal to one if the sum of all inputs times a number dedicated to the edges connecting the neuron are larger than a given threshold. In all other circumstances the sum equals zero:

$$Output = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq threshold. \\ 1 & \text{if } \sum_j w_j x_j > threshold. \end{cases}$$

Perceptrons are used in networks of other perceptrons, and thus the output of one perceptron will be received by one or multiple other ones, creating a network of edges. This kind of network can produce NAND gates and are therefore universal for computation, this in combination with weights and backpropagation discussed later in the thesis provides a robust base for artificial neural networks, though perceptrons are

not usually the kind of neurons used today, the base functionality of perceptrons is still present in other artificial neurons.[31]

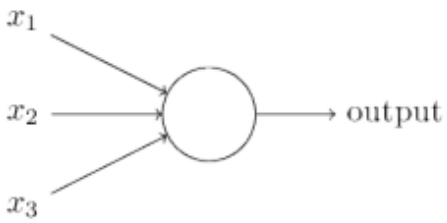


Figure 9: Model of perceptron [31]

2.3 Artificial neural networks and deep learning

2.3.1 Artificial Neural Networks

Artificial neural networks are systems of input layers of nodes or neurons, multiple hidden layers of neurons, and a final layer of output neurons as shown in figure 10 The different connections between neurons are weighted with a number that signifies the strength of the connection. The output of a neuron in the hidden layer is equal to:

$$h_i = s\sigma(\sum_{j=i}^{N} V_{ij}x_j + T_i^{hid})$$

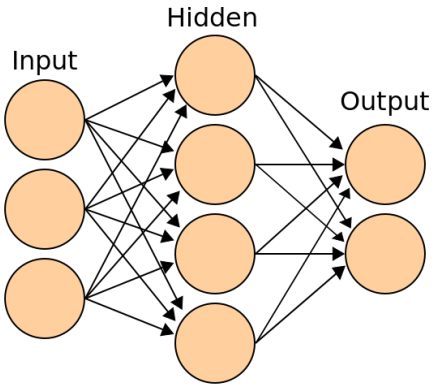


Figure 10: An example of an artificial neural network with a hidden layer [6]

Artificial neural networks learn with no preexisting rules about their target. In supervised artificial neural networks, they may be trained to identify images of cars, and only given images labeled car or no cars. The goal of the networks is then to automatically establish a set of rules that has the highest probability of achieving correct guesses. [45]

2.3.2 Deep learning

Deep learning is a field in machine learning that emphasizes multiple processing layers of data. The layers have different levels of abstraction, and each serves a purpose in the algorithm. Back-propagation is important for the algorithm to be able to learn from results and change its variables. Deep learning networks consist of three main

layers, an input layer consisting of all the input data in an array, multiple hidden layers that transform the data, and finally an output layer.

2.3.3 Back propagation

For a machine learning model to be able to learn, it needs to be able to improve upon its previous state. This is done through backpropagation. Backpropagation uses the loss function applied in the model and applies the weights of the previous nodes. This means that nodes that have a high amount of effect on the result are also given more impact on the loss. The loss can be calculated on each node individually, but the weight of the succeeding node is also taken into account. Backpropagation is a highly effective algorithm and is therefore widely used in complicated networks. The formula for Calculating the error of neuron j on layer l is:

$\delta_j^l = \frac{dC}{d_j^l}$

While the error on a single layer is calculated by:

$\delta^L = (a^L - y) \odot \sigma'(z^l)$

[31]

Nielsen describes the backpropagation algorithm as [31]:

1. **Input** x: Set the corresponding activation a_1 for the input layer
2. **Feedforward**: For each $l = 2, 3...L$ compute $z^l a^{l-1} + b^l$ and $a^l = \sigma'(z_l)$
3. **Output error** δ^L: Compute the vector $\delta^L = \nabla_a C \odot \sigma'(z_l)$
4. **Backpropagate the error:** For each $l = L-1, L-2, ..., 2$ compute $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \delta'(z^l)$
5. **Output:** The gradient of the cost function is given by $\frac{dC}{dW_j^l k} = a_k^{l-1} \delta_j^{l-1}$ and $\frac{dC}{db_j^l} = \delta_j^l$

$w_j^l k$ is the weight the edge connecting the k-th neuron in layer l-1 to the j-th neuron in layer l. a_k^l is the activation of the k-th neuron in layer l, while b_k^l is the bias of the k-th neuron in layer l. $z^l \equiv w^l a^{l-1} + b^l$. C is a cost function and $\delta$is the error.

2.3.4 Convolutional Neural Networks

Convolutional neural networks are a type of neural network commonly used for analyzing imagery and other data that has a grid-like structure. CNN recognizes patterns in data even when it is not spatially dependent, and is therefore very useful in analyzing images. CNN's reduce the number of parameters in the network, and can therefore be used to analyze much larger and complex models than a traditional neural network. [3] Convolutional derives from the mathematical operation convolution, which applies to different functions to each other to create a third function. This is comparable to the way a CNN will use multiple layers applied to each other, to create one output as shown in figure 11.

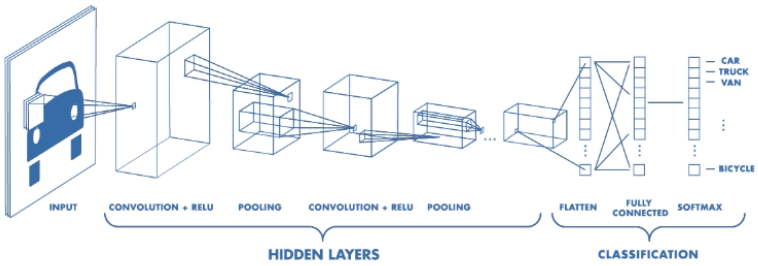


Figure 11: Typical CNN architecture [3]

[9]

CNN's are composed of a group of layers each with their own purpose within the network. The first layer is the raw input data, followed by multiple layers of convolution and pooling. The purpose of pooling is to reduce the complexity for the next layer. The convolutional layer transforms the data by moving through areas of the data, usually 3x3 or 5x5, and calculating a pixel in the next image. This pixel is calculated by using the formula:

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

Where S is the output in the next layer, I is the input image and * is the convolutional operation. [3]

2.3.5 Non-linearity

The layer after the convolution layer is non-linearity. The purpose of this layer is to cut off, remove or adjust the output of the previous layer. There are several approaches to this layer, where sigmoid and tanh have been very popular previously. In recent years Rectified Linear Unit (ReLU) has been used most often. Some of these can be seen in figure 12. These functions are often called activation functions. [3]

ReLU is a very simple function when compared to other ones, which is why it has gained popularity. It is given by:

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

This eases the back-propagation since the gradient is consistent across the entire function, unlike tanh and sigmoid. In addition, ReLU can also deliver complete zeroes, unlike tanh and sigmoid which always gives some non-zero value. This creates a sparser representation that is in favor for training. ReLU is non-differentiable at zero but can be chosen to be either 0 or 1, making no difference. [3]

Sigmoid outputs a value between 0 and 1, closer to 0 the lower x is and closer to 1 the higher x is. Sigmoid is given by:

$$\sigma(x) = 1(+e^{-x}$$

Thanh is very similiar to sigmoid, but it outputs a value between -1 and 1, averaging the values around 0. Thanh is given by:

$$\frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

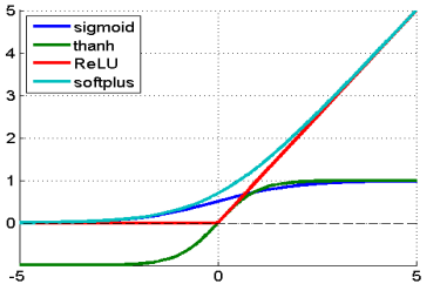


Figure 12: Common types of non-linearity [28]

2.3.6 Pooling

Next, the values are pooled, so that the complexity for further layers is reduced, while still retaining important data. One very common sort of pooling is max-pooling, in which the maximum value from a given area. This means that larger values are given much more importance to the algorithm, as smaller values are simply forgotten as shown in figure 13. One can also use average pooling, which outputs the average of the values in the average, also preserving some of the smaller data. Typically 2x2 or 3x3 is selected to carry on to the next layer. Stride is another factor of pooling, deciding how much the pooling area moves each calculation. Different strides have impacts on the result and can be chosen with for instance greater pooling are to have more overlap between multiple poolings. [3]

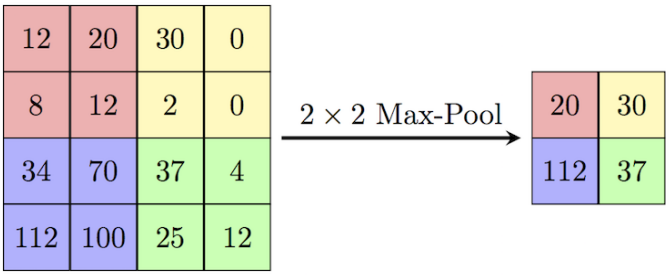


Figure 13: 2x2 max-pooling example [33]

[33]

It is possible to also use linear max pooling, such as 1x3 and 1x4 when working on linear data.

Max pooling was shown to be vastly superior to alternatives by Scherer et al. [40] in 2010. Sherer et al. worked on image-like data, specifically on the MNIST and NORB jittered-cluttered datasets.

2.3.7 Fully Connected Layer

Last comes the fully connected layer, where there is a set of nodes in separate layers, and all of the nodes are connected to each node in the previous and next layers. This is a traditional artificial neural network, and it is important that the number of connections and nodes remain relatively small, and this can be done using the dropout technique. What separates a CNN from a traditional ANN is the steps previous to the fully connected layer. [3]

2.3.8 Gradients

Vanishing gradient

A measure of how quickly a weight in an ANN changes, is the gradient. It is desirable for a gradient to reduce as the loss decreases, as larger changes in weight would forego previously achieved accuracy achieved by the weights.

As deeper neural networks are in theory able to calculate more complex tasks, it is desired to be able to experiment with deeper networks to receive greater accuracy and tackle new problems. Adding layers indefinitely does not go without problems, as more layers correlate to an increase in the amount of computing time. In addition, more layers means that the gradient for earlier layers often decreases. This is called the vanishing gradient problem. The inverse is the explosive gradient problem, in which the gradient for earlier layers keep overshooting. In general the gradient of earlier layers becomes increasingly unstable as the number of layers increase.

The vanishing gradient problem mainly occurs due to the chosen activation (nonlinear) function having a derivative smaller than 0. The gradient is calculated as follows:

$$\frac{\partial C}{\partial b_1} = \sigma'(z_1)\, w_2 \sigma'(z_2)\, w_3 \sigma'(z_3)\, w_4 \sigma'(z_4)\, \frac{\partial C}{\partial a_4}.$$

Figure 14: Gradient calculation [31]

Vanishing gradient can be counteracted heavily using ReLU instead of another activation function, as the derivative is either 1 or 0. due to it being completely linear, and is the main reason for its popularity, as using it allows one to use many more layers without the gradient becoming unstable.

2.3.9 Convolutional Neural Networks For Audio

Both audio data and image data are usually not spatially dependent, and therefore CNN works very similarly on both. Audio is not 2-dimensional, and so a 1-dimensional convolution is used instead of a 3-dimensional one. Audio data is very large, and so even with a powerful CPU, training a model takes a large amount of time.

CNN's have been used successfully on classification tasks such as identifying instruments or genres of music (here can use sources from høst)

Audio classification uses a supervised technique. Example audio are fed to the machine, and it is trained to recognize using example audio. The input can vary in format, and different techniques can provide varying results. Performing Fast Fourier Transform on the data and producing a spectogram heavily reduces the size of the data while both preserving much of the data and separating different audio streams, making combined audio easier to analyze.

2.4 Machine Learning Continued

2.4.1 Dying ReLU

Another similar problem that might appear when using ReLU, is the dying ReLU problem. Because ReLU transforms all values less than 0 to 0, some neurons might end

up having no non-zero output for all its grids. And in deeper networks, larger areas might be filled with these dead neurons.

An extreme of the dying ReLU problem is dying ReLU neural networks. This occurs when the entire network dies, and the network becomes a constant function. Lu et al. [22] ran a simulation with some of the most popular initialization techniques on 10 layers 1000 times and got over 90% dying networks on a task that should be very possible with ReLU. The probability of a network dying increases as the depth of the network increases, which is one of the reasons for the popularity of wider networks when using ReLU, though wider networks require much more processing power. Lu et al. estimated the chance for a network to be born dead based on the depth and number of layers as seen in figure 15 [22]

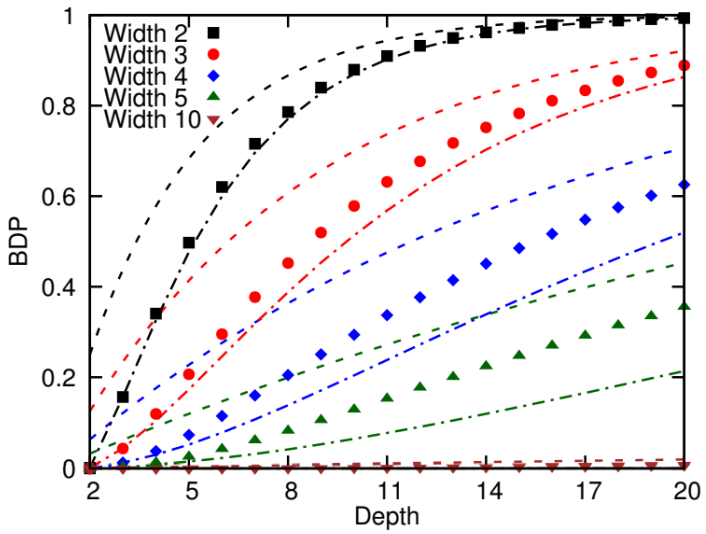


Figure 15: Probability of a ReLU Neural network to be born dead as a function of the number of layers for different widths [22]

2.4.2 Overfitting and underfitting

A concern when working on convolution neural networks is overfitting and underfitting, Burnham describes overfitting as "the essence of overfitting is to have unknowingly extracted some of the residual variation as if that variation represented underlying model structure" [7]. An overfitted model does not become applicable to new datasets because the rule set that it creates is too specific for the data provided, while an underfitted model will fail to provide enough rules at all.

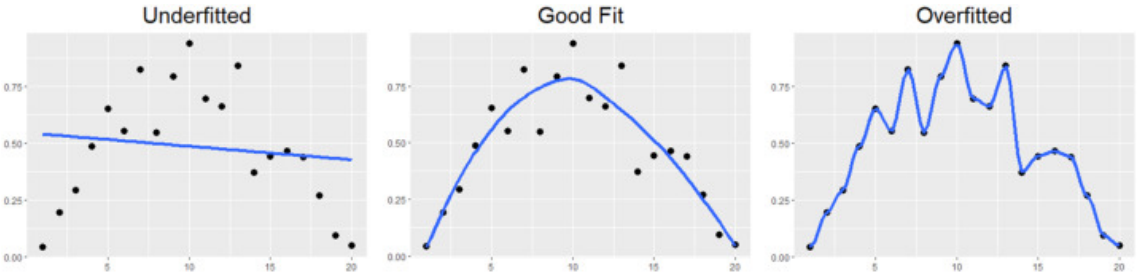


Figure 16: Example of overfitted and underfitted data [29]

Overfitting can be observed when a machine learning algorithm grows more effective at making correct guesses at its data but does not become more effective on the validation data set. Because it is possible to observe it this way, it is very desirable to

stop the learning once guesses on the validation data set stop improving. An example of overfitting can be seen in figure 16, where the loss on the overfitted graph is lower, but the curve that would fit data more generally is abandoned for a curve that fits only to this specific dataset.

Overfitting is especially a concern when working with a limited dataset. A dataset where too much data is similar will lead the model towards learning patterns relevant to what is common in the specific dataset rather than attributes that are common to the actual target class.

Underfitting is simple to observe, as it is measured by the model having too much loss. This typically means the model has failed to learn enough about the actual structure of the data, in figure 16 the under-fitted model only tries to reduce its mean square error but gains no insight into the structure of the data. Underfitting usually means the model needs some kind of change, for instance to the activation function.

2.4.3 Automatic Speech and voice Recognition

Automatic Speech Recognition (ASR) by computers is a field that has been long pursued in machine learning. Speech recognition has come a long way as new machine learning technology is researched and more effective algorithms are found.

Automatic speech recognition has come a long way, but an issue with the technology is that it is limited by data available for different languages. A large amount of data is required for good speech recognition, but some languages such as Scandinavian languages are limited by low availability.

A typical architecture of an ASR system is displayed in figure 17, Signal processing and feature extraction are similar to the processing for other audio tasks, attempting to remove noise and converting to a spectrogram. The language model and acoustic model give a score for the probability of a hypothesized word sequence and gives an output based on the sequence with the highest probability. [46]

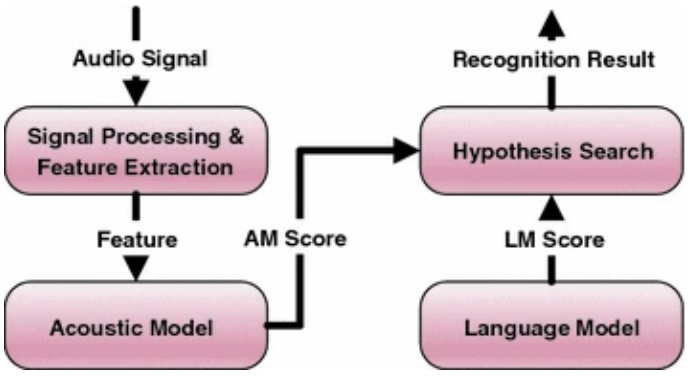


Figure 17: Architecture of ASR system [46]

Voice recognition aims to identify the user speaking, rather than identifying words and sentences from the sound. Each human has a unique tone, rhythm, frequency, and pitch to express speech, including where they stop in phrases and how quickly they speak depending on where they are in a phrase. [19]Because of this, voice recognition has applications in security and can perform well with machine learning models and models with very high accuracy exist.

2.4.4 Supervised and unsupervised learning

Supervised learning is a widely used methodology in machine learning, that drives much of the accomplishments the field has. Supervised learning is the technique of providing the computer with already labeled data, which then can be used to test its accuracy and improve further training. Data is split into a training set and a testing set, and the data must be held separate should the learning have value when used on a new data set.

Due to the data needing to be split up, supervised learning requires an unusually large data set, which for many tasks can be a challenge. For example, audio files are especially large, and compressing an audio file can lead to the loss of valuable data.

Unsupervised learning requires no test or validation data, and only uses the given data for training When doing unsupervised learning, the goal is often for the computer to find some sort of pattern in a chaotic environment.

For example, if the task is to separate data into multiple categories, also called clustering, the goal is to find similarities between data and group data that are similar to each other. The number of clusters can be decided manually but is often chosen based on getting an optimal distance between each of the groups, such that the different groups remain as concise and separate as possible.

Another form of unsupervised learning is dimensional reduction. In dimensional reduction, the goal is to find which factors have the closest correlation to another given factor or group of factors. This can often be used in combination with other machine learning techniques to create more effective models and to gain insight into the data. [5]

2.4.5 Type 1 and type 2 errors

In statistics when trying to prove a null hypothesis, there is two main types of errors that the results are divided into. Type 1 errors are a wrong rejection of the null hypothesis. Type 2 errors are the wrong acceptance of the null hypothesis. In general, there is always a chance that these kinds of errors occur, and their probability can often be calculated. Depending on the null hypothesis one might want to lean into either type 1 errors or type 2 errors having a higher chance of occurring because the consequence of one error occurring might be much more grave. Table 1 shows how these errors correlate. [41]

	H_0 is actually	
	true	false
Reject H_0	Type 1 error	correct
Accept H_0	correct	Type 2 error

Table 1: Type 1 and type 2 errors

When working with binary classification, a similar logic is applied. Should a model for instance be poor at identifying a critical scenario, the model might not behave much practical use. For instance, when classifying whether a vehicle is about to crash, the null hypothesis can be that the vehicle is going to crash. Getting a type 1 error could lead to fatal scenarios. In this instance, one would want to heavily skew the model such that type 2 errors are much more likely than type 2 errors.

2.4.6 Loss Functions

When the model makes a prediction $\hat{y}$, the output needs a form of evaluation to measure the correctness of the prediction. This output is usually called the loss of the prediction, and it is calculated using what is called a loss function. Loss functions can vary greatly depending on the task, and different loss functions can drastically change the result and consistency of a machine learning algorithm. Loss functions are also called cost functions or error functions.

When using a loss function of a network containing weights, the loss function will be calculated by taking the weights as input:

$C(W_1, W_2, W_3, W_4 ... W_n)$

Where C is the loss function and W is the n weights.

One of the most commonly used types of loss functions is quadratic loss functions, as it is symmetric and mathematically tractable. The most simple version of these are the mean squared error, defined by:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Mean squared error heavily penalizes larger errors compared to smaller ones, which is generally desired as one wants the algorithm to identify errors with a larger potential for improvement.

It is important when evaluating a machine learning algorithm using loss functions that they remain the same across different models, such that the comparison remains fair. Loss functions are inherently biased and evaluate loss differently,

2.4.7 Gradient Descent

The gradient descent algorithm is an algorithm to minimize a model's loss function. Gradient descent works by updating the parameters in the opposite way of the gradient of the function. As the curve flattens, gradient descent will take smaller steps so as to not overshoot and reach a valley.

Gradient descent has three main variations, batch gradient descent, stochastic gradient descent, and mini-batch gradient descent.

Batch Gradient descent

Batch gradient descent calculates the gradient of the entire dataset for each update and is therefore very slow to compute for datasets that do not fit in memory. [38]

Stochastic Gradient descent

Stochastic gradient descent works by calculating the gradient for each individual training example. Stochastic gradient descent performs frequent updates with high variance, which allows it to shoot to new and potentially better local minimums. [38]

Mini-batch Gradient descent

Mini-batch gradient descent mixes the two prior techniques, and takes small batches of size n and computes the batch gradient descent for that mini-batch. This allows the calculation to still use effective matrix operations to calculate the gradient. [38]

Rprop and RMSprop

Rprop is a version of mini-batch gradient descent by Hinton et al. [15] that only uses the size of the gradient, meaning it is less likely to get stuck on hills or move too far

way reaching sudden slopes.

RMSprop keeps a moving average of the square gradient for each weight. RMSprop is given by:

$MeanSquare(w,t) = 0.9MeanSquare(w,t-1) + 0.1(\frac{dE}{dw^t})^2$ [15]

2.5 Agile Development

Agile Development is a development technique that was specifically designed as a response to the rise of the waterfall technique. The waterfall technique creates long arduous development and holds a vice grip on development plans and schedules. Agile development was designed by a group of developers to be much more flexible and adaptable. The agile manifesto is as follows:

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan The items on the right are still valued, just less so than the items on the left.

Agile development typically works by using sprints; one or two-week-long development plans that works towards having a more complete product by the end of each sprint, such that customer satisfaction can be held high and so that multiple avenues for feedback are made more readily available. This allows developers to have simpler and clearer goals for each sprint and is much better for continuously tracking progress.

Having a complete product by the end of each sprint means that big changes in plans still leave a partially completed product, ensuring that work does not go entirely to waste. The waterfall technique is inflexible when it comes to schedule and takes a long time before a minimum product is produced.

3 Methodology

This chapter will discuss the methodology for this project, including methodology for finding related work, the choice of technology and general decisions regarding the project.

3.1 Scientific Method

The scientific method is a method applied for testing and constructing a hypothesis when working in sciences. The scientific method is important in almost every field of science to establish new theories and testing older and established ones.

The most common version of the scientific method is a researcher establishing a hypothesis, and then testing it. Should the hypothesis fail the tests, it is modified to fit with these new results or entirely thrown. This is then repeated until the hypothesis stay consistent with the results. These hypothesis newer truly leave this testing phase, but instead becomes more accepted as attempts continues to prove it. [37]

Because hypothesises go through this cycle, it is important that other researchers are able to repeat the same experiments, or perform similar experiments. Therefore scientific experiments need to be well documented an information about the experiment must be made available.

3.2 Method for related work

For this project, finding relevant work in audio machine learning and convolutional neural network. The relevant work was found by research on google scholar, and finding articles books or projects with multiple relevant sources, good documentation of results and multiple interesting insights. Since machine learning for adblocking in podcasts is not a widely covered subject, projects that used similar technology was prioritized.

Resources that covered adjacent technology was also reviewed, such as other forms of spectrograms or neural networks. This was done to highlight relevant technology in the field for alternatives in this project and especially as strong contenders for future work on the same dataset.

The theory was written based on work that was read during the semester and knowledge acquired through reading relevant sources. The sources chosen to cite in the theory section has been mostly scientific papers describing the relevant theory in high detail.

Choice of technology in creating the dataset was chosen by necessity. Splitting the files is relatively fast, but transforming the files into .wav files through ffmpeg is a lengthy process if one wants a larger dataset. Alternatives may exist, but there's a lack of well documented technology for this purpose.

3.3 Development plan

At the beginning, it was found that the project had four important parts, and a four-step plan for the project was created.

1. January: Start-up, vision document, work contracts, planning for what obstacles we expect to find, and how to solve these.

2. February: Get familiar with what technologies that will be used, develop tools that help solve the obstacles we are meeting. Download podcasts that fit the criteria set.
3. Mars: Convert podcasts files to usable sorted data, split the files into smaller chunks and separate the advertisements from the podcast.
4. April: Create the CNN model, have it train on the data and tweak the model and the data so that it achieves results.
5. May: Work on rapport, concluding the results and minimal new development if necessary. Delivery on may 20.

Because the time each sprint would take was not strictly defined, the scope of the results was dependent on how much time was left to allocate as the due date of the project was approaching. A more in depth look at how the group has worked can be found in our time sheet.

3.4 Gathering data

There are currently no available datasets that we could use for training our model. We had to figure out how to gather a sufficient amount of podcasts, split out the advertisements, and convert them to readable data for training.

3.4.1 Gathering data

The project required a large amount of data to work on, and currently, there are no such datasets available for us. Consequently, we had to create our dataset. The first technology we created was a data-scraper; a data-scraper is a process used to download lots of data from the web to local storage. We combined an RSS-reader (real simple syndication) that can read the RSS feed of a website with a file downloader that would look for .mp3 files in the XML we got. A popular way to host podcasts is using Acast.com, and when hosted there, the podcasts will be in an RSS feed. This way, we had lots of podcasts to download. For the amount, we settled on having 50 or more of every podcast in every language. Some podcasts were much larger than others but did not contain more advertisements therefore, there was not much emphasis on how big the podcast was compared to others, but we did want to have some difference as there might be data that the model would pick up on.

3.4.2 Choosing samples

Another reason to use acast feed is that acast will implement advertisements into the podcasts. These advertisements are known as localized advertisements which means they are different based on where you are localized, as we live in Norway we get Norwegian advertisements. This is useful for our dataset as we can change our localization with a VPN (Virtual Private Network), which means we can connect to other countries and get their advertisements as well. For our project we chose to get Norwegian advertisements since we are from here, American advertisements as they are known for having a lot of advertisements, and Dutch advertisements as having different languages is something that might be useful for the machine learning model to learn on.

We also used the Tor software for the last download, installing tor on a Linux computer lets us connect through it and use it like a vpn. The reason to use tor as well like this,

is that with tor we can download while hiding our localization. From Acasts perspective it will then not give us any advertisements, and by extension we get a clean podcast that can be used both as an example for the model for what isn't advertisement as well as using a diff to compare with the other podcasts so we find the position of the advertisements in data form.

3.4.3 Converting to readable data

Now that the data files are sorted into ”Advertisements” and ”Not-Advertisements” they then need to be processed into information that the computer can read. Computers don't understand sounds like we do, only code. The process of conversion is to first convert the waveform files with FFT, which then can be converted into mel-spectrograms which are pictures. Pictures are simpler for the computer to work with, as they lose the dimension of time, and also colors can be directly read by the computer. We now read the pixels of the picture as data and store this as small individual segments of the bigger soundfile. These new files of data can be directly read by the model which can now start training.

Our podcasts were mainly selected from Acast.com, as Acast allowed for an easily available dataset to work with. Podcasts were chosen based on them containing advertising at some point, and when deciding which podcasts to use, the initial set was found using Google. Acast podcasts are easily found på searching with Acast and some other popular terms such as "news”or "videogames”, and ads are often placed in the beginning of podcasts, so finding podcasts containing ads were relatively simple.

Podcasts were also downloaded from different regions using a virtual private network (VPN) to compare the algorithm with different languages, and using a trained model to test its efficiency when performed cross-language. The three languages chosen were Norwegian, English and German as they are languages that are linguistically very close (need source), and therefore could yield interesting results when testing results.

The data gathered from acast was also downloaded using tor (should explain further), since there is a loophole in Acasts advertising that made Acast unable to insert advertising when downloaded through tor. This meant that a set of podcasts with advertising and one without was gathered.

The mp3 files were then compared with a byte differentiator algorithm that compared the audio files containing advertisements with the version not containing any. Parts that were unequal in the two files, must be ads since that is the only difference between the tor and normal versions of the podcast. MP3 files are stored such that they can be split at any point in the file without damaging the data and still remaining playable. Thus the start and end of the intersections are stored in an array, and the file could be split into correctly labelled mp3 files containing either only podcast or advertisement. In this way, advertisements were filtered out and placed into their own folder. This provided a set for the model to use for training.

3.5 Creating the model

With readable data we now require a model that will train on it. We imported the TensorFlow library and set up a model using its functions. TensorFlow is a machine learning library developed by google and gives us easy access to useful functions for splitting the dataset, training and testing.

3.5.1 Training the model

Now that there is data gathered and stored on a file, we split the data for training using the train_test_split function. This way, some of the data is kept for testing so that the model does not test on training data. The reason this is done is that the model should not know the answer to what it is tested on. Instead, the test data does not affect the training on the model but instead gives an insight into how well the model handles new data, which is what it would be working on as a product. The model itself is built with convolution layers, where the data will be portrayed through the layers as explained in the theory. The first layer is a size of 32. After testing, it seems to peak at higher accuracy with a 64 layer. However, it is also significantly more inconsistent with the results when running it with cross-validation, expand in the results section. We use Max-pooling to downsample to smaller matrices and then flatten the model back to 1 dimension for the results. The model did not accept new layers after the first max-pooling as it would overfit and then not give any results to train on. For learning rate, we had the best results when using 0.0001, while most models we looked at used 0.001. For this reason, doing additional epochs was also important as the learning rate was lower, but the results were significantly different.

3.5.2 K-fold Cross Validation

When testing the model and with the data, different questions needed answers. Firstly can the model extract useful information for distinguishing the advertisements and podcast parts of the podcast we have found for the dataset? However, it does not necessitate that it works on podcasts not in the dataset, even if this works. A common problem in machine learning is a bias toward the data being trained on, leading to unfortunate results like facial recognition that does not work on certain races. [30] In conclusion, training on only one dataset would possibly make it biased towards that data. There are a few ways to fight this. As discussed earlier, the dataset consists of advertisements in different languages. Therefore one way is to exclude one language while training and see if it still recognizes the advertisements. Another way is to exclude a podcast and see if it still finds the advertisements while listening to a new voice. Lastly, K fold cross-validation is used to iterate through the dataset to make all the data samples be test data and training data at some point. K fold cross-validation works by training the model multiple times. The first round uses the first 1/5 of the data, while the next round will change the test data to the second 1/5 of the data. This process iterates until it has used all the parts of the training data as test data, then it takes the average scores, which will be a more reliable score than just running the training once.

3.5.3 Testing specific training data

The model can train on its data and get good results, but there isn't that much variation in the data. To check if the model can find advertisements it hasn't seen before or work with podcasts that it hasn't trained on. Its also important to see how language affect the model, since all the advertisements from Norway is in Norwegian (it isn't like that with the dutch advertisements) we made a separate dataset that will test exclusively on Norwegian advertisements and train on non-Norwegian. This will bring insight into how important it is to train on languages. It would also be useful to see how it works on an different podcast than what it is training on, so we also made a training set where it would not train on Vicegaming, and rather test on it in the end.

4 Results

In this chapter we will present how we achieved some of the work goals we had during the project, particularly about the data gathering phase. This chapter will also present the empiric results from our testing, what we were testing for and short about what we read from that data.

4.1 Goals

Initially, the goal was to have the model find and cut advertisements from new podcasts when downloaded. We found that we did not have time to reach this goal. We have reached a trained model that makes predictions about which sound bites are advertisements and podcasts with a significant accuracy. A task to reach this goal was to create a dataset that the model could train on. Currently, there are no widely available datasets that are fit for creating such a model.

4.1.1 Data gathering

Creating a dataset with varied data required multiple programs created for the project. A data scraper was created and integrated with an XML reader to read RSS-feeds and download the mp3 files listed in the feed. After finding it difficult to balance the number of podcasts, we limited it to 50 downloads per feed.

The mp3 with advertisements was compared to the same podcast without using a diff program, which noted the position of where they differentiated. The advertisements were then separated from the podcast, and then it was converted to .wav format to decompress the sound data.

At the start of the project, gathering data effectively was a goal, such that the project should be reproducible. to this end, multiple smaller programs that work towards this end has been developed. As outlined earlier, one program that reads and downloads from rss feeds normally and using Tor has been developed. The program also had functionality to download from different languages for use later.

In addition programs that programs that filtered out the advertisements from the previously downloaded files and stored them separately on the computer was developed. Finally, a program that converted the files into the .wav format was used, as this would be necessary for the machine learning.

4.1.2 Machine learning

Another goal of the project was to make one or more machine learning models that got some sort of significant result, whether it showed the non-viability or viability of the project. To this end, a machine learning model based upon mel spectograms and deep learning was written, with an accuracy of above 80% was achieved. This is above what what the project aimed for, and thus that goal was achieved.

4.2 Data gathering

Using the data-scraper we ran it for some days and gathered an amount of data, which is displayed in table 2, notably the vice gaming podcast data is relatively larger in size, but the amount of episodes is comparable to the other podcasts.

Dataset	size	files
nor	9.7GB	237
eng	9.7GB	237
de	9.0GB	189
tor	9.5GB	237
BBC food	2.6GB	153
Vice Gaming	23.2GB	223
Vice Guide	5.5GB	325
Dnd is for nerds	6.8GB	200
All languages and podcasts	38.1GB	900

Table 2: Total amount of podcast audio before processing

4.3 Converting the data

During conversion from mp3 to waveform some files became unusable. The reason some files were not usable is that during the data conversion something could go wrong with the differentiation, causing some files listed under advertisements to be either a huge amount of podcast or even some entire podcasts. Those files got removed as they were not purely advertisements, so the model would learn wrong information from them. The data is displayed in table 3. The data has grown in size, but this is because the .wav format is entirely uncompressed.

Dataset	size	files
Advertisements	4GB	596
Non-advertisements	108GB	388

Table 3: Data after conversion to wav and filtering into advertisements and non-advertisements

4.4 Training the model

Preferably the model would work on a somewhat equal amount of each class it would classify, so we cut files from the non-advertisements until we had equal sizes. This process also helps with time, as converting the entire dataset to mel-frequencies would take multiple days, and it had to be done multiple times because of errors we experienced further down the line. The training on 8GB of data with 50 epochs takes 5 hours. Using these hyperparamteres gave us the best result we got with above 91 accuracies. We did run with fewer epochs and got slightly worse results. 89 accuracy at 10 epochs which runs in an hour, is not considerably far off for way less time investment. Running this many epochs is necessary because the learning rate is relatively low at 0.0001. The model was tested with an entire dataset and then with some missing categories so that the model would experience those missing for the first time. To account for randomness in the training and testing set, we used k fold for cross-validation, then took the average accuracy of five runs and tests. Cross-validation gives a more reliable accuracy score than running just once, as we have used all the data as tests at some point. The result from this training is presented in table 4

Fold	Loss	Test Accuracy
1	0.8260	0.9171
2	0.8973	0.9121
3	0.7794	0.9184
4	0.8338	0.9166
5	0.8151	0.9157

Table 4: Training when performed on all data

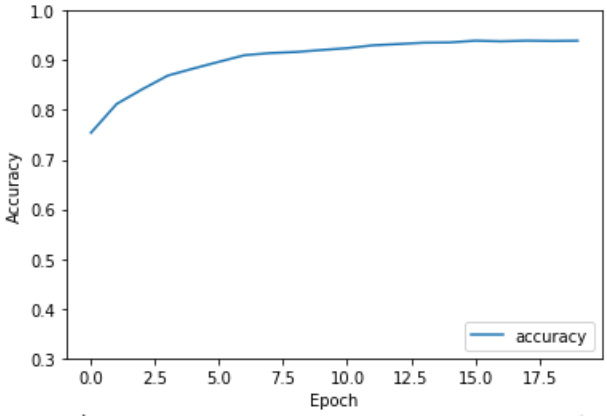


Figure 18: Graph from training fold 1.

4.5 Changing epochs and layers

Two of the five folds had lower accuracy than expected when running with multiple layers and folding, while the other three had slightly better accuracy. This indicates that the extra layers cause overfitting. Therefore, the other tests were done with only one layer. Table 5 presents the data when running with two layers.

Fold	Loss	Test Accuracy
1	0.7300	0.9048
2	0.5757	0.9249
3	0.7802	0.8708
4	0.5570	0.9260
5	0.5763	0.9287

Table 5: Testing done with 2 layers and 20 epochs

4.6 MFCC vs mel-spectrograms

We also tested using MFCC instead of mel-spectrograms, this training set runs faster as there are fewer properties, but may end up not learning anything.

Fold	Loss	Test Accuracy
1	0.9033	0.9347
2	7.5352	0.5115
3	7.7997	0.4885
4	0.9249	0.9337
5	0.8469	0.9378

Table 6: Results from MFCC training set, using 2 layers and 20 epochs.

As presented in these results, in fold 2 and 3 something went wrong with the learning as it got stuck around 0.50 accuracy. This is unlike the two weaker folds from table 5 where the results were slightly lower.

4.7 Training and testing for new data

4.7.1 Vicegaming podcast

When testing on a dataset that misses certain datasets we can't use folding. This is because those certain datasets is exactly what is supposed to be tested, therefore its no use in folding through the training set. In this trainingset we set aside the Vicegaming podcast, and trained on the others. When the training was done, we checked how well it would test on Vicegaming as it had never seen it before.

Epochs	Loss	Test Accuracy
5	1.0082	0.8296
10	0.9034	0.8494
20	0.9883	0.8686
50	1.1292	0.8781
100	0.6961	0.8953

Table 7: Training done with the vicegaming podcast used exclusively as testset.

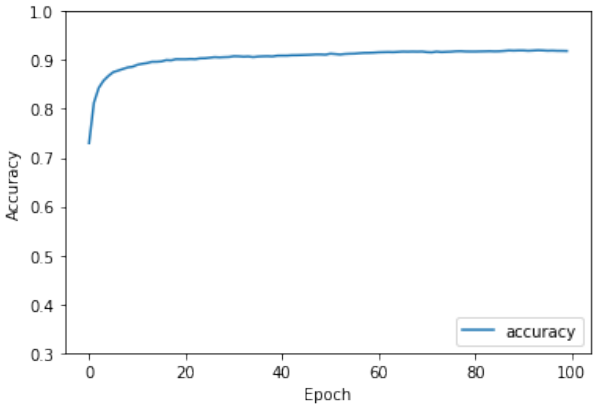


Figure 19: Graph from training without vicegaming, 100 epochs, 2 layers

Relatively good results, considering the podcasts is entirely unknown to the model, the number of epoch are not that high and the model is simple.

4.7.2 Norwegian advertisements

We trained using podcasts and advertisements from English and dutch podcasts to evaluate generalization for different languages while testing on Norwegian. Norwegian advertisements have significantly different MFCC features. In Table 8 we can observe that there is no change in test accuracy when training additional epochs, suggesting that the model requires to be familiar with the language.

Epochs	Loss	Test Accuracy
5	2.7165	0.7001
10	3.4758	0.6786
20	3.8018	0.6807
50	4.1673	0.7066
100	4.3496	0.6723

Table 8: Norwegian podcasts used as test set.

Despite the model not learning to differentiate Norwegian in testing, the training data changed compared to when training on the full dataset. In Table 9 we can observe a significantly higher accuracy compared to Table 4.

Epochs	Training Accuracy
5	0.9309
10	0.9510
20	0.0.9640
50	0.9719
100	0.9749

Table 9: Norwegian podcasts omitted from training

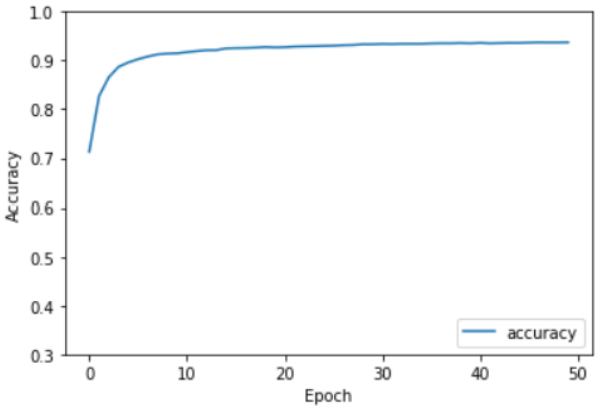


Figure 20: Graph from training without Norwegian, 50 epochs 2 layers

5 Discussion

In this chapter we will discuss why the results turned out the way they did and what our sources of error are and what we would do different if we were to tackle this task again. We will also discuss how the group work turned out.

5.1 Discussing results

It is important to have enough data when working with machine learning, more is almost always better for the model, but it has to be balanced by time spent training. Using one layer or two layers didn't make a huge difference in the results, but the time spent training was 6x longer It would be useful when creating a finished product, but for these testing purposes it doesn't seem necessary, as long as the model can prove that it is possible to extract data about where there are advertisements in the podcast with a significant accuracy, it has done its job.

5.1.1 Results from gathering data

The data gathering script was able to gather data fort the program very efficiently. Large amounts of data can be downloaded relatively quickly, and transforming the data into ads and waveform goes faster than the download itself.

Due to time constraints, in this project the size of data gathered was limited, as time had to be allocated to other parts of the project.

5.1.2 Results from training

The model performed well on the dataset, and a high accuracy was generally achieved across variations of the dataset, proving the efficiency of spectrogram based machine learning on the dataset. MFCC wasn't perfect, while it was able to produce results, it would also have models that did not learn anything. When it did produce results they were similar to mel-spectrograms, but took a quarter the time to train.

5.1.3 Generalization of the model

We had two tests for generalization, one where we tested if it could work with new podcasts that it had not seen before, but in the same languages that it had trained on. And one where the advertisements were in a new language that it had not trained on. In the test on a new podcast we used the Vicegaming podcast as test (table 7), while the accuracy is slightly lower than the accuracy on the full dataset tests, it is still clear that the model can distinguish between the advertisements and the podcast.

When testing on the norwegian dataset (table 8) we had the opposite results, where it has a hard time understanding what these new sounds are. This is also reflected on the data from when it was training without Norwegian (table 9), we achieved the highest accuracy compared to all other tests. This indicates that the model require more data from languages to train.

5.2 Sources of error and weaknesses

5.2.1 Inexperience

The most significant source of error for us is inexperience. Most of the technologies that are used is new to us and we had to learn to traverse using it. In addition some of the challenges that is !! with this project is had to be solved by making our own tools. Datascrapers, mp3 to wav converters, that we had to design from ground up. While this was fun and interesting for us to learn, it is also something that would be done more effectively by someone whom had met tasks such as these before.

While we had experience with machine learning from last semester, that was using datasets that we could load in from python libraries and that we didn't have to process any of the data on. With this project there was a need to create the data from scratch. There was no previous experience working with sound data in the group, figuring out how the computer should read the data was a choice we had to make, as we could choose between mel-spectrograms and simply reading the waveform [14]. We went for mel-spectrograms as they seem to be the standard, even though the research indicated there might be advantages to using the waveform.

5.2.2 Dataprocessing

A problem with the gathering itself is that podcasts naturally have magnitudes more data that isn't advertisements than what is. This meant that we had to remove a lot more of the podcast data before training to fight over representation. There might have been more structured ways for us to achieve this, but time and inexperience in data handling made us not do this as methodically as we might have liked.

There also was a slight problem with the detection of advertisements with the diff where we would get full podcasts in the advertisement folder, we are unsure why this happened, but as we still had hundreds of files that we knew were advertisements without podcast we kept those and removed those that were long enough for it to include podcast. This had to be done as if the non-advertisement data was kept, the model would train on data that would teach it the opposite of what we want it to learn. And since the data labeling was automated by the tools that were developed, there wasn't an time effective way to go through the files and find specifically those that there was something wrong with.

MFCC could have been a useful path to follow up on more, but as we had a working model with the mel-spectrogram datasets we discontinued the MFCC testing since we met problems in the training. It is possible that there are minor tweeks that could be done to the model to fix these problems but we decided against that because of time constraints.

When doing generalization of new podcasts, we only tested on one dataset. We should have tested with multiple datasets where we could change which podcast was the unseen one.

5.2.3 Choosing data

The dataset is not entirely randomly chosen, and therefore there could be bias in the podcasts in the data set. Acquiring a large enough dataset for there to be no bias when using the data set, would need a substantial increase in the amount of data gathered, such that the variance in available data would be higher. The podcasts used for this dataset are few, and therefore whether the performance would stay consistent on

testing on more substantial or just different data sets can not be entirely predicted before tests are done.

As seen in the gathered data, the vice gaming podcast is quite heavily represented in the gathered data. Much of that data is just normal podcasts which is cut away before training, but there is still possibly more advertisements during these long 2-3 hour podcasts than there would be for one that is 1 hour or less.

The low number of podcasts was also chosen so that a large enough amount of data for a single podcast to be used as validation or testing data. Doing so would mean the data could be tested for overfitting, considering our data size and data source.

Using only Acast.com as the source for podcasts and advertisements, means that there could be very low variance among the acquired advertisements. Podcast hosting websites such as Spotify and Acast often have advertisements for their own platforms, and may also have a lower amount of advertisements available total, leading to similar and perhaps equal advertisements. This was not taken into consideration in the model, which means there could be duplicate data included in training. This increases the chance that the model overfit, and testing whether the data is overfitted is difficult when there is not enough variance in the available dataset. A better testing set should be taken from another source than Acast to achieve better testing for overfitting.

5.3 Teamwork and development

Going in to the project, the group had clear goals to work towards but no idea how easy or difficult they would be to achieve. As the group have different strengths and weaknesses on the team it was usually easy to delegate tasks to where it would be most efficient to work. Though this also meant that it was not a balanced workload throughout the project. But if one of us felt that we were overworked on the task we were working on, then the group had a short meeting explaining how things are going and what we required help with. As the group met hardships personally or professionally we were able to support each other well.

There were new concepts for the group to tackle in this project, such as data gathering and processing. When the group felt like the group started losing track of how to solve a current task, we contacted our supervisor and followed the advice he gave as he had experience with these tasks.

Covid-19 is a current worldwide problem that has had the potential to affect the work the group did. In the beginning of the year, the group worked a lot from home, which felt isolating and lonely and tore on the motivation a lot. We got through it but we feel that it impacted us somewhat. As the school opened up more we were able to meet up and had a better work environment.

5.4 Ethics

Advertisements in podcasts are put there as a way for the creators to support themselves without locking their content behind a paywall. This means that it is easier to spread their own content and draw in more fans, and the advertisements makes these fans have a income value outside of being exposure. In other words it makes the creators care more about all the fans instead of mainly the once that pay some subscription. As a listener however, advertisements are irritating and breaks the flow of what you are listening to. It lessens the experience of the podcast. There is also a question about making you stream data that you don’t want, in TV historically you haven’t had a choice, but for online content you have had the option in forms web

based ad-blockers, they usually work by stopping web-requests from being sent to other websites that hosts the advertisement, as the request is stopped you will not have any advertisement sent to your computer. Shine, an Israeli ad-blocking company claims that a 5 minute long section of mobile gaming went from 50KB to 5MB when ad-blocking software was not used. [32]

Removing advertisements in podcasts work slightly different from other online content. When you download a podcast the advertisements are put into the file itself that you are downloading, currently we can not block this from happening, advertisements consume a huge amount of internet speed and data. The research has been to find a way to remove those parts of the file, which is a slightly different ethical question. As you have downloaded the file you should be free to use it as you please, you yourself choose what part of the audio you wish to listen to. If there is advertisements, you are free to fast forward through it so you do not have to listen. This projects aim has been to simply remove that part of the audio that you do not want. Ethically we feel this is justifiable as this is your data to deal with as you want. There might be many reasons you do not wish to have the advertisements. They can be interruptive for the flow of the podcast you are listening to, or inappropriate content for you.

On the other side, if such technology for blocking advertisements becomes more avaliable, it is cutting into the monetisation of the content for the creators, making entering into such a field more inaccessible. Overall this seems to be the direction online content is headed, anytime advertisements increase, answers to how to remove them rises as well, which is what we have done here.

Machine learning as a whole has had multiple uses as a result of better hardware and technology in recent years. Machine learning has been used to detect cancer, figure out errors in devices and much more. Using machine learning in audio is still an unsolved field that has room for development. However there is fear around AI controlling our lives more than we would want. [21] Contributing to this development can be seen as unethical by someone with this fear.

6 Conclusions and Further Work

This chapter will answer the research questions based on the discussion and results from the previous chapters. We will also talk about how well we achieved the goals we wanted to accomplish at the beginning of the project and what further work could be done if the project was continued.

6.1 Research question answers

6.1.1 Can machine learning be used to effectively detect advertisements in podcasts?

Yes, based on the testing results from table 4 where we tested the full dataset with folding, as well as table 9 where we tested with mostly English speaking advertisements. The results were accurate enough that we can conclude that the model learnt to distinguish advertisements from the podcast to a satisfactory degree.

6.1.2 Are MFCC features sufficient to classify between advertisement and podcast?

It is clear from the vicegaming test (table 7) that the model is able to generalize and learn to recognize advertisement and podcast sections outside of the training data, only using the sound data from the podcasts. However MFCC requires extra attendance compared to mel-spectrograms, incase the training creates too many dead neurons.

6.1.3 Can a model that has trained on one language detect advertisements in another language?

No, the model does not handle new languages well, results (Table 8) indicate that it is close to random guessing when classifying Norwegian advertisements for the first time. However this might also be because of other MFCC features in the Norwegian advertisements in addition to the language.

6.1.4 Is there enough available data training a machine learning model based on podcasts and advertisements?

There are no readily datasets for training on podcasts and advertisements. However it is possible to gather enough data and processes it into a workable dataset using a set of tools as we did. Perfecting the function of the tools, possible making a user interface for them could let us fully streamline the process, but we did not develop all the way there as that was not part of the scope for this project.

6.2 Further Work

The dataset used in this thesis is entirely derived from Acast. Testing with a much wider dataset could provide interesting results, and information about ads could also be researched. The same method of acquiring podcasts both with and without ads are

possible from other sources such as Spotify, and would therefore be not too difficult to implement.

Testing with bigger datasets are desired, but downloading, preparing and doing machine learning on bigger datasets takes a vast amount of time.

The dataset doesn't have to be specifically limited to podcasts. Other audio based entertainment providers using ads such as YouTube would have new challenges, and might be too varied for the approach used in this thesis.

Another type of AI that we would like to test for a project like this is Speech processing. Speech processing would let us find phrases and words that would indicate that something is advertisements or not. It would be interesting to see if it is more or less accurate at detection than the sound based approach. Ideally they would be combined to give more secure results where at the very least all of the podcast would still be available, with less-to-no false-negative results.

During the process of development we have made some steps simple because of inexperience and simplifying means we could continue towards results more effectively. We would like to go more in dept on the conversion of data from the wav format to mel to an array of values. Currently as of delivery this is quite automatically done by the librosa library, but as our understanding of the data has developed, we believe it possible that the data results could be even better if the data was curated even more during this conversion. An example of this would be to see if cutting of certain frequencies would have an impact on the learning. There is also interesting research about not converting to mel at all, and using raw waveforms and using small filters. [16]

When training and testing on the datasets, we found that even if we managed to balance the amount of podcast to advertisement ratio, it didn't mean that the data was fully balanced. It was still sorely lacking in terms of language for example. As we could see in the results, Norwegian holds the model back quite a lot in testing, it is inaccurate when classifying so it probably doesn't have enough data to extract from, giving the model on the full dataset considerably less accuracy than when training without it. Working more on balancing the datasets is something also would have done better if we continued.

In general, since the data gathering technique is available, testing with other models and comparing their results would be the next step.

References

[1] July 2009. url: https://web.archive.org/web/20130208164732/http://lists.mcgill.ca/scripts/wa.exe?A2=ind0907d&L=auditory&P=389 (visited on 4th May 2022).

[2] Alif Bin Abdul Qayyum, Asiful Arefeen and Celia Shahnaz. 'Convolutional Neural Network (CNN) Based Speech-Emotion Recognition'. In: *2019 IEEE International Conference on Signal Processing, Information, Communication Systems (SPICSCON)*. 2019, pp. 122–125. doi: 10.1109/SPICSCON48833.2019.9065172.

[3] Saad Albawi, Tareq Abed Mohammed and Saad Al-Zawi. 'Understanding of a convolutional neural network'. In: *2017 International Conference on Engineering and Technology (ICET)*. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[4] Sören Becker et al. 'Interpreting and explaining deep neural networks for classification of audio signals'. In: *arXiv preprint arXiv:1807.03418* (2018).

[5] Olivier Bousquet, Ulrike von Luxburg and Gunnar Rätsch. *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*. Vol. 3176. Springer, 2011.

[6] Colin M.L. Burnett. *An example artificial neural network with a hidden layer*. 2006. url: https://commons.wikimedia.org/wiki/File:Artificial_neural_network.svg.

[7] KP Burnham. 'In KP Burnham & DR Anderson'. In: *Model selection and inference: A practical information theoretic approach* (2002).

[8] Dhanith Chathuranga and Lakshman Jayaratne. 'Automatic music genre classification of audio signals with machine learning approaches'. In: *GSTF Journal on Computing (JoC)* 3.2 (2013), pp. 1–12.

[9] Chandra Churh Chatterjee. *Basics of the classic CNN*. July 2019. url: https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add.

[10] Dejan Ćirić et al. 'Audio signal mapping into spectrogram-based images for deep learning applications'. In: *2021 20th International Symposium INFOTEH-JAHORINA (INFOTEH)*. IEEE. 2021, pp. 1–6.

[11] Monika Dörfler, Roswitha Bammer and Thomas Grill. 'Inside the spectrogram: Convolutional Neural Networks in audio processing'. In: *2017 International Conference on Sampling Theory and Applications (SampTA)*. 2017, pp. 152–155. doi: 10.1109/SAMPTA.2017.8024472.

[12] *Feature extraction and supervised learning on fMRI: from practice to theory - Scientific Figure on ResearchGate.* url: https://www.researchgate.net/figure/The-technique-of-KFold-cross-validation-illustrated-here-for-the-case-K-4-involves_fig10_278826818%20[accessed%2020%20May,%202022].

[13] *FFT of Cosine Summation Function*. url: https://commons.wikimedia.org/w/index.php?curid=86139361.

[14] Shawn Hershey et al. 'CNN architectures for large-scale audio classification'. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 131–135. doi: 10.1109/ICASSP.2017.7952132.

[15] Geoffrey Hinton et al. 'Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups'. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 82–97. doi: 10.1109/MSP.2012.2205597.

[16] Taejun Kim, Jongpil Lee and Juhan Nam. 'Comparison and Analysis of SampleCNN Architectures for Audio Classification'. In: *IEEE Journal of Selected Topics in Signal Processing* 13.2 (2019), pp. 285–297. doi: 10.1109/JSTSP.2019.2909479.

[17] Shashidhar G. Koolagudi et al. 'Advertisement detection in commercial radio channels'. In: *2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS)*. 2015, pp. 272–277. doi: 10.1109/ICIINFS.2015.7399023.

[18] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. 'ImageNet Classification with Deep Convolutional Neural Networks'. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. url: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[19] Guus de Krom. 'Consistency and reliability of voice quality ratings for different types of speech fragments'. In: *Journal of Speech, Language, and Hearing Research* 37.5 (1994), pp. 985–1000.

[20] Yi Ren Leng et al. 'Alternative frequency scale cepstral coefficient for robust sound event recognition'. In: *Twelfth Annual Conference of the International Speech Communication Association*. 2011.

[21] Jian Li and Jin-Song Huang. 'Dimensions of artificial intelligence anxiety based on the integrated fear acquisition theory'. In: *Technology in Society* 63 (2020), p. 101410. issn: 0160-791X. doi: https://doi.org/10.1016/j.techsoc.2020.101410. url: https://www.sciencedirect.com/science/article/pii/S0160791X20300476.

[22] Lu Lu et al. 'Dying relu and initialization: Theory and numerical examples'. In: *arXiv preprint arXiv:1903.06733* (2019).

[23] Stéphane Mallat. 'Group invariant scattering'. In: *Communications on Pure and Applied Mathematics* 65.10 (2012), pp. 1331–1398.

[24] *Mel Frequency Cepstral Coefficient (MFCC) tutorial*. url: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/.

[25] Shervin Minaee et al. 'Ad-Net: Audio-Visual Convolutional Neural Network for Advertisement Detection In Videos'. In: *CoRR* abs/1806.08612 (2018). arXiv: 1806.08612. url: http://arxiv.org/abs/1806.08612.

[26] T. M. Mitchell. 'Does Machine Learning Really Work?' In: *AI Magazine* 18 (3) (1997), pp. 11–20.

[27] Abdel-rahman Mohamed, George Dahl, Geoffrey Hinton et al. 'Deep belief networks for phone recognition'. In: *Nips workshop on deep learning for speech recognition and related applications*. Vol. 1. 9. 2009, p. 39.

[28] Martin Musiol. *Speeding up Deep Learning Computational Aspects of Machine Learning*. Jan. 2016.

[29] Simukayi Mutasa, Shawn Sun and Richard Ha. 'Understanding artificial intelligence based radiology studies: What is overfitting?' In: *Clinical Imaging* 65 (2020), pp. 96–99. issn: 0899-7071. doi: https://doi.org/10.1016/j.clinimag.2020.04.025. url: https://www.sciencedirect.com/science/article/pii/S0899707120301376.

[30] Alex Najibi. 'Racial Discrimination in Face Recognition Technology'. In: (). url: https://sitn.hms.harvard.edu/flash/2020/racial-discrimination-in-face-recognition-technology.

[31] Michael Nielsen. *Neural Networks and Deep Learning*. 2019. Chap. 2.

[32] Lara O'Reilly. *This ad blocking company has the potential to tear a hole right through the mobile web - and it has the support of carriers*. May 2015. url: https://www.businessinsider.com/israeli-ad-blocker-shine-could-threaten-mobile-advertising-2015-5?r=US&IR=T.

[33] *Papers with code - max pooling explained*. Feb. 2018. url: https://computersciencewiki.org/index.php/File:MaxpoolSample2.png#file.

[34] Paul Pedersen. 'The Mel Scale'. In: *Journal of Music Theory* 9.2 (1965), pp. 295–308. issn: 00222909. url: http://www.jstor.org/stable/843164 (visited on 4th May 2022).

[35] Hendrik Purwins et al. 'Deep learning for audio signal processing'. In: *IEEE Journal of Selected Topics in Signal Processing* 13.2 (2019), pp. 206–219.

[36] Leland Roberts. *Understanding the mel spectrogram*. Mar. 2020. url: https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53.

[37] Kara Rogers. *Scientific method*. Sept. 2018. url: https://www.britannica.com/science/scientific-method.

[38] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2016. doi: 10.48550/ARXIV.1609.04747. url: https://arxiv.org/abs/1609.04747.

[39] R Vijaya Saraswathi et al. 'Voice Based Emotion Detection using Deep Neural Networks'. In: *2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*. 2021, pp. 1–6. doi: 10.1109/SMARTGENCON51891.2021.9645905.

[40] Dominik Scherer, Andreas Müller and Sven Behnke. 'Evaluation of pooling operations in convolutional architectures for object recognition'. In: *International conference on artificial neural networks*. Springer. 2010, pp. 92–101.

[41] Martyn Shuttleworth and Lyndsay T Wilson. *Type I error and type II error*. url: https://explorable.com/type-i-error.

[42] Savitha Srinivasan, Dragutin Petkovic and Dulce Ponceleon. 'Towards Robust Features for Classifying Audio in the CueVideo System'. In: *Proceedings of the Seventh ACM International Conference on Multimedia (Part 1)*. MULTIMEDIA '99. Orlando, Florida, USA: Association for Computing Machinery, 1999, pp. 393–400. isbn: 1581131518. doi: 10.1145/319463.319658. url: https://doi.org/10.1145/319463.319658.

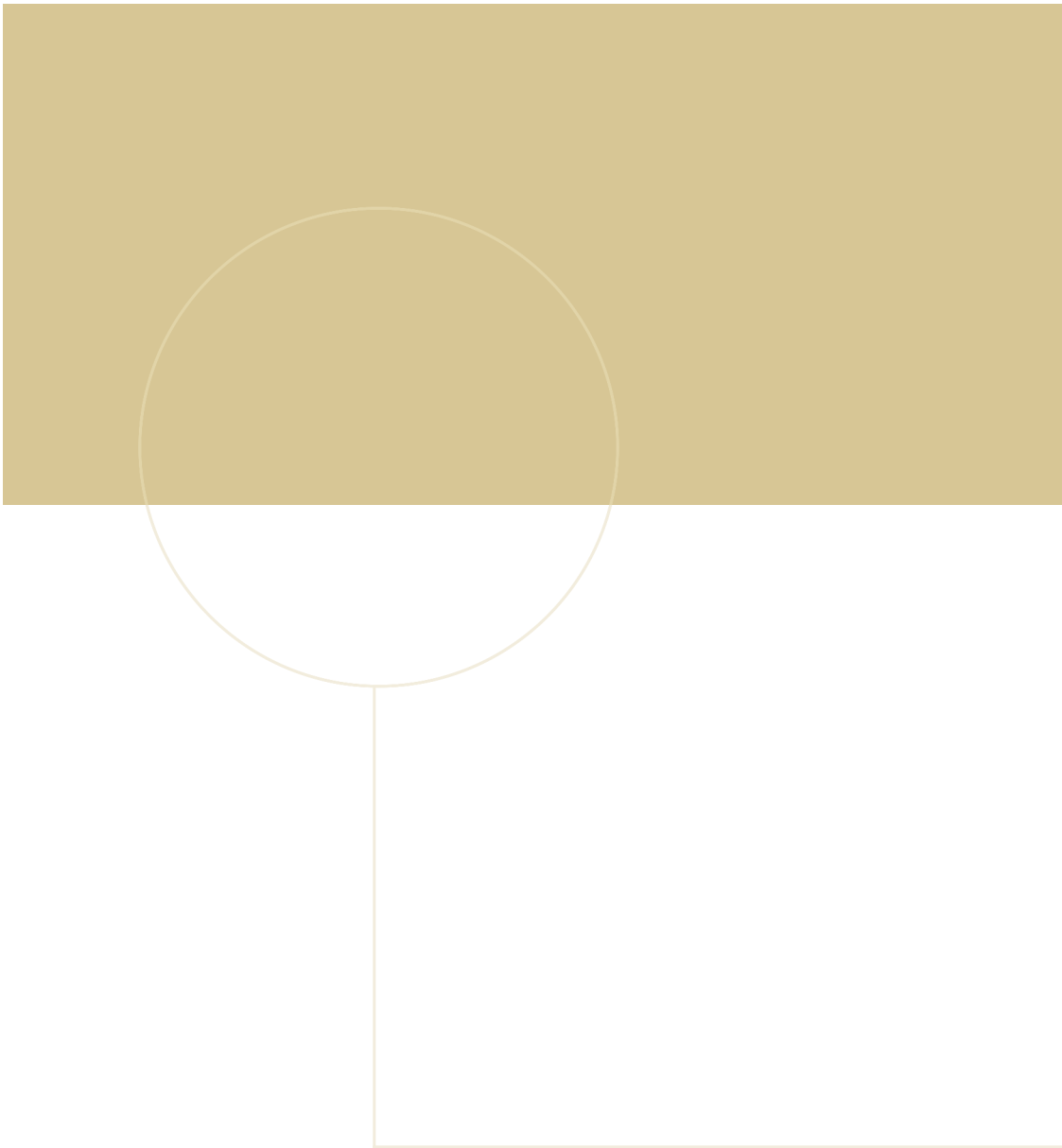
[43] Stanley Smith Stevens, John Volkmann and Edwin Broomell Newman. 'A scale for the measurement of the psychological magnitude pitch'. In: *The journal of the acoustical society of america* 8.3 (1937), pp. 185–190.

[44] Suraj Tripathi et al. 'Learning Discriminative Features using Center Loss and Reconstruction as Regularizer for Speech Emotion Recognition'. In: *Proceedings of IJCAI 2019 3rd Workshop on Artificial Intelligence in Affective Computing*. Ed. by William Hsu. Vol. 122. Proceedings of Machine Learning Research. PMLR, Oct. 2020, pp. 44–53. url: https://proceedings.mlr.press/v122/tripathi20a.html.

[45] Sun-Chong Wang. 'Artificial Neural Network'. In: *Interdisciplinary Computing in Java Programming*. Springer, Boston, MA, 2003, pp. 81–100.

[46] Dong Yu and Li Deng. *Automatic speech recognition*. Vol. 1. Springer, 2016.

[47] Fang Zheng, Guoliang Zhang and Zhanjiang Song. 'Comparison of different implementations of MFCC'. In: *Journal of Computer science and Technology* 16.6 (2001), pp. 582–589.



NTNU

Norwegian University of
Science and Technology