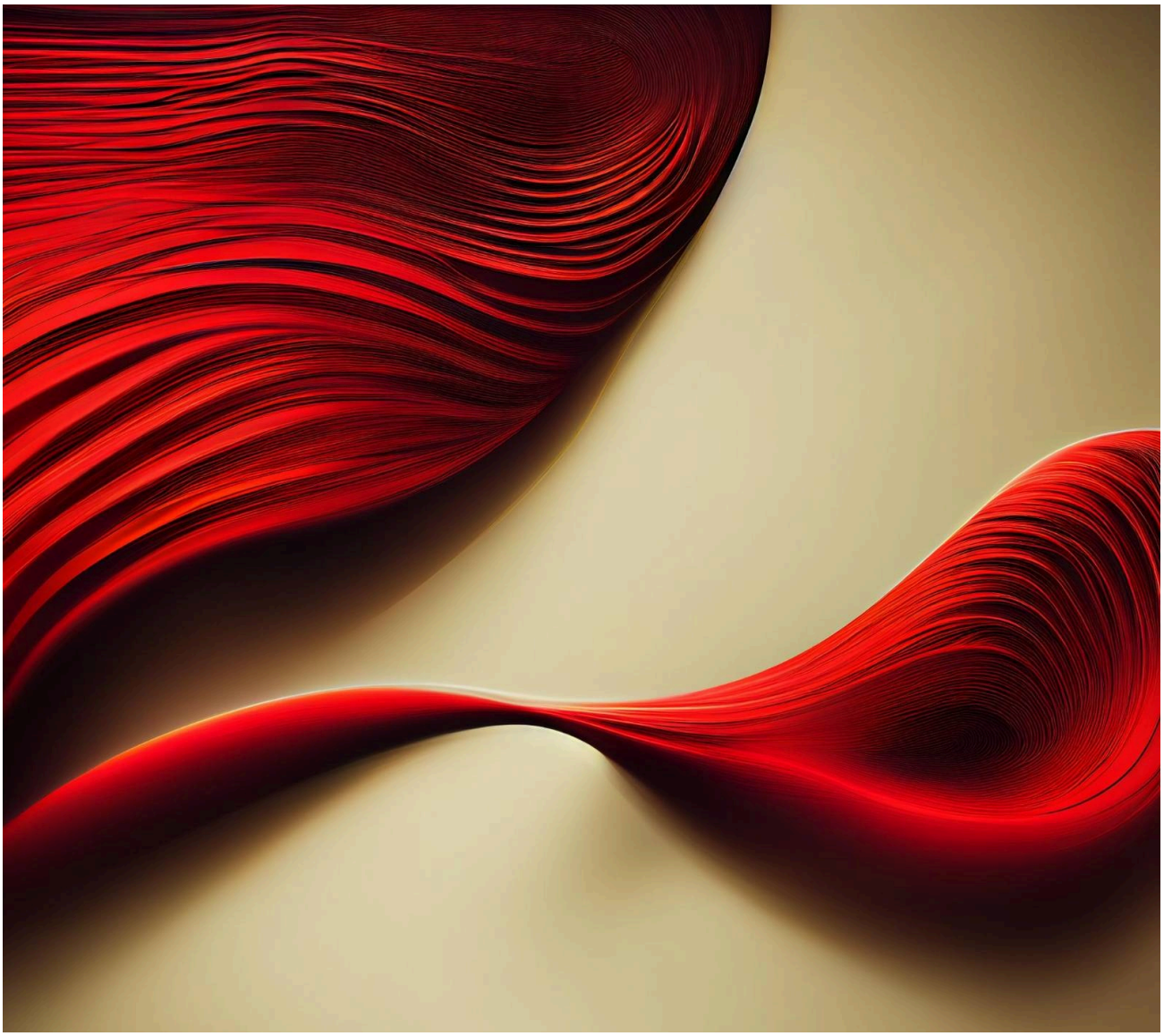


Master Thesis | December 2022

Self-supervised representation learning from electrocardiogram data for medical applications.

Matilda Andersson

LUND
UNIVERSITY

Centre for Mathematical Sciences
Lund University
Box 118
SE-221 00 LUND
Sweden

Lund 2022

Abstract

Cardiovascular diseases are the leading cause of death worldwide, increasing yearly. However, many abnormalities in heart cycles can be discovered and treated years before the onset of diseases. But in most societies, regular health checkups are a concept reserved for cars, not humans. In order to save lives, our healthcare systems must adopt a preventative rather than a reactive approach. To that end, there have been several attempts to produce automated ECG-based heartbeat classification methods over the last few decades. But their performance is hindered by limited access to high-quality labeled data, restricting their usage to secondary diagnostic purposes. In this regard, a self-supervised learning framework could provide a viable solution, as it decouples deep learning progress from the dependence on large volumes of annotated data, and instead uses unlabelled samples. In this thesis, we present an assessment of self-supervised representation learning on 12-lead clinical ECG data to examine whether self-supervised learning methods can be applied to electrocardiogram signals to produce meaningful feature representations from only unlabelled data. We implement the self-supervised learning methods SimCLR, BYOL, and VICReg and compare their performances to a supervised learning method. In doing so, we find that self-supervised learning produces meaningful representations of ECG signals. When following each method's recommended implementation protocol, the performance equals those of a conventional supervised model, initially suggesting that self-supervised pre-training offers no additional benefits to downstream tasks. However, by increasing the length of the ECG signal and adjusting the data augmentation strategy, self-supervised pre-trained models outperformed their supervised counterparts in all evaluation settings. In light of our experiments, we find that a suitable augmentation protocol is crucial for high downstream classification performance.

Acknowledgements

First, immense thanks to Douglas Adams, in my opinion, one of the greatest minds that humanity has ever produced. Writing the Hitchhikers guide to the galaxy, he showed us that there is no need to limit our thinking to categories, dimensions, or norms. His words describe so well how I feel about my journey up until today and how I hope my story will continue to play out:

"I may not have gone where I intended to go, but I think I have ended up where I needed to be."

My dear mother and father, thank you for your DNA; the code that laid the foundation for what I am. But more importantly, thank you for all the love you have shown, and all you have given that has made me into what I am. My brother Erik, thank you for always being yourself and meeting me in laughter, fantasies, playfulness, and dreaming without limits.

My fantastic grandparents. You have all been a steady rock in my childhood. You sowed a seed of strength in me and the foundation of responsibility for being put here on earth. To my cousins, aunts, and uncles. Thank you for providing me with a sense of family and community that has been paramount in my life.

To my whole family, I love you all.

Thank you, Italy, Bali, China, South Africa, Australia, Germany, and all the other countries where I have lived, the cultures and people that have reshaped my mind over and over again.

John Cleese, Michael Palin, Graham Chapman, Terry Gilliam, Eric Idle, Terry Jones. Thank you for your geniuses.

Vanja, Alexandra, and Katarina from Pink Programming. Words are not enough to express my gratitude to you. You introduced me to a new world where I now know I belong, which I love to explore and fly higher and higher. You are amazing.

Penguins. Dogs. Evolution. Robots. Thank you for that.

Thank you to everyone I have met along the way who, when I have shared my dreams, thoughts, and ideas, have shown interest and given me encouragement to go for it. Your openness and support have given me strength, and for that, I am eternally grateful.

Hazelnuts. Fuck you.

Maria, Morten, Luna, and Freja. You are joy and warmth in human form. My chosen family. Thank you for always being there, and embracing me with an open mind, without ever demanding anything in return. I love you.

Marvin, thank you for always cheering me up.

Thank you to all my classmates and teachers at LTH for providing me with a wealth of knowledge.

Elon Musk, thank you for pushing the envelope and showing the human race what is possible. As well as what we should not do.

Kpe, thank you for all our hectic discussions and intellectual waltzing. I love every minute of it. Thank you for expanding my horizons and sharing your wisdom.

To the brilliant minds of Yann LeCun, the team at MLST, Lex Fridman, Andrej Karpathy, Geoffrey Hinton, Yoshua Bengio, Andrew Ng, and Ilya Sutskever (and many more whose contributions should have been noted). Thank you for continuously stimulating my intellect and feeding my curiosity about deep learning so that I can keep advancing my knowledge and pushing my boundaries.

42. Thank you too.

Deep thanks to Kalle Åström and Gabrielle Flood at LTH for supervising this work. Your knowledge and support made this thesis possible. Andreas Jakobsson, I dedicate the section on statistical significance to you.

To all my fantastic colleagues at Cur8, thank you so much for giving me a place where I feel at home, where thoughts, ideas, and perspectives are allowed to fly, and crash, and where I absolutely love to be.

Mattias Nilsson, thank you for all your support, for sharing your mind with me, and for encouraging me every day. You are a true mensch. I absolutely love working with you and you make every day a Carpe Diem day.

To my Keplarian Kye. This planet is not big enough to contain all the life force you have within. In you, I will always see a universe of possibilities and a light that will never be extinguished. As you journey through new galaxies, within yourself and out in the cosmos, always remember that I will be soaring alongside you.

To future me. Thank you for never stopping.

Contents

1

Introduction

1.1 Background

Cardiovascular disease is the leading cause of death worldwide, killing 17.9 million people each year [WHO, 2021]. The term cardiovascular disease refers to a group of diseases affecting the blood vessels and the heart, including a variety of conditions such as stroke, heart failure, hypertensive heart disease, rheumatic heart disease, cardiomyopathy, abnormal heart rhythms, and others [Mendis et al., 2011]. Eight out of ten deaths are caused by heart attacks and strokes and one-third of these deaths occur prematurely among people under the age of 70 [WHO, 2021]. Thus, it is crucial to detect cardiovascular disease at an early stage to ensure appropriate counseling and treatment – and to prevent premature death.

Unfortunately, our current healthcare system is not equipped to provide the preventive healthcare that is needed. With the non-availability of medical diagnosing tools and limited access to medical experts, early diagnosis and vital treatments remain absent. As many abnormalities are detectable years before the onset of symptomatic cardiovascular events, implementing a widespread and cost-effective screening and disease detection program could prevent a number of premature deaths.

There are a number of industries and fields in which the application of machine learning has proven to be a successful solution to a wide range of problems. Machine learning-based systems have for example increased machine longevity and operational efficiency in factories by monitoring and predicting machine failures. The reason for this is that machine learning systems are capable of analyzing a large amount of data and detecting minuscule patterns that are largely unrecognizable by humans. The use of machine learning could perhaps contribute to the development of similar medical systems. These systems, instead of predicting the failure of machines, would predict the failure of the human body.

In fact, various medical fields are benefiting from machine learning, and early detection of diseases, as well as individualized treatment plans, are now

possible [Muhammad et al., 2020a]. Additionally, machine learning-assisted healthcare systems may be able to meet the growing demand for affordable, high-quality cardiac screenings, enabling improved and enhanced detection of cardiovascular diseases.

Increasing the prospects for the development of early detection systems, electrocardiograms, or ECGs for short, have become increasingly used in cardiovascular screening. Due to its simple and noninvasive nature, it enables the detection of many cardiovascular abnormalities of the heart rhythm as it records and analyzes the electrical signal of each heartbeat. Each heartbeat is seen as a cardiac action potential waveform produced by cardiac cells during the contraction and relaxation of the heart. Section 3.1 will provide a more detailed description of electrocardiograms. The last couple of decades have brought with them several attempts to produce automatic ECG-based heartbeat classification methods [Siontis et al., 2021a; Ribeiro et al., 2020a]. However, limited access to high-quality labeled data hinders their performance and restricts their usage to secondary diagnostic purposes. By combining the latest development of deep learning with the availability of electrocardiograms, there is hope for efficient and accurate detection systems to be developed.

Situation at present

As a result of deep learning, medical systems have improved considerably in recent years. These models have learned to recognize patterns in large amounts of carefully labelled data, and they have a proven track record of performing extremely well on the task for which they have been trained. The unprecedented progress in artificial intelligence, and especially deep learning, has demonstrated the immense potential for automated, algorithmic detection and diagnosis of cardiovascular diseases. However, supervised learning alone is not sufficient to advance machine-aided systems in this field. These deep learning systems are challenging to train due to their inherent reliance on large amounts of annotated data. The low availability of datasets with high-quality labels is an omnipresent challenge in machine learning in general but is especially pressing in the health domain. The medical labeling process is particularly expensive and clinical ground truth is many times hard to define. In addition, systems are further compromised by mismatches in distribution due to differences in equipment, data collection procedures, and local population and application environments.

In order to decouple deep learning progress from its dependence on large amounts of annotated data, novel learning frameworks, such as self-supervised learning, are being explored in current research. Self-supervised learning is presented as a method that allows systems to "learn" using unlabelled data but is applied in settings where a supervised model normally would be used. In this respect, self-supervised learning may be a viable solution to the issue

of data labeling as it permits a label-efficient approach to training [Spathis et al., 2022].

1.2 Related Research

In this thesis, the focus is on self-supervised representation learning of electrocardiogram signals. The following is a summary of related research that has played an important role in the progress of this field.

Self-supervised learning

Many fields, including computer vision, natural language processing, and speech processing, have in the past decades been able to witness a revolutionary development of specialized deep learning systems. However, as the success of these learning models greatly depends on massive amounts of carefully labelled training data, advancements in certain domains are suppressed due to the scarcity of qualitative data. Within the last years, even fields flourishing with millions of publicly available data samples have seen progress hindered by data shortages. This is partly due to the cumbersome data labeling process, and labeling everything in the world is as unlikely as walking on the surface of a black hole. To overcome this supervised learning bottleneck and to get around the need for data labels, recent research is proposing a novel learning framework. In this framework, deep learning systems obtain supervisory signals directly from the data itself, in contrast to the previously used human annotations. Known as self-supervised learning [LeCun and Misra, 2021], it was first introduced to the field of natural language processing, where self-supervised trained models such as BERT, RoBERTa, and XLM-R [Devlin et al., 2018; Liu et al., 2019; Conneau et al., 2019] entailed significant performance improvements without the increasing need for labelled data.

The domain of computer vision was not far behind in adopting this novel approach of using self-defined pseudo labels as supervision for learning general representations applicable to several deep learning tasks. Specifically, contrastive learning has recently become a dominant component in computer vision with self-supervised learning methods such as PIRL, SwAV, SimCLR, MoCo and CPC [Misra and Maaten, 2020; Caron et al., 2020; Chen et al., 2020; He et al., 2020; Oord et al., 2018] being developed. This discriminative approach aims at grouping feature representations of similar samples close together in a latent space while diverse sample representations are placed far from each other. To achieve this, the approaches require negative, nonsimilar, samples to be explicitly defined – a task which is not always feasible. SimCLR learns representations by maximizing agreement between differently augmented views of the same data example via a contrastive loss in the latent space. Other methods, like MoCo, utilize dynamic dictionaries of negative

sample representations. Non-contrastive approaches include methods such as BYOL, SiamSiam, Barlow Twins and VICReg [Grill et al., 2020; Chen and He, 2021; Zbontar et al., 2021; Bardes et al., 2021]. Instead of explicitly defining negative samples, they introduce asymmetry in the network architecture to solve the representation problem. BYOL uses two neural networks to learn, the online and the target networks, while Barlow Twins and VICReg introduce instance contrasting in their respective loss functions.

Machine learning in cardiovascular disease detection

The application of machine learning to the electrocardiogram is an example of the ongoing transformative effect of artificial intelligence in cardiovascular medicine. ECG signals can now be interpreted by machines with high accuracy thanks to deep learning. With signals and patterns largely unrecognizable to human interpreters being detectable by deep learning networks, the ECG becomes a powerful, non-invasive tool for disease detection [Raghunath et al., 2020]. State-of-the-art solutions mainly follow convolutional neural network based architectures trained in a supervised manner [Li et al., 2020; Muhammad et al., 2020b; He, 2020; Siontis et al., 2021b] and offer valuable insights into cardiovascular health and disease detection. However, interpretation and annotation of ECG signals require considerable human expertise, resulting in costly, time-consuming, and at times, even infeasible labeling procedures. Current systems are highly dependent on large amounts of labelled data and further development of clinical AI-based ECG disease detection systems is hindered by its scarcity.

Self-supervised representation learning in cardiovascular disease detection

Given recent achievements of self-supervised learning in other fields, present research in the domain of cardiovascular disease detection aims to apply self-supervised learning methods to decouple system performance from the need for excessive amounts of labelled data. Mehari and Strodthoff [Mehari and Strodthoff, 2022] apply a selection of self-learning methods to 12-lead ECG data and evaluate their representational performance in a multi-label classification task setting. They find an adjusted version of the contrastive predictive coding, CPC [Oord et al., 2018], and the SimCLR approach to show the highest performance results. Research presented by [Kiyasseh et al., 2021] also considers BYOL and SimCLR approaches for ECG representation learning, but used a very shallow encoder network architecture with only five convolutional layers. Conclusively, work by [Liu et al., 2021] highlights the potential for using self-supervised learning methods in ECG representation learning by presenting a careful comparison of the effects of different self-supervised learning methods on linear evaluation and fine-tuning evaluation. They con-

clude that self-supervised learning methods relying on negative sample pairs are able to achieve excellent results with only a small number of labelled data samples, allowing for these methods to approach the disease detection level of human experts with a greatly reduced need for labelled samples.

1.3 Aim of this thesis

This thesis examines the potential for self-supervised representation learning of electrocardiogram signals for the detection of cardiac diseases. As a result, the objective of this thesis is to increase and improve the understanding and knowledge of self-supervised learning methods within the medical field. More specifically, this thesis will investigate whether self-supervised learning methods can be applied to electrocardiogram signals to learn meaningful feature representations from a large set of unlabelled data that later helps boost the performance of supervised models on downstream tasks with only a few labelled examples. Building upon previous research, the instance-based self-supervised learning methods SimCLR, BYOL, and VICReg are adapted and directly compared. SimCLR and BYOL are investigated due to their demonstrated potential in the work conducted by [Mehari and Strodthoff, 2022] and [Liu et al., 2021], while the exploration of VICReg is motivated by its novelty, simplicity, and theoretical transparency.

Research questions

Based on the thesis objective and the current status of research on self-supervised representation learning for cardiovascular health assessments, two research questions are formulated. Following the introductory chapter, the remaining chapters will examine, discuss, and attempt to answer the following questions:

- Can self-supervised learning methods be utilized to create ECG signal representations relevant to clinical downstream tasks?
- Can we improve upon previous self-supervised learning approaches to present novel methods in the domain of medical ECG signal representation learning?

Embarking on a quest to answer our research questions, we seek to advance the understanding of self-supervised representation learning in the context of cardiovascular anomaly detection.

Delimitations

With the previous research questions in mind, it is worth exploring the delimitating choices made for the study at hand. In light of the results of recent

and related research, this thesis concentrates on the branch of self-supervised learning related to instance-based pretext tasks, as it has shown the most promising results. Consequently, it will not investigate the performance of pretext tasks based on methods such as clustering-, generative-, or adversarial-based methods. More specifically, the following methods are those which will be implemented and evaluated: SimCLR, BYOL, and VICReg. Architectural constraints are imposed on the encoder network as all the aforementioned methods are restricted to adopt a one-dimensional ResNet-50 architecture. This is in agreement with earlier research where the choice of a ResNet-50 encoder network has yielded good results [Mehari and Strodthoff, 2022].

2

Theoretical framework

In this chapter, the reader is introduced to the underpinning theories upon which the concept of self-supervised learning is built. Descriptions of the central theoretical- and conceptual frameworks are presented along with research approaches related to the thesis. The chapter begins with an overview of information theory in the context of self-supervised learning, followed by a description of representational learning and transfer learning before concluding with a discussion of the framework for self-supervised learning.

2.1 Information theory

Quantifying the amount of information present in a signal is the main focus of the branch of mathematics called information theory [Thomas and Joy, 2006]. A field that is part of the fundamental theoretical framework of deep learning and has made substantial contributions to its development. Deep learning can be formulated as an information-theoretic trade-off between compression and prediction. In supervised learning, the goal is to find a representation $T(x)$ of the input x that enables an accurate prediction of the output label y [Thomas and Joy, 2006]. This is obtained by finding an efficient way to learn the patterns from the unknown joint distribution $P(X;Y)$, while at the same time retaining the ability to generalize. Formally, the mutual information between two random variables X and Y is

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X),$$

with H being the entropy $H(X) = -\sum_x p(x)\, log\, p(x)$; the average amount of information needed to specify the state of a random variable. The mutual information between X and Y can be understood as how much knowing X reduces the uncertainty in Y, or vice versa. Given a function f that takes x and y as its input, the goal of supervised training is to find parameters of the function f that maximize $I(x;y)$. As the input variable in most deep learning cases is of a much higher dimensionality compared to the output

variable, most of the entropy (information) of the input variable X is not informative about Y, meaning that the features of X relevant for the task are highly distributed and difficult to extract. The goal becomes to statistically decouple input units and minimize information redundancy, without the loss of relevant information. That is, remove the information irrelevant to the prediction of Y while retaining only the most relevant information.

Information bottleneck theory

Following this definition, the information bottleneck theory [Tishby et al., 2000] was introduced. It is an information-theoretic principle for extracting the sought-for-task-relevant information an input random variable $X \in \mathcal{X}$ contains about the output random variable $Y \in \mathcal{Y}$. This random variable X induces a probability distribution on Y that the neural network aims to learn. Thus mathematically, the problem is that of estimating the values of the unknown conditional probability $P_{Y|X}(X)$ for all elements of the dataset. Given their unknown joint distribution $p(X;Y)$, the relevant information is defined as the mutual information $I(X;Y)$, where statistical dependence between X and Y is assumed. Y implicitly determines which features in X are meaningful, and an optimal representation of X would consist solely of the relevant features. Consequently, compressing X by discarding irrelevant parts which do not contribute to the prediction of Y. In statistical terms, the relevant part of X with respect to Y, denoted by U, is a minimal sufficient statistic of X with respect to Y. The problem of finding this representation U can be written as the following Lagrangian optimization

$$\min_{P_{U|X}} I(X,U) - sI(Y,U),$$

where $I(\cdot\,;\cdot)$ denotes the mutual information and s is a Lagrange-type parameter controlling the trade-off between accuracy and regularization. When $s \to 0$, the representation U is at its most compact form, $|U| = 1$. By gradually increasing the parameter s, the emphasis on the relevance term $I(Y;U)$ increases, and at a critical value of s, the optimization focuses on not only the compression but also the relevance term. Namely, it is the simplest mapping of X that captures the mutual information $I(X;Y)$ [Tishby and Zaslavsky, 2015]. This formulation is closely related to that of Rate-Distortion Theory [Equitz and Cover, 1991], with a distortion function that measures how well Y is predicted from a compressed representation compared to its direct prediction from X. This interpretation provides a general algorithm for solving the information bottleneck trade-off and finding an optimal feature representation.

In the context of self-supervised learning

There is no information available regarding the output label Y in self-supervised learning. Thus, the information bottleneck problem cannot be formulated on the input variable X and the output variable Y. Instead, X and Y can be defined as two augmented versions, referred to as two views, of the same data sample [Tsai et al., 2020]. These two views are mapped to a latent embedding space by neural networks defined in the self-supervised learning setup. In this latent space, the two views are represented by various features. The aim of introducing an information bottleneck is to maximize the mutual information between the features extracted from these multiple views of the same data sample [Tschannen et al., 2019].

An analogy can be drawn to a child learning to represent observations generated by a shared cause, e.g. the sights, scents, and sounds of baking. This learning is driven by a desire to predict other related observations, e.g. the taste of cookies. For a more concrete example, the shared context could be an ECG recording. The multiple views of this context could be produced by repeatedly applying data augmentations to the ECG signal. The main idea is that maximizing mutual information between features extracted from these multiple views forces the features to capture information about higher-level factors that broadly affect the shared context. However, this formulation relies on the core assumption that only the shared information between the input and augmented views contributes to solving the task. In addition, the data augmentations should not affect the true labels of input X. In section 2.4, this topic is further elaborated.

2.2 Representation learning

The concept of representation learning within the context of self-supervised learning is closely related to the maximization of mutual information between multiple semantically similar views, as discussed in Section 2.1. In a broader context, representation learning is a class of machine learning approaches where the objective is to represent input data by a latent representation vector that contains the features required for solving a machine learning task [Goodfellow et al., 2016]. Deep learning networks convert data into complex mathematical representations, where each input is assigned a numerical position within a high-dimensional space. These representations, also called embeddings or features, can be seen as a distilled summary of the unique characteristics of the data input. For an ECG recording, the dimensions might quantify minuscule patterns in the signal, the length of each heart cycle, or information on the QRS duration. In representation learning, the goal is to learn which traits distinguish one group of data from another. For example, a question to answer might be what characteristics are always relevant for a

normal ECG recording. Or if a certain cardiovascular anomaly can be represented by a set of features extracted from the heart cycle? Extracting certain features from an ECG signal might enable the network to map the QRS duration to various cardiovascular states. This might enable the network to learn that a particular ECG signal should be considered to come from an unhealthy heart. Being able to understand which features are relevant for representing the input data is the essence of representation learning.

Shared representations are useful to handle multiple data modes or domains, or to transfer learned parameters to tasks for which few or no examples are given. Feed-forward networks trained by self-supervised learning methods can be seen as performing a kind of representation learning where the pre-trained network learns representations assumed to be relevant for solving the actual problem. Broadly speaking, a good representation from a pre-trained network is one that makes a subsequent learning task easier. Thus, how to define a useful representation will most often depend on the actual problem to solve. Tying representation learning back to the information bottleneck theory, a good representation in the self-supervised learning context will maximize the mutual information between augmented views of a shared context and can be used to perform a given task [Bengio et al., 2013].

2.3 Transfer learning

Transfer learning, domain adaptation, and pre-training make use of the idea that network parameters learned in one setting (e.g. data distribution P_1) can be chosen as initial parameters for a neural network in another setting. In principle, a model exploits the knowledge gained from one task and transfers it to improve the performance of another. Pre-training allows for a significant improvement of optimization and generalization capabilities in the new domain (e.g. data distribution P_2), as the transferring of network parameters initializes the model in a multidimensional latent state that otherwise might be inaccessible. An example of an unreachable area could be a region on the mathematical function the network is optimizing, that is surrounded by areas where the cost function varies so much from one example to another that mini-batches of data samples give only a very noisy estimate of the gradient. Or it could be a latent region reachable only with a larger amount of sampled data than is available from P_2, the setting in which we aim to deploy the model. The fact that many of the features explaining variations in P_1 are also relevant for explaining variations in P_2 then becomes a crucial assumption. If there is no, or very little, data sampled from P_2 while there is an abundance of data sampled from the first domain P_1, features that are learned about this first set of data might be used to represent data coming from the second setting. This allows for deep learning models to be used in

low-data regimes where these methods otherwise would not be applicable. In conclusion, transfer learning, multitask learning, and domain adaptation can all be obtained via representation learning when there are underlying factors that are relevant to the various settings or tasks.

Although transfer learning has shown great success in a variety of deep learning domains, there are fundamental differences in data, features, and task specifications between datasets generally used for pre-training and those stemming from the medical domain. The effects of transfer learning within the domain of medical tasks are not widely researched and there is little understanding of the effects of transfer learning from fundamental models to clinical applications [Raghu et al., 2019]. For example, features learned for the classification of everyday objects like images of dogs or cars might be hindering representation learning in the medical domain as the network might have been trained to discard information later needed for the medical application. Despite the impending widespread deployment of foundation models, the current lack of clear understanding of their inner workings might furthermore introduce unwanted model biases [Bommasani et al., 2021].

2.4 Self-supervised learning

Self-supervised learning is a strategy for learning feature representations, using only unlabelled data examples. By training models on unlabelled data, self-supervised learning aims to learn general representations that are easily adaptable for usage on other tasks. Before continuing, the reader should be introduced to two key concepts: the pretext task and the downstream task. A pretext task is a self-supervised learning problem in which the model constructs feature representations from unlabeled data inputs. The pretext task is often followed by a downstream task. This downstream task is the primary task to be solved and it often involves real-world applications and human annotations of the input data. Figure 2.1 presents an illustration of the learning strategy and its two major phases: the unsupervised learning phase which is followed by the supervised fine-tuning process [Zhai et al., 2019]. During the unsupervised representation learning step, supervisory signals are formed directly from the data, contrary to supervised representation learning where explicit annotations or labels act as feedback signals. By forcing a model to solve a deliberately designed pretext task it learns to extract task-agnostic feature representations of the data. The feature vectors after this initial step could in some cases be directly useful for the downstream task; this potential usefulness is measured by the so-called linear evaluation procedure. However, the self-supervised learning routine continues to improve the representations by applying a second step; fine-tuning. Here the model weights are further updated by supervised training on a few labelled data samples. This multi-step

procedure allows for self-supervised learning to be seen as an evolved unification of unsupervised learning, semi-supervised learning, and transfer learning. Most importantly, it allows neural networks to learn useful representations of a data modality that has no labelled examples.

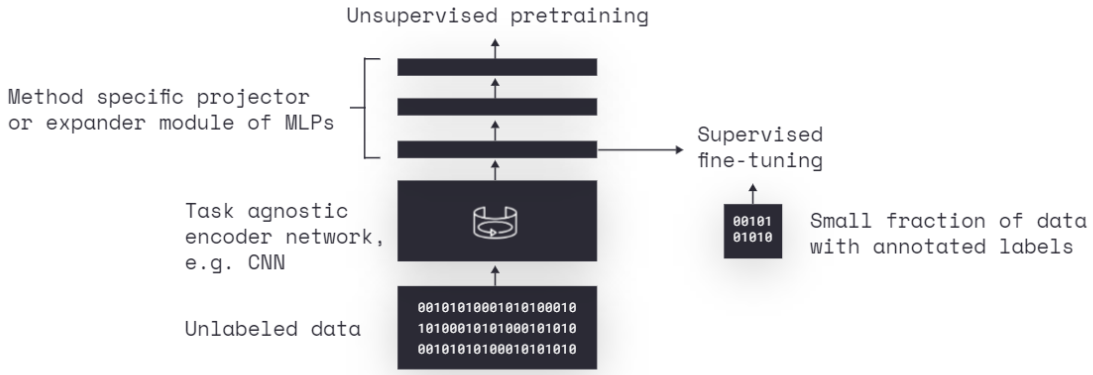


Figure 2.1 Schematic image of the two major phases in self-supervised learning; pre-training with unlabelled data and fine-tuning with labelled data.

Moving away from the need for massive amounts of labelled data, self-supervised learning enables intelligent systems to utilize the abundance of unlabelled data already available, but unusable, given supervised learning techniques. This is a crucial step for the development of deep learning as it seeks to understand more subtle patterns of our world [LeCun and Misra, 2021].

Pre-training and downstream task: The fundamental idea

In defining the pretext-, and downstream tasks much depends on the data modality and the true task which must be completed. A common pretext task in the domain of natural language, for example, is to randomly remove words from a sentence and then let the network predict the missing words. A similar pretext task could also be applied to the visual domain where pixels in an image are randomly masked out, only to have the network fill in the blanked-out pixels based on surrounding pixel values. Other pretext tasks in the visual domain can be constructed by applying transformations to an image, such as rotations, and letting the network predict the degree of rotation that has been applied to the image. Furthermore, [Oord et al., 2018] presents a pretext task that is performed in the audio domain. Here, the aim of the network is to predict latent feature representations of the audio samples when given a sequence of past samples. Moreover, downstream tasks can be any task that is regularly used in supervised learning. Typical problem for-

mulations include classification-, detection-, recognition-, segmentation-, and data-generation tasks.

A formal definition. Given a large set of unlabelled samples $\mathcal{D} = (\mathbf{x}_i)_{i=1}^{U}$, and a small collection of labelled data samples, $\mathcal{S} = (\mathbf{x}_{si}, \mathbf{y}_i)_{i=1}^{L}$, with $L \ll U$ and where $\mathcal{S}$ may be a subset of the dataset $\mathcal{D}$, $\mathcal{S} \subset \mathcal{D}$. The goal of the pretext task in a self-supervised learning algorithm Ψ is to learn parameters θ of a function f_θ that maps a data sample $\mathbf{x}$ to vector representations in a latent space. With a model architecture f, the parameters θ are therefore learnt as $\theta = \Psi_f(\mathcal{D})$. Representations are fine-tuned and evaluated on various supervised downstream tasks $\mathcal{T} = \{(\bar{x}_1, y_1), \ldots, (\bar{x}_M, y_M)\}$, where pairs of labelled input data and output labels $(\mathbf{x}_{si}, \mathbf{y}_i)$ from the labelled dataset $\mathcal{S}$ is used.

Self-Supervised representation learning. State-of-the-art self-supervised learning methods form representations through joint-embedding architectures. Using Siamese networks, representations are learned by maximizing agreement between embeddings of different views of the same data example via a loss function in the latent space [He et al., 2020; Caron et al., 2020; He, 2020; Zbontar et al., 2021; Bardes et al., 2021]. In general, data views are constructed by applying different augmentations to the input data. The goal of Siamese networks is to learn an encoder $f(\cdot)$ that produces similar vector embeddings for two views of the same data sample. Intuitively, the encoder f is sufficient if $f(\mathbf{x}_i)$ has kept all the information about $f(\mathbf{x}_j)$ that the learning objective requires, and symmetrically, if $f(\mathbf{x}_j)$ has kept all the significant information about $f(\mathbf{x}_i)$, i.e. the encoding procedure is lossless $I(\mathbf{x}_i; \mathbf{x}_j) = I(f(\mathbf{x}_i); f(\mathbf{x}_j 2))$. Among the sufficient encoders, minimal encoders only extract information relevant to the learning objective and throw away other irrelevant information. The sufficient encoder f of $\mathbf{x}_i$ is minimal if and only if $I(f(\mathbf{x}_i); \mathbf{x}_i) \leq I(F(\mathbf{x}_i); \mathbf{x}_i), \forall F$ that is sufficient. Given a minimal sufficient encoder network $f(\cdot)$ and two augmented views $\mathbf{x}_i$ and $\mathbf{x}_j$ of a data sample $\mathbf{x}$, the encoder independently maps each view to a latent representation vector $\mathbf{z}_i$ and $\mathbf{z}_j$. The learning objective of the joint-embedding network is to learn the minimal sufficient encoder that maximizes the similarity between these augmented views while minimizing redundancy within the representation vectors in order to become insensitive to differences between views [Tsai et al., 2021].

Furthermore, an optimal representation will maintain the smallest complexity, containing no other information about the input besides the one required for the downstream task. With a downstream task $\mathcal{T}$ whose goal is to predict a semantic label y from the input data $\mathbf{x}$, the optimal representation $\mathbf{z}^*$ encoded from $\mathbf{x}$ is the minimal sufficient statistic with respect to y. Thus, $\mathbf{z}^* = f(\mathbf{x})$ has all the information necessary to predict y as accurately as if it were to access $\mathbf{x}$. The objective of the self-supervised pretext task becomes

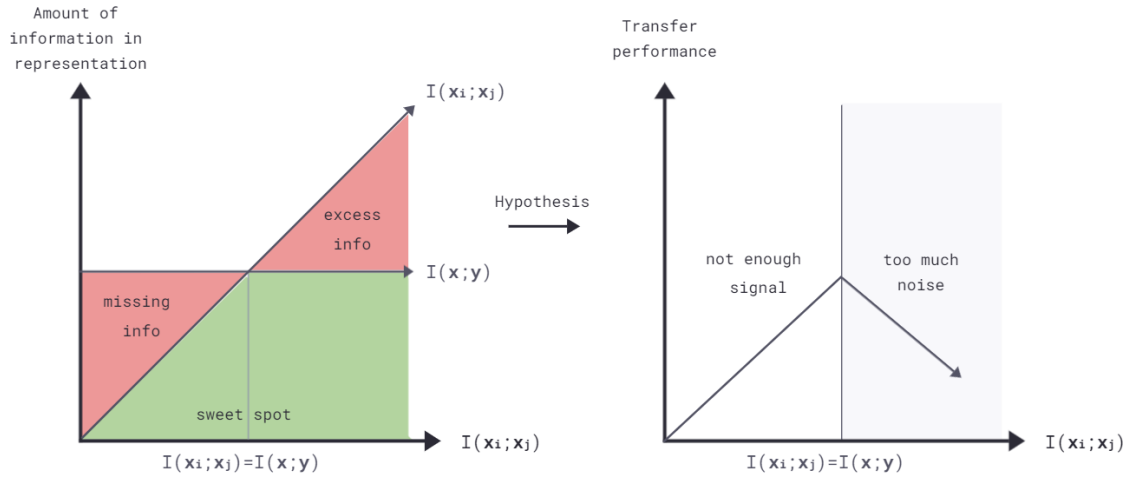


Figure 2.2 Given two views $\mathbf{x}_i$ and $\mathbf{x}_j$ of a data sample $\mathbf{x}$ and a task label y, the information captured in a representation will be optimal at the sweet spot, where the only information shared between $\mathbf{x}_i$ and $\mathbf{x}_j$ is task-relevant.

to learn the optimal representation where the only information shared between the views $\mathbf{x}_i$ and $\mathbf{x}_j$ is task-relevant and there is no irrelevant noise, $I(\mathbf{x}_i; y) = I(\mathbf{x}_j; y) = I(\mathbf{x}_i; \mathbf{x}_j) = I(\mathbf{x}; y)$. Figure 2.2 presents a graphical illustration of the trade-off between capturing too much information or too little information in a representation.

Embedding collapse

Current self-supervised learning approaches map data to vector representations in a multidimensional embedding space. In this space, nearly identical data should be represented in a very similar way, meaning that they should be placed close to each other in this hyperspace. As the training process proceeds, the aim is to create clusters of data representations, where data-sharing characteristics are placed in the same cluster. Although this method has proven to be highly successful in several domains, an improperly designed pretext task might bring about a complete representation collapse in which all embedding vectors are represented by a trivial solution at a single point. In other words, if representation collapse is not prevented, models can create the exact same vector representation for each input, and in the attempt to maximize the likeness between similar representations the model ends up treating all data samples as if they were the same. Another form of collapsing-problem is that of dimensional collapse, whereby the embedding vectors end up spanning a lower-dimensional subspace instead of the entire available embedding space. Figure 2.3 presents a graphical illustration of the collapsing problems.

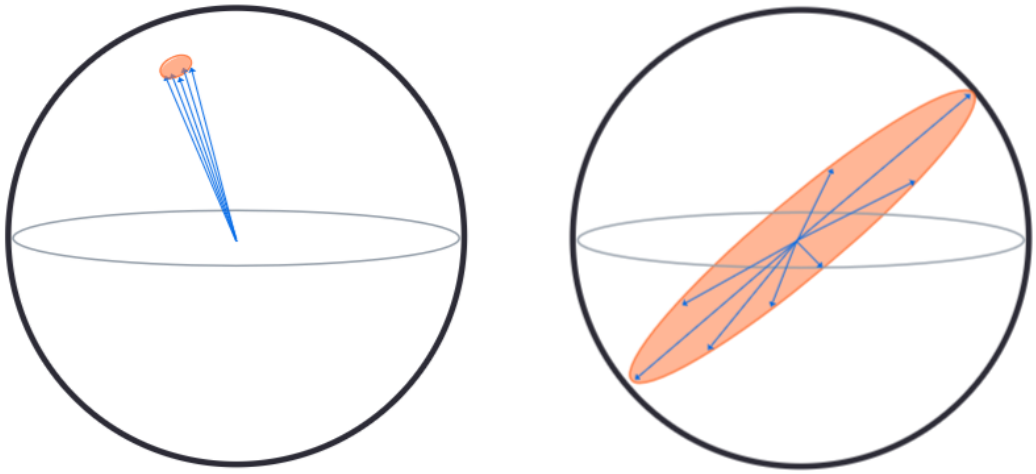


Figure 2.3 Trivial collapse brings all embedding vectors to same point. For dimensional collapse, the embedding vectors only span a lower dimensional space.

Self-supervised learning methodologies

Several approaches have been suggested in an attempt to overcome these collapsing problems. They can be divided into contrastive and non-contrastive learning methods. Contrastive losses explicitly push away embeddings of different samples while non-contrastive, e.g. the so-called asymmetric approaches, introduce asymmetric network architectures to prevent collapse, e.g., stop-gradient operations and a momentum encoder. Other non-contrastive approaches try to decorrelate the vector components of the embedding dimensions to minimize redundancy across samples.

Contrastive learning. Contrastive self-supervised learning methods are based on the idea of instance discrimination. Instead of predicting the exact class of a data sample, the objective is to predict whether pairs of inputs belong to the same or different classes. To formalize the process, multiple views of the inputs are created via a data transformation process $\mathcal{T}$, and their representations are compared as either positive or negative pairs in a latent embedding space [Jaiswal et al., 2021]. Given a data sample $\mathbf{x}$, two views are created by augmentation to form $\mathbf{x}_i = \mathcal{T}_i(\mathbf{x})$ and $\mathbf{x}_j = \mathcal{T}_j(\mathbf{x})$ and form a positive pair. One view, $\mathbf{x}_i$ is chosen to be the anchor and is contrasted with its positive pair $\mathbf{x}_j^+$. The anchor is also contrasted with negative samples, which are views of other samples $\hat{\mathbf{x}}$ in the same batch, $\hat{\mathbf{x}}_i^- = \mathcal{T}_i(\hat{\mathbf{x}})$ and $\hat{\mathbf{x}}_j^- = \mathcal{T}_j(\hat{\mathbf{x}})$. Theoretically, the positive pairs can be seen as coming from a joint distribution over views $p(\mathbf{x}_i, \mathbf{x}_j^+)$, and the negative pairs from a product of marginals

$p(\mathbf{x}_i)p(\mathbf{x}_j^-)$. The objective is to minimize the distance in representation space between positive sample pairs while maximizing the distance between negative sample pairs [Caron et al., 2020]. Figure 2.4 illustrates an example of a joint-embedding network that applies a contrastive learning approach.

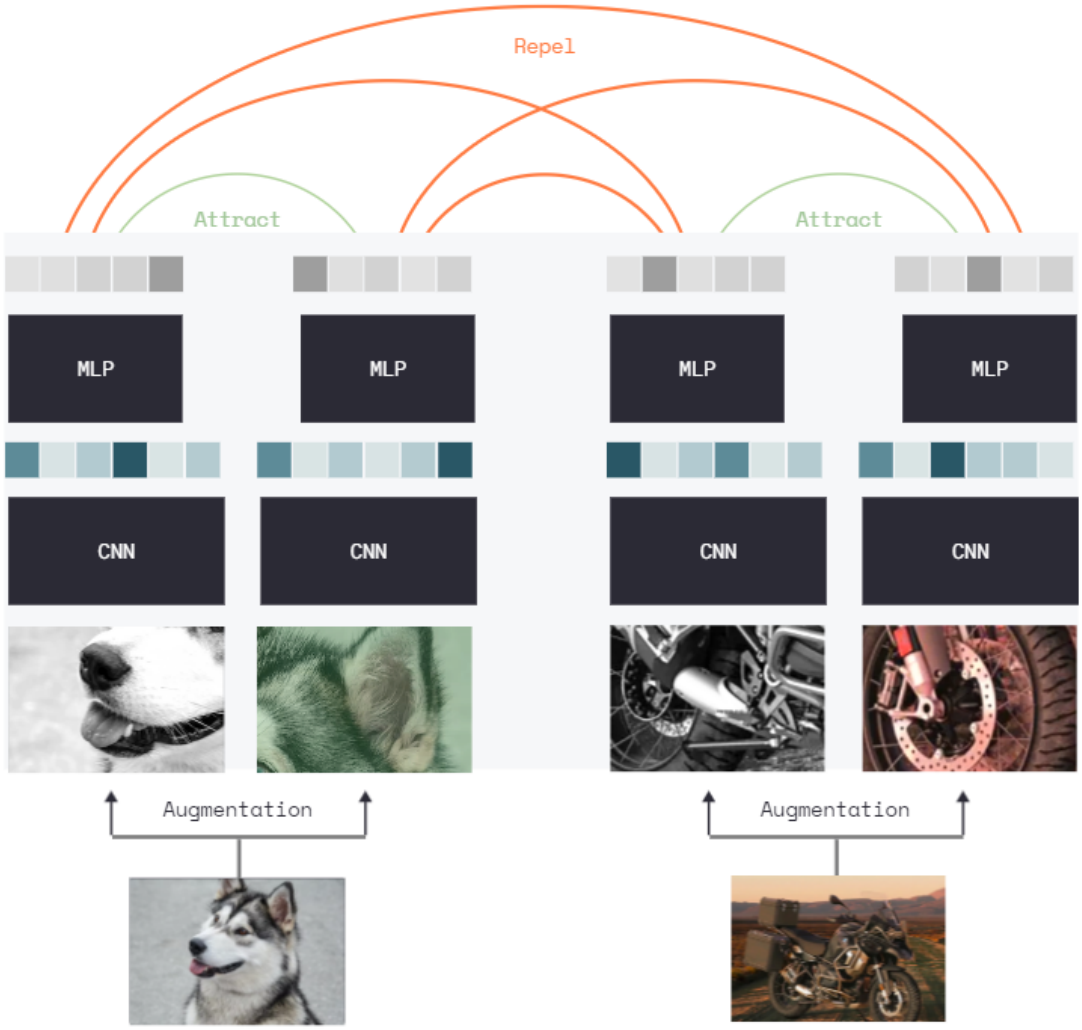


Figure 2.4 Overview of the SimCLR method which adopts a Siamese network and a contrastive loss.

A commonly used contrastive loss function, InfoNCE [Oord et al., 2018] learns statistical properties of a target distribution by comparing the positive samples from the target distribution $p(\mathbf{x}_i; \mathbf{x}_j)$ to the negative samples in the batch. The negative samples are seen to be sampled from a noise distribution, and thus, the method is known as negative sampling in some contexts. In practice, given an anchor point $\mathbf{x}_i$, the InfoNCE loss is optimized to score the correct positive $\mathbf{x}_j^+ \sim p(\mathbf{x}_j|\mathbf{x}_i)$ higher compared to a set of K distractors

$\mathbf{x}_j^- \sim p(\mathbf{x}_j)p(\mathbf{x}_i)$

$$I(X;Y) \geq \mathbb{E}\left[\frac{1}{K}\sum_{i=1}^{K}\log\frac{e^{f(x_i,y_i)}}{\frac{1}{K}\sum_{j=1}^{K}e^{f(x_i,y_j)}}\right] \triangleq I_{\mathrm{NCE}}(X;Y),$$

where the expectation is over K independent samples $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^K$ from the joint distribution $p(\mathbf{x}, \mathbf{y})$ [Poole et al., 2019]. InfoNCE is in practice estimated using Monte Carlo estimation by averaging over multiple batches of samples. The function f will for each x_i try to predict which of the K samples $y_1, \ldots, y_k$ it was jointly drawn with, i.e. its positive pairing. This is done by assigning high values to the jointly drawn sample, and low values to all other pairs. It has been shown that minimizing the InfoNCE loss also maximizes a lower bound on mutual information [Oord et al., 2018]. This approach is based on the idea of the information bottleneck theory as described in Section 2.1, whereas the objective of contrastive learning becomes to maximize the mutual information between embeddings of the positive pairs.

In order to avoid the collapsed embedding problems, contrastive learning methods require large quantities of negative samples so that the learning objectives obtain the maximum similarity and have the minimum similarity with negative samples. Representative contrastive learning methods such as contrastive predicting coding (CPC) [Henaff, 2020], SimCLR [Chen et al., 2020] and MoCo [He et al., 2020] have demonstrated that it is possible to produce features that surpass the supervised feature representations on many downstream computer vision tasks.

Non-contrastive learning. Non-contrastive learning methods, unlike contrastive methods, learns nontrivial representations using only positive sample pairs. Representational collapse is prevented by introducing asymmetry in the network architecture. For example, asymmetrical encoding for the two input views [Grill et al., 2020] or minimizing redundancy via the cross-correlation between features [Zbontar et al., 2021; Bardes et al., 2021]. See Figure 2.5 for an illustration of an asymmetric network architecture. Although, it is not fully understood how these methods avoid collapse, theoretical and empirical studies point to the crucial importance of batch-wise or feature-wise normalization [Tian et al., 2021].

Another method building on the non-contrastive framework is VICReg [Bardes et al., 2021]. Here, collapse is prevented by introducing to the loss function a simple regularisation term on the variance of the embeddings along each individual dimension. This term forces the embedding vectors of samples within a batch to be different, thus preventing each dimension from collapsing to the same point. In addition to the variance term, a decorrelation mechanism based on redundancy reduction and covariance regularisation is introduced. This term attracts the covariances over a batch between every

pair of embedding variables towards zero while decorrelating the variables of each embedding and preventing them from being correlated. Decorrelation of the covariance is also used in the Barlow Twins method [Zbontar et al., 2021], where it prevents informational collapse caused by redundancy between the embedding variables.

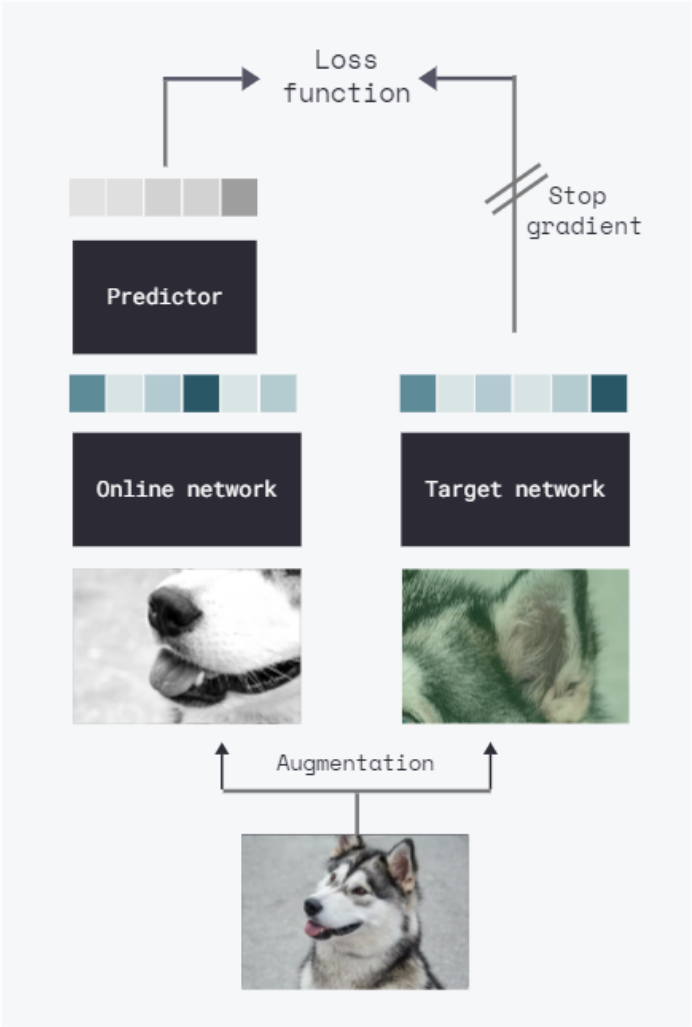


Figure 2.5 Overview of an asymmetrical Siamese network not using a contrastive loss function. Stop gradients and exponential moving average updating of network weights are for example used to prevent embedding collapse.

Common methods. Figure 2.6 illustrates the architecture of four state-of-the-art methodologies for self-supervised learning. All of the methods are Siamese networks that consist of two network arms, or branches, where the input to each arm is one of the augmented versions $\mathbf{x}_i$ or $\mathbf{x}_j$ of the data sample $\mathbf{x}$. By feeding the two views $\mathbf{x}_i$ and $\mathbf{x}_j$ to an encoder f, embedding

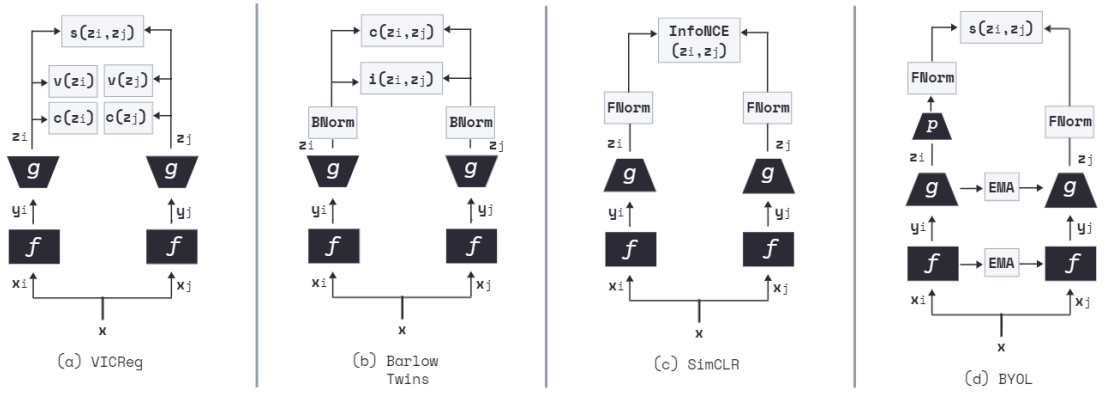


Figure 2.6 Schematic image comparing the architecture of four self-supervised methods. In VICReg and Barlow Twins, module g is an expander network while in SimCLR and BYOL g is a projector network. BNorm in the figure is short for batch-wise normalized embeddings while FNorm stands for feature-wise normalized embeddings. EMA is short for exponential moving average.

representations $\mathbf{h}_i$ and $\mathbf{h}_j$ are formed. These embeddings are further transformed by a network g, where g can be a projector network that reduces the dimensionality of the representations, or an expander network that increases the dimensionality. The final embeddings $\mathbf{z}_i$ and $\mathbf{z}_j$ are used to minimize the methods loss function $\mathcal{L}$.

Further highlighting a few similarities and differences between the various approaches we begin examining method (a), VICReg [Bardes et al., 2021]. The variance and covariance of each branch in VICReg are regulated in the loss function by the regularizing terms v and c. These terms prevent representational embedding collapse and the distance between embeddings of the two branches is minimized with a mean-squared error loss s. The Barlow Twins method (b) [Zbontar et al., 2021] is closely related to the VICReg method as they use similar mechanisms for decorrelating features of each embedding vector in order to remove redundancy between the embedding variables. VICReg penalizes the off-diagonal terms of the covariance matrix computed on the embeddings from each network arm as the terms are attracted toward zero. Barlow Twins instead uses the cross-correlation matrix where each matrix entry is the computed cross-correlation between the two representation vectors $\mathbf{z}_i$ and $\mathbf{z}_j$. Furthermore, Barlow Twins applies the invariance loss i that drives the diagonal elements of the cross-correlation matrix towards 1. Thus, making the embeddings of the two augmented sample views invariant to the applied augmentations. Another non-contrastive method is the method (d), BYOL [Grill et al., 2020]. Instead of preventing embedding collapse by introducing regularization terms to the loss function as VICReg and Barlow Twins, BYOL introduces an asymmetric network architecture. The weights of one arm are updated by applying an exponential moving average to the

weights of the other arm. A stop gradient is also applied to one of the arms to further prevent embedding collapse. Barlow Twins, VICReg, and BYOL all use non-contrastive learning frameworks where no explicit negative sample pairs are used in the formation of latent representations. Method (c), SimCLR [Chen et al., 2020] in contrast uses negative samples in combination with the contrastive loss InfoNCE to form latent representations of the input data. Here, representations of positive pairs are brought close to each other in the embedding space while representations of negative pairs are pushed far apart.

3

Data

In this chapter, we will introduce the data modality, datasets, and data transformations that have been used for the experiments in this thesis. The data consist of short-duration 12-lead ECG signals obtained from 6 publicly available datasets. In total, 49,662 unlabelled ECG signals are used for the pretext task and 21,837 labelled signals are used for evaluation. In this chapter, the reader is also provided with a comprehensive description of the ECG signals and accompanying annotations. In addition, Section, 3.3, extends the previous section by describing the applied data augmentation techniques and their role in the self-supervised learning framework.

3.1 Electrocardiogram

Analysis of an electrocardiogram (ECG) plays a significant role in the diagnosis and screening of cardiac diseases. By placing electrodes on the skin, the heart's electrical activity is recorded as an electrogram of voltage versus time, giving the ECG signal. The electrodes placed on the skin detect small electrical changes present in each cardiac cycle, which are caused by the cardiac muscle's depolarization and repolarization process. Numerous cardiac abnormalities such as cardiac rhythm disturbances (e.g. atrial fibrillation and ventricular tachycardia), inadequate coronary artery blood flow (e.g. myocardial ischemia and myocardial infarction) and electrolyte disturbances (e.g. hypokalemia and hyperkalemia) can be seen in the ECG signal as deviations from its normal pattern.

In a conventional ECG, 12 electrodes are used to measure the magnitude of the heart's electrical potential. Two electrodes are placed on the limbs and the ten remaining are placed on the surface of the chest. Recorded over a period of time, usually lasting around ten seconds, the overall magnitude and direction of the heart's electrical depolarization is then captured at each moment throughout the cardiac cycle.

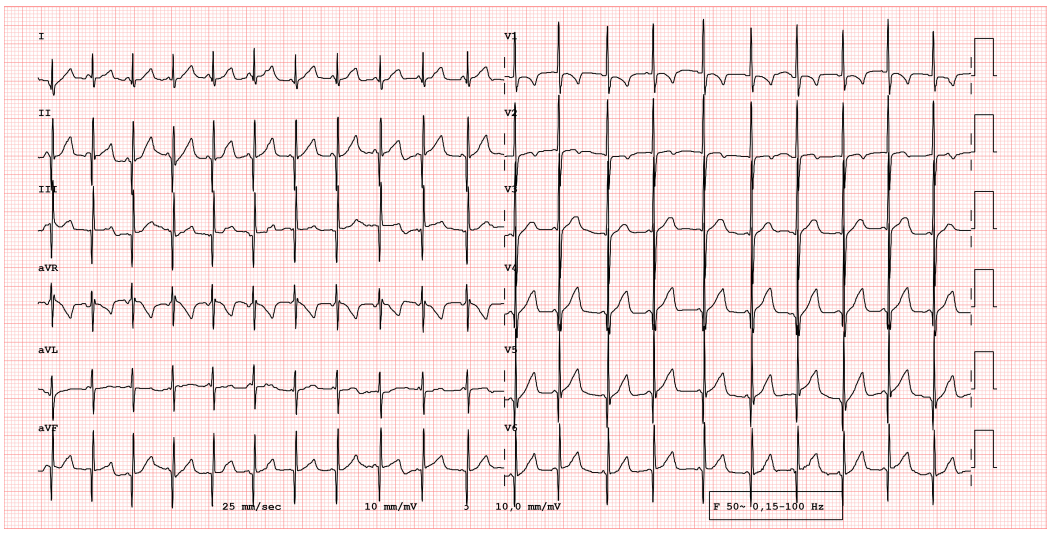


Figure 3.1 Example of a 12-lead ECG with normal sinus rhythm recorded on an adult human.

An ECG signal can be divided into three main sections based on the heart's electrical activity. The first is the P wave which represents the depolarization of the atria. The second section represents the depolarization of the ventricles and is referred to as the QRS complex. Lastly, the T wave represents the repolarization of the ventricles. Each waveform contains significant information that can be used to understand the cardiac state of individuals.

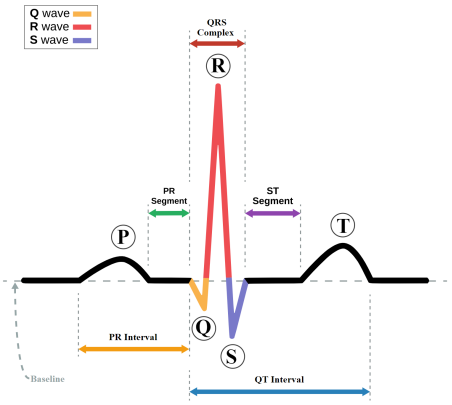


Figure 3.2 ECG of a heart in normal sinus rhythm [Atkielski, n.d.]

3.2 Datasets

All ECG recordings used during this thesis have been obtained from publicly accessible databases. Presented below is a detailed overview of the datasets, their acquisition methods, relevant annotations, and a description of the ap-

plied preprocessing. Moreover, Figure 3.3 presents a graphic summary of the data that was utilized.

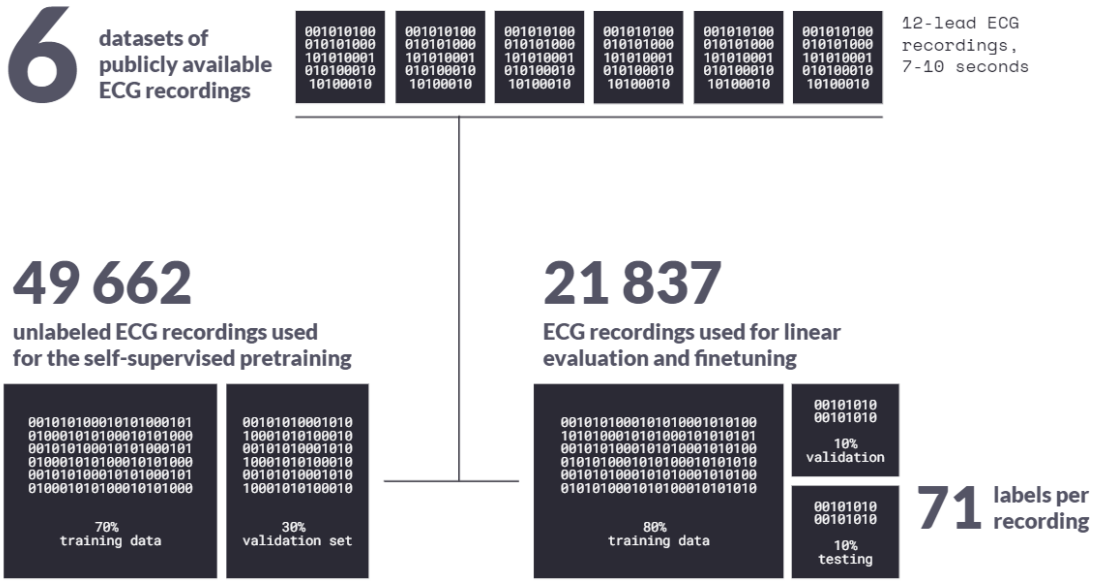


Figure 3.3 Graphic summary of the data used in this thesis.

All datasets consist of short-duration (7-10 seconds) standard 12-lead ECG recordings in combination with patient-specific metadata e.g. age, gender, weight, height, etc. During the experiments conducted throughout this thesis work, only ECG recordings are used and the metadata is discarded. Furthermore, all ECG recordings and associated annotations are obtained from in-clinic exams conducted by clinical cardiologists. Specific information relating to each dataset is found in the respective subsections below.

Following the work of [Mehari and Strodthoff, 2022] and [Strodthoff et al., 2020], the ECG recordings used throughout these experiments were restricted to ECG data at a sampling rate of 100Hz. Although, the work of [Kwon et al., 2018] suggests that 100 Hz ECG is sufficient for models operating in the time domain, but recommends a higher sampling frequency for frequency domain-based models. Experiments conducted by [Strodthoff et al., 2020] however found no compelling evidence for this apprehension, as no significant gain in performance in any diagnostic task was detectable when processing ECG data with a sample rate of 500 Hz. Furthermore, using ECG data with a higher sampling rate would introduce other more obvious issues affecting performance, e.g. dealing with all sorts of artifacts and label noise, preprocessing and more resilient and effective training procedures. Signals were segmented into windows of length T, where $T = 250$ or $T = 1000$ depending on whether the used signal length was 2.5 seconds or 10 seconds.

We would like to emphasize that the annotations of the ECG recordings used only for pre-training have not been described in this report. Due to the fact that data label information in self-supervised pre-training is irrelevant per definition, the label information in the datasets was discarded.

Table 3.1 Summery of the unlabelled data used for pre-training.

Dataset	**# recordings**	**Length** *(s)*	f_{sample} *(Hz)*
Ribeiro	**827**	7 or 10	500
Zheng	**10646**	10	500
Cinc2020	**38189**		
CPSC & CPSC-Extra	*10330*	*6-60*	*500*
Incart	*74*	*1800*	*257*
PTB & PTB-XL	*17441*	*10*	*1000/500*
Georgia	*10344*	*10*	*500*
Total	**49662**		
Total with length $\geq$ 10s	**49048**		

Datasplitting

A collection of three datasets were used for the pre-training process: Computing in Cardiology Challenge 2020 (CinC2020), Ribeiro, and Zheng. They are throughout this thesis denoted $\mathcal{D}_{Pre-train}$. The evaluation procedure, however, was carried out exclusively with the PTB-XL dataset, denoted as $\mathcal{D}_{PTB}$. Notice that the dataset used for the CinC2020 itself is a compilation of five different datasets. In particular, it includes the PTB-XL dataset $\mathcal{D}_{PTB} \subset \mathcal{D}_{Pre-train}$, where only the dataset's suggested training data was utilized for the pre-training process. The PTB-XL suggested validation and test data were removed from the pre-training process and only incorporated into the evaluation procedure. Table 3.1 presents a summary of the unlabelled data $\mathcal{D}_{Pre-train}$ used for pre-training.

During pre-training, data were randomly split into stratified training and validation sets, ensuring recordings sampled from the same patient were not mixed between training and validation sets. 70% of the data were used for training and the other 30% as validation data. For the evaluation phase, the suggested training, validation, and test splits in the PTB-XL dataset were used, combined with respective annotations. See Section 3.2 for a detailed description of the evaluation dataset and Table 3.2 lists the number of ECG signals in the different datasets.

Table 3.2 Size of datasets used for pre-training and evaluation.

Pre-training	**# ECG recordings**
Training set	34763
Validation set	14899
Evaluation	
Training set	17441
Validation set	2193
Test set	2203

Computing in cardiology challenge 2020

The dataset used for the Computing in Cardiology Challenge 2020 [Alday et al., 2020], here referred to as CinC2020, is a collection of data from multiple sources. In total there are 20,740 recordings of which 74 are shorter than 10 seconds.

CPSC database and CPSC-extra database. The first source is the public (CPSC Database) and unused data (CPSC-Extra Database) from the China Physiological Signal Challenge in 2018 [Liu et al., 2018]. This dataset consists of two sets of 6,877 (male: 3,699; female: 3,178) and 3,453 (male: 1,843; female: 1,610) 12-ECG recordings lasting from 6 seconds to 60 seconds. Each recording was sampled at 500 Hz.

INCART database. The second source set is the public dataset from St Petersburg INCART 12-lead Arrhythmia Database [Tihonenko et al., 2008]. This database consists of 74 annotated recordings (male: 17; female: 15 women) extracted from 32 Holter records. Each record is 30 minutes long and contains 12 standard leads sampled at 257 Hz.

PTB and PTB-XL database. The third source from the Physikalisch Technische Bundesanstalt (PTB) comprises two databases: the PTB Diagnostic ECG Database [Bousseljot et al., 1995] and the PTB-XL [Wagner et al., 2020]. These are two large publicly available electrocardiography datasets. The first PTB database contains 516 records (male: 377, female: 139). Each recording was sampled at 1000 Hz. The PTB-XL contains 21,837 clinical 12-lead ECGs (male: 11,379 and female: 10,458) of 10-second length with a sampling frequency of 500 Hz.

The Georgia 12-lead ECG challenge (G12EC) database. The fourth source is a Georgia database which represents a unique demographic of the Southeastern United States. This training set contains 10,344 12-lead ECGs (male: 5,551, female: 4,793) of 10-second length with a sampling frequency of 500 Hz.

Zheng

This dataset [Zheng et al., 2020] contains 12-lead ECGs of 10,646 patients collected from Chapman University and Shaoxing People's Hospital (Shaoxing Hospital Zhejiang University School of Medicine) with a 500 Hz sampling rate. It features 11 common rhythms and 67 additional cardiovascular conditions, all labelled by professional experts. For each subject, a sample size of 10 seconds is available. The database consists of 10,646 ECG recordings of 10 seconds in length, including 5,956 males and 4,690 females. Among those patients, 17% had normal sinus rhythm and 83% had at least one abnormality.

Ribeiro

The data in this database was collected from different patients of 811 counties in the state of Minas Gerais/Brazil from the Telehealth Network of Minas Gerais (TNMG) [Ribeiro et al., 2020b]. The data were obtained between 2010 and September 2018 and were sampled at 400 Hz. There are 827 recordings of lengths of 7 or 10 seconds. In order to have the same size for all recordings, the ECG signals are filled with zeros on both sizes i.e. zero-padded, resulting in a signal with 4,096 samples for each lead. There is no available information regarding the number of males and females from whom the ECG signals were recorded.

Physikalisch-Technische Bundesanstalt

The PTB-XL [Wagner et al., 2020] ECG dataset is a large dataset of 21,837 clinical 12-lead ECGs from 18,885 patients of 10-second length, collected from Schiller AG over the course of nearly seven years between October 1989 and June 1996. Out of the 18,885 patients, 52% were male and 48% female with ages covering the whole range from 0 to 95 years (median 62 and interquartile range of 22). See Figure 3.4 for a graphical representation of data demographics. The value of the dataset results from the comprehensive collection of many different co-occurring pathologies, combined with a large proportion of healthy control samples. All are meticulously annotated by clinical cardiologists.

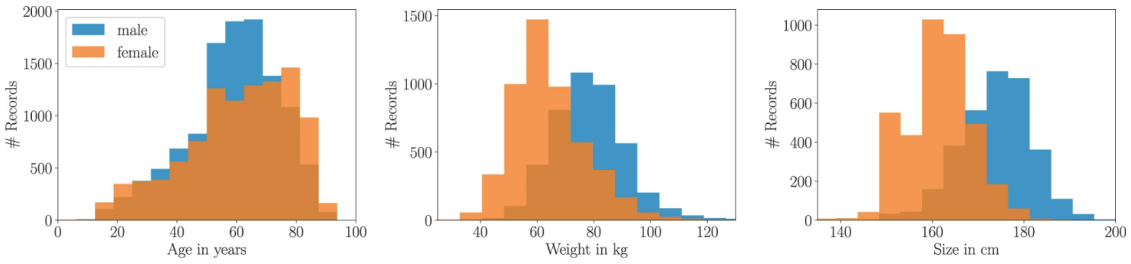


Figure 3.4 Demographic overview of patients in PTB-XL [Wagner et al., 2020].

All ECG recordings are provided at two sampling frequencies, the original 500 Hz and a down-sampled version of 100 Hz. The latter is the data used for this thesis. In addition to the recordings, each ECG instance is paired with an extensive description covering demographics data, infarction characteristics, and likelihoods for diagnostic ECG statements as well as annotated signal properties. To further facilitate the usage of machine learning algorithms, the dataset is presented with a recommended 10-fold train-test split obtained via stratified sampling. Respecting patient assignments, all records of a particular patient were assigned to the same fold. To achieve not only a balanced label distribution but also a balanced age and sex distribution, ECG labels, sex, and age are considered for the stratification process. Data in folds 9 and 10 underwent at least one human evaluation and are therefore of particularly high label quality. Therefore, fold 9 is proposed as a validation set and fold 10 as a test set. As a training set, folds 1-8 are recommended; a suggestion that we chose to follow.

ECG annotations and labels

The following section presents a comprehensive overview of the ECG annotations used for the evaluation of the downstream performance of the ECG representations. Figure 3.5 presents an overview of SCP-ECG acronym descriptions for the super- and subclasses present in the dataset.

Table 3.3 Distribution of diagnostic superclass labels in PTB-XL dataset.

#Records	Superclass	Description
9528	NORM	Normal ECG
5486	MI	Myocardial Infarction
5250	STTC	ST/T Change
4907	CD	Conduction Disturbance
2655	HYP	Hypertrophy

At the most fine-grained level, the PTB-XL dataset includes 71 labels, all of which are included in the multi-label classification problem proposed in this thesis. These labels cover a wide variety of diagnostic, form, and rhythm statements that can be used for a comprehensive evaluation of ECG analysis algorithms. The 44 diagnostic statements can be categorized in terms of five superclasses (normal/conduction disturbance/myocardial infarction/hypertrophy/ST-T change), the 19 form statements relate to mostly morphological changes in specific ECG segments such as an abnormal QRS complex, and the 12 rhythm statements comprise statements characterizing normal cardiac rhythms as well as arrhythmia. The raw waveform data has been annotated by clinical cardiologists. In addition to the ECG recordings, the dataset is complemented by extensive metadata on demographics,

		Acronym	SCP statement Description
Superclasses		NORM	Normal ECG
		CD	Conduction Disturbance
		MI	Myocardial Infarction
		HYP	Hypertrophy
		STTC	ST/T change
Subclasses	NORM	NORM	Normal ECG
	CD	LAFB/LPFB	left anterior/left posterior fascicular block
		IRBBB	incomplete right bundle branch block
		ILBBB	incomplete left bundle branch block
		CLBBB	complete left bundle branch block
		CRBBB	complete right bundle branch block
		_AVB	AV block
		IVCB	non-specific intraventricular conduction disturbance (block)
		WPW	Wolff-Parkinson-White syndrome
	HYP	LVH	left ventricular hypertrophy
		RHV	right ventricular hypertrophy
		LAO/LAE	left atrial overload/enlargement
		RAO/RAE	right atrial overload/enlargement
		SEHYP	septal hypertrophy
	MI	AMI	anterior myocardial infarction
		IMI	inferior myocardial infarction
		LMI	lateral myocardial infarction
		PMI	posterior myocardial infarction
	STTC	ISCA	ischemic in anterior leads
		ISCI	ischemic in inferior leads
		ISC_	non-specific ischemic
		STTC	ST-T changes
		NST_	non-specific ST changes

Figure 3.5 SCP-ECG acronym descriptions for super- and subclasses [Wagner et al., 2020].

infarction characteristics, and likelihoods for diagnostic ECG statements as well as annotated signal properties. This metadata is however not used for the thesis experiments. If for convenience, diagnostic statements are restricted to their aggregated super-classes, the distribution of cardiovascular diagnosis present in the dataset is as presented in Table 3.3. Note that the sum of statements exceeds the number of records as there might be multiple labels associated with a single record. Figure 3.6 and Figure 3.7 present a graphical summary of the dataset in terms of diagnostic superclasses and subclasses. Figure 3.8 displays a plot of ECG signals of normal or abnormal character.

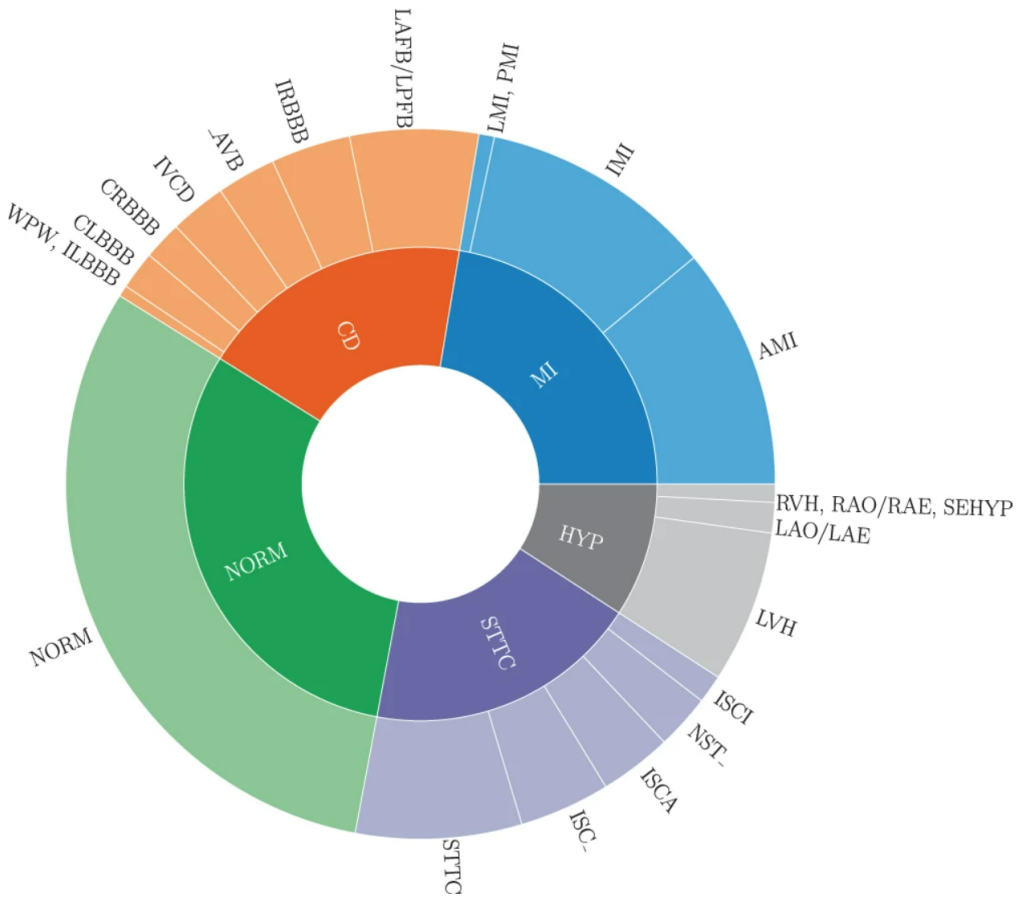


Figure 3.6 Graphical summary of the PTB-XL dataset in terms of diagnostic superclasses and subclasses, see Figure 3.5 for a definition of the used acronyms [Wagner et al., 2020].

3.3 Data augmentations

In deep learning, data augmentation is typically used as a regularization technique to reduce overfitting. As a result of producing slightly altered copies of existing data or creating synthetic data, the volume of available training data increases. Although data augmentation has been widely applied in both supervised and unsupervised representation learning [Krizhevsky et al., 2012; Henaff, 2020; Bachman et al., 2019], it serves a different purpose in self-supervised learning, where the composition of data augmentation operations is crucial for achieving useful representations.

Data augmentation for self-supervised learning. In self-supervised learning, the choice of views is what controls the information captured in the representation. Self-supervised techniques encourage representations to discard information regarding the augmentations applied to the input data, thereby becoming invariant to the set of chosen augmentations. By, for example, minimizing the mean squared error between two latent representations

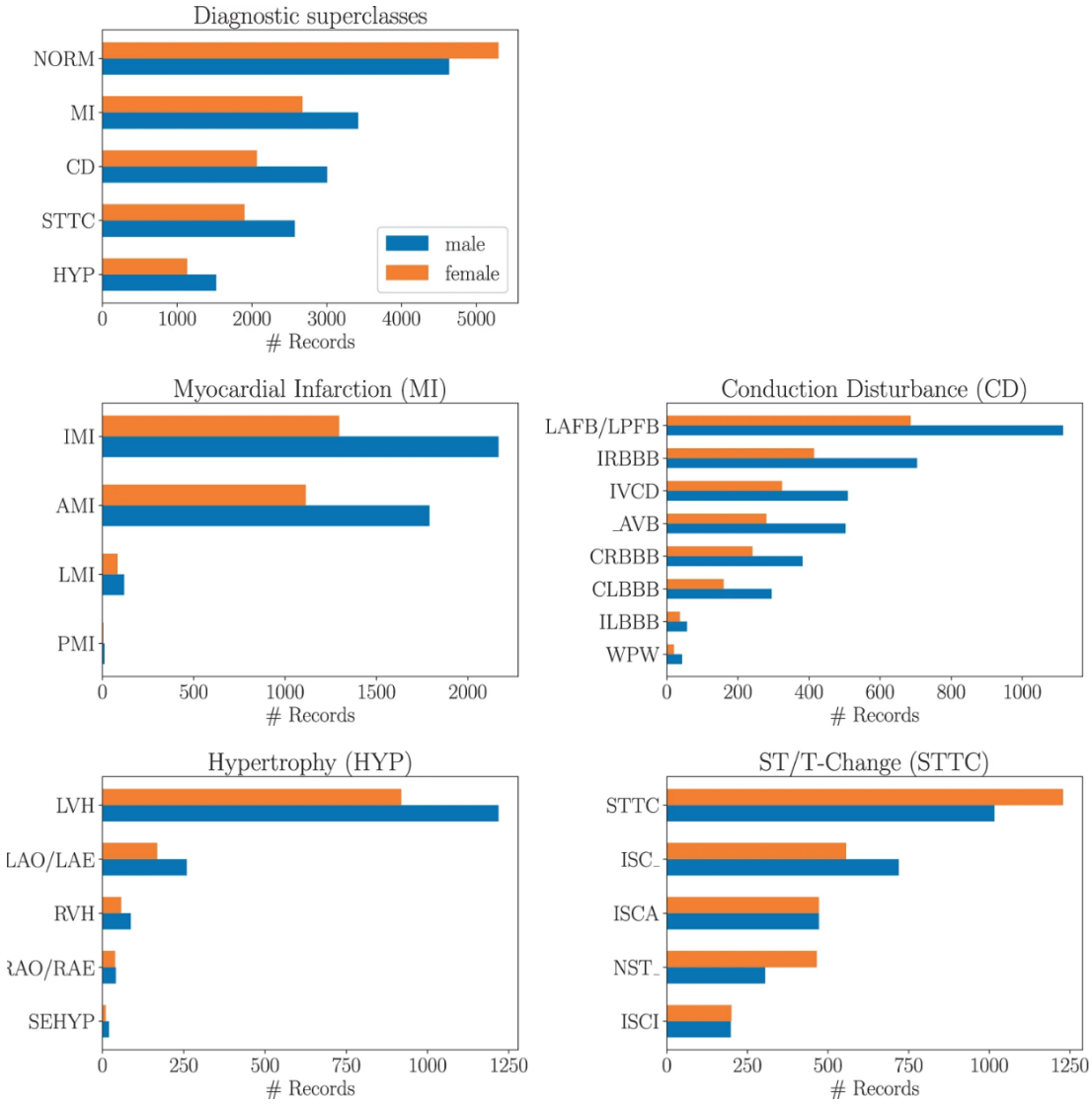


Figure 3.7 Distribution of diagnostic subclasses for given diagnostic superclasses [Wagner et al., 2020].

of the augmented data samples, the model will be able to learn that "these two representations are in fact just different views of the same thing", and that information that does not pertain to the semantics of the data can be eliminated. Just as observing a dog from left to right, it remains the same dog.

Augmentations in contrastive learning can be seen through the perspective of the information bottleneck theory described in Section 2.1. The aim of the pretext task is to maximize the mutual information between features extracted from multiple views of a shared context. The pre-training phase serves to decouple the correlations of irrelevant features between the representations of positive samples. Whereby the neural networks will ignore the decoupled features and learn from the similarities of features that are more resistant to

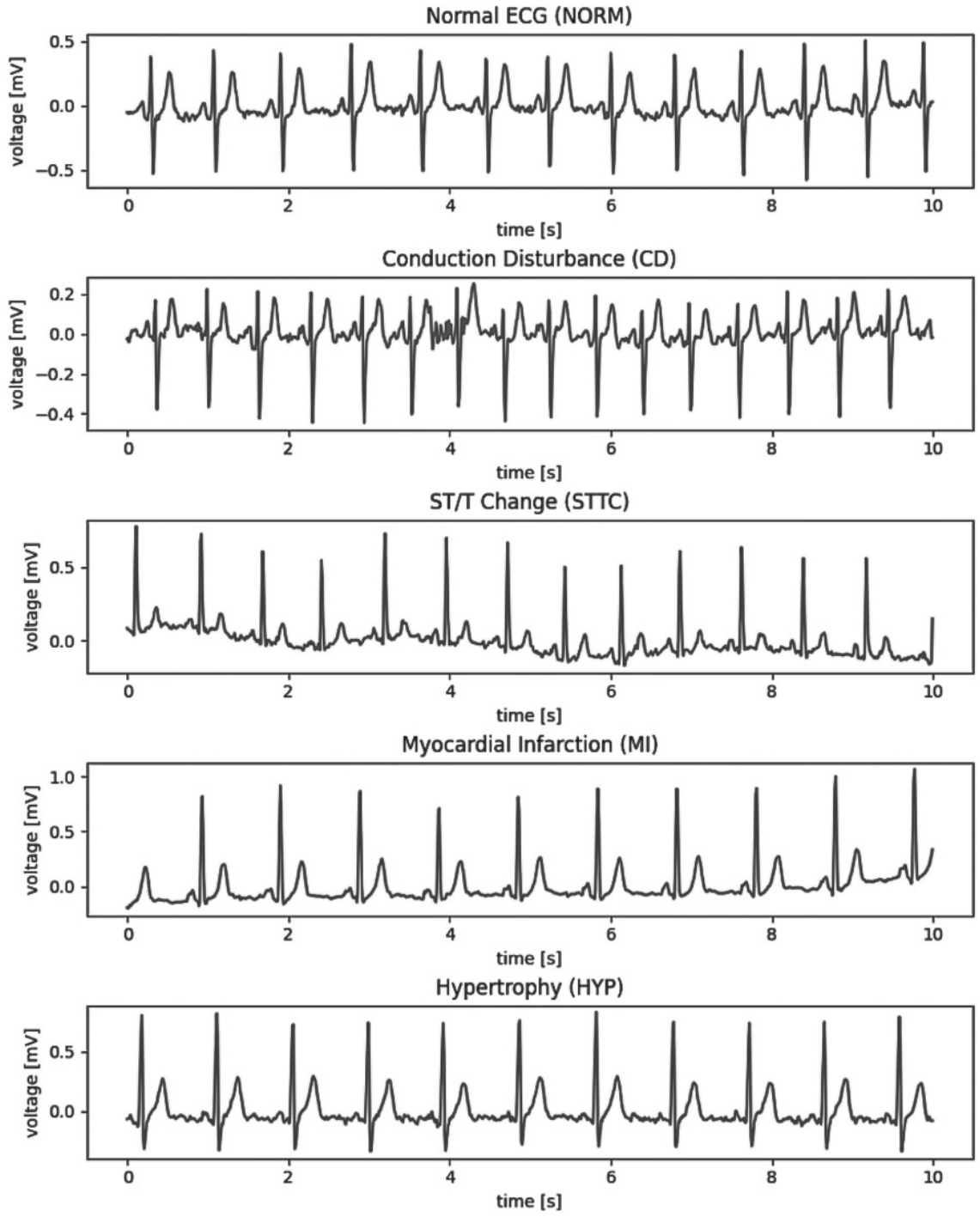


Figure 3.8 Plots of normal and abnormal ECG signals present in the PTB-XL dataset [Śmigiel et al., 2021].

data augmentations. Therefore, the representations become invariant under the chosen augmentations. Formally, this translational symmetry is the invariance of a system under any translation [Agrawal et al., 2015; Cohen and Welling, 2016]. Given a group of transformations G, $T_g(\mathbf{x})$ for any $g \in G$ denotes the function with which g transforms an input signal $\mathbf{x}$. E.g. if G is the

group of time-out augmentations, $T_g(\mathbf{x})$ will set a portion of the signal $\mathbf{x}$ to zero. The encoder network f which maps the input signals to the latent representation space, $f(\mathbf{x})$ promotes the property of invariance under the given translation group G. Formally stating $f(T_g(\mathbf{x})) = f(\mathbf{x})$, which means that the output representation $f(\mathbf{x})$ does not vary with T_g.

Consequently, the choice of appropriate transformations for inducing two semantically equivalent views on the original instance is crucial for the effectiveness of the method. In spite of this, there is not yet a well-defined strategy for selecting data transformations that produce representations containing just the information we require.

How to choose transformations. Naturally, the question becomes: to what transformations should the ECG signal representations be insensitive when used for multi-label classification of ECG characteristics? Recent research, though not extensively studied, has shown that intuition for the ideal transformations to use is indeed dependent on the downstream task [Tian et al., 2020].

The design of augmentations, or the view-selection distribution $\mathcal{T}$, is critical due to its influence on the representation invariances learned. For example, aggressively adding additional noise as augmentation in $\mathcal{T}$ may lead to representations invariant to certain parts of the signal. Therefore, the aim is to create $\mathcal{T}$ which forms representations with enough invariance to be robust to inconsequential variations but not so much as to discard information required by the downstream task. Experiments carried out by [Chen et al., 2020] show that self-supervised learning benefits from stronger data augmentation than supervised learning.

Implementation. Throughout the experiments performed for this thesis work, the view-selection distribution $\mathcal{T}$ is defined following the experiments carried out by [Mehari and Strodthoff, 2022]. In order to find a suitable combination of transformations during pre-training, they performed a grid search based on six transformations. These were partly inspired by computer vision and partly by time series analysis. These transformations were Gaussian noise, Gaussian blur, channel resize, time out, random resized crop, and dynamic time warp. The interested reader is referred to [Mehari and Strodthoff, 2022] for detailed descriptions. Figure 3.9 depicts a heat map of the linear evaluation performance measured in macro-AUC on the PTB-XL validation set of a one-dimensional ResNet-50 after 500 epochs of pre-training using the SimCLR framework. Diagonal entries correspond to a single transformation and off-diagonal entries correspond to the sequential composition of two transformations. As can be seen in Figure 3.9, the combination of random resized crop with time-out augmentation appears to be the most effective transformation pair as it shows the highest evaluation performance. Based on these results, the data transformation module $\mathcal{T}$ is constructed to sequentially apply ran-

dom resized crop augmentation followed by time-out augmentation. The two augmentation techniques are further described in the section below.

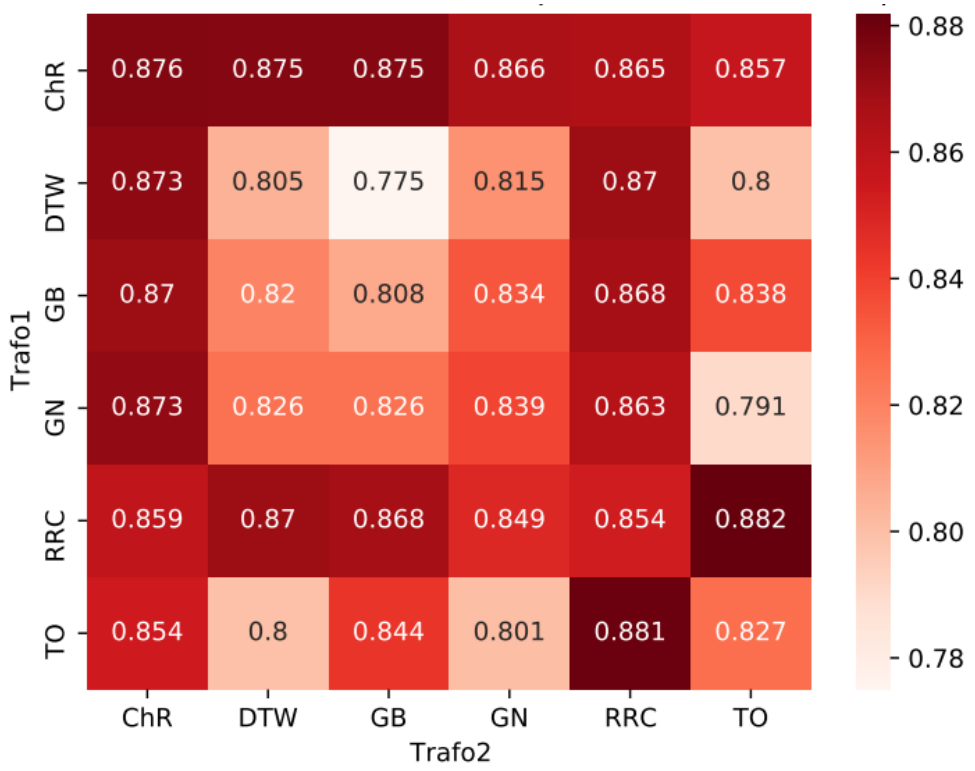


Figure 3.9 Heat-map illustrating linear evaluation performance measured in macro-AUC when applying grid search over different data transformations [Mehari and Strodthoff, 2022].

Random resized crop

Random resized crop cuts a random contiguous segment of the signal and rescales it to its original size. A crop parameter p is sampled uniformly from the range (l, m), where (l, m) are the parameters of the transformation. The default values used in the experiments were $(l, m) = (0.5, 1.0)$. This means that the signal is cropped into portions between 50% and 100% percent and then rescaled to 100 Hz. To give an intuition of the augmentation, applying a random resize crop can be seen as "zooming in" on a random part of the signal. Figure 3.10 shows an example of an original ECG signal and the same transformed version.

Time out

The temporal specific transformation, timeout augmentation [DeVries and Taylor, 2017] sets a random contiguous segment of the signal to zero. The range of the cutout window is determined by the parameters (t_l, t_u), from

Figure 3.10 Example of an original ECG signal and the same signal with random resize crop augmentation and time-out augmentations applied.

which the timeout parameter t is uniformly sampled. The parameter describes how much of the original signal will be set to zero. Throughout our experiments, the default values used were set to $(t_l, t_u) = [0.0, 0.5]$. Thereby a stochastically chosen window with a maximum length of 50% of the original signal was set to zero. Figure 3.10 illustrates an example of an unaltered ECG signal and the same time-out transformed version.

4

Method and experimental implementation

In the following chapter, we describe the experimental setup and implementation details. The chapter begins with Section 4.1, which introduces the baseline implementation consisting of a supervised one-dimensional ResNet-50 model. A comprehensive review of the various pretext tasks and self-supervised methods used in this thesis is then discussed in Section 4.2. Having reviewed the various pre-training methods, Section 4.3 introduces the real task at hand, the downstream task, which involves a supervised multi-label classification task with signal representations extracted from the pre-trained encoder networks. Section 4.4 continues with a discussion of the evaluation methods and metrics we used to assess the usefulness of the ECG representations on the downstream task.

4.1 Supervised baseline

During the course of this thesis work, we compared the downstream performance of the self-supervised representations with a 1-dimensional ResNet-50 model trained on labelled data using a standard binary cross-entropy loss. In this section, we will provide a description of the model architecture, implementation details, and training approach used. As an illustrative point to note, the model trained in this supervised manner is architecturally identical to the encoder networks that were used for the various pretext tasks. Therefore, the supervised performance can be viewed as a benchmark performance of "the performance we will get if we do not pre-train with a self-supervised method". In fact, the architectural identity of the ResNet-50 model with the encoder module is what has led us to use a 1D ResNet-50 model as a representative of supervised learning performance. The pretext models consist of an encoder part as well as an adaption for the network specific to the method, and the pre-trained models are stripped down to consist only of this 1D ResNet-50

model after pre-training. In Section 4.4, the self-supervised learning framework will be fully described; however, it is briefly described here in order to clarify the rationale for selecting a 1D ResNet-50 network to determine baseline performance.

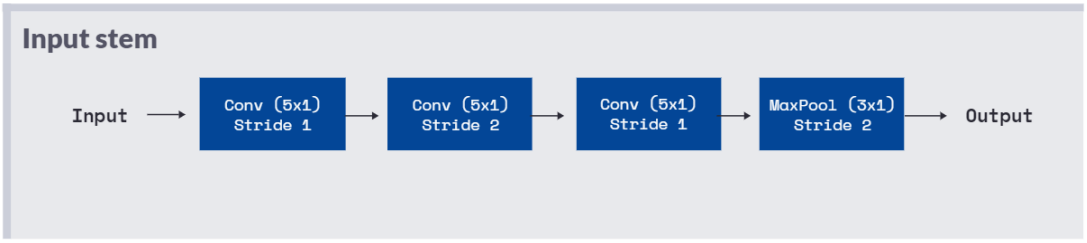


Figure 4.1 The input stem is the first part of the ResNet-50 model. The 12-lead ECG signal is passed through three convolutional layers and one max pooling layer, thus reducing the length of the input signal by a factor of four and increasing the channel size to 64.

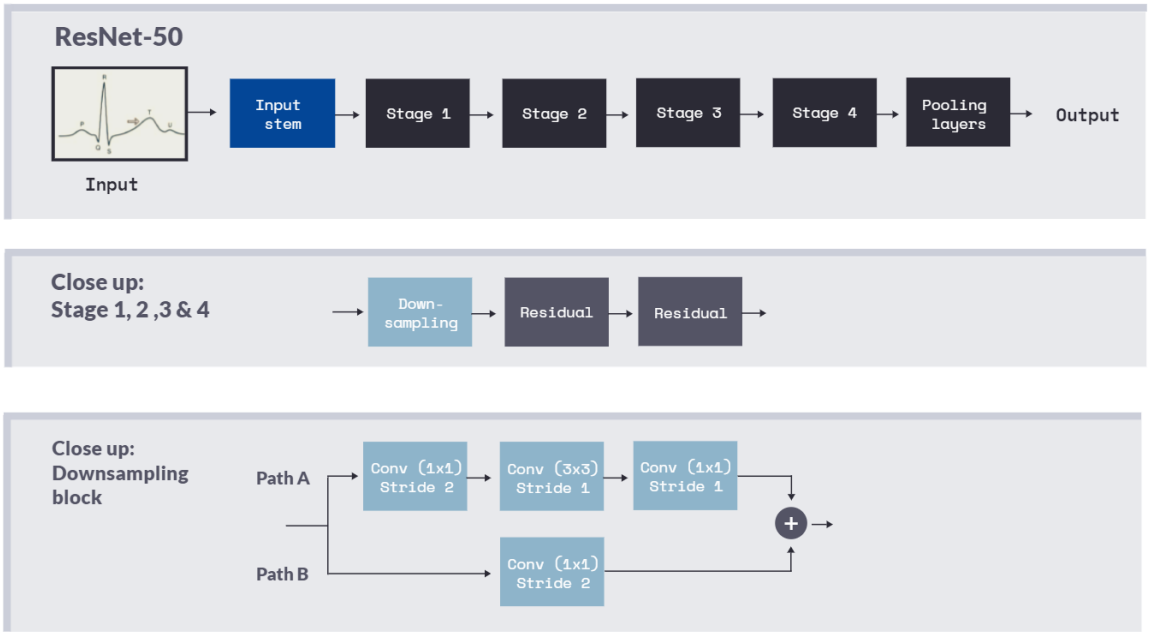


Figure 4.2 The figure presents an overview of the blocks in the ResNet-50 model. It consists of one input stem, four stage blocks and one pooling block.

ResNet50 architecture

Adopting a convolutional residual network [He et al., 2016] with 50 layers (ResNet-50), the network was adapted to fit the one-dimensional ECG recordings with its 12 channels and was trained from random initialization using the

same procedure as for fine-tuning using the full PTB-XL training dataset, see section 4.4.

The implemented ResNet network consisted of an input stem and four subsequent stages followed by a final output layer. Figure 4.2 illustrates the architecture. The input stem consisted of three 5x1 convolutions, shown in Figure 4.1, where the first and second convolutions used 32 output channels. Additionally, the first convolution used a stride of 1, and the second convolution a stride of 2. Using a stride of 1, the last convolution layer gave an output channel depth of 64. Following was a 3×1 max pooling layer which used a stride of 2. The given input stem reduced the input length by 4 times and increases its channel size to 64. Starting from stage 2, each stage began with a downsampling block, followed by several residual blocks. The downsampling block consisted of two paths, A and B.

In path A there were three convolutions with kernel sizes are 1×1, 3×3, and 1×1, respectively. These layers were constructed so that a bottleneck structure was formed. This was done by setting the stride of the first convolution to 2, in order to halve the input length, and letting the output channels of the last convolution be 4 times larger than the previous two. Path B instead used a 1×1 convolution with a stride of 2, transforming the input length to match the output length of path A; allowing for the outputs of both paths to be summed and outputted by the downsampling block. The residual block was similar to the downsampling block, only differing in stride length, where this block only used convolutions with a stride of 1. The final network layers consisted of a 1D adaptive average pooling layer and one 1D adaptive max pooling layer.

Implementation details

During training, a constant learning rate of 0.008 was used to optimize a binary cross-entropy loss. AdamW optimizer was used in combination with a weight decay regularization of 0.001 [Loshchilov and Hutter, 2017]. When training the network using ECG recordings of 10 seconds in length, the batch size was set to 512, whereas a batch size of 2048 was used for networks trained with ECG recordings of length 2.5 seconds. Model performance was measured using macro-AUC, computed from the 71 labels on the most fine-grained level in PTB-XL [Wagner et al., 2020]. The selected model was the one that during training obtained the highest macro-AUC score when evaluated using validation data. Reported metrics are the respective test set score of this selected model.

4.2 Self-supervised pretext task

As presented in Section 2.4, many self-supervised learning methods are based on a joint-embedding architecture. That is also the case for the methods which

were implemented during our experiments. All our adopted self-supervised learning frameworks, regardless of whether they use contrastive or non-contrastive learning approaches, share an architectural pipeline. The pipeline begins with the data augmentation module, which samples transformations $\mathcal{T}$ that will be applied to the data samples. The second module consists of the encoder network, which maps the different views to a latent representation space. Following this step, the representation will either be projected into a lower dimensional space or expanded into a higher dimensional space, depending on the method employed. During the last step of the pipeline, a loss function is minimized in this final representation space.

In the following subsections, the three self-supervised representation approaches that were implemented, trained, and evaluated during our thesis research will be discussed in detail.

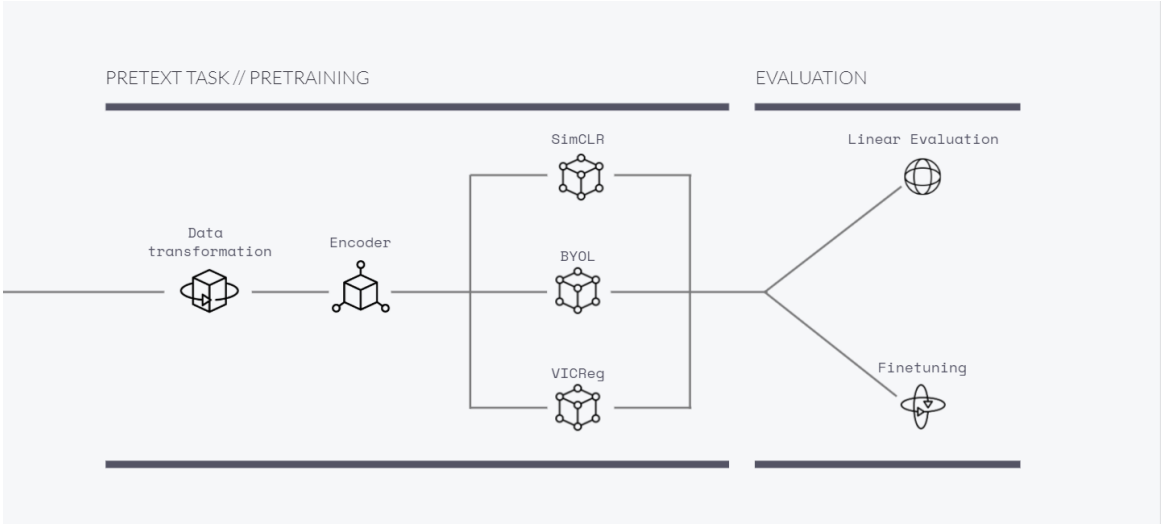


Figure 4.3 Schematic image of the self-supervised learning pipeline. The first phase consists of four steps: 1) Transformation module 2) Encoder module 3) Method-specific module (projection, projection+predictor, expander) 4) Loss function. The second phase is the evaluation process, where either linear evaluation or fine-tuning is used.

Data transformation

The first phase of the learning pipeline consists of a stochastic data transformation module. This module randomly transforms any given data sample to produce two correlated views of this same data instance. The two views are here denoted as $\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j$, and make together up what is considered a positive pair.

The module procedure follows as: Given a dataset $\mathcal{D}$, an ECG signal $\mathbf{x}$ is uniformly sampled as $\mathbf{x} \sim \mathcal{D}$. Two data transformations t and t' are then sampled from the transformation distribution $\mathcal{T}$. By applying respective data

transformation $t \sim \mathcal{T}, t' \sim \mathcal{T}$ to the ECG signal $\mathbf{x}$, the two different signal views are produced as $\tilde{\mathbf{x}}_i = t(\mathbf{x}), \tilde{\mathbf{x}}_j = t'(\mathbf{x})$. These transformations are stochastic resizing crops of the signal, followed by setting a portion of the signal to zero. The distribution $\mathcal{T}$ along with the augmentation details are described in Section 3.3.

Encoder

Even though the self-supervised learning paradigm still is in its infancy, choosing the convolutional residual network architecture, ResNet, [He et al., 2016; Chen et al., 2020] as encoder module has already become a custom. One possible explanation for this is the neural architecture of the ResNet model. Deep neural architectures are often associated with the vanishing gradient problem (the back-propagated gradient becomes so small that the weight stops changing its value) and the degradation problem (adding too many layers to a model will increase the training error). In order to mitigate these issues, the ResNet model uses skip connections, which allow input from one layer to be passed on to a layer further down in the network. With a ResNet model, it is possible to train up to hundreds of layers while still achieving a high level of performance [He et al., 2016].

Following current state-of-the-art methods [Mehari and Strodthoff, 2022] the ResNet-50 architecture was here chosen as the encoder network for all three self-supervised learning approaches implemented in this thesis work. Following the same architectural details as described in Section 4.1, the ResNet-50 network acts as the second module of the representation learning pipeline, succeeding the data transformation module.

Given the two sample views $\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j$ received from the sample transformations applied in the data transformation module $\tilde{\mathbf{x}}_i = t(\mathbf{x}), \tilde{\mathbf{x}}_j = t'(\mathbf{x})$, the base encoder network denoted as $f(\cdot)$ extracts representation vectors from respective data views as $\mathbf{h}_i = f(\tilde{\mathbf{x}}_i) = ResNet(\tilde{\mathbf{x}}_i), \mathbf{h}_j = f(\tilde{\mathbf{x}}_j) = ResNet(\tilde{\mathbf{x}}_j)$. The output from the last layer of the network is denoted as $\mathbf{h}_i, \mathbf{h}_j \in \mathbb{R}^d$ and d is the dimension of this output vector, here set to 512. Hence, signals are after the encoder network represented by an embedding vector in a 512-dimensional latent space. For the Siamese network architectures, as used in SimCLR and VICReg, the encoders on both branches share the same set of weights, see Figure 4.4 and 4.6, while BYOL updates the encoder weights of the target branch according to a moving average of the online branch. A topic Section 4.2 will dive deeper into.

SimCLR

SimCLR [Chen et al., 2020] proposes a framework for contrastive learning by employing a Siamese joint embedding network that learns representations by maximizing the similarity between augmented views of the same data

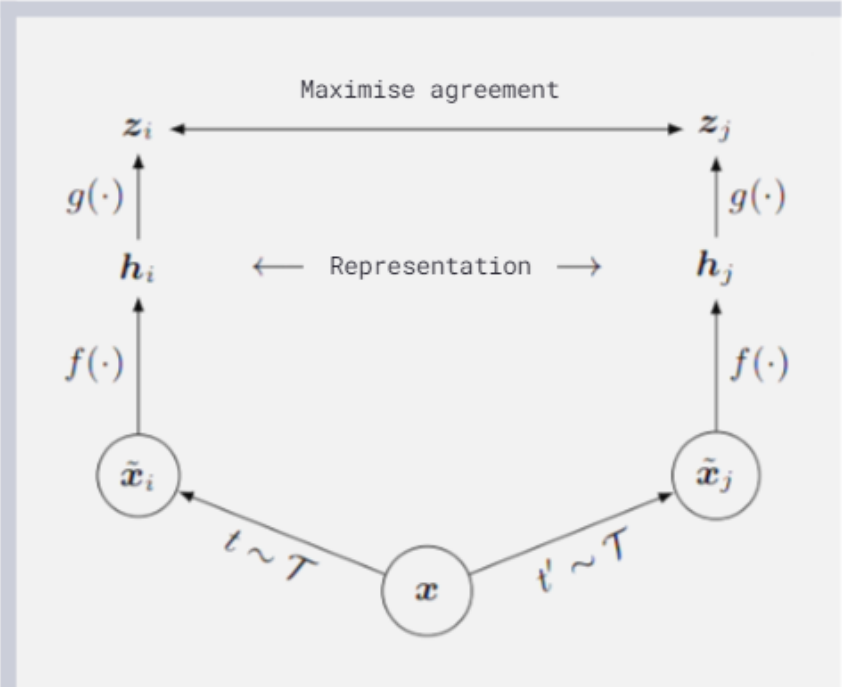


Figure 4.4 Illustration of the SimCLR framework [Chen et al., 2020].

samples in the latent space, using a contrastive cost function. The SimCLR framework is illustrated in Figure 4.4. Given the Siamese network architecture, the two branches are identical and the set of encoder- and subsequent projection module weights, are shared between branches. In formal terms, given two augmented views $\tilde{\mathbf{x}}_i = t(\mathbf{x}), \tilde{\mathbf{x}}_j = t'(\mathbf{x})$ of the same ECG signal $\mathbf{x}$, a set of encoder weights θ, and encoder network f_θ, signal representations are obtained by $\mathbf{h}_i = f_\theta(\tilde{\mathbf{x}}_i) = ResNet(\tilde{\mathbf{x}}_i), \mathbf{h}_j = f_\theta(\tilde{\mathbf{x}}_j) = ResNet(\tilde{\mathbf{x}}_j)$.

The contrastive prediction task is defined on the pairs of augmented signals randomly sampled to form a minibatch of N samples. As each sample $\mathbf{x}$ is represented by two views $\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j$ each minibatch consist of $2N$ data points. The contrastive learning approach requires explicit negative samples to be used during the network's learning phase, albeit negative signals are not deliberately sampled. Instead, the negative samples are defined to be the remaining $2(N-1)$ data points present in the minibatch when the positive pair is not considered.

Projection head. Following the encoder network f_θ, the 512-dimensional signal representations $\mathbf{h}_i$ and $\mathbf{h}_j$ are passed through another neural network; the projection head $g(\cdot)$, which projects the encoder embeddings to a 128-dimensional latent space. This network module consists of two multilayer perceptrons that map the signal representations to the latent space where contrastive loss is applied. Using multilayer perceptrons with one hidden layer

and σ being a ReLU non-linearity, the final signal representations are obtained by $\mathbf{z}_i = g(\mathbf{h}_i) = W^{(2)}\sigma(W^{(1)}\mathbf{h}_i)$ and $\mathbf{z}_j = g(\mathbf{h}_j) = W^{(2)}\sigma(W^{(1)}\mathbf{h}_j)$. Where $W^{(i)}$ is a matrix consisting of the weights and biases of layer i.

Loss function. Given a set of data points $\{\mathbf{x}_k\}$ which includes the positive pair of views $\mathbf{x}_i$ and $\mathbf{x}_j$, the aim of the contrastive prediction task is to identify $\mathbf{x}_j$ in $\{\mathbf{x}_k\}_{k\neq i}$ for a given representation $\mathbf{x}_i$.

The contrastive loss function defined for the contrastive prediction task in the SimCLR framework has previously shown success when adopted by [Sohn, 2016; Wu et al., 2018; Oord et al., 2018]. This loss function is called, NT-Xent, which is short for *the normalized temperature-scaled cross-entropy loss*. Letting $\text{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^\top\mathbf{v}/\|\mathbf{u}\|\|\mathbf{v}\|$ denote the cosine similarity between the vectors $\mathbf{u}$ and $\mathbf{v}$, i.e. the dot product between ℓ_2 normalized vectors, the loss function for a positive pair of samples (i, j) is then defined as in Equation 4.1

$$\ell_{i,j} = -\log \frac{\exp\big(\text{sim}(\mathbf{z}_i, \mathbf{z}_j)\big)/\tau}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\big(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)\big)/\tau}, \tag{4.1}$$

where $\mathbb{1}_{[k\neq i]}$ is an indicator function evaluating to 1 iff $k \neq i$ and τ denotes a temperature parameter set to 0.5 for our experiments, as suggested by [Chen et al., 2020]. The final loss is computed across all positive pairs in a minibatch, both (i, j) and (j, i).

Implementation details. The NT-Xent loss function described in the section above was optimized using layer-wise adaptive rate scaling, LARS [You et al., 2017], with a learning rate of 0.00229 which decayed with the cosine decay schedule without restarts, as described in [Loshchilov and Hutter, 2016]. Furthermore, pre-training was done using a weight decay regularization of 10^{-6}, loss function temperature set to 0.5, and training length of 2000 epochs. Due to hardware memory constraints, a batch size of either 2048 or 512 was used depending on ECG signal length. Additionally, 16-bit floating points were used to reduce memory footprint during model training. The mentioned parameters were the default settings used for the majority of the experiments. Deviating settings are described in chapter 5 for the respective experiments.

BYOL

BYOL [Grill et al., 2020] introduces a self-supervised learning framework that does not require the utilization of explicit negative data pairs. Instead, BYOL replaces the explicit negative samples with an architectural network asymmetry consisting of two neural networks. This is in contrast to the SimCLR method which depends on the explicit definition of positive and negative pairs. As mentioned above, SimCLR forms data representations by reducing

distances between representations of augmented views of the same data samples and increasing distances between representations of augmented views from different data samples. BYOL instead uses two networks which are referred to as online and target networks. Feature representations are created by letting the two networks interact and learn from each other. As in SimCLR, two augmented views of a data sample are used as input data. However, instead of optimizing for similarity between positive pairs and dissimilarity between negative pairs, BYOL trains the online network to predict the representation from the target network when each network respectively is given one of the augmented views from the same positive data pair. At the same time, the weights of the target network are updated with a slow-moving average of the weights of the online network. The BYOL learning framework is illustrated in Figure 4.5.

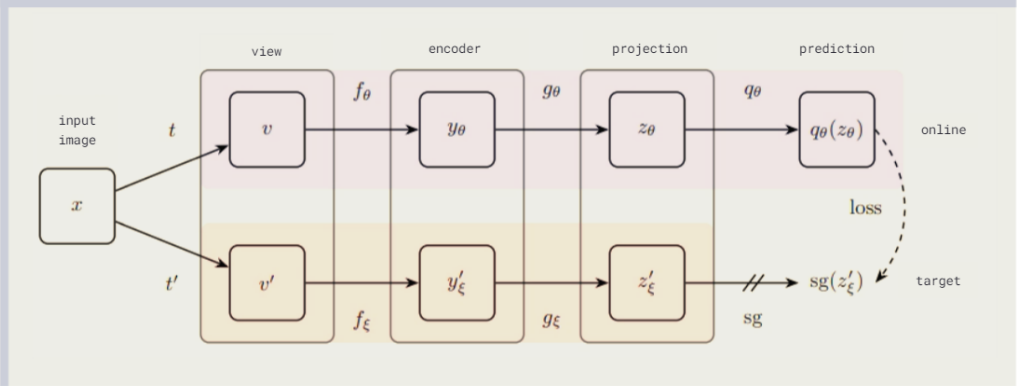


Figure 4.5 Illustration of the BYOL framework [Grill et al., 2020].

In order to learn a representation y_θ, BYOL will, as stated, use the two neural networks: the online and target networks. The online network is defined by a set of weights θ and consists of three stages: an encoder f_θ, a projector g_θ, and a predictor q_θ. The target network copies the architecture of the online network but uses a different set of weights ξ. The online network is trained using the regression targets provided by the target network. The target networks parameters ξ are an exponential moving average of the online parameters θ, and given a target decay rate $\tau \in [0, 1]$, the update in $\xi \leftarrow \tau\xi + (1 - \tau)\theta$ is performed after each training step.

The BYOL learning framework begins in the same manner as that of SimCLR. The data transformation module gives two augmented views $\tilde{\mathbf{x}}_i = t(\mathbf{x}), \tilde{\mathbf{x}}_j = t'(\mathbf{x})$ of the same ECG signal $\mathbf{x}$. The online and target networks, respectively, take one of the two views as input. From the first augmented view $\tilde{\mathbf{x}}_i$, the online network θ firstly represents the ECG signal as a 512-dimension embedding vector by passing it through the encoder module f_θ, $\mathbf{h}_\theta = f_\theta(\tilde{\mathbf{x}}_i) = ResNet_\theta(\tilde{\mathbf{x}}_i)$. The target network ξ uses the second augmented view and its

encoder module f_ξ to form the embedding vector $\mathbf{h}_\xi = f_\xi(\tilde{\mathbf{x}}_j) = ResNet_\xi(\tilde{\mathbf{x}}_j)$.

Projection and prediction. From the signal representations $\mathbf{h}_\theta$ and $\mathbf{h}_\xi$, the online and target projections are created through the projection modules g_θ and g_ξ, $\mathbf{z}_\theta = g_\theta(\mathbf{h}_\theta)$ and $\mathbf{z}_\xi = g_\xi(\mathbf{h}_\xi)$ for the two networks. The projection networks g_θ and g_ξ each consist of a multilayer perceptron that projects the signal representations to a smaller latent space. The multilayer perceptron consists of a linear layer followed by batch normalization [Ioffe and Szegedy, 2015] and a rectified linear unit, ReLU, before a final linear layer. This gives the latent representations $\mathbf{z}_\theta = g_\theta(\mathbf{h}_\theta) = W^{(2)}\sigma(W^{(1)}\mathbf{h}_\theta)$ and $\mathbf{z}_\xi = g_\xi(\mathbf{h}_\xi) = W^{(2)}\sigma(W^{(1)}\mathbf{h}_\xi)$.

Passing the latent representations $\mathbf{z}_\theta$ through a prediction network q_θ, the online network further transforms the latent representation into a prediction vector $q_\theta(\mathbf{z}_\theta)$ which is a prediction of the target representation $\mathbf{z}_\xi$. The prediction network g_θ is architecturally identical to the projection module g_θ. The prediction $q_\theta(\mathbf{z}_\theta)$ and the target representation $\mathbf{z}_\xi$ are both ℓ_2-normalized to $\bar{q_\theta}(\mathbf{z}_\theta) = q_\theta(\mathbf{z}_\theta)/\|q_\theta(\mathbf{z}_\theta)\|_2$ and $\bar{\mathbf{z}}_\xi = \mathbf{z}_\xi/\|\mathbf{z}_\xi\|_2$.

Loss function. The loss function of the BYOL framework is then defined as the following mean squared error between the normalized predictions and target projections, i.e.

$$\mathcal{L}_{\theta,\xi} = ||\bar{q_\theta}(\mathbf{z}_\theta) - \bar{\mathbf{z}}_\xi||_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(\mathbf{z}_\theta), \mathbf{z}_\xi \rangle}{||q_\theta(\mathbf{z}_\theta)||_2 \cdot ||\mathbf{z}_\xi||_2}. \tag{4.2}$$

The loss $\mathcal{L}_{\theta,\xi}$ is symmetrized by subsequently separately feeding view $\tilde{\mathbf{x}}_j$ to the online network and $\tilde{\mathbf{x}}_i$ to the target network to compute $\tilde{\mathcal{L}}_{\theta,\xi}$. This forms the final loss as $\mathcal{L}_{\theta,\xi}^{\text{BYOL}} = \mathcal{L}_{\theta,\xi} + \tilde{\mathcal{L}}_{\theta,\xi}$, which is minimized at each training step by performing a stochastic gradient optimization step with respect to θ, as illustrated by the stop-gradient for ξ in Figure 4.5. The learning dynamics of the BYOL framework can then be summarized as

$$\theta \leftarrow \text{optimizer}(\theta, \nabla_\theta \mathcal{L}_{\theta,\xi}^{\text{BYOL}}, \eta), \xi \leftarrow \tau\xi + (1-\tau)\theta, \tag{4.3}$$

where optimizer is an optimizer and η is the learning rate.

Implementation details. The loss $\mathcal{L}_{\theta,\xi}^{\text{BYOL}}$ was optimized using the LARS optimizer [You et al., 2017] with a cosine learning rate decay schedule, without restarts, over 2000 epochs, with a warm-up period of 10 epochs. A setup that is identical to the optimization scheme for SimCLR.

The base learning rate was set to 0.2 and scaled linearly with the batch size. Furthermore, a global weight decay parameter of $1.5 \cdot 10^{-6}$ was used for regularization, while excluding the biases and batch normalization parameters from both LARS adaptation and weight decay. Applying an exponential moving average parameter τ which starts from $\tau_{\text{base}} = 0.996$

and is then increased to one during training. More precisely, τ is set to $\tau = 1(1 - \tau_{\text{base}} \cdot (\cos(\pi k/K) + 1)/2$, where k the current training step and K the maximum number of training steps. As for SimCLR, a batch size of either 2048 or 512 was used depending on ECG signal length. Additionally, 16-bit floating points were used to reduce memory footprint during model training. The mentioned parameters were the default settings used for the majority of the experiments. Deviating settings are described in Chapter 5 for the respective experiments.

VICReg

The final self-supervised learning framework used in this thesis was VICReg [Bardes et al., 2021]. The VICReg framework is a self-supervised method for training joint embedding architectures and is based on the principle of preserving the information content of the embeddings. Similar to SimCLR, VICReg learns representations by maximizing similarity in the latent space between augmented views of the same data samples. In order to prevent a representational collapse, VICReg introduces two regularization terms. Separately applied to both embeddings of a positive data pair, the loss function in addition to the similarity term introduces a term that maintains the variance of each embedding dimension above a threshold combined with a term that decorrelates each pair of embedding features. Although the VICReg framework does not require a Siamese network, meaning the two branches are not required to share the same parameters, architecture, or input modality, we chose to use a Siamese network architecture. The two branches thereby share network architecture and weights, as illustrated in Figure 4.6.

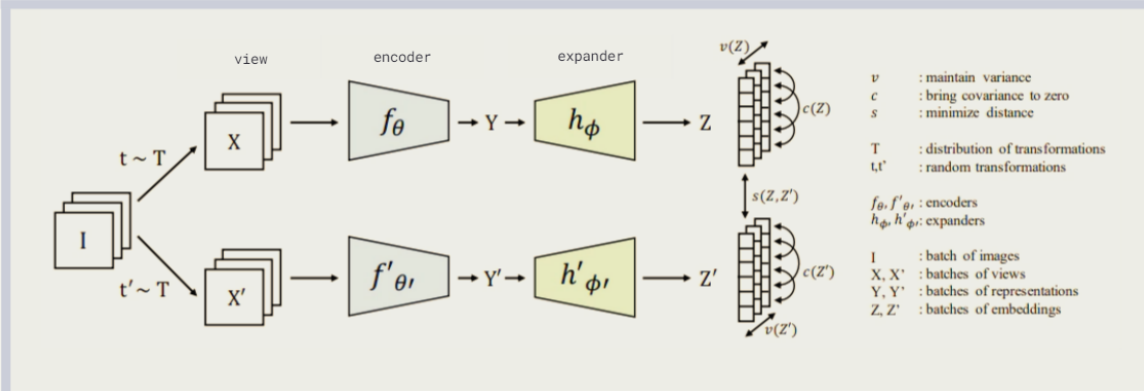


Figure 4.6 VICReg: joint embedding architecture with variance, invariance, and covariance regularization [Bardes et al., 2021].

The main aspect of VICReg is the variance preservation term and the covariance criterion. The variance preservation term explicitly prevents a collapse due to a shrinkage of the embedding vectors towards zero, while the

covariance criterion, borrowed from Barlow Twins [Zbontar et al., 2021], prevents informational collapse caused by information redundancy between the embedding features. With the introduction of these terms, VICReg forms a loss function that aims to preserve the information content of the embeddings of augmented views while keeping superfluous information over dimensions to a minimum.

Sharing the data transformation module and base encoder network with the methods of SimCLR and BYOL, VICReg defines two augmented views of the same ECG recording $\mathbf{x}$ as $\tilde{\mathbf{x}}_i = t(\mathbf{x}), \tilde{\mathbf{x}}_j = t'(\mathbf{x})$. Given the encoder network $f(\cdot)$, representation vectors are extracted from respective data views as $\mathbf{h}_i = f(\tilde{\mathbf{x}}_i) = ResNet(\tilde{\mathbf{x}}_i), \mathbf{h}_j = f(\tilde{\mathbf{x}}_j) = ResNet(\tilde{\mathbf{x}}_j)$, as described in Section 4.2.

Expander. Each Siamese branch ends with an expander module g_ϕ that maps the signal representations outputted from the encoder networks to a continuous latent space where the loss function is computed. Introducing an expander network will eliminate the information differing between the two representations as well as expand the dimensions in a non-linear fashion that decorrelates the embedding variables and reduce the dependencies between the features of the representation vector. Architecturally, the expander network consists of two fully-connected layers with batch normalization [Ioffe and Szegedy, 2015] and ReLU activation σ, and a third linear layer. The sizes of all 3 layers were set to 2048, expanding the encoder space with a factor of 4. This gives the final signal representations $\mathbf{z}_i = g_\phi(\mathbf{h}_i) = W^{(3)}\sigma\big(\mathrm{BN}\big(W^{(2)}\sigma\big(\mathrm{BN}(W^{(1)}\mathbf{h}_i)\big)\big)\big)$ and $\mathbf{z}_j = g_\phi(\mathbf{h}_j) = W^{(3)}\sigma\big(\mathrm{BN}\big(W^{(2)}\sigma\big(\mathrm{BN}(W^{(1)}\mathbf{h}_j)\big)\big)\big)$. Here $W^{(i)}$ is defined to be the matrix consisting of the weights and biases of layer i.

Loss function. Given the two representations $\mathbf{z}_i$ and $\mathbf{z}_j$ given as the output from the expander networks g_ϕ of the two branches, the loss function is computed in this embedding space. VICReg uses a loss function consisting of the following three terms:

- Invariance: learns invariance to data transformations by computing the mean square distance between the embedding vectors.

- Variance: maintains the standard deviation (over a batch) of each embedding feature above a given threshold by using a hinge loss. This term forces the embedding vectors of samples within a batch to be different.

- Covariance: a term that decorrelates the features of each embedding and prevents an informational collapse in which the variables would vary together or be highly correlated. It attracts the covariances (over a batch) between every pair of centered embedding features toward zero.

VICReg's objective function can be understood through the lens of information theory and the information bottleneck objective described in Section 2.1. Applied to this self-supervised learning method, the information bottleneck objective consists in finding a representation that conserves as much information about the sample as possible while being the least possible information about the specific distortions applied to that sample by reducing the information redundancy over feature dimensions.

Given a batch of samples, two augmented versions of the sample batch are denoted as $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_n]$ and $\mathbf{Z}' = [\mathbf{z}'_1, \ldots, \mathbf{z}'_n]$ and are composed of n embedding vectors of dimension d, taken as the output from the two branches of the Siamese network. Denote $\mathbf{z}^j$ to be the vector consisting of each value at dimension j in all vectors in $\mathbf{Z}$. The variance regularization term v is defined as a hinge function on the standard deviation of the embeddings along the batch dimension, as

$$v(\mathbf{Z}) = \frac{1}{d}\sum_{j=1}^{d} \max(0, \gamma - S(\mathbf{z}^j, \epsilon)). \tag{4.4}$$

Here γ is a constant target value for the standard deviation, fixed to 1 as suggested by the VICReg authors [Bardes et al., 2021]. Furthermore, ϵ is a small scalar preventing numerical instabilities and $S(x, \epsilon) = \sqrt{\mathrm{Var}(x) + \epsilon}$ is the regularized standard deviation. Further defining the terms of the loss function, the covariance regularization term c is defined to be the sum of the squared off-diagonal coefficients of the covariance matrix of $\mathbf{Z}$,

$$c(\mathbf{Z}) = \frac{1}{d}\sum_{i \neq j} \big[C(\mathbf{Z})\big]^2_{i,j}, \tag{4.5}$$

where

$$C(\mathbf{Z}) = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{z}_i - \bar{\mathbf{z}})(\mathbf{z}_i - \bar{\mathbf{z}})^\top, \text{ and } \bar{\mathbf{z}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{z}_i. \tag{4.6}$$

Finally, the invariance criterion s is defined as

$$s(\mathbf{Z}, \mathbf{Z}') = \frac{1}{n}\sum_{i=1} ||\mathbf{z}_i - \mathbf{z}'_i||_2^2. \tag{4.7}$$

Combining the tree terms, the overall loss function becomes a weighted average of the invariance, variance, and covariance terms

$$\ell(\mathbf{Z}, \mathbf{Z}') = \lambda s(\mathbf{Z}, \mathbf{Z}') + \mu\big[v(\mathbf{Z}) + v(\mathbf{Z}')\big] + \upsilon\big[c(\mathbf{Z}) + c(\mathbf{Z}')\big] \tag{4.8}$$

with λ, μ, and υ being hyper-parameters controlling the weighting of each term in the loss. Following the original implementation of [Bardes et al., 2021],

the hyper-parameters are set to $\upsilon = 1, \lambda = \mu = 25$. The overall loss function evaluated over a full dataset $\mathcal{D}$ then becomes

$$\mathcal{L} = \sum_{I \in \mathcal{D}} \sum_{t,t' \sim \mathcal{T}} \ell(\mathbf{Z}^B, \mathbf{Z}'^B), \tag{4.9}$$

where Z^B and Z'^B are the batches of vector representations corresponding to the batch of data samples B transformed in the data transformation module by t and t'.

Implementation details. The VICReg network is pre-trained using the same training protocol as for BYOL [Grill et al., 2020]. It is trained for 2000 epochs using LARS optimizer [You et al., 2017] with a weight decay of 10^{-6} and a learning rate with a cosine decay learning rate schedule with a base learning rate set to 0.2, starting from 0 with 10 warmup epochs and with a final value of 0.002. The loss functions hyper-parameters are set to the values presented in the section above, with the additional $\epsilon = 0.0001$ in equation 4.4. Batch sizes are chosen in accordance with the SimCLR and BYOL implementations to be either 2048 or 512.

4.3 Downstream task

At the end of the day, self-supervised learning, and deep learning in general, are used to solve a given problem. During the course of this thesis, the question of representation learning is merely a minor barrier to overcome in the pursuit of solving the final problem: the assertion of optimal cardiovascular health. As a representative of this larger undertaking, a smaller, well-defined downstream task is formulated. Within the scope of this thesis, the downstream task is framed as a multi-label classification problem, allowing for the 71 labels on the most fine-grained level in PTB-XL to be fully utilized [Wagner et al., 2020]. In sum, the downstream task is intended to assess how well the signal representation can solve the main problem presented in this thesis.

The training set for multi-label classification consists of data samples, each associated with a set of labels. After training or fine-tuning a model with the training data, the assumption is that the model should have learned the mapping of inputs x to binary vectors y. That is, it assigns a value of 0 or 1 for each label in y, where 0 means that the label is not associated with this input sample, and naturally, a value of 1 signals a positive label association. The task is then to predict the label sets of unseen data samples. Formulating the downstream task as a clinical multi-label ECG classification problem, the PTB-XL dataset with its 71 annotation labels is adopted for the evaluation process. See Section 3.3 for a detailed description of the dataset and Figure 4.7 for a schematic image of the classification task. During the

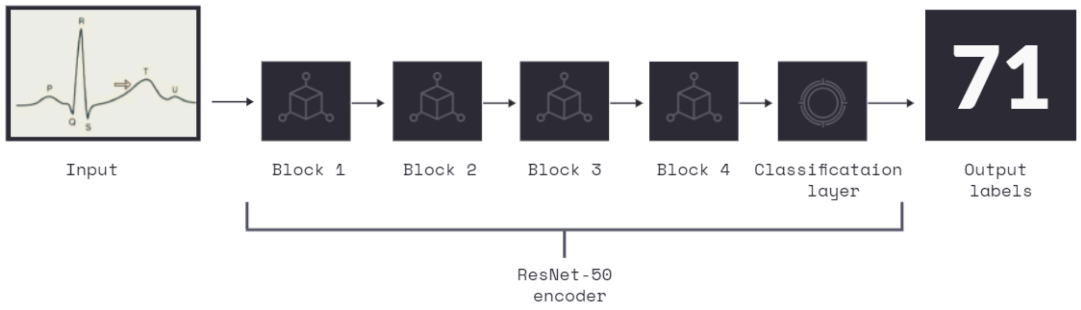


Figure 4.7 Schematic image of the downstream task. It is framed as a multi-label classification problem, allowing for the 71 labels on the most fine-grained level in PTB-XL to be fully utilized.

multi-label classification process, a binary cross-entropy loss is optimized. This binary cross-entropy loss consists of a sigmoid activation function followed by a computation of the cross-entropy loss

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i)\,, \tag{4.10}$$

where N is the number of labels, $\hat{y}_i$ is the ith label in the output vector and y_i the corresponding target value. This loss is independent for each vector component i.e. label, meaning that the loss computed for every output vector component is not affected by other component values. Hence, this is the common choice when solving multi-label classification tasks.

Further discussion of how the ECG representations are evaluated is provided in the following section.

4.4 Evaluation

To validate the effectiveness of the learned signal representations, the model is evaluated using the widely adopted evaluation approaches [Kolesnikov et al., 2019; Wu et al., 2018; Oord et al., 2018; Bachman et al., 2019]:
(i) linear evaluation protocol, where the aim is to access the quality of the learned representations through their linear separability.
(ii) fine-tuning protocol, in which the usefulness of the learned representations for downstream tasks will be investigated by evaluating the model performance on clinically relevant classification- and anomaly detection tasks.

Following pre-training, the method-specific network adjustments are removed, leaving only the ResNet-50 encoder network for use as a feature extractor. See Figure 4.8 for a graphical illustration of the adjustments made to the learning framework as the pre-training phase passes over to evaluation.

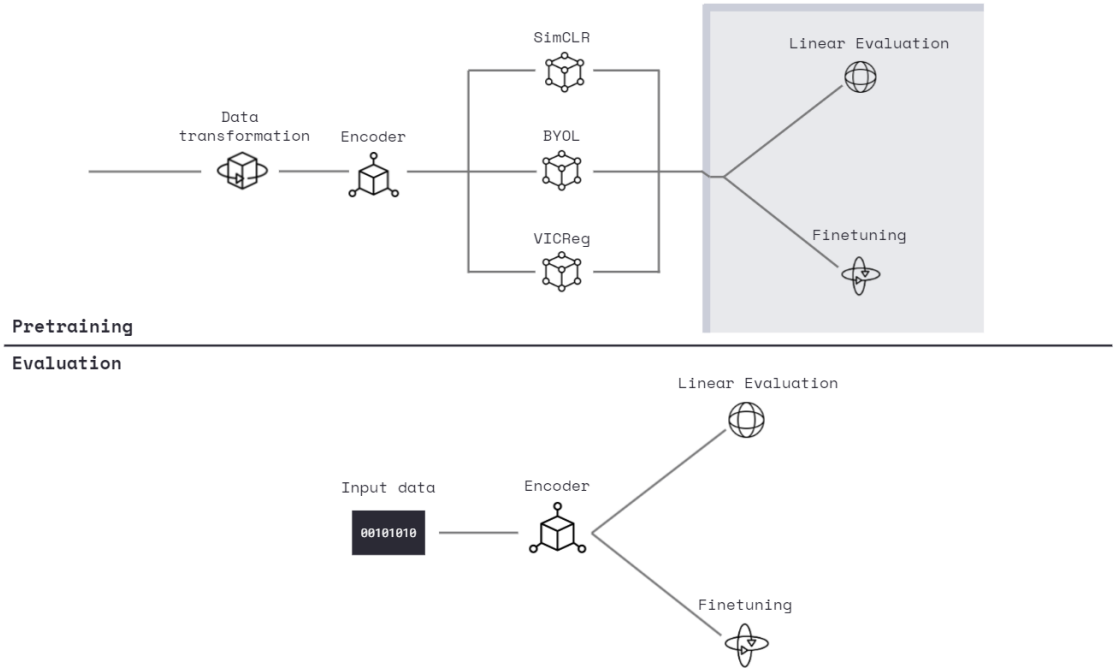


Figure 4.8 After the pre-training phase, all modules except the encoder network are removed. The output from the encoder network is used as a latent representation of the ECG signal.

The ECG signals are passed through the encoder network and the signal features, or representations, outputted by the encoder network are evaluated. In the evaluation process, an output layer containing multiple nodes is added at the end of the network. Each node maps to a distinct ECG label and serves the purpose of outputting a score for the probability that the given input sample is associated with that node label.

Linear evaluation

The linear evaluation protocol aims to assess the quality of the learned representations through their linear separability. To this end, the classification head of the network is replaced by a single linear layer. All other layers, as well as batch normalization statistics, are frozen before the linear classifier is trained on top of the frozen base network. Figure 4.9 presents a schematic of linear evaluation and fine-tuning. The linear layer is trained for 100 epochs with the labelled PTB-XL training dataset. Using AdamW optimizer[Loshchilov and Hutter, 2017], the learning rate is set at 0.008 with a weight decay regularization of 0.001 as suggested by [Mehari and Strodthoff, 2022]. As a measure of the quality of the representation and the usefulness of the signal representations in clinical tasks, the test macro-AUC score is used.

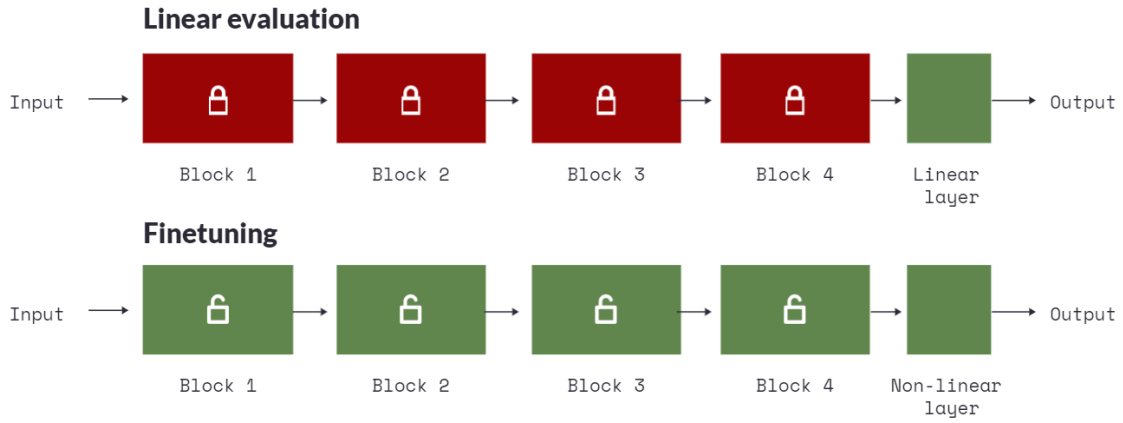


Figure 4.9 Illustration of linear evaluation and fine-tuning procedures. The red color indicates a frozen layer during training, while green represents a tunable layer.

Fine-tuning

Within the fine-tuning protocol, the usefulness of the signal representations for downstream tasks is investigated. This evaluation protocol, in contrast to linear evaluation, does not freeze the network layers but allows for a fine-tuning of all the network weights. For this semi-supervised fine-tuning, the classification head as well as all layers of the pre-trained model are unfrozen and the model is trained now using the labelled data. Figure 4.9 presents a schematic image of linear evaluation and fine-tuning.

Fine-tuning can be further divided into two procedures. One involves fine-tuning the encoder based on the entire labelled training dataset, and the other involves using only a subset of the labelled training dataset. This allows for model performance to be measured as a function of data subset size, where the subset consists of 1%, 10%, 25%, 50%, 70% and 100% of the labelled training data, see Figure 4.10. As much of the motivation behind self-supervised learning lies in the improvement of model performance in the low-data regime, this evaluation routine reflects the expectation that self-supervised learning will represent data better than its supervised counterpart.

During fine-tuning, as with linear evaluation, binary cross-entropy is optimized using AdamW optimizer [Loshchilov and Hutter, 2017] with a constant learning rate of 0.008 and a weight decay regularization of 0.001. Performance is measured using macro-AUC from 71 labels in PTB-XL [Wagner et al., 2020] as with the fully supervised model and the linear evaluation protocol. The selected model is the one that during training obtains the highest macro-AUC score when evaluated on the validation data. The reported metrics are the test results for this selected model.

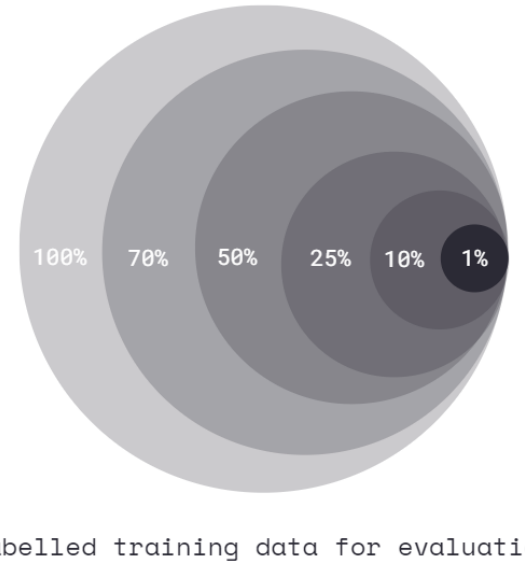


Figure 4.10 Fine-tuning is also performed with subsets of 1%, 10%, 25%, 50%, 70% and 100% of the labelled training data.

Multi-label classification metrics

Presented below are the metrics used to measure the performance on the downstream task and a brief discussion of the multi-label classification metric that was chosen.

In supervised learning, the generalization performance of the learning system is evaluated using metrics such as accuracy, F-measure, area under the ROC curve with the closely related AUC score, etc. However, performance evaluation in multi-label learning is more challenging than in traditional single-label settings, as each example can be associated with multiple labels simultaneously. Consequently, conventional metrics cannot be used as a means of measuring performance. Instead, a number of evaluation metrics specific to multi-label learning have been proposed. Figure 4.11 displays a simplified schematic of the taxonomy of multi-label metrics, where example-based metrics [Ghamrawi and McCallum, 2005] and label-based metrics [Tsoumakas and Vlahavas, 2007] compromise the major groups. The common approach for evaluating the performance of tasks of this sort is to treat each label as its own binary task; n possible labels can be seen as n binary classifiers [Zhang and Zhou, 2013], but the groups differ in the subsequent computational algorithms. Where example-based metrics work by evaluating performance on each test example separately and then returning the mean value across the whole test set, label-based metrics instead evaluate a model's performance on each class label separately, before returning the macro-/micro-averaged value across all class labels. The latter approach is used in this thesis, and

macro-AUC serves as the evaluation metric.

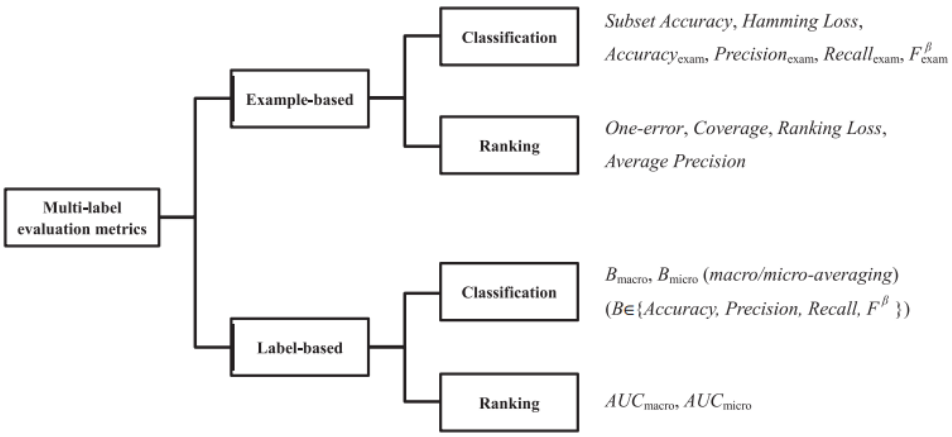


Figure 4.11 Summary of major multi-label evaluation metrics.

The use of a metric based on raw classifier outputs allows a better understanding of the discriminatory power of the classification model. As no thresholding is applied, the selection of an appropriate classifier is disentangled from the issue of threshold optimization. A process that is highly dependent on the clinical application context in which the model will be deployed. Therefore, macro-AUC is used as a performance measure in this thesis. A choice further motivated by its compensation for the discrepancy between the distribution of pathologies in the dataset and the natural distribution of the population. Macro-averaging takes into consideration the expected dataset label imbalance and hence, the score is not dominated by a few, from a statistical perspective, exceedingly large classes resulting from the data collection process. Averaging label-wise AUC scores over all 71 labels in the PTB-XL dataset gives the term centric macro-AUC; the metric used as the primary evaluation metric.

4.5 Defining positive pairs

The topic of how to define the two context views remains open for debate. Not only is the issue of which augmentations to apply and how, but also that of how to select the positive contrastive pair. So far, these choices have been mostly driven by intuition, with little formal understanding of why certain choices may be preferable, and how these choices can be generalized [Patrick et al., 2021].

Recall that the role of positive pairs in self-supervised learning is to define multiple views of a shared context, from which the mutual information between the feature representations of every positive pair is maximized. Maximizing mutual information between features extracted from these views re-

quires capturing information about high-level factors whose influence spans multiple views. The usefulness of the feature representations is thereby dependent on the information present in each view of a positive pair.

Current state-of-the-art approaches define a positive pair as two augmented copies $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_j$ of a data sample $\mathbf{x}$ [Chen et al., 2020; Grill et al., 2020; Bardes et al., 2021; Zbontar et al., 2021; He et al., 2020]. Throughout our experiments, the default approach for defining a positive pair was to apply the sampled transformations t and t' to a signal $\mathbf{x}$, creating two augmenting copies of the same signal, $\tilde{\mathbf{x}}_i = t(\mathbf{x}), \tilde{\mathbf{x}}_j = t'(\mathbf{x})$. The two augmented views were then considered a positive pair $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$. However, for some of the experiments, positive pairs were defined such that one augmented copy $\tilde{\mathbf{x}}_i = t(\mathbf{x})$ was paired with the original signal $\mathbf{x}$, giving a positive pair as $(\tilde{\mathbf{x}}_i, \mathbf{x})$.

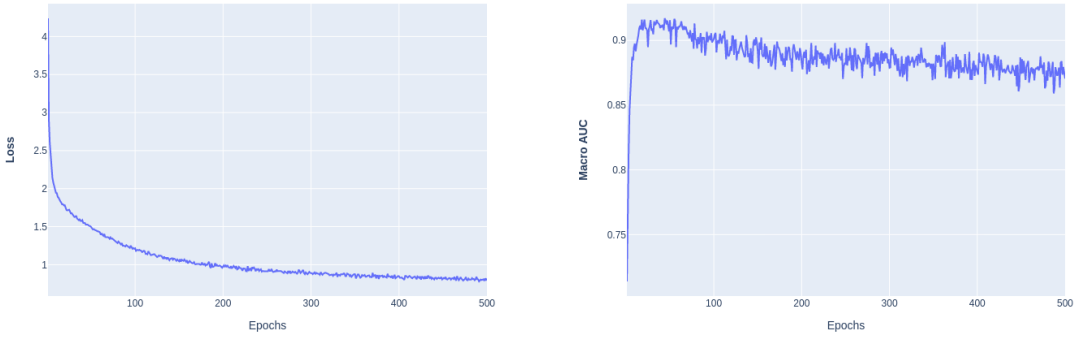
5

Experiments and results

This chapter presents experimental motivations in combination with the most important results and evaluations of the implemented self-supervised representation learning methods. Initially, the chapter presents the experiments that have been conducted in order to establish a benchmark score for how well a non-pre-trained model may perform. In the next experiment, we aim to evaluate whether the performance of the models can be improved through self-supervised pre-training. Following, we conduct another experiment that investigates how performance varies as feature representations are extracted from different stages of the network. Next, the performance of self-supervised pre-trained and non-pre-trained models is evaluated when fine-tuned using smaller subsets of the labelled data. The last experiments investigated how alterations to the learning framework affect model performance when they are fine-tuned on both the full labelled dataset and smaller subsets of the data. To this end, 10-second ECG signals, stronger augmentation strategies, and a redefinition of the positive contrastive pairs are investigated. Lastly, a one-sided Welch's test is performed to analyze the statistical significance of the performance improvements resulting from these framework alterations.

5.1 Supervised baseline results

To begin, we trained a randomly initialized ResNet-50 model in a supervised manner in order to obtain a performance benchmark. This supervised performance can be thought of as the obtainable performance score of a ResNet-50 model without any prior self-supervised training. Figure 5.1 presents training loss and macro-AUC on test data for each of the total 500 training epochs. The model was trained using the full training dataset of PTB-XL, $\mathcal{D}_{PTB}$, and the task was formulated as a multi-label classification task using the 71 labels. Macro-AUC was evaluated on the test dataset. Observing Figure 5.1b we see that the highest macro-AUC is obtained after $\sim$20 training epochs. After that, performance diminishes.



(a) Training loss vs epoch. **(b)** macro-AUC on testdata vs. epoch.

Figure 5.1 Loss and test metrics as a function of training epoch for a ResNet-50 model trained on the full PTB-XL dataset in a supervised manner.

These findings led us to train another model with fewer training epochs. The same setting was used as for the previous experiment, but this time we only let the model's weights be updated for 20 epochs. Its performance was evaluated using linear evaluation and fine-tuning and the results are presented as the first row in Table 5.1. In this supervised setting where no pre-training has been conducted, linear evaluation means that the network is as usual randomly initialized, but its weights are then directly frozen. The linear classification layer added for the linear evaluation procedure is then the only layer for which weights are updated. Fine-tuning in this supervised setting is performed as regular supervised training of a model. Table 5.1 displays that a model trained with all the available labelled data at its highest reaches a macro-AUC of 0.9157.

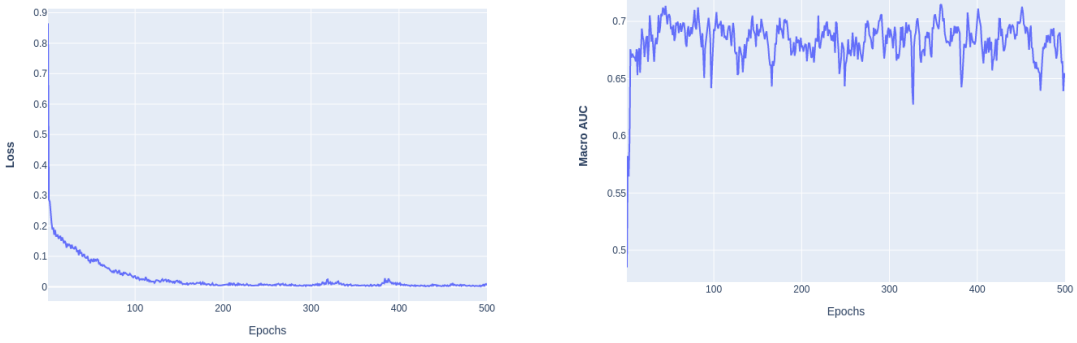
We then carried out a second baseline experiment. This time, a ResNet-50 model was trained using longer ECG signals. This new model was trained with the same setting and hyper-parameters as for the previous baseline experiment. Only this time, the model was trained with 10-second long ECG signals instead of the previous length of 2.5 seconds. The performance results after 20 training epochs are presented as the second row in Table 5.1. It can be seen that performance is improved both in the linear and fine-tuning evaluation settings. When performing two identical model fittings with the only variable being the length of the ECG signals, performance is improved by 2.97 percentage points for linear cases and by 8‰ per mill for fine-tuning cases.

The premise of self-supervised learning is to find representations of unlabelled data that can be effectively applied to various prediction tasks with no, or little adjustments to the feature representations. This framework is mostly used when the availability of labelled data is scarce. Therefore, it is of interest to investigate how a supervised model performs when very few labelled examples are available. This would indicate how well a model is able to perform in

Table 5.1 Supervised ResNet-50 macro-AUC on testdata. Evaluated using linear evaluation and fine-tuning.

Signal length *(s)*	Linear evaluation	Fine-tuning
2.5	0.6606(.0019)	0.9157(.0037)
10	0.6903(.0018)	0.9165(.0032)

that setting without any prior self-supervised training. To simulate this scenario, a ResNet-50 model was trained on 1% of the initial, full dataset. Given the training dataset's original size of 17,441 samples, we this time trained the model using only 174 labelled examples. The chosen recordings were uniformly sampled from the full training dataset at the beginning of the training procedure. Meanwhile, validation and test datasets remained at their original sizes. Figure 5.2 displays training loss and test macro-AUC for each of the 500 training epochs. If we compare Figure 5.2a to Figure 5.1a, it is evident that training loss is minimized after fewer training epochs when available training data is less. However, model performance measured in macro-AUC in Figure 5.2b remains at a more constant level than in Figure 5.1b, although with a higher fluctuation. Given the plots in Figure 5.2, a reasonable trade-off between macro-AUC performance and model generalizability is found after 20-50 training epochs.



(a) Training loss vs. training epoch.

(b) Macro-AUC on testdata vs. training epoch.

Figure 5.2 Loss and test metrics as a function of training epoch for ResNet-50 model trained in a supervised manner on 1% of the dataset.

Our last benchmark experiment examined the performance of the supervised model in relation to the number of labelled examples. To this end, a randomly initialized ResNet-50 model was trained multiple times, each time using an increasing number of data samples. As such, model performance is

given as a function of data availability. The plot in Figure 5.3 shows macro-AUC performance on test data for a model trained on a given % of the full training dataset. Using only 1% of the dataset, a macro-AUC of 0.6980 was reached with 20 epochs of training.

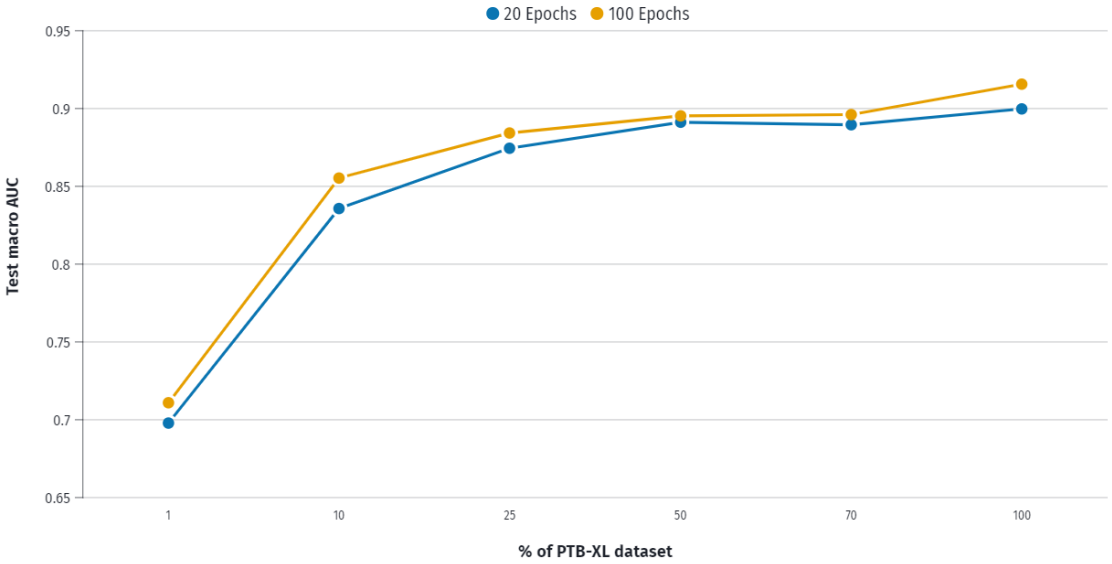


Figure 5.3 Performance of supervised ResNet-50 model as function of % labelled training data.

5.2 Reproducibility and VICReg implementation

Following the initial benchmarking experiments, a second experiment was conducted to determine if self-supervised pre-training would result in higher downstream performance compared to the non-pre-trained supervised model. In particular, the experiment examined the potential of self-supervised learning methods in the context of ECG signals. To this end, and to validate the results obtained by [Mehari and Strodthoff, 2022], we began by implementing the two learning frameworks SimCLR [Chen et al., 2020] and BYOL [Grill et al., 2020]. They were adapted to the one-dimensional ECG data, as outlined in Section 4.2. Both the SimCLR and BYOL models were pre-trained for 2000 epochs using the full unlabelled dataset $\mathcal{D}_{Pre-train}$. Section 3.2 describes the dataset in detail. The only difference in the setup between [Mehari and Strodthoff, 2022] and ours was the training batch size. The reason for this was due to the smaller RAM memory provided by our GPUs.

The pre-trained models were then evaluated using the linear evaluation and fine-tuning protocols. For each evaluation process, the unfrozen layers of the pre-trained encoder network were tuned for 100 training epochs. For the

evaluation setting, we used the labelled PTB-XL training data, $\mathcal{D}_{PTB}$. Please refer to Section 4.4 for detailed information regarding evaluation procedures.

Table 5.2 and 5.3 presents mean macro-AUC on the test data over 10 evaluation runs for models selected by either the lowest pre-training validation loss or by the last training epoch. Despite the smaller batch sizes, we see that performance of our implemented models is in accordance with the results and conclusions provided by [Mehari and Strodthoff, 2022]. Thus, confirming the applicability of self-supervised learning methods in the ECG domain. Nevertheless, since supervised learning yields similar results to those of pre-trained methods, self-supervised pre-training with the current setup does not provide any significant advantage.

Table 5.2 Model selected by lowest validation score. Mean and standard deviation of test macro-AUC over 10 evaluation runs.

Method	Linear evaluation	Fine-tuning
SimCLR	**0.8852(.0112)**	0.9105(.0159)
BYOL	0.8715(.0163)	**0.9107(.0125)**
VICReg 4x	0.8731(.0116)	0.9068(.0120)
VICReg 2x	0.8802(.0086)	0.9040(.0096)
VICReg 1.5x	0.8818(.0120)	0.9091(.0109)

Table 5.3 Model selection by last epoch. Mean and standard deviation of test macro-AUC over 10 evaluation runs.

Method	Linear evaluation	Fine-tuning
SimCLR	**0.8828(.0032)**	0.9148(.0035)
BYOL	0.8659(.0010)	**0.9158(.0033)**
VICReg 4x	0.8690(.0019)	0.9095(.0056)
VICReg 2x	0.8719(.0025)	0.9130(.0047)
VICReg 1.5x	0.8791(.0063)	0.9107(.0027)

Our next experiment was inspired by the recent success of the self-supervised learning method VICReg [Bardes et al., 2021]. In addition to the previously used SimCLR and BYOL methods, we now also implemented the VICReg framework. This was done in order to investigate whether self-supervised pre-training using the VICReg framework would result in improved downstream performance. In this regard, an ECG-compatible modification of

the original network was implemented and trained following the same specifications as for the SimCLR and BYOL methods. That is, 2,000 epochs of training using the unlabelled dataset $\mathcal{D}_{Pre-train}$. This model referred to as VICReg 4x, introduces a dimensional expansion of factor 4 between the encoded representation and the expander network described in Section 4.2. From each pre-training round, we chose to save and evaluate two different models. These were selected as the model with the lowest pre-training validation loss and the model at training epoch 2,000. Table 5.2 and 5.3 presents the mean macro-AUC score on the test data after 10 evaluation runs in the linear evaluation and fine-tuning settings.

As previously mentioned, the VICReg method expands the latent space by, after the encoder, introducing the expansion module. To determine whether a different expanding factor would perform better given our current context, we implemented two additional VICReg versions where we varied the expansion dimensions. The latent dimensions were expanded by a factor of 2 or 1.5 resulting in the names VICReg 2x and VICReg 1.5x. Their respective linear evaluation and fine-tuning performances are also found in Table 5.2 and 5.3.

Based on the evaluation of the initial method implementations, no substantial improvements in performance are observed as a result of self-supervised pre-training. Choosing a BYOL model from the last training epoch gives us a mean macro-AUC of 0.9158, only 1‰ better than using a self-supervised approach. With the current setup, self-supervised pre-training does not provide any meaningful advantages over supervised training alone.

5.3 Extracting feature representations

The presented and implemented methods discard their projector, or expander, networks after pre-training, leaving the encoder network as a feature representation extractor. Adding a classification layer on top of this encoder network, the models are then adapted to solve the downstream task of interest. The ability to perform pre-training with module-based network designs by varying the architectural choices has resulted in highly useful data representations for a variety of tasks. There is, however, a limited understanding of how network architecture influences representation formation in self-supervised learning. More specifically, how does the choice of network modules affect the final learned representations? Do these varying architectures also learn different intermediate hidden layer features with discernible differences in the representations?

This experiment was conducted in order to gain a better understanding of how feature representations were formed. The experiment assessed the downstream performance of feature representations extracted from different stages of the ResNet-50 encoder network. In the previous experiment, representa-

tion vectors were obtained from the last layer of the ResNet-50 model. The last layers of a convolutional neural network are likely to contain task-specific features, as convolutional neural networks acquire complex feature representations by a layer-hierarchical process. Each layer forms a feature representation that builds upon the feature representation from the previous layer. In our setting, these feature representations are shaped by the pretext task we choose to employ. Thus, one can speculate that the feature representations most useful for our given downstream task may not necessarily be extracted from the last ResNet layer. With the last layers perhaps being too biased and adjusted towards the pretext task, the earliest layers will extract more task-agnostic features.

Therefore, for our next experiment, we pre-trained the models as we did in the previous experiment. After pre-training their projector/expander modules were discarded, leaving only their encoder networks to be used for feature extraction. Following the routine linear evaluation and fine-tuning procedures, the full ResNet model was employed as an encoder network by placing a classification layer at the end. This meant that feature representations from the last stage of the network were used for the downstream task. Figure 4.2 displays the different stages of the ResNet model. If, however, we place the classification layer after a layer in the earlier parts of the network, the feature representations from this stage are used for the downstream task. Table 5.4-5.7 present test macro-AUC over 10 evaluation runs using feature representations from varying stages of the network. In the result tables, "stage 2+3" is a concatenated feature vector where feature vectors from network stage 2 and 3 are extracted and concatenated to form a single feature vector of 1024 dimensions. Similarly for the entry "stage 3+4".

Table 5.4 Model selection by lowest validation score. Mean and standard deviation of macro-AUC on test data over 10 evaluation runs. Evaluated using linear evaluation.

Method	Stage 4	Stage 3	Stage 2	Stage 1	Stage 3+4	Stage 2+3
SimCLR	0.8852(.0112)	**0.8913(.0024)**	0.8789(.0021)	0.8652(.0027)	0.8849(.0032)	0.8886(.0023)
BYOL	0.8715(.0163)	0.8739(.0011)	0.8590(.0016)	0.8473(.0020)	0.8671(.0021)	0.8618(.0030)
VICReg 4x	0.8731(.0116)	0.8856(.0017)	0.8789(.0027)	0.8577(.0025)	0.8712(.0043)	0.8788(.0032)
VICReg 2x	0.8802(.0086)	0.8856(.0017)	0.8744(.0024)	0.8536(.0030)	0.8784(.0035)	0.8775(.0047)

Results from Tables 5.4-5.7 show that linear evaluation performance for all methods is improved when using representations extracted from earlier parts of the network. In the regular linear evaluation setting with representations from stage 4, the macro-AUC of 0.8852 obtained by SimCLR was the highest score. By instead extracting representations from stage 3, macro-AUC performance is elevated to 0.8991. Furthermore, linear evaluation performance is increased for all methods when using representations from stage 3 rather than

Table 5.5 Model selection by last epoch. Mean and standard deviation of macro-AUC on test data over 10 evaluation runs. Evaluated using linear evaluation.

Method	Stage 4	Stage 3	Stage 2	Stage 1	Stage 3+4	Stage 2+3
SimCLR	0.8828(.0032)	**0.8991(.0018)**	0.8869(.0022)	0.8651(.0020)	0.8916(.0023)	0.8988(.0018)
BYOL	0.8659(.0010)	0.8671(.0013)	0.8596(.0016)	0.8378(.0015)	0.8670(.0014)	0.8625(.0020)
VICReg 4x	0.8690(.0019)	0.8815(.0026)	0.8800(.0027)	0.8614(.0031)	0.8736(.0031)	0.8757(.0049)
VICReg 2x	0.8719(.0025)	0.8884(.0025)	0.8775(.0027)	0.8523(.0028)	0.8792(.0035)	0.8851(.0026)

Table 5.6 Model selection by lowest validation score. Mean and standard deviation of macro-AUC on test data over 10 evaluation runs. Evaluated using fine-tuning.

Method	Stage 4	Stage 3	Stage 2	Stage 1	Stage 3+4	Stage 2+3
SimCLR	0.9105(.0159)	0.9139(.0028)	0.9144(.0032)	0.9139(.0038)	0.9133(.0045)	0.9154(.0030)
BYOL	0.9107(.0125)	0.9152(.0026)	0.9138(.0032)	0.9123(.0040)	0.9168(.0017)	**0.9171(.0030)**
VICReg 4x	0.9068(.0120)	0.9074(.0019)	0.9098(.0028)	0.9108(.0043)	0.9112(.0024)	0.9101(.0027)
VICReg 2x	0.9040(.0096)	0.9107(.0031)	0.9097(.0020)	0.9114(.0039)	0.9123(.0040)	0.9110(.0046)

Table 5.7 Model selection by last epoch. Mean and standard deviation of macro-AUC on test data over 10 evaluation runs. Evaluated using fine-tuning.

Method	Stage 4	Stage 3	Stage 2	Stage 1	Stage 3+4	Stage 2+3
SimCLR	0.9148(.0035)	0.9141(.0022)	0.9142(.0041)	0.9140(.0036)	0.9145(.0028)	0.9156(.0029)
BYOL	0.9158(.0033)	0.9146(.0024)	0.9124(.0040)	0.9147(.0058)	**0.9179(.0044)**	0.9155(.0036)
VICReg 4x	0.9095(.0056)	0.9076(.0028)	0.9119(.0033)	0.9087(.0028)	0.9104(.0035)	0.9132(.0030)
VICReg 2x	0.9130(.0047)	0.9088(.0016)	0.9085(.0036)	0.9129(.0028)	0.9135(.0039)	0.9128(.0044)

from stage 4. In the fine-tuning setting, all methods are not benefited from earlier feature extraction, but the highest achieved macro-AUC is increased. Previously, using representations from stage 4, BYOL obtained a macro-AUC of 0.9158. Instead, with a concatenated representation of feature vectors from stages 3 and 4, a macro-AUC of 0.9179 is reached.

5.4 Fine-tuning with subsets of labelled data

As Section 5.1 points out; the goal of self-supervised learning is to construct meaningful feature representations of the input data without using data labels. In order to investigate how our self-supervised models perform when there are very few labelled examples available for fine-tuning, we chose to carry out the following experiments.

Using a modified version of the simulated scenario presented for the supervised experiment, the pre-trained models were fine-tuned with a smaller amount of labelled training data than that used for the previous fine-tuning evaluations. This meant that after pre-training models with SimCLR, BYOL,

or VICReg approaches, the respective encoder networks were fine-tuning on smaller subsets of the original, full dataset $\mathcal{D}_{PTB}$. Data samples were uniformly sampled from the full training dataset at the beginning of the training procedure. Validation and test sets remained at their original sizes. Pre-training, however, was still performed on the entire unlabelled dataset $\mathcal{D}_{Pre-train}$, as this experiment assumed access to a larger set of unlabelled data while labelled examples were scarce. After pre-training models for 2000 epochs, the last model was extracted and fine-tuned for 20 epochs using either 1%, 10%, 25%, 50%, 70% or 100% of the full training PTB-XL dataset. Mean test macro-AUC after 5 evaluation runs are presented.

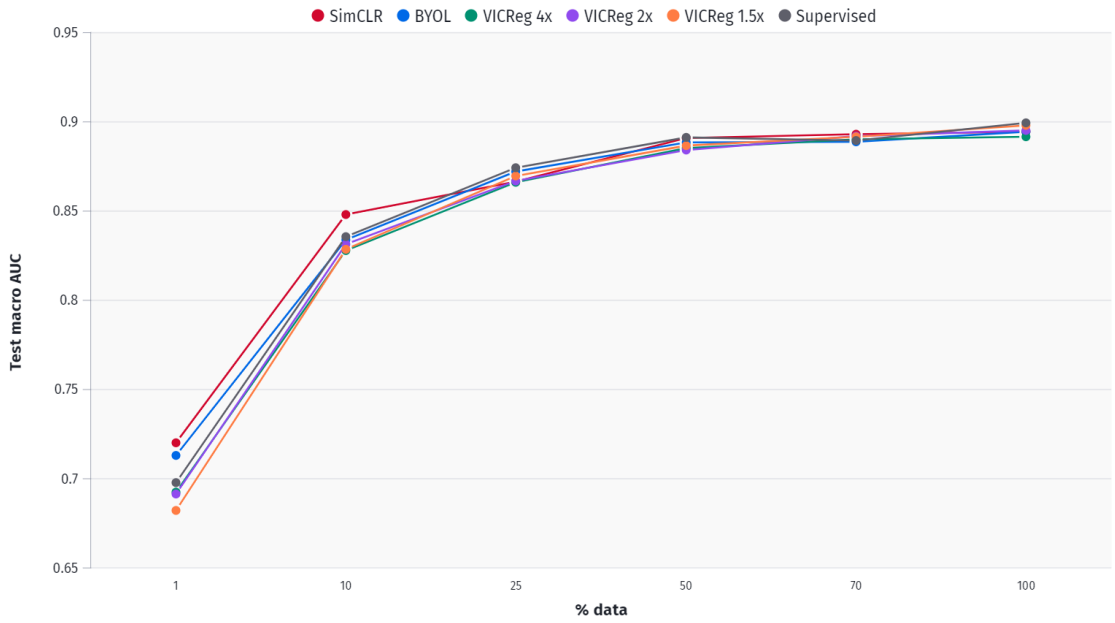
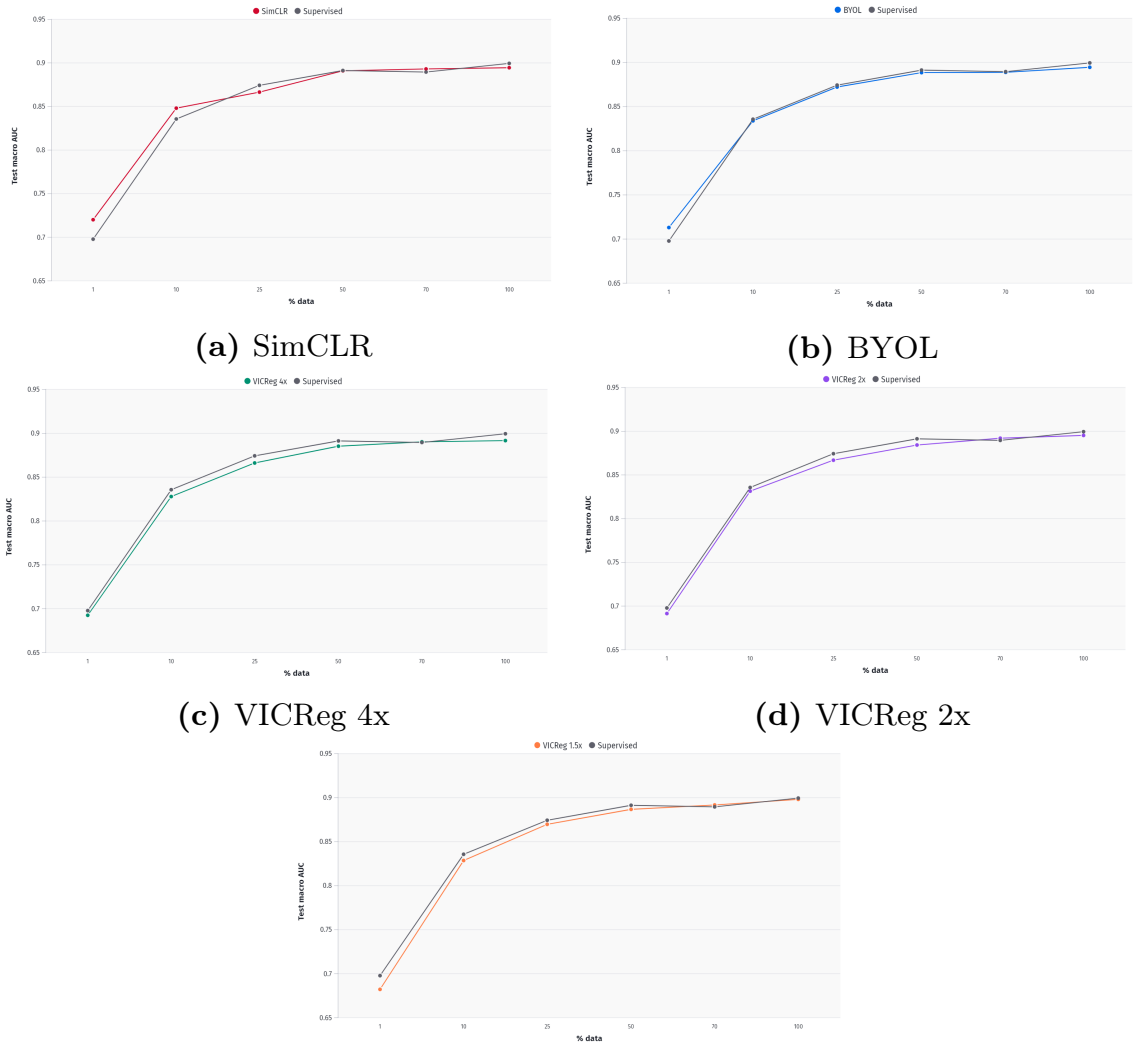


Figure 5.4 Performance of models as function of % labelled fine-tuning data. Fine-tuned for 20 epochs. Mean over 5 evaluation runs.

Table 5.8 Model from last epoch. Mean macro-AUC on test data over 5 evaluation runs. Fine-tuned for 20 epochs with varying % subset of labelled data.

	% of full dataset used for fine-tuning					
Method	1%	10%	25%	50%	70%	100%
Supervised	0.6980(.0128)	0.8358(.0047)	0.8743(.0025)	0.8914(.0088))	0.8894(.0056)	0.8996(.0033)
SimCLR	0.7203(.0143)	0.8482(.0127)	0.8666(.0060)	0.8910(.0029)	0.8932(.0026)	0.8946(.0053)
BYOL	0.7131(.0106)	0.8338(.0045)	0.8721(.0056)	0.8885(.0026)	0.8888(.0034)	0.8946(.0026)
VICReg 4x	0.6926(.0156)	0.8278(.0032)	0.8663(.0067)	0.8854(.0027)	0.8901(.0019)	0.8918(.0029)
VICReg 2x	0.6915(.0135)	0.8314(.0103)	0.8670(.0083)	0.8844(.0047)	0.8922(.0054)	0.8954(.0017)
VICReg 1.5x	0.6822(.0181)	0.8285(.0050)	0.8696(.0098)	0.8868(.0035)	0.8917(.0040)	0.8982(.0019)

In Figure 5.4, performance results are displayed for all models when performance is evaluated as a function of available data. Figure 5.5 shows per-



(a) SimCLR

(b) BYOL

(c) VICReg 4x

(d) VICReg 2x

(e) VICReg 1.5x

Figure 5.5 Performance as function of % labelled data, fine-tuned for 20 epochs. Results are mean macro-AUC and standard deviation over 5 evaluation runs on test data.

formance plots for all methods and compares them to the supervised model performance. Table 5.8 presents a detailed table view of the plot in Figure 5.4. Over every data subset size, the supervised model performs better than any pre-trained method.

We then investigated the topic further by letting the fine-tuning training last for 100 epochs instead of the aforementioned 20. The supervised model in Figure 5.2 displayed a loss close to zero and highly fluctuating macro-AUC at this training epoch. Although this might be an indication of overfitting to the training data and thereby a lack of generalization, a performance comparison between supervised and pre-trained models in the low-labelled example setting was still carried out. Adopting the same procedure as before, we fine-tuned the pre-trained models for 100 epochs using varying sizes of the training data. As before, the mean test macro-AUC after 5 evaluation runs are reported. Figure 5.6 displays performance as a function of dataset size for all methods, while Figure 5.7 presents individual plots of all methods when compared to the supervised model performance. Table 5.9 displays the detailed table view of the plot in Figure 5.6. In spite of the additional fine-tuning epochs, none of the pre-trained methods were able to adjust their representations to the downstream task any better than the supervised model. Consequently, in this low-labeled data setting, no self-supervised pre-training approach proved effective for improving downstream performance.

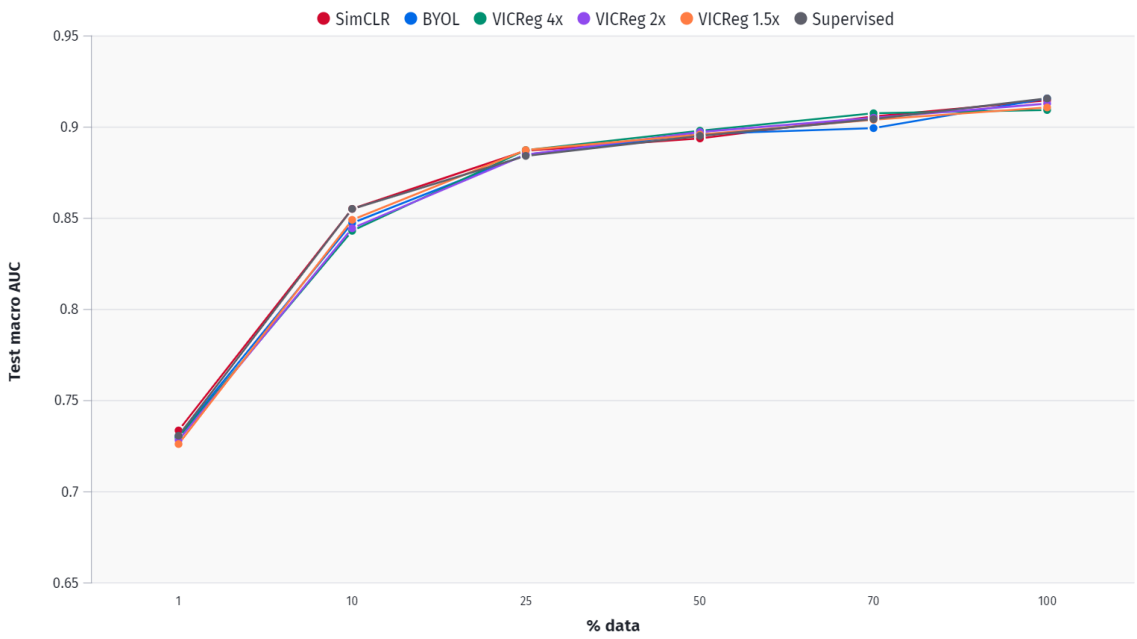


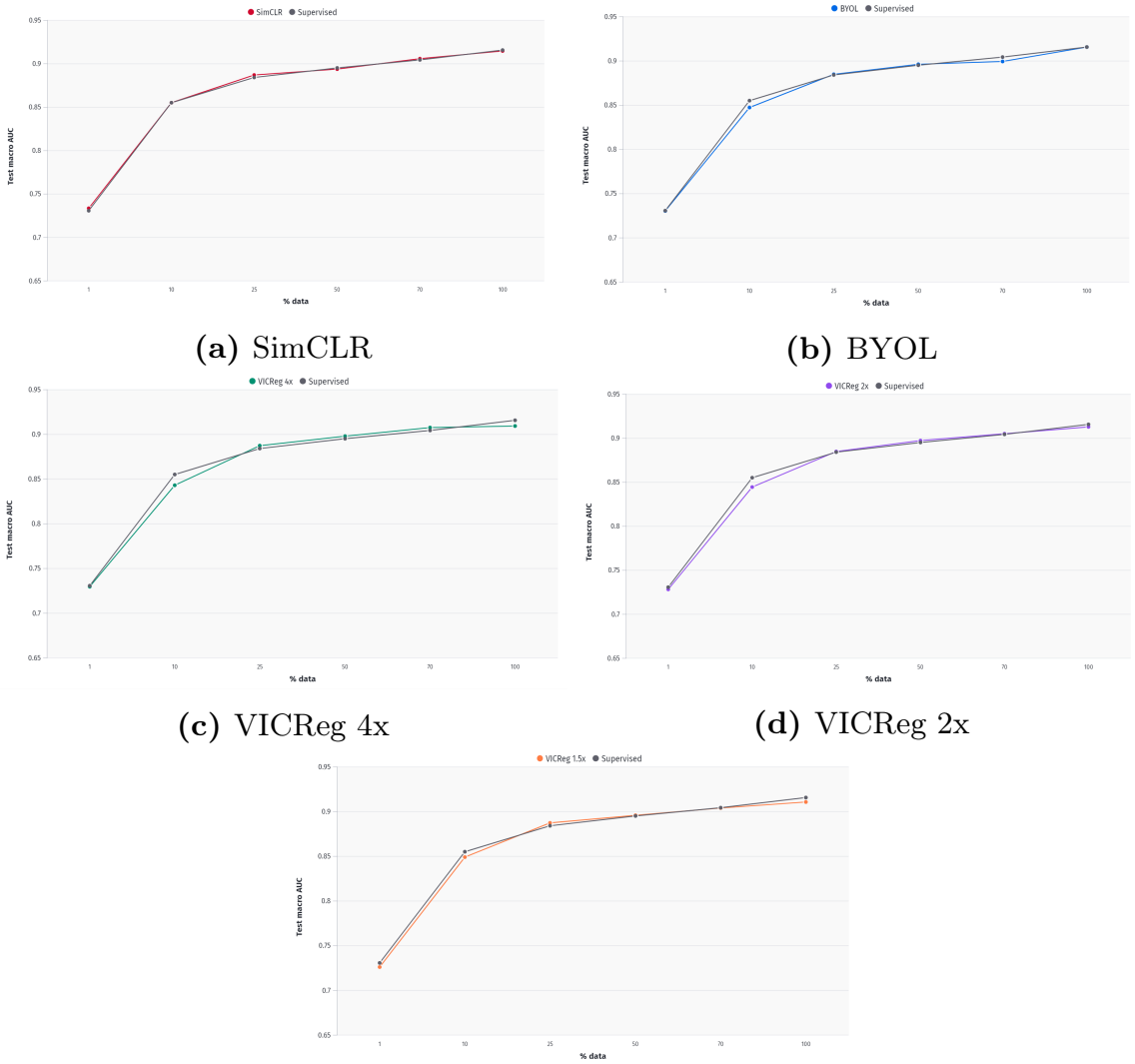
Figure 5.6 Performance by varying pre-training methods as a function of % labelled fine-tuning data. Fine-tuned for 100 epochs. Model selected by last epoch. Mean and standard deviation over 5 evaluation runs.

5.5 Increasing length of ECG signals

For a healthy heart with a typical heart rate of 70 to 75 beats per minute, each cardiac cycle, or heartbeat, lasts for more or less 0.8 seconds [Gersh, 2000].

Table 5.9 Model from last epoch. Mean macro-AUC on test data over 5 evaluation runs. Fine-tuned for 100 epochs with varying % subset of labelled data.

	% of full dataset used for fine-tuning					
Method	1%	10%	25%	50%	70%	100%
Supervised	0.7309(.0086)	0.8552(.0062)	0.8842(.0077)	0.8953(.0045)	0.9043(.0041)	0.9157(.0037)
SimCLR	0.7335(.0114)	0.8553(.0061)	0.8870(.0074)	0.8937(.0054)	0.9059(.0038)	0.9148(.0035)
BYOL	0.7305(.0091)	0.8474(.0068)	0.8850(.0081)	0.8963(.0045)	0.8996(.0023)	0.9158(.0033)
VICReg 4x	0.7296(.0129)	0.8430(.0075)	0.8875(.0065)	0.8980(.0054)	0.9078(.0018)	0.9095(.0056)
VICReg 2x	0.7284(.0137)	0.8444(.0065)	0.8851(.0048)	0.8975(.0045)	0.9053(.0048)	0.9130(.0047)
VICReg 1.5x	0.7263(.0096)	0.8492(.0064)	0.8873(.0046)	0.8960(.0054)	0.9042(.0039)	0.9107(.0027)



(a) SimCLR **(b)** BYOL

(c) VICReg 4x **(d)** VICReg 2x

(e) VICReg 1.5x

Figure 5.7 Performance as function of % labelled data, fine-tuned for 100 epochs. Results are mean macro-AUC over 5 evaluation runs on test data.

Therefore, 2.5 seconds of an ECG signal entails about three complete cardiac

cycles. It may not be sufficient information to train a classifier on three full cardiac cycles, given that some cardiovascular diseases, such as arrhythmia, can only be detected sporadically.

To investigate how the length of the ECG signals affects model performance, we set out to redo the previous experiments. This time, using 10-second long ECG signals for each of the specified self-supervised learning methods. Following the implementation details presented in Section 4.2, we pre-trained the models for 2000 epochs following the already described learning approaches of SimCLR, BYOL, and VICReg 4x. For these experiments, pre-training was performed solely on the training data of the PTB-XL dataset. However, without the use of any data labels. This was primarily done to shorten the training time and allow for more experiments to be conducted. As in previous experiments, we evaluated the models using the fine-tuning evaluation protocol. Fine-tuning was performed for 20 epochs using varying percentages of the full labelled dataset $\mathcal{D}_{PTB}$. Table 5.10 presents the mean model performance on test data over 5 evaluation runs, on models pre-trained with 2.5-seconds or 10-seconds ECG signals. Additionally, the effect of the number of pre-training epochs on performance was investigated by extracting models after 250, 500, 1000, 1500, and 2000 epochs of pre-training. These models were fine-tuned given the previous explanation, and the results are presented in Table 5.10.

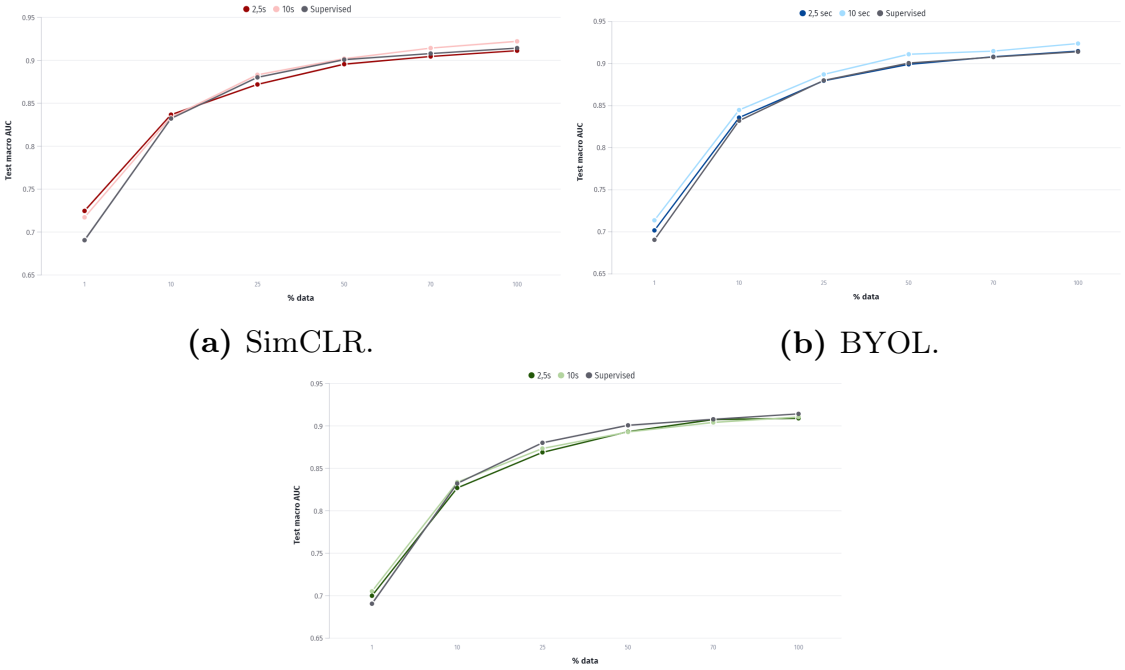
Table 5.10 Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 20 epochs with 1% of training data. Models extracted after varying numbers of pre-training epochs.

	250 epochs		500 epochs		1000 epochs		1500 epochs		2000 epochs	
	2.5s	10s	2.5s	10s	2.5s	10s	2.5s	10s	2.5s	10s
SimCLR	0.7169(.0172)	**0.7189(.0127)**	0.7053(.0267)	0.7150(.0199)	0.7043(.0152)	0.7050(.0110)	0.7027(.0196)	0.7031(.0107)	0.7061(.0097)	0.7090(.0120)
BYOL	0.7016(.0106)	0.7130(.0052)	0.7022(.0097)	0.7106(.0074)	0.7007(.0098)	0.7056(.0031)	0.7020(.0211)	0.7087(.0108)	0.6979(.0128)	0.7040(.0077)
VICReg4x	0.6867(.0285)	0.6945(.0175)	0.6919(.0127)	0.6971(.0171)	0.6917(.0136)	0.7012(.0084)	0.6829(.0199)	0.6870 (.0191)	0.6816(.0179)	0.7015(.0073)

Recall that the performance of the supervised model in the setting of 1% training data was 0.6906 macro-AUC after 20 epochs. Additionally, the highest test macro-AUC obtained by pre-trained models using 2.5-second signals was 0.7169. By only changing the length of the signals, SimCLR instead reached a macro-AUC of 0.7189. Furthermore, the performance of all extracted models independent of the pre-training method, was higher after fine-tuning on 1% on training data when pre-trained on 10-second signals rather than 2.5-second signals, as seen in Table 5.10.

As we increase the size of the labelled fine-tuning dataset, we obtain the results shown in Figure 5.8, where model performance is plotted as a function of available labelled data. We compared the performance of pre-trained models with supervised models in settings with varying amounts of data available.

We pre-trained SimCLR, BYOL, and VICReg models for 1000 epochs with either 2.5-second or 10-second ECG signals from the PTB-XL dataset. The



(a) SimCLR. **(b)** BYOL.

(c) VICReg4x.

Figure 5.8 Performance as function of % labelled fine-tuning data, using 2.5 and 10-second ECG signals. Fine-tuned for 20 epochs. Results are mean macro-AUC over 5 evaluation runs on test data.

models were then fine-tuned for 20 epochs using 1%, 10%, 25%, 50%, 70%, or 100% of the PTB-XL labelled training dataset. In the previous scenario where performance was investigated as a function of % available labelled examples, no pre-trained model showed better results than the supervised model over any subset size. With the 10-second ECG signals and the BYOL pre-training method, however, we were able to obtain higher macro-AUC scores for all sizes of labelled datasets than they would be with a supervised model alone.

Finally, we sought to test and evaluate the models when fine-tuned on the entire labelled dataset of 10-second signals. To this end, each pre-trained model was fine-tuned for 100 epochs using the full labelled PTB-XL dataset $\mathcal{D}_{PTB}$. Table 5.11 presents mean macro-AUC on test data over 5 evaluation runs for the various methods. It is presented both the results of training with 2.5-second and 10-second ECG signals. Using 10-second signals improves the performance of all methods, although the VICReg approach only sees an increase of 1‰.

5.6 Rethinking contrastive pairs

In pursuit of an optimal signal representation, the mutual information between the latent representations of different sample views is maximized. This

Table 5.11 Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 100 epochs.

	Signal length	
	2.5s	10s
Supervised	0.9157(.0037)	0.9165(.0032)
SimCLR	0.9150(.0035)	0.9224(.0050)
BYOL	0.9219(.0047)	**0.9263(.0037)**
VICReg 4x	0.9163(.0033)	0.9164(.0035)

allows for view-invariant representations to be learned, and consequently, the choice of views controls what information is captured in the representation. As the learning framework results in representations that focus on the shared information between different transformed versions of the same ECG signal, the definition of a contrastive pair becomes central to the representation's usefulness. While this is of significant importance, the question of viewing conditions to be invariant to still remains an important question in need of an answer. As previous research on self-supervised learning in the ECG domain has defined positive contrastive pairs as two augmented versions of a given data sample, described in Section 4.5, this experiment was conducted to investigate how different formulations of positive pairs affect the learned representations. We now deviate from the double augmented pairing, where two augmented versions of the original data samples are defined to form a positive pair by applying respective data transformation $t \sim \mathcal{T}, t' \sim \mathcal{T}$ to the ECG signal x. This results in the two views $\tilde{x}_i = t(x), \tilde{x}_j = t'(x)$, forming the positive pair $(\tilde{x}_i, \tilde{x}_j)$. Instead, one augmented view is paired up with an identity mapping of the original data sample to define the contrastive pair $(\tilde{x}_i, x)$, here referred to as *single augmentation*.

We begin the experiments by pre-training models based on the SimCLR, BYOL, and VICReg frameworks. They are pre-trained using this updated definition of positive pairs and then fine-tuned with 1% of the labelled evaluation data. Table 5.12 presents the results of models pre-trained on the unlabelled PTB-XL training dataset, using varying signal lengths and different definitions of contrastive pairs. During pre-training, models are extracted after 250, 500, 1000, 1500, and 2000 pre-training epochs and fine-tuned for 20 epochs on 1% of the full training dataset $\mathcal{D}_{PTB}$. Mean macro-AUC results after 5 fine-tuning evaluation runs are reported.

Our results are presented in Table 5.12, which shows that the SimCLR model pre-trained using single augmentation and 10-second ECG signals obtained the highest macro-AUC. This value of 0.7264 is an improvement of 3.58

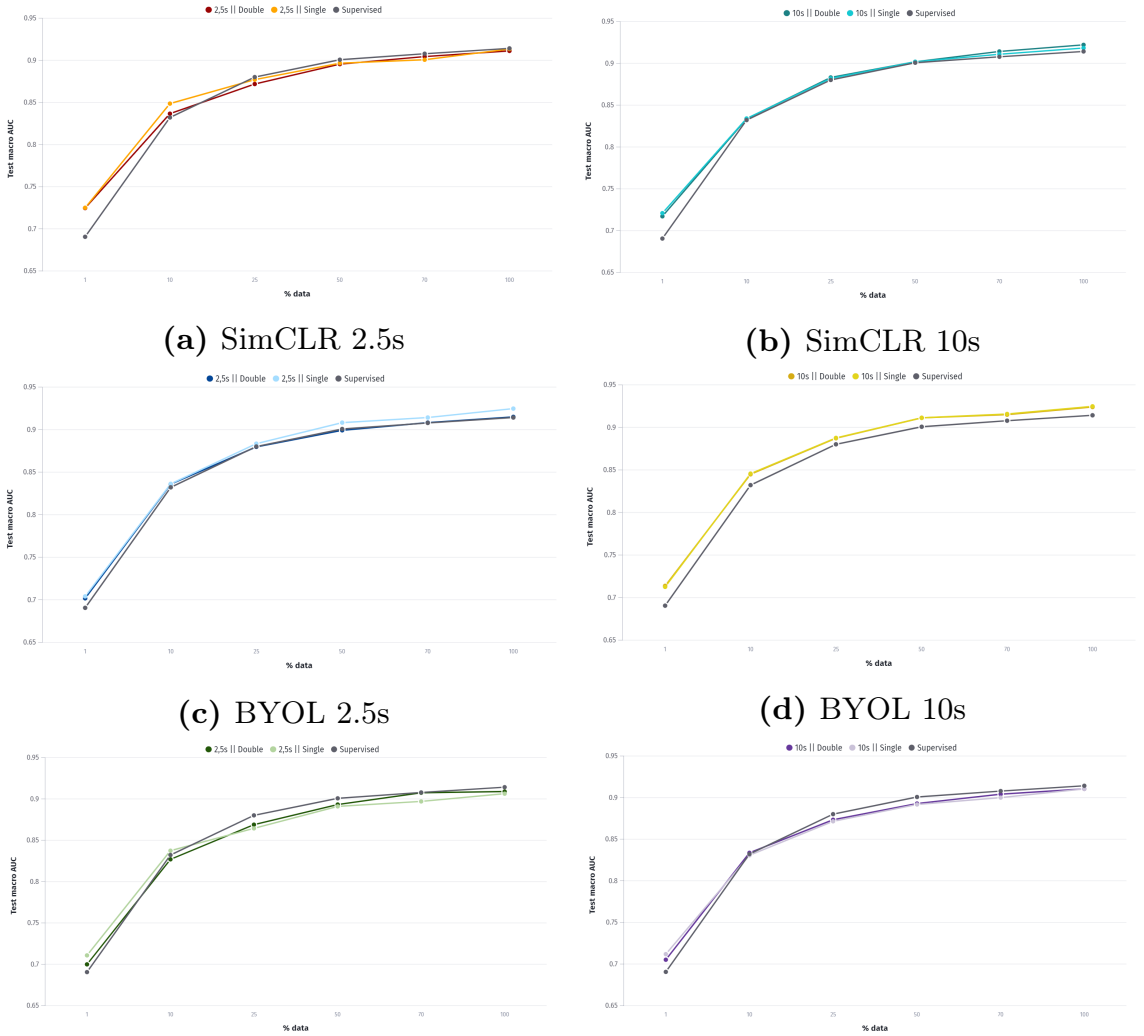
percentage points over the corresponding supervised model performance. As a result of using the single augmentation strategy rather than the double augmentation formulation, all the models demonstrate an increase in performance.

Table 5.12 Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 20 epochs with 1% of training data. Models extracted after varying numbers of pre-training epochs.

		SimCLR		BYOL		VICReg 4x		Supervised	
Epoch	Augment.	2.5s	10s	2.5s	10s	2.5s	10s	2.5s	10s
								0.6879(.0161)	0.6906(.0076)
250	Single	0.7179(.0262)	**0.7264(.0132)**	0.7103(.0076)	0.7136(.0129)	0.6970(.0224)	0.6978(.0112)		
	Double	0.7169(.0172)	0.7189(.0127)	0.7016(.0106)	0.7130(.0052)	0.6867(.0285)	0.6945(.0175)		
500	Single	0.7101(.0063)	0.7170(.0106)	0.7080(.0120)	0.7113(.0125)	0.6967(.0018)	0.7063(.0141)		
	Double	0.7053(.0267)	0.7150(.0199)	0.7022(.0097)	0.7106(.0074)	0.6919(.0127)	0.6971(.0171)		
1000	Single	0.7072(.0147)	0.7132(.0111)	0.7053(.0095)	0.7068(.0077)	0.6968(.0215)	0.7072(.0123)		
	Double	0.7043(.0152)	0.7050(.0110)	0.7007(.0098)	0.7056(.0031)	0.6917(.0136)	0.7012(.0084)		
1500	Single	0.7053(.0194)	0.7175(.0096)	0.7029(.0134)	0.7090(.0105)	0.6973(.0220)	0.7023(.0185)		
	Double	0.7027(.0196)	0.7031(.0107)	0.7020(.0211)	0.7087(.0108)	0.6829(.0199)	0.6870(.0191)		
2000	Single	0.7084(.0173)	0.7099(.0180)	0.7050(.0043)	0.7047(.0116)	0.6913(.0177)	0.7091(.0082)		
	Double	0.7061(.0097)	0.7090(.0120)	0.6979(.0128)	0.7040(.0077)	0.6816(.0179)	0.7015(.0073)		

Further extending the investigation of performance in the low labelled data regime, we pre-trained models for 1000 epochs using unlabelled training PTB-XL data and fine-tuned them using an increasing number of labelled samples. Models were pre-trained using 2.5-second or 10-second ECG signals. To form contrastive pairs, single or double augmentation strategies were applied. As in previous experiments following a similar evaluation set-up, the pre-trained models were fine-tuned for 20 epochs with 1%, 25%, 50%, 70% or 100% of the PTB-XL dataset $\mathcal{D}_{PTB}$. Figure 5.9 presents plots of model performance given different subsets of the training data. As seen, all methods showed improved performance when single augmentation pairs were used. Compared with the original pair definition, only a slight performance improvement was observed with the BYOL model trained with the single augmentation strategy; however, both 10-second ECG models performed well above the supervised baseline.

For the next experiment, we pre-trained models for 1000 epochs using the unlabelled PTB-XL training data. We also used the same combinations of signal length and contrastive pair formulations as described for previous experiments. These models were then fine-tuned for 100 epochs using the full labelled PTB-XL dataset, $\mathcal{D}_{PTB}$. Mean test macro-AUC over 5 evaluation runs are reported in Table 5.13. In accordance with findings in the aforementioned experiments, our most recent results from applying single augmentation strategies present consistent performance increases across all methods. For example, a BYOL model pre-trained with the single augmentation strategy and 10-second ECG signals was able to obtain a macro-AUC of 0.9270. When compared to its supervised counterpart, the supervised model reaches a macro-AUC of 0.9165.



(a) SimCLR 2.5s

(b) SimCLR 10s

(c) BYOL 2.5s

(d) BYOL 10s

(e) VICReg4x 2.5s

(f) VICReg4x 10s

Figure 5.9 Performance as a function of % labelled fine-tuning data, using 2.5 and 10-second ECG signals. Pre-trained with single or double augmentation. Fine-tuned for 20 epochs. Results are mean macro-AUC over 5 evaluation runs on test data.

Table 5.13 Methods pre-trained with 2.5 or 10-second signals and with single or double augmentations. Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 100 epochs with the full dataset.

	SimCLR		BYOL		VICReg 4x		Supervised	
Augment.	2.5s	10s	2.5s	10s	2.5s	10s	2.5s	10s
							0.9157(.0037)	0.9165(.0032)
Single	0.9175(.0047)	0.9246(.0021)	0.9262(.0019)	**0.9270(.0050)**	0.9195(.0022)	0.9168(.0023)		
Double	0.9150(.0035)	0.9224(.0030)	0.9219(.0047)	0.9263(.0037)	0.9163(.0033)	0.9154(.0035)		

5.7 Increasing augmentation strength

Data augmentation has been widely used in neural network representation learning. However, it has not been considered a systematic way to define the

contrastive prediction task. Experiments conducted by [Chen et al., 2020] show that unsupervised contrastive learning benefits from stronger data augmentation than supervised learning. Moreover, data augmentation that does not yield accuracy benefits for supervised learning can still help considerably with contrastive learning and the definition of contrastive pairs. Following this line of reasoning, our next experiments were designed to investigate the effect on performance when applying stronger data augmentations.

The default augmentation described in Section 3.3, sets the random resize cropping parameter to $(l, m) = (0.5, 1.0)$, and the time out parameters to $(t_l, t_u) = [0.0, 0.5]$. This augmentation strength is referred to as *RRC 50-100% TO 0-50%*. Given the context, an increase in augmentation strength specifically means an increase in the permitted cropping window. The default augmentation randomly cuts a contiguous segment of the signal, with the largest allowed cropping window being half of the original signal. Thereafter, a stochastically chosen window with a maximum length of 50% of the original signal is set to zero. Additional augmentation strengths used for the following experiments allowed cropping windows of size 30-100% for the random resize crop and time-out windows of 0-80% or 20-60% of the original signal. These augmentation strengths are referred to as *RRC 50-100% TO 0-80%* and *RRC 30-100% TO 20-60%*.

We once again pre-trained models on the unlabelled data and fine-tuned the models with the labelled data. The models were pre-trained with different combinations of hyper-parameter settings. Either 2.5 or 10-second ECG signals were used. Stronger or "default" augmentation strength was applied and finally, positive pairs were defined using either the single or double augmentation strategy. As in the previous experiments, pre-training was conducted with the unlabelled PTB-XL training data.

To get a better estimate of the effect of augmentation, models were extracted after 250, 500, 1000, 1500, and 2000 epochs of pre-training. Fine-tuning these models for 20 epochs with 1% of the PTB-XL training dataset, mean test macro-AUC over 5 evaluation runs are presented in Table 5.14, 5.15 and 5.16. To simplify the table views, results for each pre-training method are split into their own tables. As seen, the highest performance score for each self-supervised framework was achieved when using the strongest augmentation strategy. Additionally, all methods when evaluated in the 1% labelled data setting achieve test macro-AUC results higher than the supervised model. As can be seen in the table, the supervised model reaches a macro-AUC of 0.6906, while the SimCLR model obtains a macro-AUC of 0.7217. It is thus possible to improve performance in the low-labelled data regime by first pre-training the models on available unlabelled data.

Following up on the previous experiment where 1% of the full dataset was used for fine-tuning, we continued by investigating model performance as a function of % available data. Models were pre-trained for 1000 epochs and with

Table 5.14 Methods pre-trained with 2.5 or 10-second signals, single or double augmentations and with varying augmentation strength. Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 20 epochs with 1% of test dataset.

SimCLR		RRC 50-100% TO 0-50%		RRC 30-100% TO 20-60%		Supervised	
Epoch	Augment.	2.5s	10s	2.5s	10s	2.5s	10s
						0.6879(.0161)	0.6906(.0076)
250	Single	0.7079(.0262)	0.7264(.0132)	0.7160(.0147)	0.7250(.0075)		
	Double	0.7169(.0172)	0.7089(.0127)	0.7144(.0059)	0.7162(.0090)		
500	Single	0.7101(.0063)	0.7170(.0106)	0.7118(.0117)	0.7163(.0084)		
	Double	0.7053(.0267)	0.7150(.0199)	0.7063(.0126)	0.7153(.0091)		
1000	Single	0.7072(.0147)	0.7132(.0111)	0.7093(.0157)	0.7168(.0133)		
	Double	0.7043(.0152)	0.7050(.0110)	0.7059(.0212)	0.7155(.0102)		
1500	Single	0.7053(.0194)	0.7175(.0096)	0.7091(.0104)	**0.7217(.0099)**		
	Double	0.7027(.0196)	0.7031(.0107)	0.7066(.0136)	0.7076(.0085)		
2000	Single	0.7084(.0173)	0.7099(.0180)	0.7058(.0225)	0.7157(.0050)		
	Double	0.7061(.0097)	0.7090(.0120)	0.7022(.0179)	0.7149(.0152)		

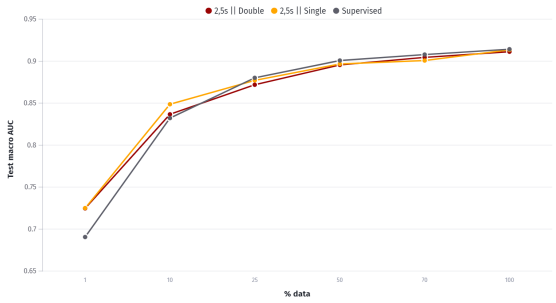
Table 5.15 Methods pre-trained with 2.5 or 10-second signals, single or double augmentations, and with varying augmentation strength. Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 20 epochs with 1% of the test dataset.

BYOL		RRC 50-100% TO 0-50%		RRC 30-100% TO 20-60%		Supervised	
Epoch	Augment.	2.5s	10s	2.5s	10s	2.5s	10s
						0.6879(.0161)	0.6906(.0076)
250	Single	0.7103(.0076)	0.7136(.0129)	0.7105(.0036)	0.7143(.0091)		
	Double	0.7016(.0106)	0.7130(.0052)	0.7091(.0134)	0.7135(.0061)		
500	Single	0.7080(.0120)	0.7113(.0125)	0.7164(.0086)	**0.7179(.0075)**		
	Double	0.7022(.0097)	0.7106(.0074)	0.7041(.0080)	0.7115(.0091)		
1000	Single	0.7053(.0095)	0.7068(.0077)	0.7068(.0072)	0.7076(.0107)		
	Double	0.7007(.0098)	0.7056(.0031)	0.7041(.0173)	0.7061(.0118)		
1500	Single	0.7029(.0134)	0.7090(.0105)	0.7070(.0080)	0.7097(.0109)		
	Double	0.7020(.0211)	0.7087(.0108)	0.7023(.0057)	0.7094(.0142)		
2000	Single	0.7050(.0043)	0.7047(.0116)	0.7069(.0109)	0.7092(.0114)		
	Double	0.6979(.0128)	0.7040(.0077)	0.7037(.0137)	0.7077(.0074)		

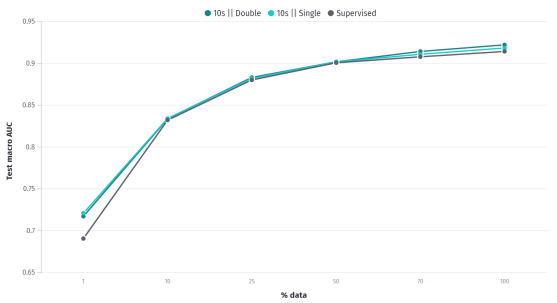
Table 5.16 Methods pre-trained with 2.5 or 10-second signals, single or double augmentations, and with varying augmentation strength. Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 20 epochs with 1% of the test dataset.

VICReg 4x		RRC 50-100% TO 0-50%		RRC 50-100% TO 50-80%		RRC 30-100% TO 20-60%		Supervised	
Epoch	Augment.	2.5s	10s	2.5s	10s	2.5s	10s	2.5s	10s
								0.6879(.0161)	0.6906(.0076)
250	Single	0.6970(.0224)	0.6978(.0112)	0.6983(.0127)	0.7042(.0146)	0.6988(.0168)	0.7105(.0058)		
	Double	0.6867(.0285)	0.6945(.0175)	0.6894(.0142)	0.7016(.0169)	0.6889(.0079)	0.7024(.0132)		
500	Single	0.6967(.0018)	0.7063(.0141)	0.7030(.0139)	0.7066(.0095)	0.7091(.0128)	0.7127(.0035)		
	Double	0.6919(.0127)	0.6971(.0171)	0.6908(.0117)	0.7051(.0118)	0.6914(.0127)	0.7119(.0023)		
1000	Single	0.6968(.0215)	0.7072(.0123)	0.6997(.0119)	0.7104(.0078)	0.7089(.0075)	0.7121(.0143)		
	Double	0.6917(.0136)	0.7012(.0084)	0.6986(.0061)	0.7066(.0125)	0.7005(.0073)	0.7116(.0075)		
1500	Single	0.6973(.0220)	0.7023(.0185)	0.6904(.0079)	0.7081(.0154)	0.7068(.0077)	**0.7134(.0097)**		
	Double	0.6829(.0199)	0.6870(.0191)	0.6853(.0143)	0.7064(.0137)	0.6890(.0161)	0.7042(.0070)		
2000	Single	0.6913(.0177)	0.7091(.0082)	0.7041(.0087)	0.7110(.0099)	0.7088(.0110)	0.7120(.0106)		
	Double	0.6816(.0179)	0.7015(.0073)	0.6894(.0095)	0.7064(.0066)	0.6998(.0131)	0.7102(.0195)		

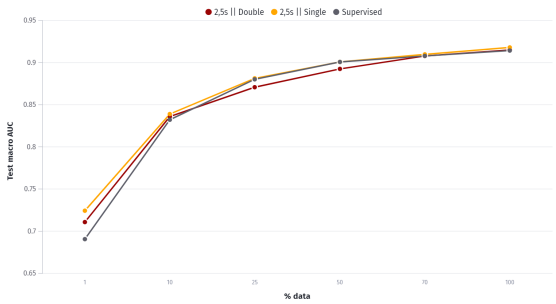
the same specification as described for the previous experiment. After pre-training the models were fine-tuned for 20 epochs using 1%, 25%, 50%, 70% or 100% of the PTB-XL dataset $\mathcal{D}_{PTB}$. Figure 5.10, 5.11, and 5.12 presents plots of the models' performance. For ease of comparison, the performance of the supervised model is also included. It has already been demonstrated that BYOL pre-trained with 10-second ECG signals performs better than the supervised model, but this performance gain is even more pronounced when pre-training with the stronger augmentation strategy. In addition, when models are pre-trained on 10-second ECG signals, with stronger augmentations and a single augmentation strategy in the contrastive pair, the performance of all self-supervised frameworks is superior to that of the supervised model.



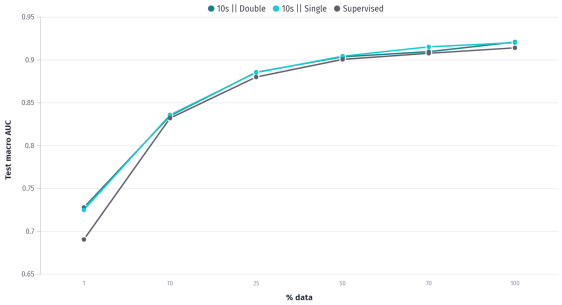
(a) SimCLR: 2.5s RRC 50-100% TO 0-50%



(b) SimCLR: 10s RRC 50-100% TO 0-50%



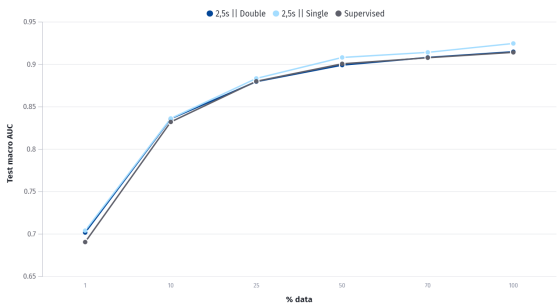
(c) SimCLR: 2.5s RRC 30-100% TO 20-60%



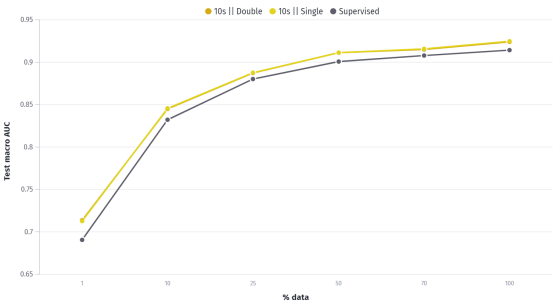
(d) SimCLR: 10s RRC 30-100% TO 20-60%

Figure 5.10 Performance as a function of % labelled fine-tuning data, using 2.5 and 10-second ECG signals. Pre-trained with single or double augmentation. Fine-tuned for 20 epochs. Results are mean macro-AUC over 5 evaluation runs on test data.

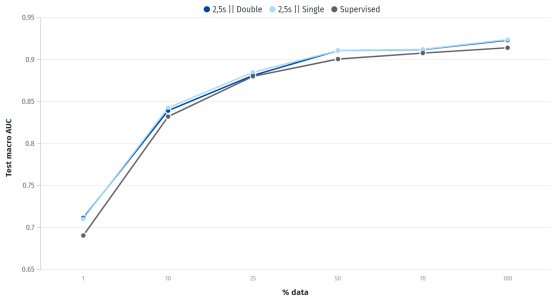
As a final experiment, we set out to investigate model performance when we utilized the full labelled dataset for fine-tuning. Applying the same pre-training setting as aforementioned, we fine-tuned the models for 100 epochs using the whole $\mathcal{D}_{PTB}$ dataset. Test macro-AUC over 5 evaluation runs are presented in Table 5.17.



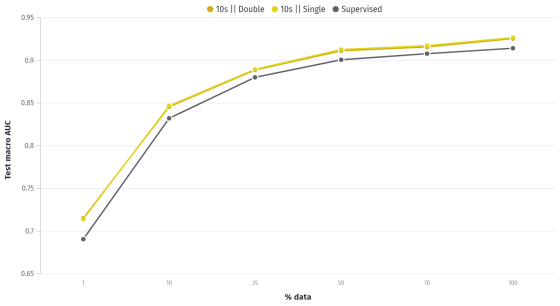
(a) BYOL: 2.5s RRC 50-100% TO 0-50%



(b) BYOL: 10s RRC 50-100% TO 0-50%



(c) BYOL: 2.5s RRC 30-100% TO 20-60%



(d) BYOL: 10s RRC 30-100% TO 20-60%

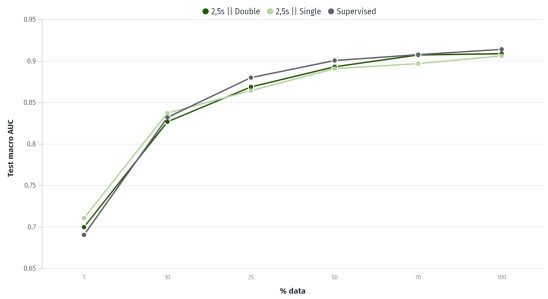
Figure 5.11 Performance as a function of % labelled fine-tuning data, using 2.5 and 10-second ECG signals. Pre-trained with single or double augmentation. Fine-tuned for 20 epochs. Results are mean macro-AUC over 5 evaluation runs on test data.

Table 5.17 Methods pre-trained with 2.5 or 10-second signals, single or double augmentations, and with varying augmentation strength. Mean and std macro-AUC on test data over 5 evaluation runs. Fine-tuned for 100 epochs with the full test dataset.

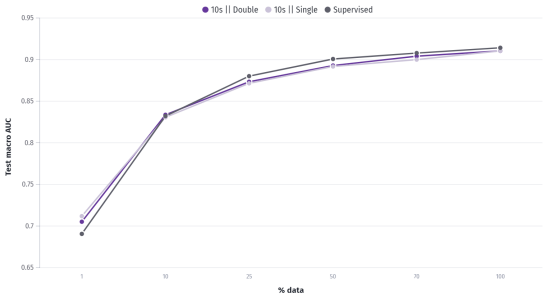
		RRC 50-100% TO 0-50%		RRC 50-100% TO 50-80%		RRC 30-100% TO 20-60%		Supervised	
Method	Augment.	2.5s	10s	2.5s	10s	2.5s	10s	2.5s	10s
								0.9157(.0037)	0.9165(.0032)
SimCLR	Single	0.9175(.0047)	0.9246(.0021)	-	-	0.9198(.0055)	0.9258(.0081)		
	Double	0.9150(.0035)	0.9224(.0030)	-	-	0.9184(.0053)	0.9254(.0032)		
BYOL	Single	0.9162(.0019)	0.9270(.0050)	-	-	0.9270(.0015)	**0.9304(.0063)**		
	Double	0.9139(.0047)	0.9263(.0037)	-	-	0.9257(.0022)	0.9269(.0036)		
VICReg 4x	Single	0.9195(.0022)	0.9168(.0023)	0.9160(.0053)	0.9162(.0061)	0.9179(.0067)	0.9263(.0025)		
	Double	0.9153(.0033)	0.9154(.0035)	0.9150(.0045)	0.9152(.0029)	0.9152(.0097)	0.9186(.0037)		

5.8 Statistical significance

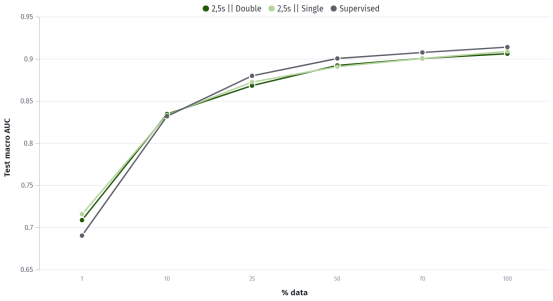
Although we have fewer observations of each model's performance than custom advice for conducting a statistical analysis of their significance, we nevertheless undertake an analysis. As we have computed the mean value and standard deviation of each model after performing the fine-tuning and test evaluation steps five times (see 4.4 for a description of the fine-tuning evaluation procedure) we can evaluate if the true mean of the model before our



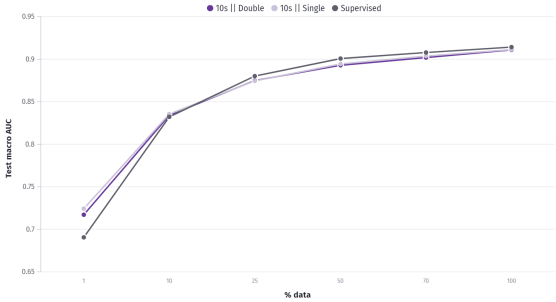
(a) VICReg 4x: 2.5s RRC 50-100% TO 0-50%



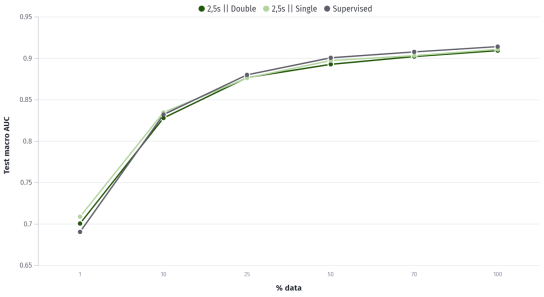
(b) VICReg 4x: 10s RRC 50-100% TO 0-50%



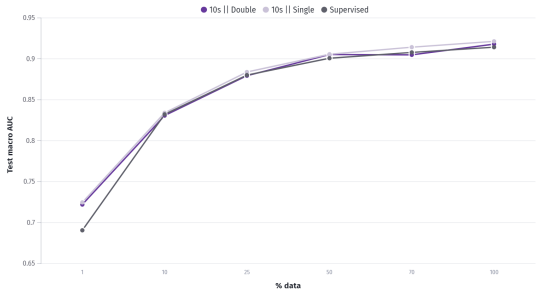
(c) VICReg 4x: 2.5s RRC 50-100% TO 50-80%



(d) VICReg 4x: 10s RRC 50-100% TO 50-80%



(e) VICReg 4x: 2.5s RRC 30-100% TO 20-60%



(f) VICReg 4x: 10s RRC 30-100% TO 20-60%

Figure 5.12 Performance as a function of % labelled fine-tuning data, using 2.5 and 10-second ECG signals. Pre-trained with single or double augmentation. Fine-tuned for 20 epochs. Results are mean macro-AUC over 5 evaluation runs on test data.

adjustments is smaller than the true mean of the model after our adjustments. To this end, we performed a one-tailed Welch's t-test, assuming that the performance means being compared between two models are normally distributed.

We performed the Welch's t-test between the performance measured in macro-AUC of the models using 2.5-second ECG signals and the RRC 50-100% TO 0-50% augmentation strength on both network arms, and the models using 10-second ECG signals and RRC 30-100% TO 20-60% augmentation strength on only one network arm. The Welch's t-test was performed with the

results presented in Table 5.17, where the values are also summarized in Table 5.18. The null hypothesis was formulated as the difference between the mean performance of the first model and the mean performance of the adjusted model was less than or equal to zero $\mu_2 - \mu_1 \leq 0$. It was tested against the alternative hypothesis stating that the mean difference was greater than zero, $\mu_2 - \mu_1 > 0$, or in other words, that the mean performance of the adjusted model (μ_2) was greater than the mean performance of the "original" model (μ_1). Performing a right-tailed Welch's t-test with a sample size of five for each model respectively (as each model was fine-tuned and evaluated five times) and at a significance level $\alpha = 0.05$ we obtained the results summarized in Table 5.18.

In light of the computed p-values, there is evidence for rejecting the null hypothesis and stating that the true mean values of the two model versions are different for all three frameworks. Nonetheless, considering the small sample size used to obtain these p-values, these results may better be viewed as an indication to reject the null hypothesis rather than strong evidence.

Table 5.18 Summary of the values used to perform a one-tailed Welch's t-test and the obtained t-, and p-values. The null-hypothesis $\mu_2 - \mu_1 \leq 0$ was tested against the alternative hypothesis $\mu_2 > \mu_1$.

Method	μ_1	μ_2	σ_1	σ_2	t-value	p-value
SimCLR	0.9150	0.9258	0.0035	0.0081	2.7369	0.01872
BYOL	0.9139	0.9304	0.0047	0.0063	4.6940	0.00096
VICReg 4x	0.9153	0.9263	0.0033	0.0025	5.9412	0.00022

6

Discussion

The purpose of this chapter is to discuss the findings presented in the previous chapters in a more in-depth manner.

6.1 Research question 1

Can self-supervised learning methods be utilized to create ECG signal representations relevant for clinical downstream tasks?

Through our conducted experiments, the feasibility of extracting discriminative representations from ECG data via self-supervised learning stands proven. Performing linear evaluation on pre-trained models using default implementations of the SimCLR, BYOL, and VICReg methods we obtain macro-AUC scores of 0.8882, 0.8715, and 0.8731, see experiment 5.2. If we, however, stochastically initialize a ResNet network and perform linear evaluation, meaning that the network layers are immediately frozen, and only the linear classification layer is trained, a macro-AUC score of 0.6606 is obtained, see experiment 5.1. This shows that self-supervised pre-training on unlabelled data can extract ECG representations that are more useful in a clinical setting than if we were to use a random representation.

However, the performance of fine-tuned models that are pre-trained following the implementations of previous research does not surpass that of a supervised model. After fine-tuning, SimCLR, BYOL, and VICReg models pre-trained using the suggested hyper-parameters e.g., augmentation strength, and framework choices, e.g. positive pair construction, obtain macro-AUC scores of 0.9148, 0.9158, and 0.9095. See Tables 5.2 and 5.3. Compared to the supervised performance of 0.9158, see Table 5.1, self-supervised pre-training does not add any performance advantage. On the contrary, the pre-training step would introduce additional and unnecessary computational costs. Moreover, even in the low label regime supervised models show overall higher performance when measured as a function on % available labelled data. See the

experiment and results presented in Section 5.4. With a macro-AUC performance of 0.6980 after supervised training on 1% of the labelled data, the models pre-trained using the SimCLR and BYOL frameworks performed slightly better with macro-AUC scores of 0.7203 and 0.7131. VICReg however only scores 0.6926. However, when increasing the size of the labelled training data, the supervised model obtains better results, as seen in the previous chapter.

Although these results establish that it is possible to obtain feature representations useful for clinical downstream representations, methods previously presented in research do not perform better than a supervised model when applied in the context of ECG signals.

6.2 Research question 2

Can we improve upon previous self-supervised learning approaches to present novel methods in the domain of medical ECG signal representation learning?

Given the results obtained after implementing SimCLR, BYOL, and VICReg methods using the frameworks presented in their original paper, or as presented in relevant research [Mehari and Strodthoff, 2022], we set out to investigate how the pre-training phase could be improved. As the premise of self-supervised learning is to find task-agnostic representations that will be feasible for downstream tasks, a few changes to the pre-training routines were introduced. By only changing the length of the ECG signal, fine-tuning performance as a function of labelled data was improved for all pre-trained methods. BYOL method even showed results better than the supervised counterpart for all sizes of labelled data. As a result, self-supervised pre-training was helpful. See experiment 5.5. Additionally, when fine-tuning using the full evaluation dataset, model performance rose up to 0.9224, 0.9219, and 0.9164 for SimCLR, BYOL, and VICReg, respectively. The supervised performance increased to a macro-AUC of 0.9165, as presented in Table 5.11. Increasing ECG signal length, thus adding additional heart cycles to each input sample, shows improved performance in all evaluation procedures. Due to the additional cardiac cycles, the ECG signal may contain additional information that is relevant to the classification. Certain cardiac abnormalities are not present in every cardiac cycle and choosing a too narrow segment might exclude relevant information or incidentally a particularly noisy segment might be selected.

Further experiments investigating how augmentation strength and definition of positive pairs affect performance prove alteration of these factors increased pre-trained model performance. Results presented in Sections 5.6 and 5.7 show that pre-training using SimCLR, BYOL or VICReg methods

adds performance improvements over the supervised model in all evaluation settings. Given 1% labelled training data, the supervised model reached a macro-AUC of 0.6906. SimCLR, by contrast, showed a macro-AUC of 0.7217 with BYOL and VICReg obtaining 0.7179 and 0.7134 macro-AUC, respectively. Additionally, even with larger subsets of training data, the pre-trained models outperform the supervised model, as seen in Figures 5.10d, 5.11d, and 5.12f. Although pre-trained models in the first experiments gave no performance advantage over supervised models, applying stronger augmentations to only one of the contrastive views yielded performance results now triumphing those of supervised training. Training a ResNet-50 model using the whole labelled dataset and 10-second ECG signals, a supervised model at best obtained a macro-AUC of 0.9165. If however, a network were pre-trained on the unlabelled data using the BYOL method, the model after being fine-tuned with the full dataset, reached a macro-AUC of 0.9304. Models pre-trained with SimCLR reach a macro-AUC of 0.9258 and pre-training with VICReg, a macro-AUC of 0.9263. Thus, by changing the signal length and augmentation strategies, all implemented pre-training approaches display performance advantages over supervised training when evaluated on the chosen downstream task. In addition, Figure 5.11d demonstrates that the improvements achieved through pre-training directly translate into improved label efficiency, as the pre-trained model outperforms the supervised model using only half of the data. Sections below present further discussion based on these results.

6.3 Evaluation methods

The given evaluation procedures of linear evaluation and fine-tuning are only two ways of measuring representation performance. We introduce a multi-label classification task to act as a proxy for evaluating the representations' clinical relevancy. Although this task incorporates the widely comprehensive set of ECG labels, the classification performance might not be enough to measure the clinical applicability of a model. Therefore, the choice of evaluation procedures and metrics becomes of high significance. If an evaluation task does not cover the full scope of clinical usage, it may produce good results in theory while failing to actually save any lives in practice.

Although our experiments demonstrate high classification performance, additional evaluation methods would further improve the qualitative assessment. For example, distance-based methods acting in the latent embedding space could investigate the model's abilities to approximate signal distributions. Given a network that models the multi-dimensional distribution of ECG signals recorded on normal cardiac cycles, an anomaly detection task could be introduced to enhance the evaluation of clinical relevance.

6.4 How to extract ECG representations

There is no general approach for choosing from which network layer representations ought to be extracted. The common procedure in recent self-supervised learning methods is to use embedding vectors taken from the last stage of the encoder network. In our experiments, the classification performance of representations obtained from this last layer, stage 4 of the ResNet50 model, is competitive with the supervised model. However, in the linear evaluation setting, all pre-trained models show that extracting feature vectors from stage 3 gives the highest performance on the downstream task. Furthermore, after evaluation of all pre-trained models in the fine-tuning setting, the concatenated feature representations from earlier stages are the representations that perform best on the downstream task.

Previous research on the transferability of features in deep neural networks reports that first-layer features appear not to be specific to a particular dataset or task but are general in that they are applicable to many problem formulations. The finding of general features in the first layers seems to occur regardless of the exact cost function and dataset. In contrast, the features computed by the last layer of a trained network are heavily dependent upon the dataset being used and the cost function being optimized [Yosinski et al., 2014; Kornblith et al., 2019]. Thus, feature representations from the last stages of a network are specific to the pre-training task. Moreover, [Yosinski et al., 2014] examines the transferability of features in deep neural networks and investigates the generality versus specificity of neurons in each layer of a deep convolutional neural network. Their findings show that the specialization of higher-layer neurons to their original task decreases performance on the downstream task after fine-tuning.

In light of the findings of the last network layer's pretext-specific features, as well as the discrepancy between pretext- and downstream tasks, it is not surprising that downstream tasks better adopt feature representations from earlier stages of the network. The lower abstraction level of features in later layers might have discarded information needed for the multi-label classification task, as it was not deemed relevant for the pretext task. Consequently, performance is improved when using feature vectors of greater generality that still contain relevant information about the input signal.

Due to the fact that self-supervised learning problem formulations depend heavily on the downstream task and the particular practical use case, it is difficult to suggest a general technique for extracting representations. Our experimental results, however, raise the issue for discussion and suggest that some representing vectors can be selected to enhance the performance of self-supervised learning.

6.5 The role and importance of data augmentation

The definition of positive pairs $(\mathbf{x}_i, \mathbf{x}_j^+)$ is central to the self-supervised learning framework. They are defined as the multiple views generated from each data sample $\mathbf{x}$ by the various transformation functions $t \in \mathcal{T}$. The applied transformations become a crucial component in the network's ability to form representations deployable in a downstream task. Given the encoder function $g(\mathbf{x})$ defined on domain $\mathcal{X}$, a representation is said to be invariant to a transformation $t : \mathcal{X} \to \mathcal{X}$ if $g(t(\mathbf{x})) = g(\mathbf{x})$, as the similarity between the view's embedding vectors is maximized. Here we assume the encoder network to be sufficient in the contrastive learning framework, meaning that the amount of information in $\mathbf{x}_i$ about $\mathbf{x}_j$, and vice versa, is lossless during the encoding procedure $I(\mathbf{x}_i; \mathbf{x}_j) = I(g(\mathbf{x}_i); g(\mathbf{x}_j))$. Thus, representations include all the information the objective of the pretext task requires. This process allows for the formation of a latent embedding space where positive samples are placed close to each other while the distance from the anchor view to the negative samples is maximized. The transformation invariance is thus introduced as the network learns to map all augmented samples of an instance to the same embedding vector. Meaningful compositions of transformations encode valuable invariance and distinctiveness that we want our representations to learn. However, some transformations might hinder performance by introducing representation invariance under a given transformation if features significant for the downstream task are discarded. It then becomes very important to ask ourselves which transformations the ECG representations ought to be invariant to.

A hint of augmentation types

Figure 3.9 created by [Mehari and Strodthoff, 2022] displays the performance of different augmentation compositions. Based on these results, we find that augmentations that do not alter the relative sample values perform best. Time-out augmentation for example will set portions of the signal to zero, thus creating a reconstruction problem as portions of the signal become "masked" without changing the value of the samples. Additionally, the random resize crop neither changes the relation between signal samples as the transformation merely "zooms in" on a section of the signal. The application of Gaussian blur and Gaussian noise however will alter sample values and may thereby distort the distribution of the ECG intervals. Additionally, [Mehari and Strodthoff, 2022] introduces ECG-specific physiological noise transformations but finds that these transformations degrade downstream performance. Finding a suitable transformation protocol is the difference between state-of-the-art performance and poor performance for various tasks. Based on our experimental findings, we suggest an exploration of transformation protocols for self-supervised learning in the context of ECG signals. It is our hope that

these discussions will provide an indication of the transformation invariances that are optimal for ECG signals.

Stronger transformations

When our experiment introduces stronger augmentations, the contrastive prediction task becomes more challenging. As embedding vectors are constrained by greater transformation invariance, the mutual information between latent representations of the multiple views decreases [Tian et al., 2020]. When maximizing mutual information between embedding vectors, the representation is forced to encode solely the underlying shared information. Consequently, local low-level information, noise, and transformation-specific information are discarded, leaving representations to consist of only task-relevant features. Stronger augmentation results in reduced mutual information between views, thereby allowing for complex representations that improve downstream performance. We hypothesize that the performance gains observed when introducing stronger augmentations are a result of this reasoning. In concurrence, [Chen et al., 2020] reports that stronger augmentations in the image domain improve the representational quality of the SimCLR framework. Additionally, learning frameworks such as Barlow Twins and VICReg [Zbontar et al., 2021; Bardes et al., 2021] puts great emphasis on minimizing redundancy between dimensions of the embedding vectors. In these methods, they introduce a regularization term that decorrelates the variables of each embedding. This prevents an informational collapse in which the embeddings store redundant information and features would vary together or be highly correlated.

Time out as signal masking

Our experimental results furthermore show that the application of strong augmentations to only one of the views gives the highest downstream performance. Applying time-out augmentation to one view can be seen as masking a signal section, framing the pretext task as a reconstruction task in the latent space. This is similar to the workings of masked auto-encoders, where representations are learned by reconstructing randomly masked patches from an input. Optimizing a reconstruction loss in ambient space requires modeling low-level signal details that are not necessary for classification tasks. As a consequence, a significant distinction is made, since the time-out transformation defines the reconstruction task in latent feature space rather than on the full signal. The reconstruction problem will drive the two embedding vectors to be similar, performing a denoising step to ensure that the global representation of the noisy view matches that of the unmasked view. That way, representations encode as much information as possible about the input signal. However, maximizing mutual information is only useful as long as the information in the embedding vectors is task-relevant. Beyond that point, rep-

resentations will increase in complexity by storing redundant information that decreases downstream performance. Consequently, the framework of mutual information maximization is only applicable when the available mutual information between embedding vectors is first minimized. [Tian et al., 2020] argue that the introduction of strong augmentations is what reduces the available quantity of mutual information. It is, therefore, speculated that the transformation pipeline must apply both random resize crop and time-out. Random resize crop augmentation will reduce the available amount of mutual information while time-out augmentation introduces the latent reconstruction task. As a prominent line of work in the formulation of self-supervised pretext tasks follows the framework of latent denoising [Devlin et al., 2018; Baevski et al., 2022; He et al., 2022; Wei et al., 2022; Assran et al., 2022; Caron et al., 2021], we further hypothesize that the performance improvements seen by the larger signal masking can be explained by the increased emphasis on the pretext task as a latent reconstruction problem.

6.6 Discarded metadata

ECG signals, human annotations, and patient-, and demographic-related metadata were all included in the public datasets from which we obtained the ECG signals. However, the documented patient-specific metadata was not included in our experiments. In our studies, we chose to disregard this information and examine the applicability of self-supervised representation learning in settings where only a single, unlabeled data mode was introduced. Using patient metadata in conjunction with ECG signals can facilitate the formation of latent representations by expanding model input. These representations are not only based on extracted signal features but are further enhanced by demographic-related features which provide additional information for the classification task. In this manner, classification is performed by using a latent representation conditioned on a demographic cohort. For a more intuitive example, certain characteristics present in the heart cycle of an old woman might be expected as a result of her age. Nevertheless, similar features in a young girl's heart cycle may be an indication of heart problems. Without the information about age, a classification algorithm would not be able to predict differently for the two heart cycles.

Furthermore, introducing patient metadata could facilitate the usage of interpretability methods, as each prediction could be traced back to identify which features lead to these predictions. By gaining a better understanding of key features associated with certain heart conditions, preventative and proactive measures can be taken.

6.7 A word on statistical significance

As part of our experimental methodology, we aimed at exploring a larger portion of the experimental space. This was to get a glimpse of performance for various parameter settings, rather than performing each experiment several times. This decision was driven by the limiting nature of the computational time required to train a model. To give the reader an idea of the needed training time, it takes around 23 hours to pre-train a BYOL model with the full unlabelled dataset of 10-second ECG signals. Thus, we chose to conduct many experiments rather than obtaining many performance observations of the same parameter settings, i.e. training and evaluating the same model setup several times. Consequently, the number of observations obtained of each model's performance will be fewer than custom advice. This is worth remembering when examining the statistical significance presented in Section 5.8, as the low sample size results in a larger margin of error.

7

Conclusions and future work

The purpose of this chapter is to outline the research presented and conducted in this thesis. Various self-supervised learning methods were implemented in the experiments for the representation learning of ECG signals. The concluding results of these experiments are discussed here, as well as possible future work.

7.1 Conclusions

In this study, we present an assessment of self-supervised representation learning on 12-lead clinical ECG data. Although self-supervised algorithms have been applied successfully in other data domains, the ECG signal is of a different data modality on which the applications of self-supervised learning have not been extensively examined. We implement and analyze three of the major self-supervised learning methods: SimCLR, BYOL, and VICReg. In doing so, we find that self-supervised learning can be used to learn meaningful representations of ECG signals. However, when following each method's suggested implementation protocol the performance results are no better than if we were to use a supervised model, indicating that self-supervised pre-training offers no additional benefits to downstream tasks. Among the many insights obtained, the most crucial led to further insights into the importance of signal length and data augmentation. By increasing the length of the ECG signal, performance results were improved for all methods in all evaluation settings. Combining increased signal length with an adjusted data augmentation strategy, self-supervised pre-trained models outperformed their supervised counterparts in all evaluation settings. In the light of our experiments, we learn that for many ways of generating data views there is a sweet spot in terms of downstream performance. This sweet spot is where the mutual information between embedded views is neither too high nor too low. Data augmentation

is one way to reduce this mutual information and finding a suitable augmentation protocol is crucial for high downstream classification performance.

Conclusively, with the sparse amount of research conducted on self-supervised learning in the domain of ECG signals, it comes as no surprise that the formulation of this optimal augmentation policy has not been extensively explored. Published experiments mostly follow the frameworks presented in the computer vision domain, though direct policy adoption could lead to weaker generalization on the downstream task. Our experimental results report performance improvements stemming from the strengthening of augmentation and redefinition of positive pairs. Hopefully, this study's findings add valuable insights to the formulation of an optimal strategy for self-supervised ECG representation learning.

7.2 Future work

Beyond our conducted experiments and discoveries, we plant seeds of thought that we hope will give rise to a variety of future research activities. Below are a few suggestions on possible research quests upon which one could embark.

- **Exploring augmentation protocols.** There are many ways of defining the augmentation protocol and constructing the positive pairs. Further research could investigate: which transformations and transformational invariances are optimal, or harmful, for ECG representations. Then, with what strength should these transformations be carried out? Should they be applied to one, or both of the views? Should the same transformations be applied to both views or can we define a separate protocol for each view?
- **How representations are chosen.** Our experiments demonstrate that how we choose to extract feature representations will affect downstream performance. Future research is suggested to examine the optimal feature representation given an ECG context. At which stage of the network are representations extracted, or should features be concatenated? Additionally, are there other encoder networks that are better suited to form meaningful ECG representations? If so, how are representations defined for these?
- **Evaluation of performance.** As discussed in Section 6.3, there exists a myriad of ways for assessing the performance of a self-supervised neural network. Defining the downstream task as a multi-label classification task is only one way, and further research could investigate model performance in the context of anomaly detection in latent space. How would a model pre-trained in a self-supervised manner perform when applied as an anomaly detector?

- **Available information.** Exploring how representation learning is affected when ECG signals are of different lengths and sampled with higher sample frequency. These choices will impact the amount of information present in the input signal and an increase in available input information might improve model performance. Additionally, pairing ECG signals with patient and demographics metadata, if available, could be investigated to further improve model performance.

- **Exploring other self-supervised learning methods.** There are many more suggestions for self-supervised learning methods proposed by research than those applied in this study. Methods based on predictive pretext tasks have shown substantial success in other signal contexts and their applicability to the ECG domain would be of interest to investigate. Furthermore, an exploration of the self-supervised generative-, and adversarial-based models is encouraged. Generative models approximating the original data distribution could be used for generating synthetic ECG signals or revisiting the application of anomaly detection.

Bibliography

Agrawal, P., J. Carreira, and J. Malik (2015). "Learning to see by moving". In: *Proceedings of the IEEE international conference on computer vision*, pp. 37–45.

Alday, E. A. P., A. Gu, A. J. Shah, C. Robichaux, A.-K. I. Wong, C. Liu, F. Liu, A. B. Rad, A. Elola, S. Seyedi, et al. (2020). "Classification of 12-lead ecgs: the physionet/computing in cardiology challenge 2020". *Physiological measurement* **41**:12, p. 124003.

Assran, M., M. Caron, I. Misra, P. Bojanowski, F. Bordes, P. Vincent, A. Joulin, M. Rabbat, and N. Ballas (2022). "Masked siamese networks for label-efficient learning". *arXiv preprint arXiv:2204.07141*.

Atkielski, A. (n.d.). *SinusRhythmLabels*. URL: `https://commons.wikimedia.org/w/index.php?curid=1560893`.

Bachman, P., R. D. Hjelm, and W. Buchwalter (2019). "Learning representations by maximizing mutual information across views". *Advances in neural information processing systems* **32**.

Baevski, A., W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli (2022). "Data2vec: a general framework for self-supervised learning in speech, vision and language". *arXiv preprint arXiv:2202.03555*.

Bardes, A., J. Ponce, and Y. LeCun (2021). "Vicreg: variance-invariance-covariance regularization for self-supervised learning". *arXiv preprint arXiv:2105.04906*.

Bengio, Y., A. Courville, and P. Vincent (2013). "Representation learning: a review and new perspectives". *IEEE transactions on pattern analysis and machine intelligence* **35**:8, pp. 1798–1828.

Bommasani, R., D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. (2021). "On the opportunities and risks of foundation models". *arXiv preprint arXiv:2108.07258*.

Bousseljot, R., D. Kreiseler, and A. Schnabel (1995). "Nutzung der ekg-signaldatenbank cardiodat der ptb über das internet".

Caron, M., I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin (2020). "Unsupervised learning of visual features by contrasting cluster assignments". *Advances in Neural Information Processing Systems* **33**, pp. 9912–9924.

Caron, M., H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin (2021). "Emerging properties in self-supervised vision transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660.

Chen, T., S. Kornblith, M. Norouzi, and G. Hinton (2020). "A simple framework for contrastive learning of visual representations". In: *International conference on machine learning*. PMLR, pp. 1597–1607.

Chen, X. and K. He (2021). "Exploring simple siamese representation learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758.

Cohen, T. and M. Welling (2016). "Group equivariant convolutional networks". In: *International conference on machine learning*. PMLR, pp. 2990–2999.

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov (2019). "Unsupervised cross-lingual representation learning at scale". *arXiv preprint arXiv:1911.02116*.

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

DeVries, T. and G. W. Taylor (2017). "Improved regularization of convolutional neural networks with cutout". *arXiv preprint arXiv:1708.04552*.

Equitz, W. H. and T. M. Cover (1991). "Successive refinement of information". *IEEE Transactions on Information Theory* **37**:2, pp. 269–275.

Gersh, B. J. (2000). In: *Mayo Clinic Heart Book: The ultimate guide to heart health*. W. Morrow, pp. 6–8.

Ghamrawi, N. and A. McCallum (2005). "Collective multi-label classification". In: *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 195–200.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press.

Grill, J.-B., F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. (2020). "Bootstrap your own latent-a new approach to self-supervised learning". *Advances in Neural Information Processing Systems* **33**, pp. 21271–21284.

He, J. (2020). *Automated Heart Arrhythmia Detection from Electrocardiographic Data*. PhD thesis. Victoria University.

He, K., X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick (2022). "Masked autoencoders are scalable vision learners". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009.

He, K., H. Fan, Y. Wu, S. Xie, and R. Girshick (2020). "Momentum contrast for unsupervised visual representation learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738.

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

Henaff, O. (2020). "Data-efficient image recognition with contrastive predictive coding". In: *International Conference on Machine Learning*. PMLR, pp. 4182–4192.

Ioffe, S. and C. Szegedy (2015). "Batch normalization: accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR, pp. 448–456.

Jaiswal, A., A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon (2021). "A survey on contrastive self-supervised learning". *Technologies* **9**:1, p. 2.

Kiyasseh, D., T. Zhu, and D. A. Clifton (2021). "Clocs: contrastive learning of cardiac signals across space, time, and patients". In: *International Conference on Machine Learning*. PMLR, pp. 5606–5615.

Kolesnikov, A., X. Zhai, and L. Beyer (2019). "Revisiting self-supervised visual representation learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1920–1929.

Kornblith, S., J. Shlens, and Q. V. Le (2019). "Do better imagenet models transfer better?" In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2661–2671.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "Imagenet classification with deep convolutional neural networks". *Advances in neural information processing systems* **25**.

Kwon, O., J. Jeong, H. B. Kim, I. H. Kwon, S. Y. Park, J. E. Kim, and Y. Choi (2018). "Electrocardiogram sampling frequency range acceptable for heart rate variability analysis". *Healthcare informatics research* **24**:3, pp. 198–206.

LeCun, Y. and I. Misra (2021). *Self-supervised learning: the dark matter of intelligence*. URL: `https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/`.

Li, Z., D. Zhou, L. Wan, J. Li, and W. Mou (2020). "Heartbeat classification using deep residual convolutional neural network from 2-lead electrocardiogram". *Journal of Electrocardiology* **58**, pp. 105–112.

Liu, F., C. Liu, L. Zhao, X. Zhang, X. Wu, X. Xu, Y. Liu, C. Ma, S. Wei, Z. He, et al. (2018). "An open access database for evaluating the algorithms of electrocardiogram rhythm and morphology abnormality detection". *Journal of Medical Imaging and Health Informatics* **8**:7, pp. 1368–1373.

Liu, H., Z. Zhao, and Q. She (2021). "Self-supervised ecg pre-training". *Biomedical Signal Processing and Control* **70**, p. 103010.

Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov (2019). "Roberta: a robustly optimized bert pretraining approach". *arXiv preprint arXiv:1907.11692*.

Loshchilov, I. and F. Hutter (2016). "Sgdr: stochastic gradient descent with warm restarts". *arXiv preprint arXiv:1608.03983*.

Loshchilov, I. and F. Hutter (2017). "Decoupled weight decay regularization". *arXiv preprint arXiv:1711.05101*.

Mehari, T. and N. Strodthoff (2022). "Self-supervised representation learning from 12-lead ecg data". *Computers in Biology and Medicine* **141**, p. 105114.

Mendis, S., P. Puska, B. Norrving, W. H. Organization, et al. (2011). *Global atlas on cardiovascular disease prevention and control*. World Health Organization.

Misra, I. and L. v. d. Maaten (2020). "Self-supervised learning of pretext-invariant representations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6707–6717.

Muhammad, Y., M. Tahir, M. Hayat, and K. T. Chong (2020a). "Early and accurate detection and diagnosis of heart disease using intelligent computational model". *Scientific reports* **10**:1, pp. 1–17.

Muhammad, Y., M. Tahir, M. Hayat, and K. T. Chong (2020b). "Early and accurate detection and diagnosis of heart disease using intelligent computational model". *Scientific reports* **10**:1, pp. 1–17.

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

Patrick, M., Y. M. Asano, P. Kuznetsova, R. Fong, J. F. Henriques, G. Zweig, and A. Vedaldi (2021). "On compositions of transformations in contrastive self-supervised learning". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9577–9587.

Poole, B., S. Ozair, A. Van Den Oord, A. Alemi, and G. Tucker (2019). "On variational bounds of mutual information". In: *International Conference on Machine Learning*. PMLR, pp. 5171–5180.

Raghu, M., C. Zhang, J. Kleinberg, and S. Bengio (2019). "Transfusion: understanding transfer learning for medical imaging". *Advances in neural information processing systems* **32**.

Raghunath, S., A. E. Ulloa Cerna, L. Jing, D. P. VanMaanen, J. Stough, D. N. Hartzel, J. B. Leader, H. L. Kirchner, M. C. Stumpe, A. Hafez, et al. (2020). "Prediction of mortality from 12-lead electrocardiogram voltage data using a deep neural network". *Nature medicine* **26**:6, pp. 886–891.

Ribeiro, A. H., M. H. Ribeiro, G. M. Paixão, D. M. Oliveira, P. R. Gomes, J. A. Canazart, M. P. Ferreira, C. R. Andersson, P. W. Macfarlane, W. Meira Jr, et al. (2020a). "Automatic diagnosis of the 12-lead ecg using a deep neural network". *Nature communications* **11**:1, pp. 1–9.

Ribeiro, A. H., M. H. Ribeiro, G. M. Paixão, D. M. Oliveira, P. R. Gomes, J. A. Canazart, M. P. Ferreira, C. R. Andersson, P. W. Macfarlane, W. Meira Jr, et al. (2020b). "Automatic diagnosis of the 12-lead ecg using a deep neural network". *Nature communications* **11**:1, pp. 1–9.

Siontis, K. C., P. A. Noseworthy, Z. I. Attia, and P. A. Friedman (2021a). "Artificial intelligence-enhanced electrocardiography in cardiovascular disease management". *Nature Reviews Cardiology* **18**:7, pp. 465–478.

Siontis, K. C., P. A. Noseworthy, Z. I. Attia, and P. A. Friedman (2021b). "Artificial intelligence-enhanced electrocardiography in cardiovascular disease management". *Nature Reviews Cardiology* **18**:7, pp. 465–478.

Śmigiel, S., K. Pałczyński, and D. Ledziński (2021). "Ecg signal classification using deep learning techniques based on the ptb-xl dataset". *Entropy* **23**:9. ISSN: 1099-4300. DOI: 10.3390/e23091121. URL: https://www.mdpi.com/1099-4300/23/9/1121.

Sohn, K. (2016). "Improved deep metric learning with multi-class n-pair loss objective". *Advances in neural information processing systems* **29**.

Spathis, D., I. Perez-Pozuelo, L. Marques-Fernandez, and C. Mascolo (2022). "Breaking away from labels: the promise of self-supervised machine learning in intelligent health". *Patterns* **3**:2, p. 100410.

Strodthoff, N., P. Wagner, T. Schaeffter, and W. Samek (2020). "Deep learning for ecg analysis: benchmarks and insights from ptb-xl". *IEEE Journal of Biomedical and Health Informatics* **25**:5, pp. 1519–1528.

Thomas, M. and A. T. Joy (2006). *Elements of information theory*. Wiley-Interscience.

Tian, Y., C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola (2020). "What makes for good views for contrastive learning?" *Advances in Neural Information Processing Systems* **33**, pp. 6827–6839.

Tian, Y., X. Chen, and S. Ganguli (2021). "Understanding self-supervised learning dynamics without contrastive pairs". In: *International Conference on Machine Learning*. PMLR, pp. 10268–10278.

Tihonenko, V., A. Khaustov, S. Ivanov, A. Rivin, and E. Yakushenko (2008). "St petersburg incart 12-lead arrhythmia database". *PhysioBank PhysioToolkit and PhysioNet*.

Tishby, N., F. C. Pereira, and W. Bialek (2000). "The information bottleneck method". *arXiv preprint physics/0004057*.

Tishby, N. and N. Zaslavsky (2015). "Deep learning and the information bottleneck principle". In: *2015 ieee information theory workshop (itw)*. IEEE, pp. 1–5.

Tsai, Y.-H. H., S. Bai, L.-P. Morency, and R. Salakhutdinov (2021). "A note on connecting barlow twins with negative-sample-free contrastive learning". *arXiv preprint arXiv:2104.13712*.

Tsai, Y.-H. H., Y. Wu, R. Salakhutdinov, and L.-P. Morency (2020). "Self-supervised learning from a multi-view perspective". *arXiv preprint arXiv:2006.05576*.

Tschannen, M., J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic (2019). "On mutual information maximization for representation learning". *arXiv preprint arXiv:1907.13625*.

Tsoumakas, G. and I. Vlahavas (2007). "Random k-labelsets: an ensemble method for multilabel classification". In: *European conference on machine learning*. Springer, pp. 406–417.

Wagner, P., N. Strodthoff, R.-D. Bousseljot, D. Kreiseler, F. I. Lunze, W. Samek, and T. Schaeffter (2020). "Ptb-xl, a large publicly available electrocardiography dataset". *Scientific data* **7**:1, pp. 1–15.

Wei, C., H. Fan, S. Xie, C.-Y. Wu, A. Yuille, and C. Feichtenhofer (2022). "Masked feature prediction for self-supervised visual pre-training". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14668–14678.

WHO (2021). *The world health report 2001*. URL: `http://www.who.int/whr/2001/en/index.html`.

Wu, Z., Y. Xiong, S. X. Yu, and D. Lin (2018). "Unsupervised feature learning via non-parametric instance discrimination". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742.

Yosinski, J., J. Clune, Y. Bengio, and H. Lipson (2014). "How transferable are features in deep neural networks?" *Advances in neural information processing systems* **27**.

You, Y., I. Gitman, and B. Ginsburg (2017). "Large batch training of convolutional networks". *arXiv preprint arXiv:1708.03888*.

Zbontar, J., L. Jing, I. Misra, Y. LeCun, and S. Deny (2021). "Barlow twins: self-supervised learning via redundancy reduction". In: *International Conference on Machine Learning*. PMLR, pp. 12310–12320.

Zhai, X., A. Oliver, A. Kolesnikov, and L. Beyer (2019). "S4l: self-supervised semi-supervised learning". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1476–1485.

Zhang, M.-L. and Z.-H. Zhou (2013). "A review on multi-label learning algorithms". *IEEE transactions on knowledge and data engineering* **26**:8, pp. 1819–1837.

Zheng, J., J. Zhang, S. Danioko, H. Yao, H. Guo, and C. Rakovski (2020). "A 12-lead electrocardiogram database for arrhythmia research covering more than 10,000 patients". *Scientific Data* **7**:1, pp. 1–8.