Tampere University

Tapio Honka

ONE-SHOT LEARNING WITH SIAMESE NETWORKS FOR ENVIRONMENTAL AUDIO

Faculty of Information Technology and Communication Sciences
Bachelor's thesis
May 2019

ABSTRACT

Tapio Honka: One-shot Learning with Siamese Networks for Environmental Audio
Bachelor's thesis
Tampere University
Bachelor's Degree Programme in Electrical Engineering
Examiner: Aleksandr Diment
May 2019

In the recent years deep learning based approaches have dominated different types of classification problems. Usually these approaches require large amounts of training data to train a model capable of generalizing to any unseen data of the same type. However, in some applications it might be difficult to gather training data efficiently and it would be beneficial to classify new samples using only a few or even a single training example.

For us humans the knowledge from previously learned concepts is relatively easy to transfer to unfamiliar concepts, therefore many researchers have experimented with this idea in machine learning classification tasks. The idea of only using a single labelled example to classify unseen data is known as *one-shot learning* and has been successful especially in the field of computer vision. Many of the modern approaches for one-shot learning utilize a special neural network architecture named *siamese network*. This architecture can be trained to predict similarities between inputs, and can be used for a metric-based approach to one-shot learning. Siamese networks have been used for different audio related tasks before, however their usage in one-shot learning for audio classification has received less attention compared to computer vision.

The purpose of this thesis is to extend the idea of *one-shot learning* to environmental audio classification and see if this approach is feasible. The proposed system was trained and evaluated on the ESC dataset, consisting of 50 different environmental audio categories. The final one-shot evaluation was done to 5 completely unseen classes, using only a single example of each class when performing the classification. The results show that convolutional siamese networks are indeed a valid approach to the difficult one-shot classification task for environmental audio.

Keywords: one-shot learning, classification, siamese neural network, audio

PREFACE

This work is the result of a bachelor's thesis idea proposed by the audio research group at Tampere University (previously known as Tampere University of Technology). I would like to thank Aleksandr Diment for the feedback and general advice on how this thesis should be approached and also the whole staff of the former laboratory of Signal Processing for their professional teaching and mentoring.

Tampere, 4th May 2019

Tapio Honka

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

CNN	Convolutional Neural Network
CPU	Central processing unit
EI	Estimated improvement
ESC	Dataset for Environmental Sound Classification
FC	Fully Connected
FFT	Fast fourier transform
FNN	Feedforward neural network
GPU	Graphics processing unit
ms	millisecond
MSE	Mean squared error
ReLU	Rectified Linear Unit
SMBO	Sequential model-based optimization
TPE	Tree-structured Parzen Estimator

1 INTRODUCTION

As humans we are able to recognize different environmental audio events (everyday environmental sounds which are neither music or speech) relatively easily using our auditory senses. However, a computer has to rely on complicated algorithms to perform this seemingly easy task. This is traditionally done by extracting and processing useful features from an audio recording followed by a classification procedure. Manually choosing which features to extract and which classifier to use can be very dependent on the type of the audio events, thus complicating the approach. This can be avoided to a certain degree with the use of deep learning.

Deep learning approaches to environmental audio scene classification is a relative new concept. The use of deep learning in computer vision classification tasks has been widely studied in the last decades, specifically with convolutional neural network based architectures [1][2][3]. However, in audio classification deep learning is a relatively new and still constantly developing field of research [4][5][6]. Publicly available datasets, such as ESC [7] and Google's AudioSet [8], are being released to encourage research in multiple audio classification domains.

Traditional deep learning based classification requires large amounts of training data to create a model capable of generalizing to similar unseen data. This is due to the diversity and possible redundant information in the given data. However, in some cases there is not enough training data available and an alternative approach is required. The idea of *one-shot learning* is to be able to classify an unseen sample by using only a single labelled example. This can be done by first learning a general similarity function using a training dataset, which can then be extended to measure the similarity between unseen target samples and comparison samples. The dataset used for classifying completely new samples is not required to be as extensive as in traditional deep learning classifiers, which can be thought as one of the main benefits of one-shot learning.

One-shot learning can be performed with deep learning approaches, one being by utilizing a siamese neural network [9][10][11]. A siamese network architecture can be used for calculating a metric between two input samples. This means that the model has two input networks acting as encoders which are then combined and fed to an output network to produce a certain metric. This metric-based approach to one-shot learning has been proven to work e.g. for image recognition [9] and speaker identification [10], indicating that this method could also be used for environmental audio classification.

The purpose of this thesis is to explore the idea of using one-shot learning with siamese networks for environmental sound event classification. The goal is to also investigate if this method is a viable approach for audio classification tasks with small amount of data available. The approach follows previous research done in one-shot learning and siamese networks and aims to extend this to environmental sound event classification.

The remainder of this thesis is arranged as follows. Publications and research related to one-shot learning and siamese neural networks are discussed in Chapter 2. Chapter 3 explains the proposed approach, including the model architecture and its background, training procedure and the execution of one-shot classification. The experiments are introduced in Chapter 4, which covers the dataset used, preprocessing of this data, model optimization and training, and the final evaluation. Finally, Chapter 5 presents the summary of this thesis and discussion regarding possible related future work.

2 RELATED WORK

In the field of computer vision one-shot learning with siamese neural networks has been researched e.g. in handwritten character recognition [9] and object category classification [12]. However, in audio event classification, where the goal is to recognize the labels of environmental sounds, this concept has publicly received only small amount of research attention. This is most likely due to deep learning approaches being relatively new in the context of audio, only publicly appearing in the last 10 years [4][5][6].

The idea of one-shot learning originated from a publication by Li Fei-Fei et al. [13] where a Bayesian approach was used to perform one-shot learning for image classification. The authors aimed to create a Bayesian model that is able to generalize to new object classes with very few examples, possibly just one, thus naming their original algorithm *Bayesian One-Shot algorithm*. This was followed by another publication by Li Fei-Fei et al. further studying this concept [12]. One-shot learning was later also studied by Lake et al. [14] in the domain of handwritten characters where they approached one-shot learning in the view of cognitive science. Their idea was that by using knowledge from previously seen characters their generative model would be able to deduce the hidden strokes in unseen characters. They have since extended their research of concept learning and one-shot classification to multiple publications and released the Omniglot dataset as a benchmark dataset for image one-shot classification [14][15][16].

Koch et al. [9] explored the idea of using deep convolutional siamese networks for one-shot learning image classification tasks. The authors trained a siamese network model on the Omniglot handwritten character dataset to rank similarities between different input pairs, which was then used to generalize to entirely unseen classes. This metric-based approach was able to learn generic image features capable of generalizing to one-shot classification, outperforming all baselines available at the time. They also suggested to extend this approach to one-shot learning tasks in other domains [9]. Vinyals et al. [17] from Google DeepMind further developed this idea by deploying a more complex network architecture and a training procedure mimicking the test conditions. Their idea was to have a deep learning model learn an end-to-end nearest neighbour classifier by training on one-shot classification tasks rather than learning a similarity function between inputs. They were able to significantly improve one-shot classification on the Omniglot dataset compared to other competing approaches [17]. Furthermore, one-shot learning with convolutional siamese networks was also studied in the context of relation extraction between entities [11]. The authors considered this fine-grained relation extraction as a one-shot

classification problem where the goal was to predict uncommon relations between samples by only using one or a few examples. They were able to produce promising results and demonstrated the possible benefits to domain-specific information extraction [11]. Recently, Dong et al. used a quadruplet siamese deep neural network in their 2019 publication [18] for one-shot learning for visual object tracking. Their proposed network architecture differed from previous work by using a total of four input instances for one-shot learning. The authors used the combination of two different loss functions, triple and pair loss, where the triplet loss was calculated with respect to three instances and the pair loss with respect to a single pair of instances. The combination weights between these losses was selected using an additional weight layer to automatize the weight adjusting process.

One-shot learning with siamese networks has not been studied as widely in the context of audio as in computer vision. In a 2018 publication [10] Vélez et al. proposed a siamese convolutional neural network architecture for one-shot speaker identification. Their goal was to train a model to learn if two audio clips are from the same speaker even for completely new speakers. This could be then utilized as part of a service robot which constantly encounters new users. The proposed system consisted of three different parts: a generic verifier, an external database for entries of known users and a speaker selection system, where the verifier was used to inspect if two audio signals are from the same speaker. This generic verifier system was implemented as a convolutional siamese network where the input audio signals were represented as time-frequency spectra. The authors also tested several different architectures for the siamese network and reported the three best performing models. These three models were based on the well known VGG [19] and ResNet 50 [20] architectures and were able to reach an average accuracy of 81.2% in real-life environment evaluation tasks, concluding that one-shot learning with siamese networks can be extended to real-life audio classification tasks. The problem of multimodal one-shot learning was studied in a publication by Eloff et al. [21] where the authors used pairs of spoken and visual digits to investigate if a siamese model would be able to perform one-shot classification to this multimodal input data. They were able to achieve superior accuracy results compared to a nearest neighbour classifier for image pixels and dynamic time warping over speech.

Besides one-shot learning, siamese networks have been also used for other audio-related problems. Manocha et al. [22] proposed a new approach to content-based retrieval for audio with the use of siamese networks. The authors' goal was to use a siamese model to obtain encoded vector representations of audio samples, where semantically similar audio can be then retrieved easily using this vector representation. The vector formation can be thought as audio fingerprinting where semantic information is stored into a easily comparable numerical form. The results concluded by the authors showed that their approach was able to capture semantic similarities between audio samples from the same class and could be used for different audio retrieval tasks. In a recent 2019 publication by Zhang et. al [23] the authors utilized siamese convolutional neural networks in the context of sound search. Their approach was to train a siamese model capable of extract-

ing features and estimating similarity between vocal imitations and original sounds. The authors proposed two systems for this problem, symmetric IMINET and asymmetric TL-IMINET. The two encoders in IMINET followed the idea of having two identical networks which are trained from scratch, whereas TL-IMINET used pretrained encoding networks which were trained by transfer learning from spoken language recognition and environmental sound classification tasks. Their results showed that both versions were able to beat a state-of-the-art system and that using transfer learning for a siamese network noticeably improves the audio retrieval performance [23].

3 METHODOLOGY

The approach to one-shot learning for environmental audio proposed in this thesis is divided into three conceptual components:

- Proposed model
- Model training procedure
- One-shot classification process

The proposed model and it's background is introduced in Section 3.1, the training procedure and the formation of training pairs is discussed in Section 3.2, and finally the execution of one-shot classification is explained in Section 3.3.

3.1 Model

A siamese neural network composed of convolutional layers was used to perform one-shot learning for environmental audio classification. This approach followed the basic idea of using matching convolutional neural networks for one-shot learning presented by Koch et al. [9]. The principles of convolutional neural networks and siamese networks followed by the proposed model are discussed in the next three subsections respectively.

3.1.1 Convolutional neural networks

A convolutional neural network (CNN) refers to a specific kind of neural network which utilizes convolutional layers in addition to a conventional feedforward neural network (FNN) consisting of input, hidden and output units. The basic CNN architecture was first proposed by Fukushima in the 1980 published paper *Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position* [24], where convolutional and downsampling layers were first introduced. A feedforward network is neural network architecture where the information moves in one direction and the connections between units do not form any cycles of loops [25]. An example of a simple FNN is depicted in Figure 3.1.

A convolutional layer uses a mathematical operation called *convolution*, which is defined for a two-dimensional input I and a two-dimensional kernel K as [26]

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n), \tag{3.1}$$

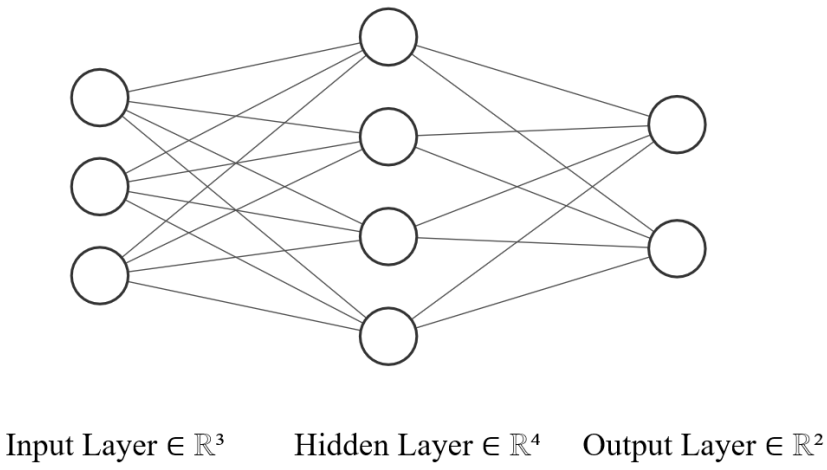


Figure 3.1. *An example of a simple feedforward neural network.*

where i and j represent the two-dimensional indices and the convolution operation is denoted with an asterisk. The kernel K is sometimes referred to as a *filter* and the output S as a *feature map*. The latter comes from the idea that each convolutional layer extracts certain features from its input and outputs these as a feature map. The convolution operation is illustrated in Figure 3.2

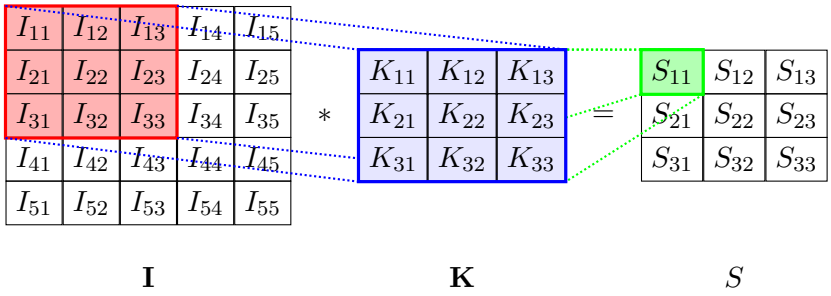


Figure 3.2. *Convolution operation.*

Convolutional neural networks are appealing in many deep learning applications due to their filtering nature, where interesting pixel patterns are extracted from input features. After the first introduction of automated gradient-based learning by LeCun et. al in their 1998 publication [1], automated weight optimization for neural networks has been implemented in most deep learning libraries. Additionally, after the first GPU implementation of a CNN by Chellapilla et al. [2], multiple libraries for GPU supported deep learning have been published, enabling larger networks and faster training compared to CPU implementations.

3.1.2 Siamese networks

A siamese network is a neural network architecture which takes two inputs instead of one and computes a metric from these inputs. The idea is to have two identical input networks acting as encoding layers, which can then be merged and fed into a single output network. Weights for both input networks have to be updated identically in order to have two identical encoding networks. This concept was first introduce by Bromley et al. in *Signature Verification using a "Siamese" Time Delay Neural Network*, where a siamese neural network was utilized for comparing an unknown signature to a known one [27]. The goal of a siamese network is to learn to extract interesting and diverse features from inputs which can be then used to form a vector representing the semantic information of each input. Encoded vectors can be then used in the output network to calculate a metric based on the encoded semantic information stored in the vectors.

Using convolutional neural networks as part of a siamese network is appealing due to their ability to extract possibly very diverse features and their lesser amount of parameters compared to a vanilla fully connected neural network. The filtering nature of convolutional layers, where a single kernel is used to all the regions of an image, also improves the translation invariance to the input data.

3.1.3 Proposed model

The model used for this thesis consists of two convolutional input networks, followed by a merging layer and a final output layer. The input networks share the same architecture and weights in order to act as identical encoding layers for both inputs. This also means that the weights are updated simultaneously for both networks during training. The proposed architecture for a single input network consists of several convolutional blocks followed by a fully connected layer. A single convolutional block consists of a convolutional layer with a given number of filters, a batch normalization layer, a rectified linear unit (ReLU) activation unit and a max-pooling layer. A batch normalization layer normalizes the outputs of the previous layer by transforming the mean of the activations close to zero and the standard deviation close to 1. ReLU activation function is defined as [26]

$$g(z) = max(0, z), \tag{3.2}$$

meaning the output is zero for all negative input values. For the fully connected layers a Sigmoid activation function was used [26]:

$$\sigma(z) = \frac{1}{1 + exp(-z)}. \tag{3.3}$$

The max-pooling layer downsamples the output of the previous layer by applying a max filter to subregions of certain size. This reduces the number of learnable parameters and helps to prevent overfitting.

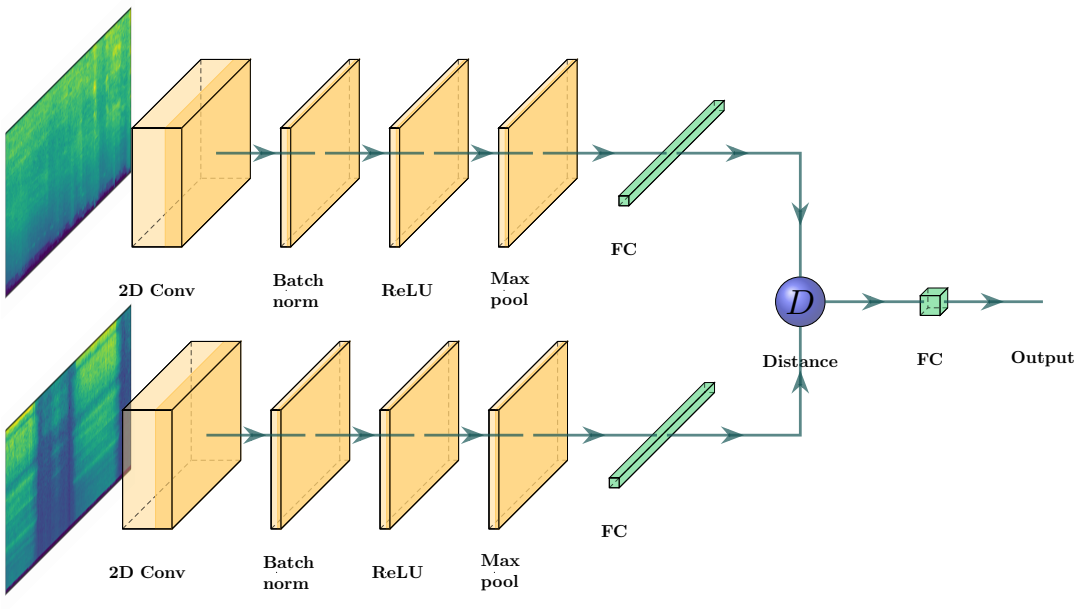


Figure 3.3. *A simple convolutional siamese network architecture for audio spectra.*

The simplest way to implement the merging layer is to calculate some element-wise distance metric between the two encoded convolutional network output feature vectors. The two metrics proposed here were the absolute difference and squared difference between the two vectors, which were both used in the parameter optimization process discussed in Section 4.3. Other metrics, such as cosine similarity, could also be used here.

The final output layer consists of a single fully connected unit with a sigmoid activation function. This acts as a simple output network for the merged difference feature vector and could be extended into a more complex network. The simplest proposed siamese network architecture is illustrated in Figure 3.3.

Due to the high number of parameters (possibly several millions) involved in a neural network, the model tends to eventually overfit to the training dataset. This means that the model performs really well on the training data but is not able to generalize to unseen test data. One way to prevent this is to use *dropout* [28] as part of the learning process. The idea of dropout is to randomly omit some amount of units during the network weight optimization, but still use all units when performing the network task. This random dropout can be implemented to both convolutional and fully connected units. The proposed model uses unique dropouts for both convolutional and fully connected layers, where the dropout amounts are chosen in the model optimization discussed in Section 4.3.

3.2 Training

Before a siamese model can be used to perform one-shot classification to unseen classes, it has to be trained on a diverse dataset to learn to extract and compare semantic informa-

tion from input data. The number of training samples is easy to keep high when forming training pairs for a siamese network with two inputs, therefore the diversity of the dataset is more important. The number of different classes used in training affects how well the final model is able to generalize to unseen samples from unseen classes.

The proposed training procedure consists of first forming input pairs used for training and their corresponding similarity labels, followed by the actual weight optimization. The similarity labels used here are either 0 or 1, where 1 indicates same class and 0 different class. The training dataset should consist of both negative and positive pairs, where a negative pair refers to samples from different classes and a positive pair to samples from the same class. Pairs are formed by taking a random sample from a dataset, pairing this with a specific number of positive and negative pairs, and finally excluding the sample to prevent duplicate pairs. This process is then iterated until no more pairs can be formed. It should be noted that the number of negative pairs can be made significantly larger than the number of positive pairs, therefore the optimal ratio of negative to positive pairs should be tested when implementing the final model.

Two possible loss functions used for weight optimization are proposed, the first one being *mean squared error* (MSE) defined as

$$L_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2. \tag{3.4}$$

This can be further simplified for a single binary output

$$L_{MSE} = (y - p)^2, \tag{3.5}$$

where p is the predicted output value and y is either 0 or 1 indicating the correct label of the input (positive or negative pair).

The second loss function, *binary cross-entropy*, is defined as [26]

$$L_{BCE} = -(y\log(p) + (1-y)\log(1-p)), \tag{3.6}$$

where again p is the predicted value and y is the correct input label.

Binary cross-entropy is commonly used for classification tasks due to its logarithmic behaviour, where the loss increases exponentially for predictions further from the ground truth. However, in the case of a siamese network the output can be interpreted as a similarity score and therefore MSE can possibly also be a valid loss function. Both of these loss functions were used as part of the hyperparameter optimization process described in Section 4.3.

For the loss function minimization process the Adam optimizer [29] is proposed. Adam is an alternative to the traditional stochastic gradient descent optimization and has been

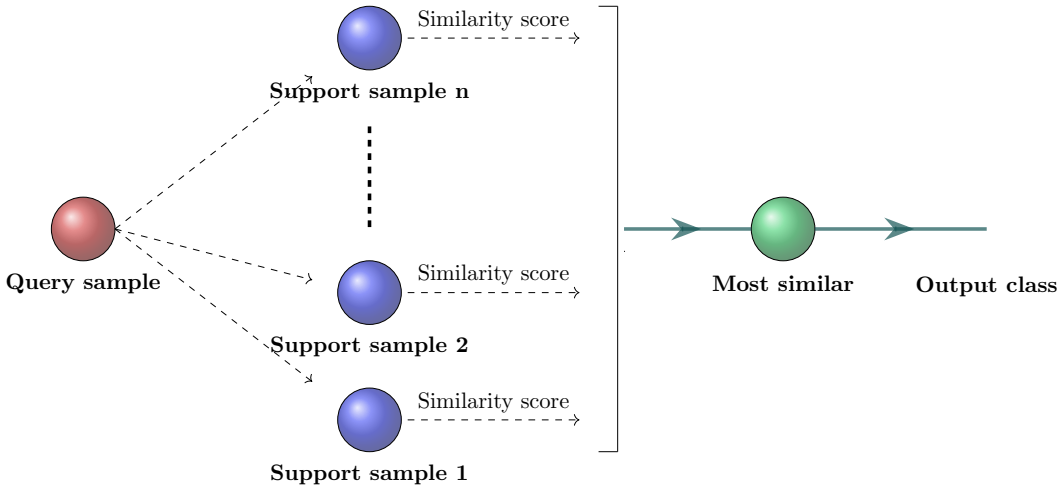


Figure 3.4. *Illustration of one-shot classification process for n classes.*

adapted in many deep learning applications. Benefits of Adam mentioned by the original authors are e.g. its computational efficiency, straightforward implementation and small memory requirements [29]. They also pointed out how the optimizer is able to solve deep learning problems more efficiently compared to other methods available at the time, which may be one reason to its popularity.

3.3 One-shot classification

The one-shot classification task can be performed by comparing a query sample to a support dataset. A support dataset used for comparison consists of samples each belonging to a different class, where one of these classes corresponds to the input samples class. Each sample from the support dataset is then paired with the query sample to produce similarity scores for the whole comparison dataset. For the final prediction the class of the support sample most similar to the query sample is chosen. The classes used for one-shot classification should be unseen by the final model to ensure the one-shot nature of the evaluation. This process is illustrated in Figure 3.4.

Mathematically this can be described as predicting the class $\hat{c}$ of a query sample $\hat{\mathbf{x}}$, where $\hat{c} \in C$ (category space) and $\hat{\mathbf{x}} \in A$ (audio spectrogram space). The discrete category space C for n categories is denoted as $C = \{c^{(i)}\}_{i=1}^{n}$. The corresponding support set for categories C can be denoted as $S = \{\mathbf{x}_s^{(i)}\}_{i=1}^{n}$, where each support audio spectrogram $\mathbf{x}_s^{(i)} \in A$ represents a single example of each class $c^{(i)}$ from C. The test sample $\hat{\mathbf{x}}$ is then matched to the support set S using a trained siamese model to produce similarity predictions for each class example $\mathbf{x}_s^{(i)}$. Each prediction can be interpreted as the probability that the given test sample $\hat{\mathbf{x}}$ belongs to the class $c^{(i)}$ of $\mathbf{x}_s^{(i)}$ or formally $P(c^{(i)}|\hat{\mathbf{x}})$. The best prediction for the unknown class $\hat{c}$ can be thus found by finding the

class from C with the highest probability given the sample $\hat{\mathbf{x}}$:

$$\hat{c} = \underset{C}{\operatorname{argmax}} P(C|\hat{\mathbf{x}}) \tag{3.7}$$

This can be easily implemented by first storing the class probabilities calculated from the support set into a vector and then finding the index which corresponds to the maximum probability. The order in which each class probability is stored is required to be known in order find the class corresponding the highest probability. Most deep learning libraries are able to take batches of inputs and export the results in a single matrix or vector. Therefore, the class probabilities can be efficiently calculated by forming the query and support set sample pairs and stacking these into an input batch. This can then be fed into the network to produce an output vector containing each class probability.

4 EXPERIMENTS

The implementation was done using the *Keras* library [30]. Keras is a neural network library for python which uses a lower level deep learning library, e.g. *Tensorflow* [31], as it's backend. It was developed to act as as a easy-to-use deep learning library enabling fast neural network experimentation.

The implemented model was trained and evaluated using the ESC-50 dataset [7] described in Section 4.1. followed by how the data was pre-processed in Section 4.2. Model optimization and training are discussed in Sections 4.3 and 4.4 respectively. Finally, the final evaluation and one-shot classification results are presented in Section 4.5

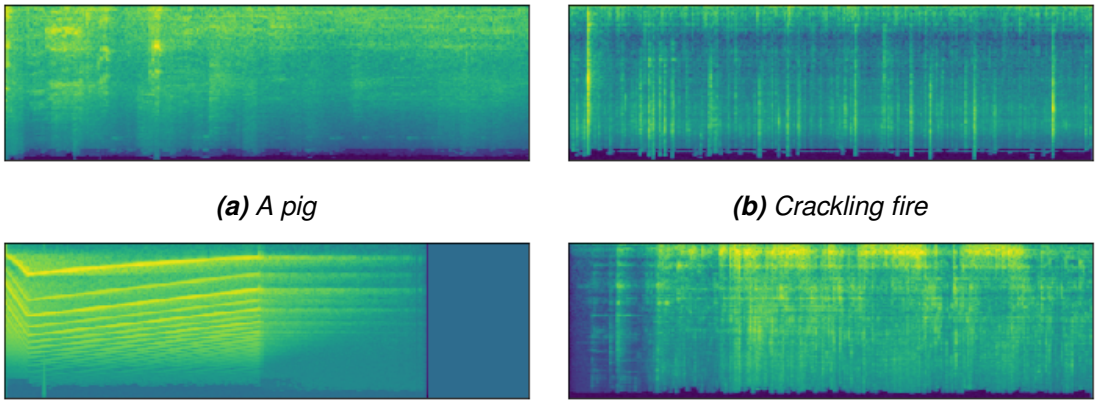
4.1 ESC Dataset

The ESC dataset is a publicly available collection of 2000 annotated 5 second audio clips divided into 50 various sound events. These sound events can be further grouped into 5 bigger categories each containing 10 classes. The dataset was created to generate more attention to research in environmental audio classification and to act as a publicly available dataset for baseline comparisons. The original publication by Piczak also provided an estimation of human accuracy and baseline approaches for environmental audio classification [7]. All major groups and their classes are presented in Table 4.1.

Table 4.1. *ESC classes*

Animals	Natural soundscapes & water sounds	Human, non-speech sounds	Interior/domestic sounds	Exterior/urban noises
Dog	Rain	Crying baby	Door knock	Helicopter
Rooster	See waves	Sneezing	Mouse click	Chainsaw
Pig	Crackling fire	Clapping	Keyboard typing	Siren
Cow	Crickets	Breathing	Door, wood creaks	Car horn
Frog	Chirping birds	Coughing	Can opening	Engine
Cat	Water drops	Footsteps	Washing machine	Train
Hen	Wind	Laughing	Vacuum cleaner	Church bells
Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
Sheep	Toilet flush	Snoring	Clock tick	Fireworks
Crow	Thunderstorm	Drinking, sipping	Glass breaking	Hand saw

This dataset has two variants used for supervised learning, ESC-50 and ESC-10, of which the former includes all of the 50 sound events and the latter includes 10 classes from three general groups. Moreover, the dataset includes an additional unlabelled dataset used for unsupervised learning. [7] For the purpose of this thesis the full 50 class dataset was used due to its higher number of different classes. A large and diverse training category



(a) *A pig* ***(b)*** *Crackling fire*

(c) *A siren* ***(d)*** *Toilet flush*

Figure 4.1. *Examples of audio spectra.*

space is beneficial when training a siamese model for this way the model is more likely to learn to extract generally useful features and generalize to new unseen classes.

4.2 Data preparation

For each sound clip with a sampling rate of 44.1 kHz, a logarithm-scaled mel-frequency spectrogram with FFT window size of 2048, hop length of 512 samples (11.6 ms) and 128 mel-bands was calculated in order to use a CNN approach for the dataset. The resulting size of each calculated mel-log audio spectrogram was 128×431. This spectrogram representation has proven to produce reasonable results when using a CNN based model for audio classification [6]. Examples of the spectrogram representation are shown in Figure 4.1.

A siamese network requires two inputs in order to produce and an output metric representing the distance between the inputs. These input pairs were formed randomly while simultaneously ensuring that no duplicate pairs are used. The pairs consisted of positive and negative pairs, where a positive pair refers to two samples of the same class and a negative pair to two samples of different classes. Different negative to positive training pair ratios were used as part of the model optimization process in Section 4.3 to study the behaviour of the model.

4.3 Model optimization

Choosing the architecture and parameters for a model is somewhat a tedious task to perform by hand. This process can be automated using a technique called *sequential model-based optimization* (SMBO) [32], which can be generally applied to instances where a minimized function is too costly to be evaluated. Model-based optimization is able to reduce the number of evaluations compared to traditional optimization strategies, thus suiting well for deep learning hyperparameter optimization. Generally, SMBO is

implemented by iteratively fitting and evaluating a model with different configurations of parameters and then using each observed model instance to decide which parameter configurations should be investigated. The main benefit of this technique is its ability to interpolate model performance between already observed configurations and the possibility to extrapolate model performance to unseen parameter configurations [32].

There are many methods for implementing SMBO, one being *Tree-structured Parzen Estimator* (TPE) algorithm [33]. The goal of TPE is to suggest a parameter configuration for the next iteration based on the search history of previous iterations. The algorithm considers each parameter in the parameter space separately, i.e. the possible correlation between multiple parameters is ignored. For a parameter x, TPE defines two densities, $l(x)$ and $g(x)$, based on the previous k observations $\{x^{(1)}, ..., x^{(k)}\}$. These density functions are formed by splitting the observations based on performance (loss) y and threshold y^* for each observation and using this to define the two densities. TPE models $p(x|y)$ using the two densities [33]:

$$p(x|y) = \begin{cases} l(x), & \text{if } y < y^* \\ g(x), & \text{if } y > y^* \end{cases} \tag{4.1}$$

TPE then uses $p(y)p(x|y)$ as the parametrization of $p(x, y)$ to optimize expected improvement (EI). The original publication shows that by using this parametrization, EI is maximized with high probability points from $l(x)$ and low probability points from $g(x)$ [33]. The tree structure of the algorithm enables easy evaluation of several parameter candidates from $l(x)$ by observing $\frac{g(x)}{l(x)}$ for each candidate. The parameter value yielding the greatest EI is returned on each algorithm iteration.

Hyperopt [34] is a python implementation of SMBO search with TPE algorithm for general model parameter optimization or other awkward search spaces. Hyperopt finds optimal parameter configurations from a parameter space based on a given objective function. The parameter space may include different model attributes, e.g. the number of convolutional layers or the learning rate of the optimizer. For choosing the parameters during the optimization, Hyperopt uses different types of distributions for each parameter and alters these distributions based on the previous search history using TPE.

The optimization process for this thesis was done by first choosing which model attributes should be used as optimized hyperparameters. These hyperparameters were then used to form a configuration space to which Hyperopt is used to search over. Each hyperparameter was given an unique value distribution which defines if the values are chosen discretely from a predefined list using TPE or from a certain continuous distribution. The search spaces consisted of discrete choices of values and logarithmic uniform distributions. Values drawn from a logarithmic uniform distribution are drawn uniformly in the logarithmic domain. Log-uniform distribution is good for modelling learning rate and decay of an optimizer for it varies across several orders of magnitude. The objective function was defined as the mean one-shot classification accuracy on the validation set subtracted

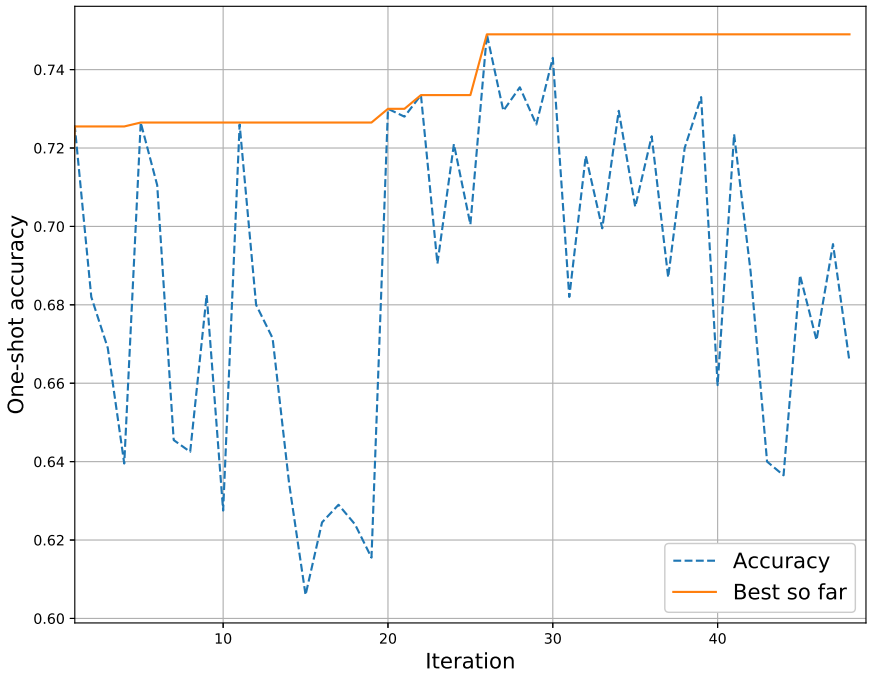


Figure 4.2. *Hyperparameter optimization process.*

from one, as Hyperopt is designed to minimize the given objective function. This way the hyperparameters are optimized based on the one-shot classification task rather than the training task.

The proposed model was optimized using 40 classes from the ESC-50 dataset and validated using other 5 classes for one-shot classification. The remaining 5 classes were left out for the final evaluation. The optimization procedure was iterated 50 times using Hyperopt [34] to find the best suitable parameters for the one-shot classification task. The number of training epochs for each instance was kept at 12 as previous experiments on the validation data showed that the general accuracy of the model could be seen after only 12 epochs. Before each training instance, the random seeds used by different libraries were set to a known value. This was to ensure that the optimization was minimally effected by randomness. The hyperparameter optimization process is visualized in Figure 4.2 where one-shot classification accuracy on the validation set for each model instance and the best accuracy so far is displayed. The figure shows how SMBO search with the TPE algorithm is able to find better hyperparameters which improve the one-shot accuracy on the validation set.

4.4 Training

After the optimization process, the hyperparameters which yielded the best one-shot classification accuracy were chosen. These hyperparameters are shown in Table 4.2. The

Table 4.2. *Hyperparameter configuration space.*

Hyperparameter	Search space	Optimization result
Number of convolutional layers	$\{1, 2, 3, 4\}$	3
Number of filters per layer	$\{16, 32, 64\}$	64
Number of fully connected units	$\{125, 250, 500\}$	500
Distance metric	{absolute difference, squared difference}	squared difference
Loss function	{MSE, binary crossentropy}	MSE
Learning rate	Log-uniform in range $[10^{-4}, 10^{-2}]$	$6.11 * 10^{-3}$
Decay	Log-uniform in range $[10^{-7}, 10^{-5}]$	$1.59 * 10^{-6}$
Dropout in convolutional layers	$\{0.0, 0.25, 0.5\}$	0.25
Dropout in fully connected layers	$\{0.0, 0.25, 0.5\}$	0.25
Negative-to-positive pair ratio	$\{1, 2\}$	1

best hyperparameters were then used to fit a model by using the same 40 training and 5 validation classes as in the optimization process. The validation classes were used after each training epoch to monitor how well the current model performs on the one-shot task rather than simply monitoring the optimized loss function. The maximum number of epochs was set to 40 while using early stopping to prevent unnecessary training epochs. Early stopping was implemented by defining a patience threshold of 10, which indicates how many epochs without one-shot accuracy improvement the training should continue before stopping. The model instance which performed best on the validation dataset was chosen to perform the final one-shot classification test on the last unseen 5 classes. This was implemented by saving the model weights each time a new best validation accuracy was achieved, and finally loading the best saved weights after the training. The two separate one-shot datasets for validation and final evaluation was done to ensure that the hyperparameter optimization and training processes would not optimize the parameters only to the validation set, but also to the unseen evaluation dataset.

4.5 Evaluation

The final evaluation was done with the remaining 5 classes which were left out in the optimization and final training process. The model trained using the optimized parameters and the same 40 and 5 classes from Section 4.4 was evaluated by performing one-shot classification to the unseen 5 classes. The final one-shot classification was performed 400 times with random unique pair combinations for each class totalling up to 2000 evaluations. This tries to ensure the generality of the results. The final results are presented in a confusion matrix in Figure 4.3, where rows indicate the ground truth and columns the model predictions.

The mean accuracy of the final one-shot classification evaluation was 79.1% which is significantly better than random guessing. This indicates that the final siamese model was able to learn to extract useful features from audio spectra which can be used to encode audio to a vector-form while preserving its semantic information. The accuracy scores for individual categories varied from 69.5% to 90.0%, concluding that the one-shot

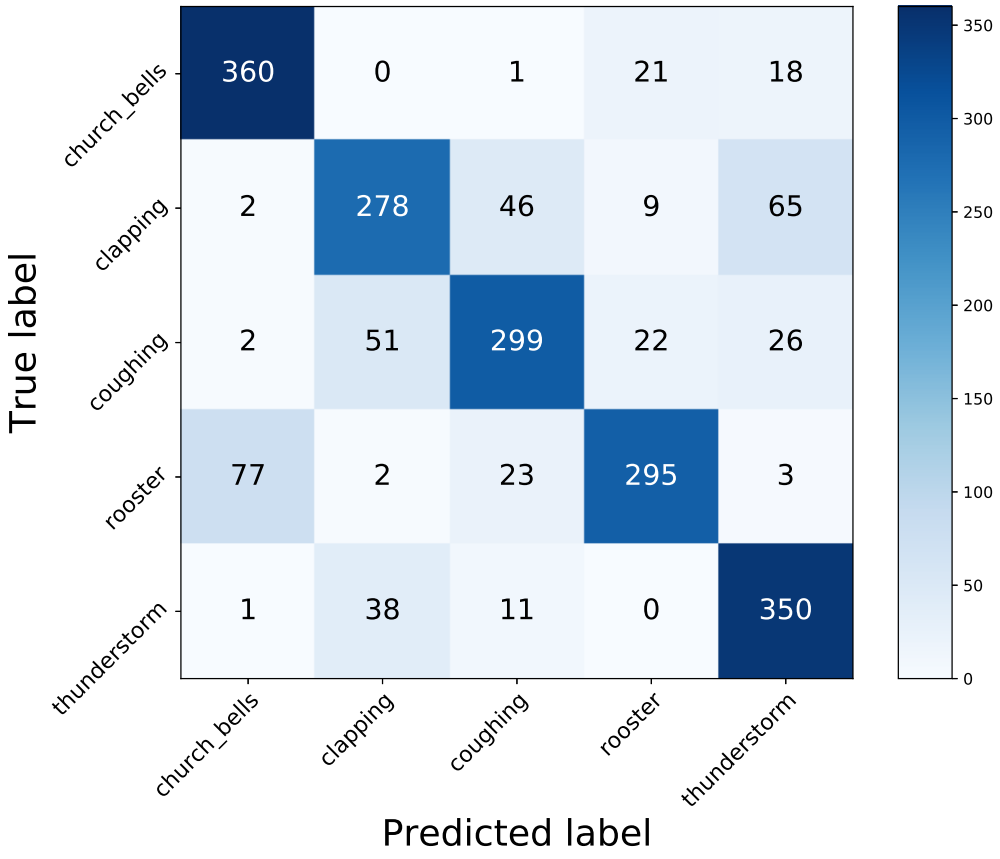


Figure 4.3. *Results presented in a confusion matrix.*

Table 4.3. *Accuracy scores for the final results.*

Category	**Accuracy**
Church bells	90.0%
Clapping	69.5%
Coughing	74.8%
Rooster	73.6%
Thunderstorm	87.5%
Mean total	*79.1%*

classification task is very dependent on the relations between the classified categories. Categories with similar features in their frequency spectra are more difficult for the model to distinguish. The accuracy scores for each category and the mean accuracy calculated from the results in Figure 4.3 are presented in Table 4.3.

The confusion matrix in Figure 4.3 shows that clapping and coughing were repeatedly mixed by the model. This may be due to both clapping and coughing consisting of single short sounds, which would also explain why especially clapping was often classified as the sound of a thunderstorm. Another interesting relation found in the results is how the sound of a rooster was multiple times wrongly classified as the sound of church bells. This could be interpreted as the result of common relations in the higher frequencies of a sound of a rooster and church bells. However, church bells are interestingly enough not classified as a rooster as often which may be due to the variability in the dataset.

5 CONCLUSION

The goal of this thesis was to study and experiment if one-shot learning with siamese convolutional networks could be utilized in environmental audio classification. The proposed approach followed previous related work where deep learning metric-based approaches were shown to produce reasonable results for both computer vision and audio related one-shot classification tasks. The proposed siamese network architecture contained two convolutional feature extracting and encoding networks and an output network capable of calculating a distance metric between the two inputs. The code for the experiments done in this thesis is available at GitHub[1].

Experiments show that the proposed approach is able to produce reasonable results when performing 2000 one-shot classifications to 5 totally unseen classes. The main difficulties in the final evaluation were related to audio clips which share common relations in their frequency spectra, indicating that the feature extraction is still lacking. However, with more distinct categories the model was able to reach up to 90% accuracy. This concludes that a convolutional siamese network is indeed a valid approach to the challenging task of one-shot classification for environmental audio.

Experiments done in this thesis are still very limited and could be extended to study the idea of one-shot classification for environmental audio with siamese networks more. The approach used should be further evaluated using cross-validation and different datasets as the results shown here are limited to a single 5-class one-shot classification evaluation. Different divisions of training and testing classes should also be experimented with to study how the proposed siamese model performs on these. Additionally, the ratio between negative and positive training pairs should be studied more, possibly even implementing some form of hard negative mining where only the negative pairs which are hard to distinguish are used as part of the training set.

In order to achieve more significant results, state-of-the-art architectures should be experimented with. As shown by Vélez et al. [10], VGG [19] and ResNet [20] based models are able to reach high accuracy in noisy real-life environments. Therefore, this idea should be also studied with environmental audio one-shot classification, possibly incorporating other famous architectures such as AlexNet [35]. Transfer learning, where the two encoding networks are pretrained on a relevant task and then transferred to the siamese network, is also a possible concept to study when striving for superior accuracy scores.

[1] https://github.com/tapioho/oneshot-learning-environmental-audio

Zhang et. al proved this to be a valid approach when implementing convolutional siamese networks for audio [23]. Transfer learning could be also coupled with the idea of using state-of-the-art architectures stated above. Finally, a more specific loss function and a better training procedure could improve the results. The training procedure should imitate the final one-shot classification task, as proposed by Vinyals et al. [17], in order to ensure that the optimized training objective is similar to the test conditions. This means that the training data would consists of query samples and support sets used for one-shot classification, instead of negative and positive similarity pairs.

REFERENCES

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In: *Proceedings of the IEEE* 86.11 (Nov. 1998), 2278–2324. ISSN: 0018-9219. DOI: 10.1109/5.726791.

[2] K. Chellapilla, S. Puri, and P. Simard. High Performance Convolutional Neural Networks for Document Processing. In: *Tenth International Workshop on Frontiers in Handwriting Recognition*. Ed. by G. Lorette. http://www.suvisoft.com. Université de Rennes 1. La Baule (France): Suvisoft, Oct. 2006. URL: https://hal.inria.fr/inria-00112631.

[3] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. June 2012, 3642–3649. DOI: 10.1109/CVPR.2012.6248110.

[4] H. Lee, P. T. Pham, Y. Largman, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In: Jan. 2009, 1096–1104.

[5] O. Gencoglu, T. Virtanen, and H. Huttunen. Recognition of acoustic events using deep neural networks. In: *2014 22nd European Signal Processing Conference (EUSIPCO)*. Sept. 2014, 506–510. URL: https://ieeexplore.ieee.org/document/6952140.

[6] K. J. Piczak. Environmental sound classification with convolutional neural networks. In: *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. Sept. 2015, 1–6. DOI: 10.1109/MLSP.2015.7324337.

[7] K. J. Piczak. ESC: Dataset for Environmental Sound Classification. In: *Proceedings of the 23rd ACM International Conference on Multimedia*. MM '15. Brisbane, Australia: ACM, 2015, 1015–1018. ISBN: 978-1-4503-3459-4. DOI: 10.1145/2733373.2806390. URL: http://doi.acm.org.libproxy.tuni.fi/10.1145/2733373.2806390.

[8] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio Set: An ontology and human-labeled dataset for audio events. In: *Proc. IEEE ICASSP 2017*. New Orleans, LA, 2017.

[9] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese Neural Networks for One-shot Image Recognition. In: 2015. URL: https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf.

[10] I. Velez, C. Rascon, and G. Fuentes-Pineda. One-Shot Speaker Identification for a Service Robot using a CNN-based Generic Verifier. In: *arXiv preprint arXiv:1809.04115* (2018). arXiv: 1809.04115. URL: https://arxiv.org/abs/1809.04115.

[11] J. Yuan, H. Guo, Z. Jin, H. Jin, X. Zhang, and J. Luo. One-shot learning for fine-grained relation extraction via convolutional siamese neural network. In: *2017 IEEE*

International Conference on Big Data (Big Data). Dec. 2017, 2194–2199. DOI: 10.1109/BigData.2017.8258168.

[12] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.4 (Apr. 2006), 594–611. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.79.

[13] L. Fei-Fei, Fergus, and Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In: *Proceedings Ninth IEEE International Conference on Computer Vision*. Oct. 2003, 1134–1141 vol.2. DOI: 10.1109/ICCV.2003.1238476.

[14] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum. One shot learning of simple visual concepts. In: *Proceedings of the 33rd Annual Conference of the Cognitive Science Society* (Jan. 2011).

[15] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Concept learning as motor program induction: A large-scale empirical study. In: (2012). URL: https://escholarship.org/uc/item/0005t82w.

[16] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. In: (2015). URL: https://science.sciencemag.org/content/350/6266/1332/.

[17] O. Vinyals, C. Blundell, T. P. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching Networks for One Shot Learning. In: *CoRR* abs/1606.04080 (2016). arXiv: 1606.04080. URL: http://arxiv.org/abs/1606.04080.

[18] X. Dong, J. Shen, D. Wu, K. Guo, X. Jin, and F. Porikli. Quadruplet Network with One-Shot Learning for Fast Visual Object Tracking. In: *IEEE Transactions on Image Processing* (2019), 1–1. ISSN: 1057-7149. DOI: 10.1109/TIP.2019.2898567.

[19] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In: (2014). arXiv: 1409.1556. URL: https://arxiv.org/abs/1409.1556.

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[21] R. Eloff, H. A. Engelbrecht, and H. Kamper. Multimodal One-Shot Learning of Speech and Images. In: *CoRR* abs/1811.03875 (2018). arXiv: 1811.03875. URL: http://arxiv.org/abs/1811.03875.

[22] P. Manocha, R. Badlani, A. Kumar, A. Shah, B. Elizalde, and B. Raj. Content-based Representations of audio using Siamese neural networks. In: *CoRR* abs/1710.10974 (2017). arXiv: 1710.10974. URL: http://arxiv.org/abs/1710.10974.

[23] Y. Zhang, B. Pardo, and Z. Duan. Siamese Style Convolutional Neural Networks for Sound Search by Vocal Imitation. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.2 (Feb. 2019), 429–441. ISSN: 2329-9290. DOI: 10.1109/TASLP.2018.2868428.

[24] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. In: *Biological cybernetics* 36 (Feb. 1980), 193–202. DOI: 10.1007/BF00344251.

[25] J. Schmidhuber. Deep learning in neural networks: An overview. In: *Neural Networks* 61 (2015), 85–117. ISSN: 0893-6080. URL: http://www.sciencedirect.com/science/article/pii/S0893608014002135.

[26] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[27] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. Lecun, C. Moore, E. Sackinger, and R. Shah. Signature Verification using a "Siamese" Time Delay Neural Network. In: *International Journal of Pattern Recognition and Artificial Intelligence* 7 (Aug. 1993), 25. DOI: 10.1142/S0218001493000339.

[28] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. In: *CoRR* abs/1207.0580 (2012). arXiv: 1207.0580. URL: http://arxiv.org/abs/1207.0580.

[29] D. P. Kingma and J. L. B. and. Adam: A Method for Stochastic Optimization. In: (2015). arXiv: 1412.6980. URL: https://arxiv.org/abs/1412.6980.

[30] F. Chollet et al. *Keras*. https://keras.io. 2015.

[31] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[32] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential Model-Based Optimization for General Algorithm Configuration. In: *Learning and Intelligent Optimization*. Ed. by C. A. C. Coello. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, 507–523. ISBN: 978-3-642-25566-3.

[33] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for Hyper-parameter Optimization. In: *Proceedings of the 24th International Conference on Neural Information Processing Systems*. NIPS'11. Granada, Spain: Curran Associates Inc., 2011, 2546–2554. ISBN: 978-1-61839-599-3. URL: http://dl.acm.org/citation.cfm?id=2986459.2986743.

[34] J. Bergstra, D. Yamins, and D. D. Cox. *Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms*. 2013. URL: http://www.coxlab.org/pdfs/2013_bergstra_hyperopt.pdf.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In: *Commun. ACM* 60.6 (May 2017), 84–90. ISSN:

0001-0782. DOI: 10.1145/3065386. URL: http://doi.acm.org.libproxy.tuni.fi/10.1145/3065386.