Computer Science Department
Quantative Data Analytics Research Group

Semi-Supervised End-To-End Learning on Raw Waveform Time-Series

Master Thesis

Joep van Genderingen

Supervisors:
David W. Romero
Dr. Jakub Tomczak

Amsterdam, September 2021

Semi-Supervised End-To-End Learning on Raw Waveform Time-Series

Vrije Universiteit Amsterdam
Computer Science Department
Quantative Data Analytics Research Group

Joep van Genderingen

Supervisors:
David W. Romero
Dr. Jakub Tomczak

Vrije Universiteit Amsterdam
Faculty of Science
De Boelelaan 1081a
1081 HV Amsterdam

Abstract

Deep neural networks have shown that they can achieve human-level performance in many speech recognition and sound classification tasks. Achieving this performance, however, requires a large amount of labeled data. Obtaining a large labeled dataset is time-consuming and costly, especially when labeling must be done by an expert. This work is aimed at exploring semi-supervised learning techniques for raw audio waveforms. Previous works on semi-supervised learning for audio rely on features derived from audio spectograms to obtain good performance. However, finding the right representation can be challenging and time-consuming, and the often heuristically designed features might not be optimal for the predictive task. Deep neural networks have demonstrated that they can learn similar feature representations from raw waveforms. In this work, we propose to adopt SelfMatch, a semi-supervised algorithm that combines the power of contrastive self-supervised learning and augmentation consistency regularization, in combination with Continuous Kernel Convolutional Networks (CKCNNs). CKCNNs solve important limitations present in conventional architectures for sequential data by formulating convolutional kernels in CNNs as continuous functions. We expand on SelfMatch by introducing various data augmentations suitable for raw audio waveforms. There currently is no implementation for SelfMatch for audio data, so we implement Temporal Convolutional Networks (TCNs) as a baseline. In experiments with varying percentages of labeled data, we show that training networks using SelfMatch significantly improves performance over supervised training with the same amount of labels. However, this improved performance is significantly lower than fully supervised performance. We hypothesise that the biggest factor in this performance gap is the combination and ordering of augmentations. Other studies on contrastive learning show that the combination and ordering of augmentations have a massive impact on the quality of learned representations and classification performance. Another plausible explanation is catastrophic forgetting, a phenomenon where a network forgets parameters from the old task in learning a new task. Contrastive learning benefits from larger batch sizes, but due to the large computational cost of modelling raw audio waveforms, we are limited to small batch sizes, which we believe has a negative effect on performance.

Preface

This master thesis is the result of my graduation project which completes my study Business Analytics at the Vrije Universiteit Amsterdam. The project was performed internally at the Computer Science department of the Vrije Universiteit Amsterdam. I would like to thank Dr. Jakub Tomczak for his time reviewing this work and I would like to express special gratitude to David W. Romero, for the opportunity and his excellent assistance in this project. Writing this thesis in times of COVID-19 has not been easy, I would like to thank my girlfriend, family, and friends for their support.

Joep van Genderingen

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, speech recognition has started to change our lives by becoming an important tool for communications and interaction with electronic devices. Products such as Google Allo, Apple Siri, and Amazon Alexa have become an integral part of interactions with mobile devices. Although humans are proficient at perceiving and understanding sounds, making algorithms perform the same task poses a challenge due to the wide range of variations in auditory features (Al-Tahan and Mohsenzadeh, 2021).

Deep neural networks have shown that they can achieve human-level performance in many speech recognition and sound classification tasks (Li et al., 2018; Oord et al., 2016). Many of these approaches are based on recurrent neural networks (RNNs) and convolutional neural networks (CNNs). RNNs have long been exclusively used for tasks handling sequential data due to their theoretical ability to keep track of arbitrary long-term dependencies of the input sequence. Their effective memory horizon however, or the number of steps the network can retain information from, has proven to be surprisingly small in practice (Bengio et al., 1994). Vanishing and exploding gradients hinder the network from learning long-term dependencies, and thereby induce a small effective memory horizon (Hochreiter, 1991). RNNs use Back-Propagation Through Time (BPTT) (Williams and Zipser, 1995), which means that they can't be trained in parallel. Recent works show that CNNs are an excellent alternative for RNNs for sequential data (Bai et al., 2018a; Oord et al., 2016; Romero et al., 2020). CNNs do not suffer from vanishing and exploding gradients and the training instability seen in RNNs. However, CNNs cannot handle sequences of unknown size and their memory horizon must be fixed a priori. CNNs perform well on sequential data as long as relevant input dependencies fall within their memory horizon, and since the extent of the memory horizon is directly attached to a proportional growth of the model size, CNNs are not parameter efficient for high dimensional data (Conneau et al., 2016).

Romero et al. (2021b) propose to replace the conventional discrete convolutional kernel formulation by a continuous one, parametrized by a small neural network. This formulation, called the Continuous Kernel Convolution (CKConv), solves previously mentioned limitations in conventional neural architectures. The continuous kernel takes as input the relative positions and outputs the value of the convolutional kernel at those positions. This way, an arbitrarily large convolutional kernel can be constructed if an equally large sequence of relative positions is given. A global memory horizon can thus be constructed within a single operation without modifying the architecture of the network or adding more parameters. CKConvs do not use Back-Propagation Through Time and can thus be trained and deployed in parallel. Since continuous kernels can be evaluated at arbitrary positions, CKConvs can handle irregularly-sampled and partially observed data. Thanks to these attributes, CKConvs obtain state-of-the-art results on discrete and continuous datasets and multiple stress-tests.

Deep networks most often achieve their strong performance through supervised learning, which requires a large labeled dataset. Labeling a large number of examples requires considerable time and cost, especially when labeling must be done by an expert. This is probably most noticeable in medical applications, where expensive experts like doctors are needed in the labeling process. *Semi-supervised learning* (SSL) is an attractive approach for training models without requiring large amounts of labeled data. SSL leverages large amounts of unlabeled data in combination with smaller amounts of labeled data. Research in this area has grown massively over recent years, following the trend observed in machine and deep learning in general (Van Engelen and Hoos, 2020). Recent advances in SSL are able to close the gap between supervised and semi-supervised learning by only using a few labels (Kim et al., 2021).

The objective of this work is to investigate semi-supervised learning for raw audio waveforms. Previous works on semi-supervised learning for audio rely on features derived from the audio spectogram, such as mel-frequency cepstrum coefficients (MFCC), to obtain good classification performance (Al-Tahan and Mohsenzadeh, 2021; Wang and Oord, 2021). However, it can be challenging and time-intensive to find the right representation in the feature-engineering process, and the often heuristically designed features might not be optimal for the predictive task (Dai et al., 2017). Neural networks can directly take raw waveforms as input and learn similar feature representations (Hoshen et al., 2015). Recent end-to-end learning works on time-series classification show similar performance to spectogram-based approaches (Li et al., 2018; Romero et al., 2020; Tokozume and Harada, 2017).

We propose to use SelfMatch (Kim et al., 2021), a promising semi-supervised learning algorithm that combines the power of contrastive self-supervised learning and consistency regularization, in combination with Continuous Kernel Convolutional Networks (CKCNNs). There currently is no semi-supervised implementation for raw audio waveforms, so we implement Temporal Convolutional Networks (TCNs) to obtain a baseline result. In comprehensive experiments, we evaluate the performance of the networks on varying percentages of labeled data in terms of classification accuracy.

The rest of this thesis is organized as follows: section 2 gives a detailed description of related work. Subsequently, section 3 entails the methods used in this work. Section 4 discusses the experimental setup and results, followed by a discussion on the results in section 5. To finalize, section 6 summarises the conclusions and future work directions.

Chapter 2

Related Work

This chapter provides a thorough review of related work on semi-supervised learning and CKConvs. We first discuss related work on semi-supervised learning. In particular, this subsection discusses two groups of methods in semi-supervised learning, self-supervised representation learning and semi-supervised classification. This subsection is followed by related work on CKConvs.

2.1 Semi-supervised learning

2.1.1 Self-supervised learning

Self-supervised learning is a learning framework that aims to learn representations via pretext tasks that are useful for solving real-world downstream tasks. Pretext tasks are pre-defined tasks for networks to solve, in order to learn useful feature representations for a subsequent downstream task. Self-supervised learning is a class of methods to unsupervised representation learning. This field has a long history and started with classical methods with well established algorithms, such as Principal Component Analysis (Jolliffe, 2011) and Independent Component Analysis (Aapo et al., 2001). Unsupervised representation learning has shown to be highly successful in Natural Language Processing (NLP), as shown by ELMo (Peters et al., 2018), GPT (Radford et al., 2018, 2019), and BERT (Devlin et al., 2018). Promising results in computer vision have only been shown recently (Chen et al., 2020; He et al., 2020; Henaff, 2020). These models learn powerful and universal representations by using self-supervised learning at the pre-training stage to encode contextual information. The representations have proven to be beneficial to performance, especially when the data of the downstream task is limited (Chi et al., 2021).

Not surprisingly, self-supervised learning as pre-training is extensively used in semi-supervised settings. Recent advances in self-supervised learning began with artificially designed pretext tasks such as solving jigsaw puzzles (Noroozi and Favaro, 2016), image colorization (Zhang et al., 2016), rotation prediction (Chen et al., 2019), relative patch prediction (Doersch et al., 2015), or by combining two or more tasks (Noroozi et al., 2018). These approaches learn representations by using objective functions similar to those used in supervised training, but train networks to perform pretext tasks where both the input and labels are derived from an unlabelled dataset. However, there is a significant drawback to these approaches. They rely on heuristics to design these artificial pretext tasks, which could limit the generality of the learned representations.

Contrastive learning. A sub-area within self-supervised learning that has gained much popularity recently is contrastive learning. Contrastive learning was first introduced by Hadsell et al. (2006) as a way of mapping a set of high dimensional input points into a low dimensional manifold where similar points in the input space are mapped to nearby points on the manifold. Contrastive learning learns representations by contrasting positive against negative pairs. Their approach

solves two drawbacks found in existing techniques. Most of the existing techniques depend on a meaningful and computable distance metric in the input space. Secondly, they do not compute a function that can accurately map new input samples whose relationship to the training data is unknown. Another benefit of this method is that it learns mappings that are invariant to transformations of the inputs. In a similar manner, Dosovitskiy et al. (2015) propose to treat each instance as a class represented by a feature vector and train the network to discriminate between a set of surrogate classes. This generic feature representation allows for good classification results for unsupervised learning on several popular datasets, but is computationally very expensive on large-scale datasets. Wu et al. (2018) use the observation that the top-5 classification error on ImageNet is significantly lower than the top-1 error, and that the second highest responding class in the softmax output of an image is likely to be visibly correlated. This shows that apparent similarity is learned from the visual data itself, and not from semantic annotations. They propose to use a non-parametric softmax function and memory bank to store the instance class representations, instead of the parametric softmax used by Dosovitskiy et al. (2015). This leads to a significant improvement in accuracy and computational efficiency. Zhuang et al. (2019) build upon this approach and extend it by adding a local non-parametric aggregation in the latent feature space, causing inputs that are naturally dissimilar to each other to move apart in the embedding space. Their approach surpasses the milestone AlexNet CNN on ImageNet trained on the supervised task. Ye et al. (2019) and Ji et al. (2019) explore the use of in-batch sampling of negative samples instead of a memory bank, yielding even better classification results and efficiency.

Contrastive learning for audio. Contrastive Predictive Coding (CPC) (Oord et al., 2018) uses a contrastive objective in self-supervised learning to learn representations specifically for speech processing tasks. CPC learns these representations by predicting a future frame in latent space by using powerful autoregressive models. Chung et al. (2019) build upon this approach and introduce Autoregressive Predictive Coding (APC), using an autoregressive model from ELMo (Peters et al., 2018) to learn even stronger speech representations. Jiang et al. (2019) introduce an unsupervised pre-training method called Masked Predictive Coding (MPC) for Transformer based models for speech recognition. Liu et al. (2020) present Mockingjay, a method where bidirectional Transformer encoders are pre-trained. Unlike CPC, APC, and MPC, Mockingjay learns speech representations by predicting the current frame through jointly conditioning on both past and future contexts. Speech representations learned by these methods significantly improve performance on both phone classification and speaker verification.

2.1.2 Semi-supervised classification

The goal of a semi-supervised learning algorithm is to utilize unlabelled data in a way that improves performance on labeled data. Over the last two decades, semi-supervised learning has grown into a mature branch of machine learning with a great diversity of approaches (Van Engelen and Hoos, 2020). These approaches differ in the assumptions they are based on, on how they utilize the unlabeled data, and in the way they relate to supervised algorithms (Van Engelen and Hoos, 2020). At the highest level, semi-supervised methods can be distinguished between two groups, inductive and transductive methods (Van Engelen and Hoos, 2020; Zhu, 2005). Inductive methods refer to the use of both labeled and unlabeled data for training, whereas transductive methods only work on the labeled and unlabeled training data, and cannot handle unseen data. Transductive methods are in almost all cases graph-based. In this section we only focus on inductive methods, as these methods form the basis of the method used in this work.

Self-training. Self-training is among the oldest and widely known algorithms for semi-supervised learning (Dattatreya and Kanal, 1986). Self-training uses a model's predictions to obtain artificial labels for unlabeled data. The generality of this approach has led it to be applied in many domains including object detection (Rosenberg et al., 2005), NLP (McClosky et al., 2006), and image classification (Lee, 2013). Pseudo-labeling is a specific variant of self-training where model predictions are converted to hard labels (Lee, 2013). A potential risk of this approach is

that classification errors can reinforce itself (Zhu, 2005). Therefore, pseudo-labeling techniques often use confidence-based thresholding to make sure that unlabeled samples are only used when the classifier is sufficiently confident. Oliver et al. (2018) suggest that pseudo-labeling is not competitive against other SSL algorithms on its own, but recent advances in SSL use pseudo-labeling as part of their pipeline to produce better results (Arazo et al., 2020; Ishii, 2021).

Co-training is an extension of self-training where two or more supervised classifiers are iteratively trained on the labeled data, adding their most confident predictions to the labeled dataset of other classifiers in each iteration (Van Engelen and Hoos, 2020). Blum and Mitchell (2000) propose to construct two classifiers that are trained on two subsets of features of the given data. Their algorithm relies on two assumptions to work: each subset of features should be sufficient to obtain good predictions on the given dataset and the subsets of features should be conditionally independent given the class label. In practice, the second assumption is generally not satisfied. Even if a natural split of features exists, it is unlikely that information contained in one view provides no information on the other view when conditioned on the class label (Du et al., 2011). Balcan et al. (2004) relax the conditional Independence assumption, showing that a weaker assumption, the expansion assumption, is sufficient. The expansion assumption states that the two views are not highly correlated, and that individual classifiers never confidently make incorrect predictions. Du et al. (2011) propose several methods for automatically splitting the feature set in two views, indicating that feature splits optimizing sufficiency and independence lead to good classifiers.

Augmentation consistency regularization. Data augmentation is a common regularization technique in supervised learning, applying transformations on the input while leaving class semantics unaffected. This artificially expands the size of a training set by generating a, theoretically, near-infinite stream of new, modified data. Data augmentation as regularization is especially common in image classification, where elastically deforming or adding noise to an input image drastically changes the pixel content of an image without altering its label (Ciresan et al., 2010; Simard et al., 2003). Consistency regularization (Bachman et al., 2014) is a data augmentation method for SSL. It leverages the idea that a classifier should output the same class distribution for an unlabeled sample even after it has been augmented. Rasmus et al. (2015) introduce the Γ-model, based on the ladder network, which is a model that is trained to simultaneously minimize the sum of supervised and unsupervised loss functions by Back-Propagation. The resulting model reached state-of-the-art in semi-supervised classification of MNIST and CIFAR-10. Laine and Aila (2016) propose to simplify the Γ-model, which they call the Π-model. The Π-model removes the parametric nonlinearity and denoising, and compares the outputs of the network instead of the pre-activation data in the final layer. Their approach results in a significant improvement in classification accuracy, but increases the variance of generated targets. Tarvainen and Valpola (2017) propose a "mean-teacher" approach to improve the target quality, outperforming the Π-model on multiple datasets. Miyato et al. (2018) solve the same problem in a method called Virtual Adversarial Training (VAT). VAT uses virtual adversarial loss as a regularization technique, a new measure of local smoothness of the conditional label distribution given the input.

MixMatch (Berthelot et al., 2019b) unifies several of the previously mentioned SSL techniques and introduce a unified loss term. This unified loss combines a supervised loss of labeled samples and unsupervised loss of guessed labels of augmented unlabeled samples. Their approach reduces entropy while maintaining consistency and remaining compatible with recent regularization techniques. ReMixMatch (Berthelot et al., 2019a) improves MixMatch by introducing two new techniques: distribution alignment and augmentation anchoring. Distribution alignment encourages the marginal distribution of unlabeled samples to be similar to the marginal distribution of ground truth samples. Augmentation anchoring forces consistency by using a weakly-augmented sample to generate an artificial label and enforcing this against a strongly-augmented view of the input sample. FixMatch (Sohn et al., 2020) can be seen as a substantial simplification of ReMixMatch. FixMatch produces artificial labels using only pseudo-labeling and consistency regularization. An

artificial label is produced based on a weakly-augmented unlabeled sample, which is then used as a target when the model is fed a strongly-augmented view of the same sample. Despite its simplicity, FixMatch outperforms ReMixMatch and obtains state-of-the-art performance on most SSL benchmarks.

2.2 CKCNNs

Recurrent neural networks (RNNs) have long been considered as the status quo for tasks handling sequential data. Due to recurrent units, connections between the output and input of neurons, RNNs can theoretically keep track of arbitrary long-term dependencies of the input sequence. RNNs are powerful dynamic systems, but training them has proven to be problematic (LeCun et al., 2015). Vanishing and exploding gradients are a well known problem in RNNs (Hochreiter, 1991), and mainly due to this problem, their effective memory horizon is surprisingly small.

Gating mechanisms. Gating mechanisms are a popular method to preserve information from the far past and enhance gradient flow. Hochreiter and Schmidhuber (1997) were the first to introduce a gating mechanism with the Long Short-Term Memory (LSTM) unit. The central idea behind the LSTM architecture is a memory cell which maintains the state over time, and input and output gates which regulate the information flow into and out of the cell. The LSTM is able to solve complex long time lags tasks that have never been solved by previous RNN algorithms. Gers et al. (2000) identify a weakness of lSTMs when processing continual input streams and introduce the forget gate. This forget gate enables a LSTM cell to learn to reset itself at appropriate times. Peephole connections (Gers and Schmidhuber, 2000) were the next addition to LSTM units to make learning precise timings possible. This is a connection from the cell to the gates, enabling the cell to control the gates. Additionally, the output activation function was omitted, as there was no evidence that it was essential for solving the tasks that LSTMs were tested on thus far. The first formulation of LSTMs trained using a mixture of Real Time Recurrent Learning (RTRL) and Back-Propagation Through Time (BPTT). Only the gradient of the cell was backpropagated through time, whereas the gradient for the other recurrent connections were truncated. Graves and Schmidhuber (2005) present a full BPTT training for LSTM networks. These modifications form the basis of what we know as the LSTM unit now. The LSTM network has had many successes since then, including being used as Google's speech recognition model on Google Voice (Beaufays, 2015).

Cho et al. (2014) propose a simplification of the LSTM unit, called the Gated Recurrent Unit (GRU). They remove the peephole connection and output activation functions, and coupled the input and forget gate into an update gate. The output gate in GRUs, called the reset gate, only gate the recurrent connection to the block input. Chung et al. (2014) compare different types of recurrent units in RNNs. Their experiments show that gated units work significantly better than traditional recurrent units, but could not make a conclusion on which gating mechanism was better. Dauphin et al. (2017) propose a gating mechanism for CNNs. Convolutional networks can be stacked to represent large context sizes. They consider models using only output gates, which allow the network to control what information should be propagated through the layers and find that they outperform LSTMs while being more efficient.

CNNs for sequential data. Recent advances show that CNNs are excellent alternatives for RNNs in handling sequential data (Cui et al., 2016). Oord et al. (2016) introduce a network of stacked dilated convolutions for generating raw audio waveforms called WaveNet. By properly defining dilation factors, one can obtain deep networks which look at all values in the memory horizon. This type of network is also known as a Temporal Convolutional Network (TCN) (Lea et al., 2016). Altough WaveNet is designed as a generative model, it can easily be adapted to discriminative audio tasks such as speech recognition. They show that layers of dilated convolutions allow the receptive field to grow longer in a much cheaper way than using LSTM units. Bai

et al. (2018a) show in an empirical evaluation of convolutional and recurrent networks for sequence modelling that TCNs substantially outperform advanced RNNs such as LSTMs and GRUs. They also show that TCNs exhibit longer memory than recurrent architectures with the same capacity. Bai et al. (2018b) present Trellis networks, a new architecture for sequence modelling. Trellis networks are TCNs with a special structure, characterized by weight tying and direct injection of the input into deep layers. Experiments show that Trellis networks outperform state-of-the-art methods on a variety of datasets and stress tests designed to test long-term memory retention. Transformer, or self-attention, models are also good alternatives for sequence modelling. Dai et al. (2019) show that Transformers have a great potential to learn longer-term dependencies, up to 80% longer than RNNs.

The next chapter entails a detailed description of the semi-supervised algorithm SelfMatch and CKCNNs. As briefly introduced in the previous chapter, we choose to use SelfMatch as semi-supervised algorithm. SelfMatch achieves state-of-the-art results on multiple benchmark datasets and is able to close the gap between fully supervised learning and semi-supervised learning with only a few labels. More specifically, SelfMatch achieves similar results with only four labels per class and fully supervised learning. In this work we specifically focus on using CKCNNs to model raw audio waveforms in a semi-supervised setting. CKCNNs solve important limitations present in conventional neural architectures and are able to handle long-term dependencies.

Chapter 3

Methodology

In this section we present our semi-supervised learning framework. The chapter is divided into three sections. The first section gives a detailed description of SelfMatch. The following sections cover Temporal Convolutional Networks and Continuous Kernel Convolutional Networks.

3.1 SelfMatch

SelfMatch (Kim et al., 2021) is a semi-supervised learning method that combines the power of contrastive self-supervised learning and augmentation consistency regularization. SelfMatch consists of two stages: (1) self-supervised pre-training based on contrastive learning and (2) fine-tuning based on augmentation consistency regularization. A schematic overview is shown in Figure 3.1. SelfMatch outperforms recent semi-supervised methods such as ReMixMatch (Berthelot et al., 2019a) and FixMatch (Sohn et al., 2020), and closes the gap between supervised learning and semi-supervised learning by only using a few labels for each class.

3.1.1 Self-supervised pre-training

The contrastive learning based self-supervised pre-training in SelfMatch is based on the SimCLR method designed by Chen et al. (2020). SimCLR is an improvement over recent contrastive self-supervised learning algorithms (Bachman et al., 2019; He et al., 2020) that does not require specialized architectures or a memory bank. SimCLR learns representations by maximizing agreement between two differently augmented views of the same sample via contrastive loss in the latent space. As shown in Figure 3.1, this framework consists of four components: stochastic data

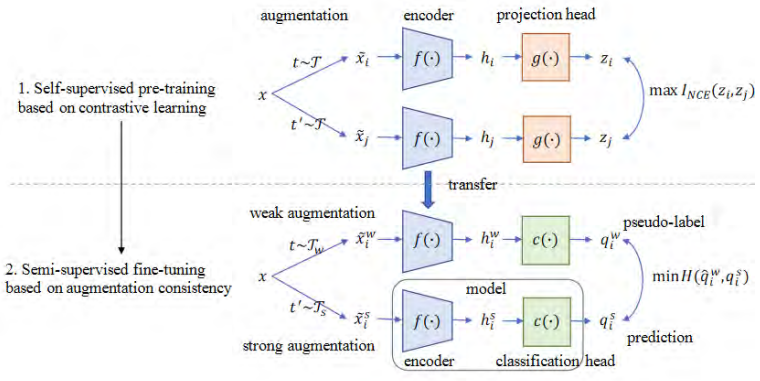

Figure 3.1: SelfMatch overview (Kim et al., 2021).

augmentation module $T(\cdot)$, neural network base encoder $f(\cdot)$, projection head $g(\cdot)$, and contrastive loss $\mathcal{L}_c$. Data augmentation module $T(\cdot)$ transforms a given input randomly resulting in two correlated views of the same sample, denoted by $\tilde{x}_i$ and $\tilde{x}_j$. In this work three augmentations on the raw audio signal are used. We randomly choose two augmentations and apply them based on the random ordering.

Audio mixing Audio mixing adds noise by mixing two audio signals. Adding small additive noise of any sort will not alter the original category of the audio signal. Given two audio signals x_1 and x_2, the mixed-up version is

$$\hat{x}_1 = \alpha x_1 + (1 - \alpha)x_2 \tag{3.1}$$

where $\hat{x}_1$ inherits the label from x_1 and α is sampled from the $\beta(5, 2)$ distribution. According to Wang and Oord (2021), this simulates various realistic noise conditions.

Shift time This augmentation shifts the raw audio signal by a certain percentage along the x-axis. In this work, we shift the raw audio signal by 15%. The shift direction is randomly sampled.

Random mask The random mask augmentation masks out roughly half of the coordinates in the data. Masking is done using the Bernoulli distribution. Let v be a vector with the same length of x where each element is sampled from the Bernoulli($\frac{1}{2}$) distribution, or $v_i \sim \text{Ber}(\frac{1}{2})$. The masked audio signal is then given by

$$\hat{x} = v^T x \tag{3.2}$$

The objective of contrastive learning is to learn features that can match two disjoint sets of the coordinates of given data points. According to Wen and Li (2021), this augmentation helps to maintain the correlations of desired signals between the disjoint coordinates and remove the undesired correlations.

Neural network base encoder $f(\cdot)$ extracts representation vectors from the augmented samples. This self-supervised framework allows various choices of network architectures without any constraints (Kim et al., 2021). The main focus of this work is to investigate the performance of CKCNNs on sequential data in a semi-supervised setting, thus CKCNNs are used as base encoders. As there currently is no baseline available for SelfMatch on sequential data, a baseline is created using TCNs. Chen et al. (2020) found that unsupervised contrastive learning benefits from bigger models. While similar findings hold for supervised learning (He et al., 2016; Szegedy et al., 2015), Chen et al. (2020) show that the difference between supervised models and linear classifiers trained on unsupervised models decreases as the model size increases.

Projection head $g(\cdot)$ maps the extracted representations from the base encoder to the latent space where contrastive loss is applied, or $z = g(h)$. Chen et al. (2020) found that it is beneficial to define the contrastive loss on z_i's in the latent space rather than on h_i's in the representation space. This is due to the loss of information induced by the contrastive loss function. The projection head is trained to be invariant to data transformation, and thus $g(\cdot)$ can remove information that may be useful for the downstream task. By leveraging the nonlinear transformation $g(\cdot)$, more information can be formed and maintained in representations h. A two layer MLP with ReLU nonlinearities is used to obtain z_i from representation h_i.

The contastive loss function encourages two views of the same sample to be similar, and two views from different samples to be dissimilar. SimCLR uses a normalized version of contrastive loss, also known as NT-Xent (normalized temperature-scaled cross entropy loss). Chen et al. (2020) find that NT-Xent, with proper temperature scaling, works much better than other commonly used contrastive loss functions. The NT-Xent loss for a positive pair of examples (i, j) is defined as

$$\mathcal{L}_c = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} 1_{(k \neq i)} \exp(\text{sim}(z_i, z_k)/\tau)} \tag{3.3}$$

where $1_{(k \neq i)} \in \{0, 1\}$ is an indicator function evaluating to 1 if $k \neq i$, τ denotes a temperature parameter, and $\text{sim}(\cdot)$ the cosine similarity measure. The cosine similarity between u and v is given by

$$\text{sim}(u, v) = \frac{u^T v}{\|u\| \|v\|} \tag{3.4}$$

A randomly sampled minibatch of N instances is fed through the data augmentation module, resulting in $2N$ data points. Negative instances are not sampled explicitly, but given a positive pair, the other $2(N-1)$ instances in the minibatch are treated as negatives. The final loss is computed across all positive pairs, i.e. (i, j) and (j, i) in a minibatch.

3.1.2 Semi-supervised fine-tuning

The semi-supervised fine-tuning based on augmentation consistency regularization in SelfMatch is based on FixMatch designed by Sohn et al. (2020). FixMatch is a combination of two semi-supervised approaches: consistency regularization (Bachman et al., 2014) and pseudo-labelling (McLachlan, 1975). Consistency regularization uses unlabelled data by relying on the assumption that the model should produce similar predictions when fed two augmented views of the same sample, while pseudo-labelling utilizes the idea of using the model itself to produce labels for the unlabelled data. FixMatch encourages a consistent prediction between weakly and strongly augmented samples, denoted by $\tilde{x}_i^w$ and $\tilde{x}_i^s$. More specifically, FixMatch uses the model output of a weakly augmented sample $q_i^w = p_{model}(y|\tilde{x}_i^w)$ as pseudo-label for a strongly augmented input $\tilde{x}_i^s$.

The loss function in FixMatch consists of two cross-entropy loss terms: a supervised loss l_s applied to the labelled samples and unsupervised loss l_u. The supervised loss is essentially just the standard cross-entropy loss on the weakly augmented labelled samples:

$$l_s = \frac{1}{B} \sum_{i=1}^{B} H(p_i, q_i^w) \tag{3.5}$$

where B is the batch size and p_i the one-hot encoded label for sample x_i. FixMatch computes artificial labels for all unlabelled samples which are then used in a standard cross-entropy loss. To obtain these artificial labels, the model's predicted class distribution given a weakly augmented version of a unlabelled sample needs to be computed first, denoted by $q_i^w = p_{model}(y|\tilde{x}_i^w)$. Then, $\hat{q}_i^w = \arg\max(q_i^w)$ is used as the pseudo-label. Unsupervised loss l_u is formulated as follows:

$$l_u = \frac{1}{\mu B} \sum_{i=1}^{\mu B} 1_{(\max(q_i^w) > c)} H(\hat{q}_i^w, q_i^s). \tag{3.6}$$

where $1_{(\max(q_i^w) > c)} \in \{0, 1\}$ is an indicator function evaluating to 1 if the maximum class probability is higher than confidence threshold c, μ is the ratio of labelled and unlabelled data samples in a minibatch, and q_i^s is the model output of strongly augmented sample $\tilde{x}_i^s$. Combining these loss terms, the final loss minimized by FixMatch is:

$$\mathcal{L}_{semi} = l_s + \lambda_u l_u \tag{3.7}$$

where λ_u is a fixed scalar hyperparameter denoting the relative weight of the unlabelled loss. In modern SSL algorithms it is typical to increase λ_u during training (Berthelot et al., 2019a,b; Tarvainen and Valpola, 2017), but Sohn et al. (2020) found that this is unnecessary for FixMatch. This may be due to the fact that $\max(q_i)$ is less than confidence threshold c early in training and as training progresses, the model's predictions become more confident and $\max(q_i) > c$ occurs

more frequently.

FixMatch uses two kinds of augmentations: *weak* and *strong*. The augmentations discussed in 3.1.1 are considered as weak augmentations. Strong augmentations are created by sequentially applying two weak augmentations to the same sample.

3.2 Temporal Convolutional Networks

Temporal Convolutional Networks (TCNs), or Dilated Causal Convolutional Networks, have shown to be an excellent alternative to RNNs for tasks handling sequential data (Bai et al., 2018a; Oord et al., 2016; Yan et al., 2020). TCNs avoid Back-Propagation Through Time, and thus do not suffer from vanishing and exploding gradients, and the training instability seen in RNNs.

The main ingredient of TCNs are causal convolutions. Providing a causal formulation to convolutions ensures that the model can't violate the ordering in which the data is modelled. The convolutional filter can thus only see past and present input values. A *centered* non-causal convolution is effectively performed between the input signal described as a sequence of finite length $X = \{x(\tau)\}_{\tau=0}^{N_x}$ and a convolutional kernel $K = \{\psi(\tau)\}_{\tau=0}^{N_x}$ described in the same way:

$$(x * \psi)(t) = \sum_{c=1}^{N_c} \sum_{\tau=-N_x/2}^{N_x/2} x_c(\tau)\psi_c(t-\tau) \tag{3.8}$$

where c is the number of convolutional channels. However, this formulation is undesirable for sequence modelling when input sequence $X = \{x(t)\}_{\tau=-N_x/2}^{-1}$ is considered, as the convolution is dependent on future observations. The causal convolution for input sequence X is given by:

$$(x * \psi)(t) = \sum_{c=1}^{N_c} \sum_{\tau=0}^{t} x_c(\tau)\psi_c(t-\tau) \tag{3.9}$$

Stacking multiple layers of causal convolutions, as shown in Figure 3.2, allows the network to build a *receptive field*. Models with causal convolutions do not have recurrent connections and are thus much faster to train than RNNs. One problem with causal convolutions however, is that they require many layers, or large filters, to increase the receptive field, resulting in a lot of weights. *Dilated* convolutions can be used to increase the receptive field without greatly increasing the number of weights. A dilated convolution is a convolution where the filter is applied over an area larger than its length by skipping input values by a certain step (Oord et al., 2016). The dilated convolution is given by:

$$(x *_\eta \psi)(t) = \sum_{c=1}^{N_c} \sum_{\tau=0}^{N_k} x_c(\eta\tau)\psi_c(t-\eta\tau) \tag{3.10}$$

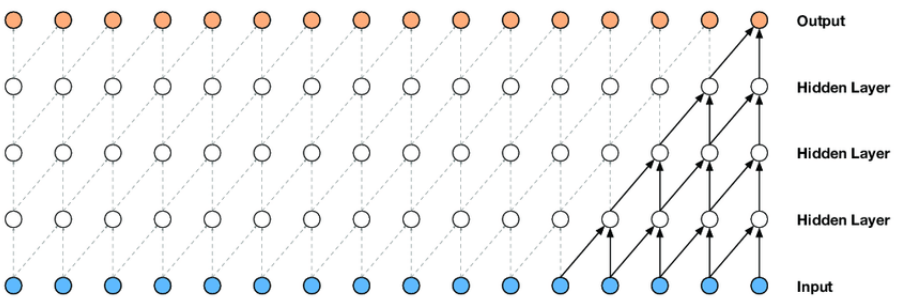

Figure 3.2: Visualisation of a stack of causal convolutional layers (Oord et al., 2016).

There is, again, one issue with this formulation. It causes extreme sparsity, input values between $x(t)$ and $x(t-\eta)$ cannot be seen by the convolutional operation. Stacking convolutions with different dilation factors is a possible solution to this problem. This also enables the network to build a very large receptive field with just a few layers. Figure 3.3 shows dilated causal convolutions for dilation factors 1, 2, 4, and 8. In this work we follow Oord et al. (2016), and double the dilation factor every layer. Exponentially increasing the dilation factor results in exponential receptive field growth with depth. Stacking these blocks increases the model capacity and the receptive field size.

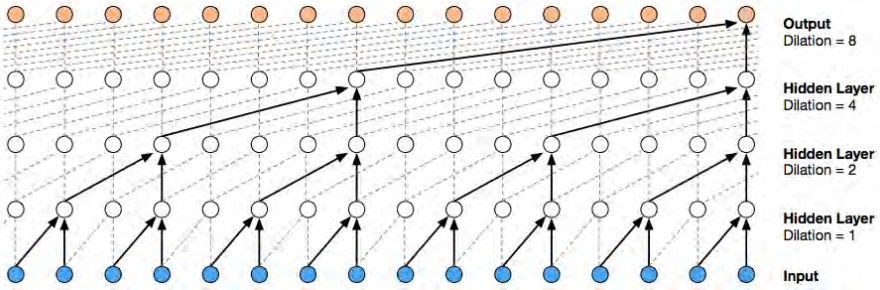

Figure 3.3: Visualization of a stack of *dilated* causal convolutional layers (Oord et al., 2016).

3.3 Continuous Kernel Convolutions

Continuous Kernel Convolutional networks (CKCNNs) solve important limitations in conventional neural architectures for sequential data. Recurrent neural networks have long ruled tasks handling sequential data, but due to vanishing and exploding gradients, their effective memory horizon, or the number of steps the network can retain information from, has proven to be surprisingly small in practice. Convolutional networks have proven to be a strong alternative for recurrent architectures by circumventing Back-Propagation Through Time altogether. However, convolutional networks cannot handle sequences of unknown size and their memory horizon must be fixed a priori. Romero et al. (2021b) propose to replace the conventional discrete convolutional kernel by a continuous one, parametrized by a small neural network, called the Continuous Kernel Convolution (CKConv). This formulation leads to the following improvements over conventional neural architectures:

- CKConvs are able to consider arbitrarily large memory horizons within a single operation, similar to RNNs.
- Contrary to RNNs, CKConvs do not rely on any form of recurrency, and thus CKCNNs do not suffer from vanishing / exploding gradients and small effective memory horizons.
- Contrary to conventional CNNs, CKCNNs detach the memory horizon from the depth of the network, dilation factor used, and parameter count of the architecture.
- CKCNNs can be trained in parallel, as they do not make use of Back-Propagation Trough Time.
- CKConvs can handle irregularly sampled data and data sampled at different sampling rates, since continuous kernels can be evaluated at arbitrary positions.

Convolutional kernel ψ is formulated as a continuous function parametrized by small neural network MLP^{ψ}. MLP^{ψ} takes as input a relative position $(t-\tau)$ and outputs the value of the convolutional kernel for that position $\psi(t-\tau)$. This formulation allows an arbitrarily large convolutional kernel $\mathcal{K} = \{\psi(t-\tau)\}_{\tau=0}^{N_K}$ to be constructed by providing an equally large sequence of relative positions $\{t-\tau\}_{\tau=0}^{N_K}$ to MLP^{ψ}. Figure 3.4 shows a visual representation of the CKConv. For the case $N_K = N_X$, a convolutional kernel of equal size to the input sequence $\mathcal{X}$ can be

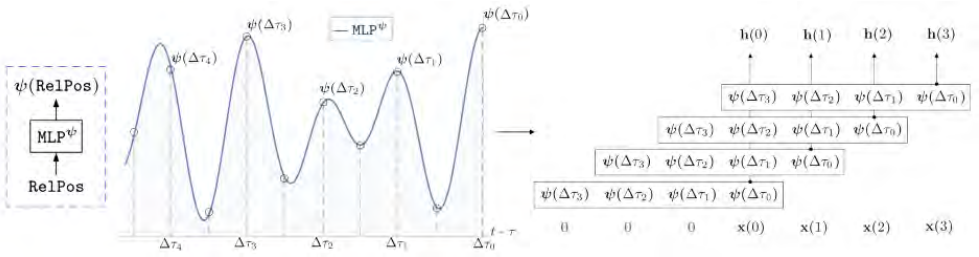

Figure 3.4: Continuous Kernel Convolution (Romero et al., 2021b).

sampled. A global memory horizon can thus be constructed without modifying the structure of the network or adding more parameters. The CKConv is given by:

$$(x * \psi)(t) = \sum_{c=1}^{N_C} \sum_{\tau=0}^{t} x_c(\tau) \text{MLP}_c^{\psi}(t - \tau) \tag{3.11}$$

with c the number of convolutional channels.

Let $\{\Delta\tau_i = (t - \tau)\}_{i=0}^{N}$ be a sequence of relative positions. The convolutional kernel MLP^{ψ} can then be parametrized by a conventional L-layer neural network:

$$h^{(1)}(\Delta\tau_i) = \sigma(W^{(1)}\Delta\tau_i + b^{(1)}) \tag{3.12}$$

$$h^{(l)}(\Delta\tau_i) = \sigma(W^{(l)}h^{(l-1)}(\Delta\tau_i) + b^{(l)}) \tag{3.13}$$

$$\psi(\Delta\tau_i) = W^{(L)}h^{(L-1)}(\Delta\tau_i) + b^{(L)} \tag{3.14}$$

with σ a point-wise non-linearity. This approach can be seen as providing implicit neural representations to the unknown convolutional kernels ψ of a conventional convolutional architecture.

Recent work by Sitzmann et al. (2020) proposes to replace ReLU by Sine activation functions to learn implicit neural representations. Romero et al. (2021b) found that this small modification allows the convolutional kernels to approximate any provided function almost perfectly. The Sine activation function is given by:

$$y = \text{Sin}(\omega_0[Wx + b]) \tag{3.15}$$

where ω_0 works as a prior on the variability of the target function. Unlike ReLU layers, Sine layers periodically bend the space. Subsequently, the same y value is obtained for all bias values $b_i' = b_i + n2\pi\|W_{i,:}\|$, $\forall n \in Z$. One unique value of b exists for ReLU layers that bends the space at the right position, while for Sine layers infinitely many values of b do so. This means that Sine layers are much more robust to parameter selection and can be tuned to approximate functions at arbitrary, and even at multiple positions in space. This attribute leads to faster convergence and much more reliable approximations.

3.3.1 Gabor Layers

Gabor filters (Gabor, 1946) are widely used in image processing and computer vision tasks. A Gabor filter can be seen as a sinusoidal plane wave with particular frequency and orientation, modulated by a Gaussian wave. This allows the filter to extract spatial frequency structures from the data that can be used for edge detection, texture analysis, and feature extraction. This attribute has shown to be highly effective for texture representation and face detection tasks (Huang et al., 2004) in images for example. Romero et al. (2021a) introduce a novel kernel parametrization based on Gabor filters for which better approximation capabilities and faster convergence properties are obtained.

Recently, Fathony et al. (2021) proposed multiplicative Gabor Networks with isotropic Gaussian envelopes and construct implicit neural representations as the linear combination of exponentially many basis functions $\mathbf{g}$:

$$\mathbf{h}^{(1)} = \mathbf{g}\big([x,y];\boldsymbol{\theta}^{(1)}\big) \qquad \mathbf{g}: R^2 \to R^{\mathrm{N_{hid}}} \tag{3.16}$$

$$\mathbf{h}^{(l)} = \big(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\big) \cdot \mathbf{g}\big([x,y];\boldsymbol{\theta}^{(l)}\big) \qquad \mathbf{W}^{(l)} \in R^{\mathrm{N_{hid}}\times\mathrm{N_{hid}}}, \mathbf{b}^{(l)} \in R^{\mathrm{N_{hid}}} \tag{3.17}$$

$$\boldsymbol{\psi}(x,y) = \mathbf{W}^{(\mathrm{L})}\mathbf{h}^{(\mathrm{L}-1)} + \mathbf{b}^{(\mathrm{L})} \qquad \mathbf{W}^{(\mathrm{L})} \in R^{(\mathrm{N_{out}}\times\mathrm{N_{in}})\times\mathrm{N_{hid}}}, \mathbf{b}^{(\mathrm{L})} \in R^{(\mathrm{N_{out}}\times\mathrm{N_{in}})} \tag{3.18}$$

where $\big\{\boldsymbol{\theta}^{(l)}, \mathbf{W}^{(l)}, \mathbf{b}^{(l)}\big\}$ represent the learnable parameters of the bases and the affine transformations. Due to the form of their parametrization, Multiplicative Filter Networks (MNFs) (Fathony et al., 2021) obtain approximations comparable to those of Sine nonlinearities, but at a faster convergence rate. Their best performing network, the Multiplicative Gabor Network, is obtained by using Gabor functions with an isotropic Gaussian envelope as basis:

$$\mathbf{g}\big([x,y];\boldsymbol{\theta}^{(l)}\big) = \exp\left(-\frac{\boldsymbol{\gamma}^{(l)}}{2}\|[x,y]-\boldsymbol{\mu}^{(l)}\|_2^2\right)\mathrm{Sin}\big(\mathbf{W}_{\mathrm{g}}^{(l)}\cdot[x,y]+\mathbf{b}_{\mathrm{g}}^{(l)}\big), \tag{3.19}$$

$$\boldsymbol{\theta}^{(l)} = \big\{\boldsymbol{\gamma}^{(l)} \in R^{\mathrm{N_{hid}}}, \boldsymbol{\mu}^{(l)} \in R^{\mathrm{N_{hid}}}, \mathbf{W}_{\mathrm{g}}^{(l)} \in R^{\mathrm{N_{hid}}\times 2}, \mathbf{b}_{\mathrm{g}}^{(l)} \in R^{\mathrm{N_{hid}}}\big\} \tag{3.20}$$

However, using a basis with isotropic Gaussian envelopes, i.e., with equal γ for both the horizontal and vertical directions, is not desirable (Romero et al., 2021a). The necessity for a particular frequency in a certain direction automatically induces that frequency on a circular range. Consequently, other bases must counteract this frequency on undesirable parts of this circular range. Romero et al. (2021a) alleviate this limitation by replacing the isotropic formulation of Fathony et al. (2021) with anisotropic Gabor functions

$$\mathbf{g}\big([x,y];\boldsymbol{\theta}^{(l)}\big) = \exp\left(-\frac{1}{2}\Big[\Big(\boldsymbol{\gamma}_{\mathrm{X}}^{(l)}\big(x-\boldsymbol{\mu}_{\mathrm{X}}^{(l)}\big)\Big)^2+\Big(\boldsymbol{\gamma}_{\mathrm{Y}}^{(l)}\big(y-\boldsymbol{\mu}_{\mathrm{Y}}^{(l)}\big)\Big)^2\Big]\right)\mathrm{Sin}\big(\mathbf{W}_{\mathrm{g}}^{(l)}[x,y]+\mathbf{b}_{\mathrm{g}}^{(l)}\big) \tag{3.21}$$

$$\boldsymbol{\theta}^{(l)} = \Big\{\boldsymbol{\gamma}_{\mathrm{X}}^{(l)} \in R^{\mathrm{N_{hid}}}, \boldsymbol{\gamma}_{\mathrm{Y}}^{(l)} \in R^{\mathrm{N_{hid}}}, \boldsymbol{\mu}_{\mathrm{X}}^{(l)} \in R^{\mathrm{N_{hid}}}, \boldsymbol{\mu}_{\mathrm{Y}}^{(l)} \in R^{\mathrm{N_{hid}}}, \mathbf{W}_{\mathrm{g}}^{(l)} \in R^{\mathrm{N_{hid}}\times 2}, \mathbf{b}_{\mathrm{g}}^{(l)} \in R^{\mathrm{N_{hid}}}\Big\} \tag{3.22}$$

Consequently, the resulting *Anisotropic Multiplicative Gabor Network* (MAGNET) obtains better control upon frequency components introduced to the approximation. We will refer to this network as the "Gabor CKCNN" in the remainder of this work.

Chapter 4

Experiments and results

In this chapter, we give a detailed description of the data used in this work, experiments that are carried out, and the results of these experiments. We end the chapter with a brief discussion on the results.

4.1 Data

4.1.1 SpeechCommands

SpeechCommands (Warden, 2018) is an audio dataset of spoken words designed to train and evaluate keyword spotting systems. Many voice programs rely on keyword spotting to start interactions. For example, saying "Hey Google" or "Hi Siri" starts a query or command on one's phone. The dataset contains 105,809 one-second recordings of both background noise and 35 spoken words sampled at 16kHz. In this work, the approach by Kidger et al. (2020) and Romero et al. (2021b) is followed to obtain a balanced classification dataset by selecting a subset of ten spoken words. This results in a dataset of 34,975 raw audio recordings. The SpeechCommands dataset is split into a training dataset consisting of 70% of the data, and a validation and test set both consisting of 15%. The data is then standardized to 0 mean and variance 1 using the statistics of the training dataset.

4.1.2 UrbanSound8K

UrbanSound8K (Salamon et al., 2014) is an audio dataset of environmental sounds in urban areas used for classification tasks. The dataset consists of 8,732 four-second (or less) stereo recordings of ten environmental sounds sampled at 44.1kHz, totalling around 10 hours of data. The data comes in ten predefined folds for cross-validation. In this work, we follow Lee et al. (2017) and train the networks on the first nine folds and report results on the tenth training fold. Although some information might be lost, we follow Dai et al. (2017) and down-sample the audio waveforms to 8kHz for computational feasibility. Not all audio signals in UrbanSound8K are four seconds long, so to ensure that every signal is the same length we pad signals that are shorter with zeros and clip signals that are longer. The data is also standardized to 0 mean and variance 1.

4.2 Experimental setup

We perform two sets of experiments on each of the two datasets. For each dataset, training was performed on varying percentages of labeled samples in the training dataset for two different approaches. We compare SelfMatch to fully supervised learning on the same labeled samples to evaluate the performance benefit of SelfMatch. For both datasets, we compare results on 1% and 10% of all labeled samples. All results shown in section 4.3 are computed over one training run and

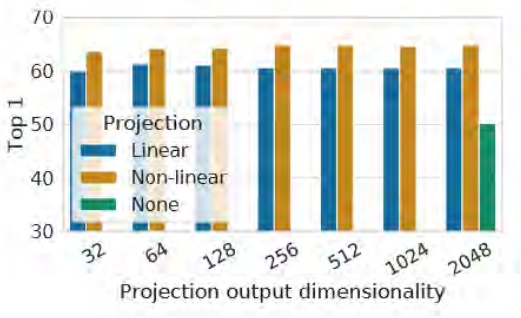

Figure 4.1: Evaluation of different projection head architectures in SimCLR (Chen et al., 2020).

based on the test set. Every experiment uses the same seed to ensure that the same set of labeled samples and augmentations are used. Pre-training is performed on the full dataset without labels, while for the fine-tuning stage, the full dataset without labels are used as unlabeled samples and a subset of this data is used as labeled samples. It is important to note that all labelled data is part of unlabelled data without their labels.

The pre-training and fine-tuning stage of SelfMatch introduce some design choices and hyper-parameters that need to be set before training can start. Chen et al. (2020) show the difference in performance of different architectures of the projection head $g(\cdot)$. Figure 4.1 shows evaluation results for three different architectures of the projection head: identity mapping, linear projection, and default nonlinear projection with ReLU activation. A nonlinear projection is better than a linear projection, and much better than no projection. Projection output dimensionality does not significantly influence final performance. For the projection head, we use a two-layer MLP with ReLU activations and a projection output dimensionality of 32 for parameter efficiency. Chen et al. (2020) also find that contrastive learning benefits from larger batch sizes. Larger batch sizes provide more negative samples, facilitating faster convergence. In this study, we are limited to a batch size of 16, while SimCLR uses a batch size of 4096 and SelfMatch 3584. This is mainly due to the large computational cost associated with modelling raw audio waveforms. Sohn et al. (2020) show the importance of the ratio of unlabeled data μ for the final performance. Figure 4.2 shows a plot of the error rates with different ratios of unlabelled data. A significant decrease in error rates is observed when using a large amounts of unlabelled data. We use a high μ as starting point in our experiments and carry out a similar ablation study for the effect of μ.

We train all networks on each dataset using the Adam optimizer and plateau learning rate scheduler with a patience of 15. For both sets of experiments, the hyperparameters of the networks were tuned by minimising the classification error on the validation set. The hyperparameters of the networks are not tuned throughout the experiments, such that every run with a different percentage of labeled data uses the same network. Table 4.1, 4.2, and 4.3 show all hyperparameters used for CKCNNs, Gabor CKCNNs, and TCNs respectively. Fully supervised training was performed on 100 epochs, while networks trained using SelfMatch pre-trained on 50 epochs and fine-tuned on 100 epochs. With this approach we follow Kim et al. (2021), and use the same number of iterations for fine-tuning the classifier. In all our experiments we follow Kim et al. (2021), and use $\lambda_\mu = 1$ in equation 3.7 and $\tau = 0.1$ in equation 3.3, as we found that these values work best after hyperparameter tuning and state-of-the-art results are obtained with these parameter values. All experiments were run on various GPU's on the computational infrastructure DAS-5 (Bal et al., 2016).

Table 4.1: Hyperparameters CKCNNs on both datasets.

Hyperparameter	Value	
	SpeechCommands	UrbanSound8K
Batch Size	16	8
Optimizer	Adam	Adam
Pre-train Learning Rate	1e-04	1e-04
Fine-tune Learning Rate	3e-04	3e-04
# Blocks	2	2
Hidden Size	30	30
ω_0	39.45	39
Dropout	0	0
Weight Decay	0.0001	0.0001
Norm Type	Layer Norm	Layer Norm
Model size	100k	100k

Table 4.2: Hyperparameters Gabor CKCNNs on both datasets.

Hyperparameter	Value	
	SpeechCommands	UrbanSound8K
Batch Size	16	8
Optimizer	Adam	Adam
Pre-train Learning Rate	1e-03	1e-03
Fine-tune Learning Rate	3e-04	3e-04
# Blocks	2	2
Hidden Size	30	30
Input Scale	256	256
Weight Scale	1	1
α	6	6
β	1	1
Dropout	0	0
Weight Decay	0.0001	0.0001
Norm Type	Batch Norm	Batch Norm
Model size	109k	109k

Table 4.3: Hyperparameters TCNs on both datasets.

Hyperparameter	Value	
	SpeechCommands	UrbanSound8K
Batch Size	16	8
Optimizer	Adam	Adam
Pre-train Learning Rate	3e-04	3e-04
Fine-tune Learning Rate	1e-04	1e-04
# Blocks	11	14
Hidden Size	30	30
Dropout	0	0
Weight Decay	0	0
Model size	138k	176k

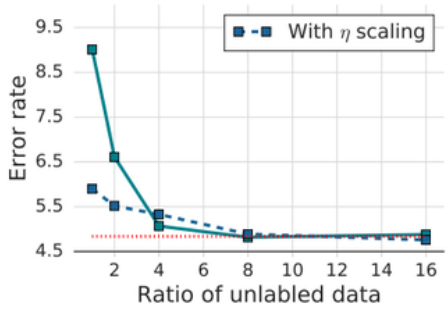

Figure 4.2: Effect of μ in FixMatch (Sohn et al., 2020). Note that we are only interested in the solid line.

4.3 Results

Table 4.4 shows the obtained classification results for the fully supervised models. We have also added M11-Nets (Dai et al., 2017) for a broader comparison. M11-Nets are very deep CNNs for raw audio waveforms that achieve near state-of-the-art results. It is important to note that the M11-Nets are non-causal CNNs, and thus not suitable for other time-series data. Nevertheless, it sheds more light on the performance of both the TCNs and CKCNNs. The TCNs significantly outperform both CKCNNs on both datasets, and just trail behind the M11-Nets. Interestingly, CKCNNs with Gabor filters outperform conventional CKCNNs on the SpeechCommands dataset, but not on the UrbanSound8K dataset. This may be due to the longer audio signals in UrbanSound8K. The CKCNNs especially lack behind in performance on the UrbanSound8K dataset. Besides having much more parameters than TCNs, M11-Nets uses multiple pooling layers, which could be an explanation to why they perform slightly better than TCNs.

Ablation study. In figure 4.3 we plot the obtained accuracy against varying ratios of unlabeled data in each minibatch for TCNs and CKCNNs on 10% of the labels of the SpeechCommands dataset. Unlike Sohn et al. (2020) (figure 4.2), we observe a decrease in performance when using more unlabeled data in a minibatch for TCNs, while CKCNNs do benefit from more unlabeled data. Figure 4.4 shows the effect of threshold parameter c in equation 3.6 on the final classification result on 10% of the labels of the SpeechCommands dataset. Interestingly, TCNs benefit from lower threshold values, and thus more but noisier data. Sohn et al. (2020) and Kim et al. (2021) find the opposite, their models benefit from less but qualitatively better data. CKCNNs, on the other hand, do not gain any significant performance when using different threshold values. The threshold value of 0.9 works well for both models, though increasing it to 0.95 hurts performance significantly. In table 4.5 we investigate the use of a cooldown scheme for the threshold parameter in equation 3.6 on 10% of the labels of the SpeechCommands dataset. The cooldown scheme starts at a threshold value of 0.9 and decreases to 0.1 within a certain number of epochs, referred to as steps in the table. Using a threshold cooldown scheme does not improve classification performance

Table 4.4: Obtained classification accuracy results fully supervised models. Best results are marked in bold letters.

Model	Param.	SpeechCommands	UrbanSound8K
TCN	134k	94.77	68.82
CKCNN	100k	71.66	40.50
Gabor CKCNN	105k	80.96	36.08
M11-Net* (Dai et al., 2017)	1.78m	**97.27**	**74.43**

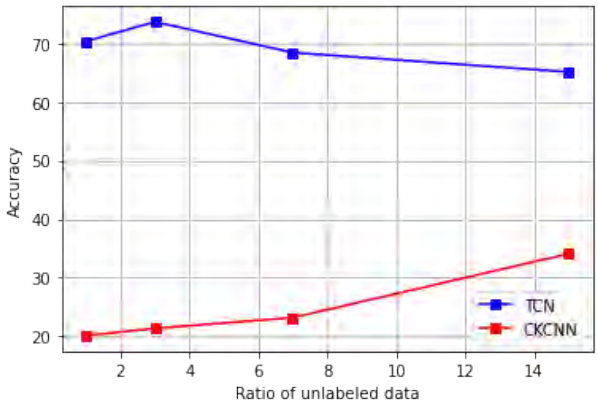

Figure 4.3: Plot of ablation study for the ratio of unlabeled data on 10% of the labels of SpeechCommands.

for both models. Unexpectedly, using a more moderate cooldown, by taking more steps, results in a worse accuracy.

Semi-supervised results. Table 4.6 shows the classification results of all networks trained using SelfMatch and supervised learning on the same amount of labels for both datasets and varying percentages of labeled data. M11-Nets significantly outperform all other models on both datasets, supervised M11-Nets trained on only 1% and 10% of the labels outperform TCNs and CKCNNs trained using SelfMatch on both datasets. Interestingly, M11-Nets trained using SelfMatch on SpeechCommands perform worse than supervised training on the same amount of labels. On the SpeechCommands dataset, using SelfMatch significantly improves classification results for TCNs on both 1% and 10% of the labels compared to supervised learning on the same amount of labels. This improved performance however, is nowhere near the performance of fully supervised learning on all labels (table 4.4). A similar observation can be made for TCNs on the UrbanSound8K dataset, the use of SelfMatch significantly improves performance when trained on 10% of the labels, but lacks in performance compared to fully supervised training. Oddly, training TCNs on only 1% of the labels of UrbanSound8K using SelfMatch results in a lower accuracy then supervised training on the same amount of labels. Both types of CKCNNs trained using SelfMatch show hardly any improvement over supervised training with the same amount of labels on the Speech-

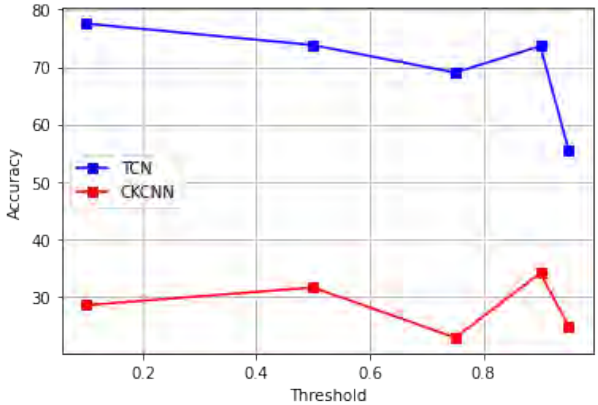

Figure 4.4: Plot of ablation study for the threshold parameter in the unsupervised loss term (equation 3.6) of the semi-supervised loss function on 10% of the labels of SpeechCommands.

Table 4.5: Ablation study on threshold cooldown for TCNs and CKCNNs on 10% of the SpeechCommands dataset.

Model	Steps	Accuracy
TCN	10	73.85
	50	60.55
CKCNN	10	26.91
	50	25.56

Commands dataset. On the UrbanSound8K dataset though, both CKCNN types trained on both methods approach the fully supervised performance with only 10% of the labels, with the Gabor CKCNN trained on SelfMatch coming especially close. Training M11-Nets using SelfMatch does result in significantly better performance, but still lacks in comparison to fully supervised training.

To put these results into perspective, Al-Tahan and Mohsenzadeh (2021) obtain an accuracy of 84% on the raw audio waveforms of SpeechCommands by combining contrastive self-supervised pre-training and supervised fine-tuning simultaneously during training on all labels. The TCN trained using SelfMatch on only 10% of the labels of SpeechCommands is able to obtain 77.58%, which is very respectable considering that the authors also used a much bigger ResNet-18 (He et al., 2016) architecture as encoder.

Discussion on the results. *Catastrophic forgetting* is a situation where in learning new tasks, the model may not use the shared parameters from the old task and "forget" them in the process (Goodfellow et al., 2013; McCloskey and Cohen, 1989). This phenomenon is especially problematic for self-supervised approaches, where self-supervised pre-training is often followed by (semi-) supervised fine-tuning. Chen et al. (2020) show that contrastive learning benefits from bigger models, which could partially be explained by this phenomenon. That is, bigger models have the capacity to maintain representations from both pre and post fine-tuning (Al-Tahan and Mohsenzadeh, 2021). This may explain why M11-Nets show less performance degradation than TCNs and CKCNNs in low label regiments. Both CKCNNs and TCNs may be suffering from catastrophic forgetting, indication for this is that convergence in the fine-tuning phase is slow compared to other contrastive learning approaches (Al-Tahan and Mohsenzadeh, 2021; Chen et al., 2020; Kim et al., 2021; Wang and Oord, 2021). These approaches show convergence within 10 epochs, while TCNs and especially CKCNNs take much longer (30+ epochs). Both TCNs and CKCNNs can be seen as small models, both have around 100k parameters, and are thus more

Table 4.6: Obtained classification accuracy results SelfMatch and fully supervised on the same amount of labels. Best results in each section are marked in bold letters. Note: the M11-Nets indicated with $*$ are non-causal.

			SpeechCommands		UrbanSound8K	
Method	Model	Param.	1% of labels	10% of labels	1% of labels	10% of labels
Supervised	TCN	134k	12.34	61.25	22.58	28.55
	CKCNN	100k	13.84	20.34	16.13	29.99
	Gabor CKCNN	105k	12.39	25.58	17.20	28.91
	M11-Net* (Dai et al., 2017)	1.78m	**78.76**	**92.56**	**27.72**	**51.61**
SelfMatch	TCN	138k	26.91	77.58	21.74	43.49
	CKCNN	104k	14.12	23.48	24.61	31.54
	Gabor CKCNN	109k	12.92	34.10	21.03	33.21
	M11-Net* (Dai et al., 2017)	1.81m	**72.19**	**91.92**	**42.77**	**62.84**

susceptible to catastrophic forgetting.

Al-Tahan and Mohsenzadeh (2021) show the impact of different augmentations and their sequential ordering on the quality of learned representations. Classification performance ranges from 50% to 89% depending on the combination and ordering of two augmentations on the raw audio signal, showing the great impact it has on final performance. The augmentations used in this work are based on augmentations from previous works on contrastive learning (Wang and Oord, 2021; Wen and Li, 2021). We hypothesise that the combination and ordering of augmentations used in this work are the biggest factor in the poor classification performance of SelfMatch. This is probably most evident from the performance of M11-Nets on the SpeechCommands dataset, where SelfMatch performs worse than supervised training on the same amount of labels. Unfortunately, due to time constraints, an ablation study on augmentations was not carried out. In the same paper, the authors find that when the amount of labeled data is decreased to only 1% in contrastive learning settings for audio, the performance of learned representations degrades and classification results suffer. They argue that this could be the result of overfitting more to the labeled data resulting in less efficient representations. This phenomenon could be an explanation for the poor performance of the TCNs and CKCNNs. Especially for CKCNNs, which have extremely large filters and are thus more sensitive to overfitting, this may effect performance on settings with larger percentages of labeled data as well. As shown by Chen et al. (2020), batch size has a significant impact on final classification performance of contrastive learning. In this work, we are limited to using a batch size of 16 for SpeechCommands and 8 for UrbanSound8K, mainly due to the large computational cost of modelling raw audio waveforms and lack of computational resources. Compared to other works on contrastive learning for audio (Al-Tahan and Mohsenzadeh, 2021; Wang and Oord, 2021), this batch size is smaller by a factor of 32 and undoubtedly has a negative effect on performance.

Chapter 5

Discussion

In this chapter we discuss the limitations of this research. More specifically, we utilize results of similar works to highlight these limitations.

The effect of augmentations. In the previous chapter we hypothesise that the biggest factor in the classification gap between SelfMatch and fully supervised learning are the augmentations. In this section we look at the work of Al-Tahan and Mohsenzadeh (2021) to support this claim. In their work, they show the impact that the combination and ordering of augmentations have on classification performance in contrastive learning for audio. Figure 5.1 shows the test performance of a ResNet-18 model on raw audio waveforms (1D) and MFCC features (2D) on the SpeechCommands dataset for different augmentations. The diagonal line represents the performance of single augmentation, while other entries represent the performance of paired augmentations. Each row indicates the first augmentation and each column shows the second augmentation applied sequentially. The last column and row in each matrix represents the averaged predictive performance of a specific augmentation. The last column depicts the average when the augmentation was applied first, while the last row shows the average when the corresponding augmentation was applied second. The bottom right element represents the average of the whole matrix. Classification performance varies significantly for different combinations and even ordering of the same augmentations. Due to time constraints, it was not possible to perform a similar ablation study on augmentations. We believe that a different combination and ordering of augmentations could lead to a large improvement in classification performance of all networks on both datasets.

The effect of batch size. Due to the large computational cost of modelling raw audio waveforms, we are limited to using a batch size of 16 on SpeechCommands and 8 on UrbanSound8K. This is smaller by a factor of 32 compared to other contrastive learning approaches for audio (Al-Tahan and Mohsenzadeh, 2021; Wang and Oord, 2021), and even smaller compared to contrastive learning for 2D data (Chen et al., 2020; Kim et al., 2021). Chen et al. (2020) find that larger batch sizes, when the number of training epochs is small, have a significant advantage over smaller ones, this can be seen in figure 5.2. Larger batch sizes provide more negative samples, which speed up convergence. On 100 training epochs for example, similar to what we are using, doubling the batch size of 256 improves accuracy by around 4%. With more training epochs, the gaps between different batch sizes decrease or disappear. Unfortunately, due to the lack of computational resources, it was not possible to increase the number of training epochs for our experiments. We believe that increasing the batch size leads to an improvement in classification performance, although we expect that this improvement is not as drastic as on different augmentations.

Computational resources. For this research, we had access to the computational infrastructure DAS-5 (Bal et al., 2016). DAS-5 is a widely used infrastructure within the Vrije Universiteit and consists of roughly 200 compute nodes with multiple GPU cards. Due to the public nature of DAS-5, it was not possible to run every experiment on the same GPU card, which may have a

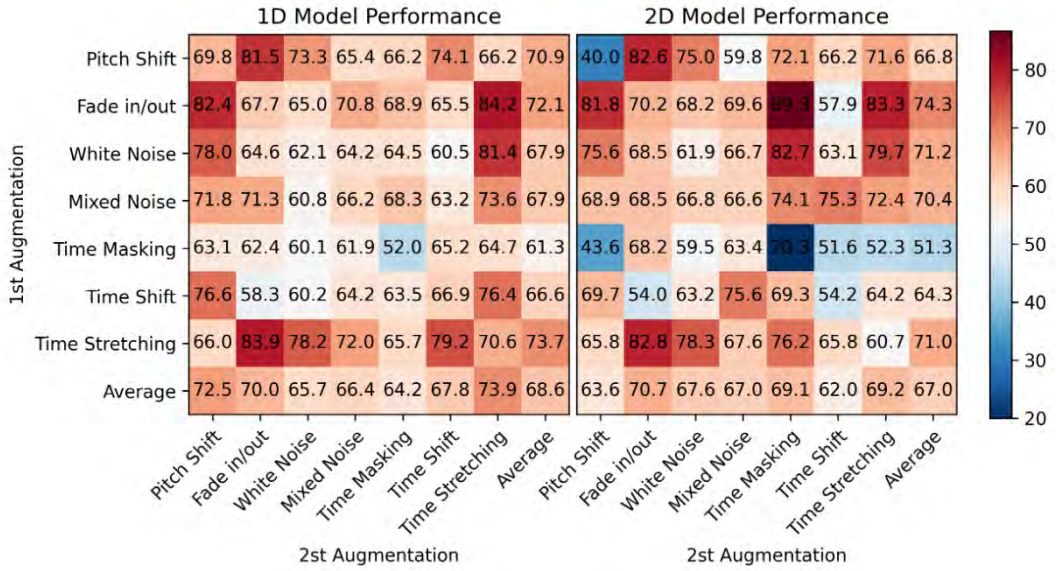

Figure 5.1: Ablation study on augmentations in contrastive learning for audio on SpeechCommands (Al-Tahan and Mohsenzadeh, 2021).

negative impact on reproducibility. Due to the large computational cost of modelling raw audio waveforms, we were limited to using three compute nodes on DAS-5 that had enough memory. Unfortunately, these compute nodes were popular among other users, meaning that we were somewhat limited in running large amounts or extremely long experiments.

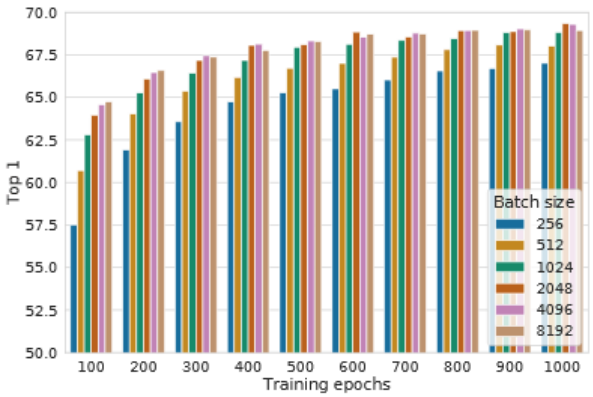

Figure 5.2: Ablation study on the effect of batch size and the number of epochs on classification performance of SimCLR on ImageNet (Chen et al., 2020).

Chapter 6

Conclusion

We conclude this report with a summary and directions for future work.

6.1 Summary

In this work we introduced a semi-supervised end-to-end learning approach for raw audio waveforms based on SelfMatch. SelfMatch combines the power of contrastive self-supervised learning and augmentation consistency regularization in two stages. The first stage is self-supervised pre-training based on contrastive learning and the second is semi-supervised fine-tuning based on augmentation consistency regularization. SelfMatch adopts SimCLR as contrastive self-supervised learning method, and aims to learn representations by encouraging two views of the same sample to be similar, and two views from different samples to be dissimilar. Fine-tuning in SelfMatch is based on FixMatch, which encourages a consistent prediction between weakly and strongly augmented views of the same sample. More specifically, FixMatch uses the prediction of the weakly-augmented view as pseudo-label for the strongly-augmented view. Augmentations are essential in both stages. SelfMatch was originally designed for 2D image data, so in order to use SelfMatch for raw audio waveforms, specific augmentations for raw audio waveforms are needed. We propose to use *audio mixing*, *random mask*, and *shift time* augmentations in the pre-training stage and as weak augmentations in the fine-tuning stage. For strong augmentations we propose to sequentially apply two randomly chosen weak augmentations to the same sample.

The main goal of this report is to investigate semi-supervised learning for sequential data using CKConvs. Formulating the convolutional kernel as a continuous function, parametrized by a small neural network, avoids drawbacks found in conventional neural architectures. The CKConv takes a set of relative positions as input and outputs the value of the convolutional kernel at those positions. A global memory horizon can thus be constructed within a single operation and without modifying the architecture of the network or adding more parameters. To the best of our knowledge, there are currently no implementations for SelfMatch for other data types than 2D image data. So, to create a baseline result for raw audio waveforms, we train TCNs using SelfMatch. Additionally to TCNs, we also introduce CKCNNs with Gabor Layers, which aim to increase robustness of learned feature representations and reduce training complexity.

In our experiments we show that TCNs and CKCNNs trained using SelfMatch on only 1% and 10% of the labels outperform supervised learning on the same amount of labels. Both networks however, are not able to obtain their fully supervised accuracy, although the Gabor CKCNNs come close on the UrbanSound8k dataset. Interestingly, supervised M11-Nets on 1% and 10% of the labels outperform M11-Nets trained using SelfMatch. M11-Nets trained using SelfMatch do however show a significant improvement, but lack in performance compared to the fully supervised baseline. Despite improving performance over supervised learning on the same amount of labels,

both types of CKCNNs lack in performance on the SpeechCommands dataset compared to TCNs and the fully supervised baseline. This gap decreases on the UrbanSound8K dataset however, and CKCNNs even outperform TCNs on only 1% of the data. Comparing our results to those of other contrastive learning approaches for raw audio waveforms however, shows that TCNs trained using SelfMatch on 10% perform quite well.

We hypothesise that this poor performance may have several causes. Other works on contrastive learning show the impact of different combinations and ordering of augmentations. In this work we consider three augmentations, of which we randomly choose two. These random combinations and ordering of augmentations might be sub-optimal and even hurt performance. Another plausible cause is catastrophic forgetting, a phenomenon where a model forgets parameters from the old task in learning a new task. Catastrophic forgetting is especially prominent in self-supervised learning and for smaller models. This partially explains why previous works on contrastive learning find that bigger models work better. Indication that our models suffer from catastrophic forgetting is slow convergence compared to other contrastive learning implementations. Another possible cause is overfitting to the labeled data resulting in less efficient representations. This could be especially problematic for CKCNNs, which have large filters and are thus more prone to overfitting. Due to the large computational cost of modelling raw audio waveforms and lack of computational resources, we are limited to a batch size of 16 on SpeechCommands and 8 on UrbanSound8K. This is smaller by a factor of 32 compared to other contrastive learning approaches for audio. As contrastive learning benefits from larger batch sizes, this limitation on the batch size undoubtedly hurts performance.

6.2 Future work

Due to time constraints, it was not possible for us to test a larger scope of experiments. In this section we summarise some exciting ideas and research directions in which this work could be extended.

- An ablation study of augmentations in both the pre-training and fine-tuning stage. As mentioned in chapter 4.3, augmentations have a significant impact on the quality of learned representations and final classification performance. We believe that a different combination and ordering of augmentations could lead to a large improvement.
- Extending the framework proposed in this work to other sequential data types. Testing SelfMatch on sequential data such as video and other time-series would undoubtedly lead to interesting results.
- Romero et al. (2021a) propose an improvement of the CKConv, called the FlexConv. The FlexConv is a convolutional operation able to learn its kernel size during training using a fixed parameter budget. Investigating how FlexConvs perform in this semi-supervised framework is another interesting research direction.

References

H. Aapo, J. Karhunen, and E. Oja. *Independent component analysis*. Wiley, 2001. 3

H. Al-Tahan and Y. Mohsenzadeh. Clar: Contrastive learning of auditory representations. In *International Conference on Artificial Intelligence and Statistics*, pages 2530–2538. PMLR, 2021. ix, 1, 2, 22, 23, 25, 26

E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020. 5

P. Bachman, O. Alsharif, and D. Precup. Learning with pseudo-ensembles. *arXiv preprint arXiv:1412.4864*, 2014. 5, 11

P. Bachman, R. D. Hjelm, and W. Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019. 9

S. Bai, J. Z. Kolter, and V. Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018a. 1, 6, 12

S. Bai, J. Z. Kolter, and V. Koltun. Trellis networks for sequence modeling. *arXiv preprint arXiv:1810.06682*, 2018b. 7

H. Bal, D. Epema, C. de Laat, R. van Nieuwpoort, J. Romein, F. Seinstra, C. Snoek, and H. Wijshoff. A medium-scale distributed system for computer science research: Infrastructure for the long term. *Computer*, 49(05):54–63, may 2016. ISSN 1558-0814. doi: 10.1109/MC.2016.127. 18, 25

M.-F. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. 01 2004. 5

F. Beaufays. The neural networks behind google voice transcription. *Google AI Blog*, Aug 2015. URL `https://ai.googleblog.com/2015/08/the-neural-networks-behind-google-voice.html`. 6

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994. 1

D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019a. 5, 9, 11

D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. Mixmatch: A holistic approach to semi-supervised learning. *arXiv preprint arXiv:1905.02249*, 2019b. 5, 11

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. *Proceedings of the Annual ACM Conference on Computational Learning Theory*, 10 2000. doi: 10.1145/279943.279962. 5

T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby. Self-supervised gans via auxiliary rotation loss. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12154–12163, 2019. 3

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. ix, ix, 3, 9, 10, 18, 22, 23, 25, 26

P.-H. Chi, P.-H. Chung, T.-H. Wu, C.-C. Hsieh, Y.-H. Chen, S.-W. Li, and H.-y. Lee. Audio albert: A lite bert for self-supervised learning of audio representation. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 344–350. IEEE, 2021. 3

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 6

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 6

Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass. An unsupervised autoregressive model for speech representation learning. *arXiv preprint arXiv:1904.03240*, 2019. 4

D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep big simple neural nets excel on hand-written digit recognition. *CoRR abs*, 1003, 2010. 5

A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*, 2016. 1

Z. Cui, W. Chen, and Y. Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016. 6

W. Dai, C. Dai, S. Qu, J. Li, and S. Das. Very deep convolutional neural networks for raw waveforms. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 421–425. IEEE, 2017. 2, 17, 20, 22

Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019. 7

G. Dattatreya and L. Kanal. Adaptive pattern recognition with random costs and its application to decision trees. *Systems, Man and Cybernetics, IEEE Transactions on*, 16:208 – 218, 04 1986. doi: 10.1109/TSMC.1986.4308941. 4

Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941. PMLR, 2017. 6

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3

C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 3

A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2015. 4

J. Du, C. X. Ling, and Z.-H. Zhou. When does cotraining work in real data? *IEEE Transactions on Knowledge and Data Engineering*, 23(5):788–799, 2011. doi: 10.1109/TKDE.2010.158. 5

R. Fathony, A. K. Sahu, D. Willmott, and J. Z. Kolter. Multiplicative filter networks. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=OmtmcPkkhT`. 15

D. Gabor. Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering*, 93:429–441, 1946. 14

F. Gers and J. Schmidhuber. Recurrent nets that time and count. volume 3, pages 189 – 194 vol.3, 02 2000. ISBN 0-7695-0619-4. doi: 10.1109/IJCNN.2000.861302. 6

F. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12:2451–71, 10 2000. doi: 10.1162/089976600300015015. 6

I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013. 22

A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks : the official journal of the International Neural Network Society*, 18:602–10, 07 2005. doi: 10.1016/j.neunet.2005.06.042. 6

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 3

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 10, 22

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 3, 9

O. Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR, 2020. 3

S. Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1), 1991. 1, 6

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 6

Y. Hoshen, R. J. Weiss, and K. W. Wilson. Speech acoustic modeling from raw multichannel waveforms. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4624–4628, 2015. doi: 10.1109/ICASSP.2015.7178847. 2

L.-L. Huang, A. Shimizu, and H. Kobatake. Classification-based face detection using gabor filter features. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, pages 397–402, 2004. doi: 10.1109/AFGR.2004.1301565. 14

M. Ishii. Semi-supervised learning by selective training with pseudo labels via confidence estimation. *arXiv preprint arXiv:2103.08193*, 2021. 5

X. Ji, J. F. Henriques, and A. Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9865–9874, 2019. 4

D. Jiang, X. Lei, W. Li, N. Luo, Y. Hu, W. Zou, and X. Li. Improving transformer-based speech recognition using unsupervised pre-training. *arXiv preprint arXiv:1910.09932*, 2019. 4

I. Jolliffe. *Principal Component Analysis*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-04898-2. doi: 10.1007/978-3-642-04898-2_455. URL https://doi.org/10.1007/978-3-642-04898-2_455. 3

P. Kidger, J. Morrill, J. Foster, and T. Lyons. Neural controlled differential equations for irregular time series. *arXiv preprint arXiv:2005.08926*, 2020. 17

B. Kim, J. Choo, Y.-D. Kwon, S. Joe, S. Min, and Y. Gwon. Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. *arXiv preprint arXiv:2101.06480*, 2021. ix, 2, 9, 10, 18, 20, 22, 25

S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 5

C. Lea, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks: A unified approach to action segmentation. In *European Conference on Computer Vision*, pages 47–54. Springer, 2016. 6

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 6

D.-H. Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013. 4

J. Lee, J. Park, K. L. Kim, and J. Nam. Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms. *arXiv preprint arXiv:1703.01789*, 2017. 17

S. Li, Y. Yao, J. Hu, G. Liu, X. Yao, and J. Hu. An ensemble stacked convolutional neural network model for environmental event sound recognition. *Applied Sciences*, 8:1152, 07 2018. doi: 10.3390/app8071152. 1, 2

A. T. Liu, S.-w. Yang, P.-H. Chi, P.-c. Hsu, and H.-y. Lee. Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6419–6423. IEEE, 2020. 4

M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989. doi: https://doi.org/10.1016/S0079-7421(08)60536-8. URL https://www.sciencedirect.com/science/article/pii/S0079742108605368. 22

D. McClosky, E. Charniak, and M. Johnson. Effective self-training for parsing. pages 152–159, 06 2006. doi: 10.3115/1220835.1220855. 4

G. J. McLachlan. Iterative reclassification procedure for constructing an asymptotically optimal rule of allocation in discriminant analysis. *Journal of the American Statistical Association*, 70 (350):365–369, 1975. doi: 10.1080/01621459.1975.10479874. 11

T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018. 5

M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *European conference on computer vision*, pages 69–84, 2016. 3

M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9359–9367, 2018. 3

A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I. J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. *arXiv preprint arXiv:1804.09170*, 2018. 5

A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016. ix, ix, 1, 6, 12, 13

A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 4

M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018. 3, 4

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018. 3

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019. 3

A. Rasmus, H. Valpola, M. Honkala, M. Berglund, and T. Raiko. Semi-supervised learning with ladder networks. *arXiv preprint arXiv:1507.02672*, 2015. 5

D. W. Romero, E. J. Bekkers, J. M. Tomczak, and M. Hoogendoorn. Wavelet networks: Scale equivariant learning from raw waveforms. *arXiv preprint arXiv:2006.05259*, 2020. 1, 2

D. W. Romero, R.-J. Bruintjes, E. J. Bekkers, J. M. Tomczak, M. Hoogendoorn, and J. van Gemert. Flexconv: Continuous kernel convolutions with differentiable kernel sizes. *arXiv preprint arXiv:XXXX.XXXXX*, 2021a. 14, 15, 28

D. W. Romero, A. Kuzinna, E. J. Bekkers, J. M. Tomczak, and M. Hoogendoorn. Ckconv: Continuous kernel convolutions for sequential data. *arXiv preprint arXiv:2102.02611*, 2021b. ix, 1, 13, 14, 17

C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. pages 29–36, 01 2005. doi: 10.1109/ACVMOT.2005.107. 4

J. Salamon, C. Jacoby, and J. P. Bello. A dataset and taxonomy for urban sound research. In *22nd ACM International Conference on Multimedia (ACM-MM'14)*, pages 1041–1044, Orlando, FL, USA, Nov. 2014. 17

P. Simard, D. Steinkraus, and J. Platt. Best practices for convolutional neural networks applied to visual document analysis. pages 958–962, 01 2003. doi: 10.1109/ICDAR.2003.1227801. 5

V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020. 14

K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020. ix, 5, 9, 11, 18, 20

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 10

A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv preprint arXiv:1703.01780*, 2017. 5, 11

Y. Tokozume and T. Harada. Learning environmental sounds with end-to-end convolutional neural network. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2721–2725, 2017. doi: 10.1109/ICASSP.2017.7952651. 2

J. E. Van Engelen and H. H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109 (2):373–440, 2020. 2, 4, 5

L. Wang and A. v. d. Oord. Multi-format contrastive learning of audio representations. *arXiv preprint arXiv:2103.06508*, 2021. 2, 10, 22, 23, 25

P. Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018. 17

Z. Wen and Y. Li. Toward understanding the feature learning process of self-supervised contrastive learning. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11112–11122. PMLR, 18–24 Jul 2021. URL `http://proceedings.mlr.press/v139/wen21c.html`. 10, 23

R. J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent. *Backpropagation: Theory, architectures, and applications*, 433:17, 1995. 1

Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018. 4

J. Yan, L. Mu, L. Wang, R. Ranjan, and A. Y. Zomaya. Temporal convolutional networks for the advance prediction of enso. *Scientific reports*, 10(1):1–15, 2020. 12

M. Ye, X. Zhang, P. C. Yuen, and S.-F. Chang. Unsupervised embedding learning via invariant and spreading instance feature. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6210–6219, 2019. 4

R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. *European conference on computer vision*, pages 649–666, 2016. 3

X. J. Zhu. Semi-supervised learning literature survey. 2005. 4, 5

C. Zhuang, A. L. Zhai, and D. Yamins. Local aggregation for unsupervised learning of visual embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6002–6012, 2019. 4