Multi-Task Self-Supervised Methods for Label-Efficient Learning

Combining Contrastive and Pretext-Based Learning
for Effective Encoders from Unlabeled and Federated Data
in Human Activity Recognition and Beyond

Master's Thesis submitted to the

Faculty of Informatics of the *Università della Svizzera Italiana*

in partial fulfillment of the requirements for the degree of

Master of Science in Informatics

presented by
Alessandro Gobbetti

under the supervision of
Prof. Marc Langheinrich
co-supervised by
Dario Fenoglio, Mohan Li

September 2025

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Alessandro Gobbetti
Lugano, 12 September 2025

Abstract

The increasing digitalization of society has led to vast amounts of data being collected across many domains. While supervised deep learning has proven highly effective in extracting value from such data, its need for large labeled datasets poses major challenges. Labeling can be costly, time-consuming, and often requires domain expertise or controlled settings. Moreover, sharing a large amount of data, labeled or not, can raise privacy concerns, especially with sensitive information. Human Activity Recognition (HAR) exemplifies these issues: sensors generate abundant data, but labeling remains difficult, and sharing can lead to unwanted release of private conditions, behaviors, habits, or contexts. Similar challenges arise in areas such as smart homes, medical research, monitoring and diagnostics, robotics, and many more.

This thesis investigates self-supervised learning (SSL) as a strategy to leverage unlabeled data to improve classification performance in low-labeled-data scenarios. Building on an analysis of the state-of-the-art, we select and evaluate promising SSL pretraining objectives, including several contrastive learning approaches and pretext tasks, and propose a modular multitask SSL framework that combines multiple objectives to enhance robustness and generalization. We also explore exploiting unlabeled data during downstream training through pseudo-labeling and assess the feasibility of applying our multitask approach in federated learning, where data remains decentralized to preserve privacy. Although designed to be general, our methods are consistently benchmarked on HAR, with preliminary validation in image recognition.

Our experiments led to several interesting findings. First, we confirm that SSL methods markedly improve downstream classification accuracy over fully supervised learning in low-label regimes (from +13.8 to +19.1 percentage points on our HAR test), and remains highly competitive when the full labeled dataset is available (-0.9 to +2.3). Second, multitask learning further enhances SSL performance (+1.1 to +3.2 across all labeled-data regimes compared to the best single task model). Importantly, the benefit of task combinations depends not only on the strength of individual tasks but also on their complementarity (for example, adding a poorly-performing task to contrastive learning yields gains of +0.9 to +1.7 points in image recognition). Third, pseudo-labeling effectively exploits unlabeled data in the downstream phase, especially in low-labeled data regimes and with larger embedding sizes (+0.4 to +3.0 on HAR). Finally, we demonstrate that self-supervised pretraining can be effectively performed in a federated learning setting, with only a minor performance drop compared to centralized training (-1.6 to +0.1 on HAR). This finding opens the door to enabling pretraining on extremely large and diverse distributed datasets, without any privacy and security concerns.

The methods and results presented in this thesis indicate a promising path for training robust and generalizable models by effectively leveraging and exploiting massive amounts of unlabeled data and possibly sensitive data. To favor replicability and further research we release all the code and training configurations as open source.

Acknowledgements

This thesis concludes my journey as a student at USI, where I had the privilege of learning from all the professors, teaching assistants, and fellow students I met during these years. I would like to thank all of them!

This work was carried out under the supervision of Prof. Marc Langheinrich who, together with Martin Gjoreski, introduced me years ago to the topics that form the core of this thesis. They not only supervised my Bachelor's project on multimodal federated learning for sensor data but also guided me into the world of research, giving me the opportunity to collaborate with them on follow-up projects and publications. I am deeply grateful for their support, guidance, and trust throughout this journey.

The thesis was co-supervised by Dario Fenoglio and Mohan Li, whose invaluable advice and feedback greatly shaped this work. I will always remember the many discussions with Dario that helped refine the ideas and concepts presented here. I am also thankful to Mohan for his precise insights and suggestions, which significantly improved the quality of this thesis.

Finally, I thank my family for the essentials: love, food, shelter, travels, bike rides, play, culture, endless talks and discussions — and, of course, the occasional GPUs.

Contents

Figures

Tables

Chapter 1

Introduction

This chapter introduces the context and motivation of the thesis, the objectives and contributions, and the structure of the thesis.

1.1 Context and Motivation

In the last few decades, our society and all scientific and engineering domains have witnessed an enormous transformation towards the computerization and digitalization of most processes. This has led to an unprecedented increase in the amount of data generated, collected, and stored, concerning virtually every aspect of human activity and all physical phenomena, opening the door to large-scale analysis [Emmert-Streib, 2021].

Raw data in itself is often not directly useful and needs to be processed to extract desired knowledge depending on needs. For example, the goal may be to classify a signal, predict a future value, or provide insights [Ahmed et al., 2023]. Over the years, several approaches have been proposed to tackle these problems, from traditional signal processing and statistical methods to modern machine learning techniques. Among these, deep learning has gained significant prominence for its ability to extract complex patterns directly from raw data [LeCun et al., 2015].

To achieve high-level performance, most deep learning models employ a supervised training paradigm, which involves learning to map inputs to outputs by training on large collections of labeled examples [Chen et al., 2024]. However, collecting large labeled datasets could be time-consuming, costly, and in some cases infeasible, especially when labeling requires expert knowledge or when privacy and security concerns limit data sharing [Yang et al., 2023; Loeffler et al., 2024]. By contrast, unlabeled data is typically easier to collect and much more abundant.

Human Activity Recognition (HAR) is a typical use case in which all the above factors play an important role. HAR is the task of automatically recognizing human activities from sensor data [Lara and Labrador, 2013]. Tracking the user in their daily lives or when performing specific tasks can produce huge quantities of measurements. However, labeling these measurements is often difficult, as it requires the user or a third party to manually segment data streams and annotate the activities performed. Moreover, sharing these measurements may pose privacy concerns, as they can contain sensitive information about the user's conditions, behavior, and context [Jung, 2020]. For these reasons, many of the available HAR datasets are small, involving limited activities and users, and collected in controlled environments, which may not generalize well to real-world applications [Wang et al., 2019; Demrozi et al., 2020]. The same situation can be found in many other domains, such as medical imaging, speech recognition, recommender systems, remote sensing, robotics, machine translation, and text classification [Ren et al., 2021].

In recent years, research has focused on techniques to overcome the problem caused by the scarcity of labeled data. One classic approach is to perform the learning, not on raw data but on a handcrafted representation. This representation is designed to be more expressive and to simplify solving the specific problem. However, constructing such representations relies on heuristic methods driven by human expertise and domain knowledge. Even if these methods can be effective for specific tasks, they are often limited to shallow features such as mean, variance, frequency, or amplitude, failing to capture complex patterns and relationships in the data [Wang et al., 2019].

More recently, interest has shifted towards learning these representations directly from the data, leveraging the abundance of unlabeled data. A prominent approach is self-supervised learning (SSL), in which training is split into two phases. In the pretraining phase, a model is trained on a large amount of unlabeled data to learn compact and expressive representations of the data using objectives directly derived from the data itself, without requiring labels. In the second phase, a downstream model is built by composing the representation encoder with a task-specific head, trained on the available small set of labeled examples [Gui et al., 2024]. The various methods differ in the way self-supervision is performed and, in particular, in how the pretraining objective is defined.

In parallel, research has also tackled the challenge of preserving privacy and security when training machine learning models on sensitive data. Federated Learning (FL), in particular, has become the prominent approach to address this problem [Zhang et al., 2021]. Privacy is preserved by distributing the training process across multiple clients, each holding their own local data.

1.2 Objectives and Contributions

This thesis explores the potential of SSL in scenarios with limited labeled data for classification tasks. In particular, we want to compare various ways of defining the objectives of the pre-training phase, and how multiple objectives can be combined to improve generalizability and robustness of the learned representations. Moreover, we want to investigate how these methods can be applied in a federated setting, where learning a complex objective might be challenging due to the limited and diverse data available on each client. We focus primarily on HAR as a case study, but our method is general and can be applied to other domains. To demonstrate this generality, we also evaluate it on image recognition as a secondary use case.

The main contributions of this thesis will be a comprehensive analysis of the performance of different SSL pretraining methods in the context of HAR, a proposal of a multitask SSL framework that combines multiple pretraining tasks to create more robust representations, an evaluation of how to exploit unlabeled data also for fine-tuning the downstream task classifier, and an exploration of how the proposed multitask approach performs in a federated setting. Moreover, we will analyze the generality of the proposed methods by designing a multitask SSL solution for image recognition.

All the benchmarks will be performed on publicly available datasets, and the code will be made available to the community to facilitate reproducibility and help future work in this area.

1.3 Structure of the Thesis

The rest of the thesis is structured as follows:

- chapter 2 provides the relevant background and analysis of related work on Human Activity Recognition, Self-Supervised Learning, Multi-Task Learning, and Federated Learning, highlighting the challenges and solutions proposed in the literature.
- chapter 3 describes the data used in the thesis and the methods employed for the experiments. Additional details are provided in Appendix A.
- chapter 4 introduces our self-supervised learning framework, identifies a general architecture for pretraining, and discusses the results of the experiments on different SSL methods.
- chapter 5 proposes a modular architecture for multitask SSL, which allows for the simple combination of multiple pretraining tasks. It compares the performance of multitask SSL with single-task approaches and presents an extensive ablation study to analyze the impact of each design choice and component on the final performance.
- chapter 6 explores how to exploit unlabeled data also for fine-tuning the downstream task classifier, presenting the results of the experiments on the proposed approach.
- chapter 7 extends the analysis to a federated setting, evaluating the performance of the multitask SSL framework in a distributed environment.
- chapter 8 discusses the generalization of the proposed methods to other domains and datasets. Specifically, we employ the methods in the field of image recognition, highlighting the applicability of the framework beyond HAR.
- chapter 9 concludes the thesis, summarizing the main findings and contributions, and discussing future work directions.

Chapter 2

Background and Related Work

Before presenting the thesis contributions, it is essential to provide an overview of the background relevant to the topics discussed and a summary of the most closely related prior works. We will first introduce the field of HAR, then discuss the challenges and solutions of working with limited labeled data, and finally see how privacy and security concerns in machine learning can be addressed through FL.

2.1 Introduction

This thesis addresses various topics, from learning with limited labeled data to HAR and FL. This chapter sets the stage, providing the necessary background and an analysis of related works.

We first define the concept of HAR from sensor data and discuss the challenges and methods commonly used in this field (section 2.2). We see how many solutions rely on the availability of large annotated datasets, which are often hard to obtain, especially in real-world scenarios. We then summarize the classes of machine learning solutions to the problem of learning with limited labeled examples, focusing on modern deep learning approaches (section 2.3). In particular, we cover self-supervised and semi-supervised paradigms, which are the foundations for the work proposed in this thesis. We then address privacy and security in machine learning, specifically concentrating on FL, which allows training models on distributed data while preserving privacy (section 2.4). Finally, we wrap up the chapter with a discussion on how these topics are interconnected and how they relate to the contributions of this thesis.

Since a detailed presentation of the background and a full coverage or related work would be too extensive, and beyond the scope of this thesis, this chapter will focus solely on the most closely relevant concepts and works. For a more general introduction to the topics, we refer the reader to available survey works on Human Activity Recognition [Lara and Labrador, 2013; Wang et al., 2019; Demrozi et al., 2020; Chen et al., 2021; Ni et al., 2024; Kaseris et al., 2024], self-supervised learning [Liu et al., 2021; Yang et al., 2023; Gui et al., 2024; Zhang et al., 2024; Chen et al., 2024], self-supervised learning applied to HAR [Haresamudram et al., 2022; Ige and Mohd Noor, 2022; Logacjov, 2024], multitask learning [Zhang and Yang, 2022; Yu et al., 2024], federated learning [Aledhari et al., 2020; Zhang et al., 2021; Kairouz et al., 2021; Huang et al., 2024; Li et al., 2025b], and federated learning with limited supervision [Jin et al., 2023].

2.2 Human Activity Recognition

HAR is a classification task that consists of recognizing activities performed by users based on collected data [Ann and Theng, 2014; Lara and Labrador, 2013]. It is a crucial task in various applications, such as health monitoring, fitness tracking, and human-computer interaction, and the types of recognized activities can vary widely depending on the applications and target domain (see Table 2.1) [Lara and Labrador, 2013].

Group	**Activities**
Ambulation	Walking, running, sitting, standing still, climbing stairs, descending stairs, riding elevator.
Transportation	Riding a bus, cycling, driving.
Phone usage	Text messaging, making a call.
Daily activities	Eating, drinking, working at the PC, watching TV, reading, brushing teeth, stretching, vacuuming.
Exercise/fitness	Rowing, lifting weights, spinning, Nordic walking, doing push ups.
Military	Crawling, kneeling, situation assessment, opening a door.
Upper body	Chewing, speaking, swallowing, sighing, moving the head.

Table 2.1. Examples of human activities recognized by HAR systems across different domains [Lara and Labrador, 2013].

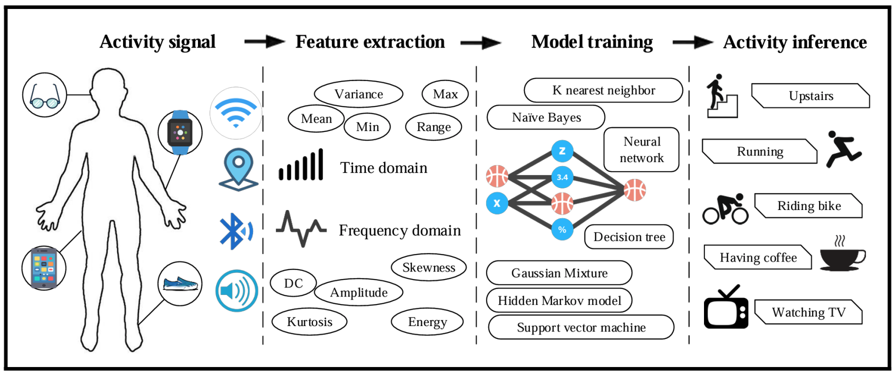
Traditionally, HAR methods can be divided into two main approaches: vision-based and sensor-based. Vision-based methods rely on cameras and computer vision techniques to analyze the movements of the observed subjects and translate them into activities. They have the advantage of not requiring body-attached sensors, and can track multiple subjects simultaneously, also capturing interactions between them. However, they are bound to specific environments (where cameras are installed), are sensitive to environmental conditions (such as lighting and occlusions), can raise privacy concerns due to continuous video recording, and often require significant computational resources [Poppe, 2010]. Sensor-based methods, on the other hand, use wearable devices that typically include sensors such as tri-axial accelerometers, gyroscopes, and magnetometers. They are generally less intrusive, more robust to environmental conditions, and can operate in real-time with limited computational resources, making them more suitable for continuous monitoring in everyday environments [Ni et al., 2024]. For this reason, they are of widespread use, also due to the large availability of affordable commodity devices such as smartphones, smartwatches, and fitness trackers, which are equipped with various sensors that can be used for HAR [Ramanujam et al., 2021; Ige and Mohd Noor, 2022].

This thesis focuses on sensor-based HAR, specifically on the use of wearable sensors to recognize activities performed by users in real-world scenarios.

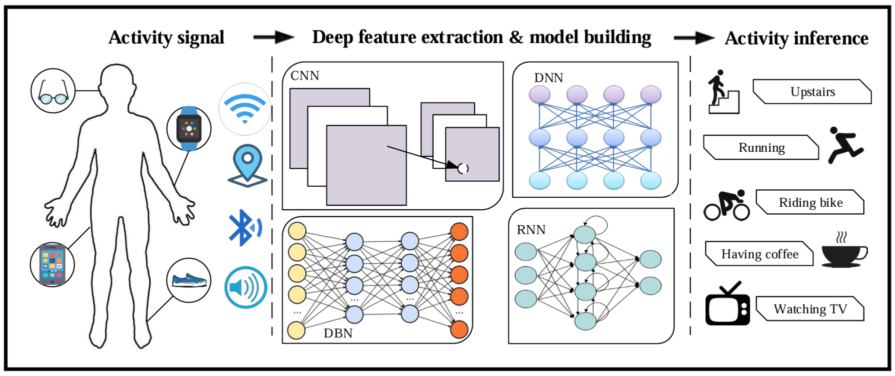
All collected data is sampled over time, the goal is to determine the intervals when a specific activity is performed. Formally the problem can be defined as follows [Lara and Labrador, 2013]: given a set of activity labels $L = \{l_0, \dots, l_{n-1}\}$ (e.g., sitting, walking, etc.) and a set $S = \{S_0, \dots, S_{k-1}\}$ of k time series, each measuring particular user properties (e.g. acceleration, angular velocity, etc.) in a specific time interval $I = [t_\alpha, t_\omega]$, the goal is to analyze the data and partition it into consecutive, non-empty, and non-overlapping time intervals $\langle I_0, \dots, I_{r-1} \rangle$ of I such that $\bigcup_{j=0}^{r-1} I_j = I$ and each interval I_j is labeled with the corresponding activity performed.

To make the problem more manageable, a common strategy is to divide the continuous data stream into fixed-duration time windows, where each window is labeled with the corresponding activity [Lara and Labrador, 2013]. This approach assumes that activities are long enough to be captured within the time windows, that the activities are stationary, meaning that they do not change significantly over time, and that their transitions are fast enough to be ignored.

Since it is extremely challenging to determine activity labels directly from raw sensor data using theoretical rules alone, HAR systems commonly rely on machine learning methods to learn patterns from labeled examples.



(a) HAR pipeline with conventional machine learning approaches.



(b) HAR pipeline with deep learning approaches.

Figure 2.1. HAR pipelines with conventional machine learning and deep learning approaches [Wang et al., 2019].

As shown in Figure 2.1a, early work in sensor-based HAR primarily focused on handcrafted features extracted from time and frequency domains, followed by the use of conventional machine learning algorithms, such as decision trees, support vector machines, and k-nearest neighbors [Avci et al., 2010; Lara and Labrador, 2013; Demrozi et al., 2020; Wang et al., 2019].

Converting raw data into feature vectors typically involves two main steps: preprocessing to reduce noise and normalize the signals, followed by feature extraction to derive meaningful attributes from the cleaned data. A recent survey [Demrozi et al., 2020] identified 26 commonly used time-domain features and 20 frequency-domain features, often combined in different ways across studies. However, to achieve good performance, these methods often require extensive feature engineering and domain expertise. Additionally, they can be resource-intensive and time-consuming at runtime, as they require the extraction of features from raw data before the classification step.

Advancements in deep learning have led to the development of end-to-end models that can automatically learn features from raw sensor data, eliminating the need for manual feature engineering (see Figure 2.1b). Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) models, were among the first deep learning techniques applied to HAR and demonstrated the ability to automatically learn spatio-temporal patterns from raw sensor data, achieving state-of-the-art results on benchmark datasets at the time of their introduction [Hammerla et al., 2016; Jiang and Yin, 2015]. Subsequently, hybrid architectures that integrate CNNs with LSTMs have been introduced to further enhance recognition accuracy [Ordóñez and Roggen, 2016]. This class of techniques, that had many follow-ups, is still widely used and ranks among the best-performing methods in HAR [Kaseris et al., 2024]. Recent studies have explored more complex architectures, such as Transformer-based models [Leite et al., 2024] and graph neural networks [Wieland and Pankratius, 2023], to better model the sequential and relational properties of multi-sensor data. Moreover, there is growing interest in robust HAR under practical constraints such as missing sensor modalities, user variability, and data heterogeneity [Woo et al., 2023; Gobbetti et al., 2024].

Despite these advancements, HAR remains a challenging task, particularly in real-world scenarios where data is often noisy, incomplete, and subject to variations in user behavior and environmental conditions. In particular, most of the discussed techniques rely on the availability of large curated datasets, which are exploited for supervised learning.

2.3 Learning with Limited Labeled Data

In many real-world applications, acquiring labeled data is often expensive, time-consuming, and sometimes impractical. This is particularly true in healthcare, which requires expert annotation, in privacy-sensitive domains such as personal sensor data, and in complex modalities like sensor recordings, where manual labeling is labor-intensive and error-prone [Loeffler et al., 2024; Sheng and Huber, 2025].

As a result, many machine learning tasks face the challenge of working with limited labeled data, which can lead to overfitting and poor generalization performance. To address this challenge, several strategies have been proposed to leverage the available labeled data more effectively and to make use of the abundant unlabeled data [Ren et al., 2021; Chen et al., 2021; Ige and Mohd Noor, 2022]. In the following, we will discuss SSL (see subsection 2.3.1) and semi-supervised learning (see subsection 2.3.2), which cover the main classes of solutions and form the basis for the methods developed in this thesis. We will then briefly summarize complementary approaches and concepts for data-efficient learning, including transfer learning, few-shot learning, and active learning (see subsection 2.3.3).

2.3.1 Self-Supervised Learning

SSL is a paradigm that aims to transform large quantities of unlabeled data into a representation containing useful knowledge on the data, before specializing it for specific downstream tasks [Gui et al., 2024].

One key idea is to design a set of pretext tasks that can be solved using unlabeled data, with the expectation that solving them will help the model capture underlying patterns relevant to the target problems [Zhai et al., 2019; Radford et al., 2021]. Different methods vary in their choice of pretext tasks, which often depend on the characteristics of the data. These tasks may involve predicting missing data segments, rearranging shuffled data (as in jigsaw puzzles), or contrasting different augmented views of the same input [Gui et al., 2024; Liu et al., 2021].

An alternative solution to acquire this knowledge is to explicitly guide the learning process to produce representations that possess some desirable properties that are expected to facilitate effective knowledge transfer. This is done prominently by contrastive approaches that aim to build compact representations where similar inputs are close together while dissimilar ones are far apart [Liu et al., 2021; Chen et al., 2020].

Independent from the way the knowledge is learned, the representation is exploited in a second stage (see Figure 2.2), where a smaller set of labeled examples is used to train a task-specific network [Zhai et al., 2019; Radford et al., 2021]. By combining general knowledge learned from the unlabeled data with specific information from the labeled examples, SSL methods can outperform traditional supervised learning approaches in scarce labeled data scenarios [Ige and Mohd Noor, 2022; Gui et al., 2024].

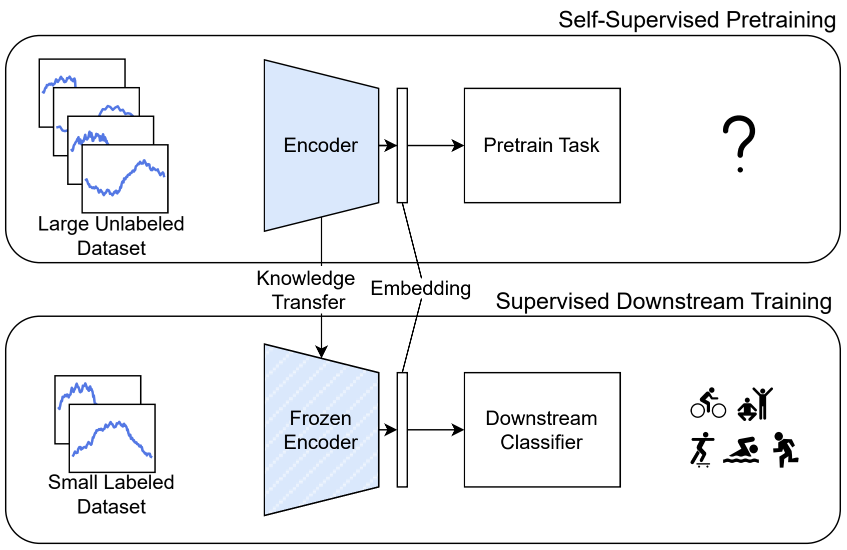


Figure 2.2. Self-supervised learning pipeline. The model is first pretrained on unlabeled data using a pretext task, then specialized on a small set of labeled examples for the downstream task. In the image the concept is illustrated for the HAR example.

In this study, we mainly focus on SSL methods for sensor signals, which are designed to learn representations from time-series data from wearable sensors, such as accelerometer and gyroscope signals. In the following sections, we first illustrate the most common pretext tasks used in this context, and then discuss the contrastive approaches.

Pretext Tasks

In typical pretext task-based methods, classification or regression tasks are used to train the network on originally unlabeled data in a supervised manner. This is achieved by generating the required supervision signal directly from the unlabeled input data itself.

Learning to encode input data into a lower-dimensional representation and then decode it back to the original input is one of the most prominent examples, which is realized through autoencoder networks. These networks proved to be effective in learning useful representations from unlabeled data, as they just need to minimize the reconstruction error between the input and the output [Zhang et al., 2024].

Masked autoencoders (MAE) and denoising autoencoders (DAE) are two popular variants of autoencoders that have been successfully applied to SSL tasks. Instead of the original input, MAEs and DAEs learn to reconstruct the original signal from a corrupted version of it. MAEs mask a portion of the input data, while DAEs add noise to the input data. These methods encourage the model to learn robust and meaningful representations that capture the underlying structure of the data [He et al., 2022; Zhang et al., 2023a; Vincent et al., 2008; Zhang et al., 2024].

Another related approach used with time-dependent data is to train the model to predict future values based on past observations. This method encourages the model to learn temporal dependencies and patterns in the data, which can be useful for various downstream tasks [Amrani et al., 2022].

Several approaches use classification rather than reconstruction as a pretext task. One of the most popular approaches to SSL is to augment the input data in various ways, such as applying transformations, cropping, or adding noise, and then train the model to predict what transformations were applied [Logacjov, 2024]. Variations of this method include the classification of the input data into a single class label that represents the applied transformation, effectively treating them as labels, or the prediction of which transformations were applied to the input data, allowing for multiple of them to be employed simultaneously [Saeed et al., 2019; Logacjov, 2024]. Also related are methods that shuffle segments of the input data and train the model to recover the original order [Misra and van der Maaten, 2020].

These methods are of general use and have found successful application to HAR problems [Saeed et al., 2019; Haresamudram et al., 2022; Amrani et al., 2022; Logacjov, 2024]. In this thesis, we will evaluate the performance of pretext tasks. We will also show how combining multiple pretext tasks and contrastive representations leads to better inference performance.

Contrastive Learning

The knowledge extracted from the pretraining step should create representations that emphasize the underlying structure of the input data. For HAR, we expect that in the representation space, similar activities should be clustered together while dissimilar ones should be separated.

In recent years, one of the most popular approaches has been contrastive learning (CL). Contrastive methods learn representations by comparing (contrasting) examples: they treat augmented views (i.e., transformations) of the same data instance as positive pairs and views of different instances as negative pairs. The goal is to maximize the similarity between positive pairs while minimizing the similarity measure between negative pairs [Liu et al., 2021; Chen et al., 2020]. Positive examples are useful to learn the structure of the data, while negative examples avoid representation collapse [Grill et al., 2020].

Early contrastive methods, such as SimCLR [Chen et al., 2020] and MoCo [He et al., 2020], have played a significant role in advancing SSL pretraining techniques, achieving, using very simple downstream classifiers, performance levels comparable to state-of-the-art supervised learning, thus demonstrating the effectiveness of this approach, especially for large-scale applications [Liu et al., 2021].

Training these types of networks requires the computation of loss functions that contrast positive and negative examples. The positive examples are generated through transformations of the original data. Negative examples, however, require access to different instances. For performance reasons, these negative examples are located within the same batch.

To ensure the presence of negative examples while avoiding the need for large batch sizes, MoCo [He et al., 2020] introduced a dynamic dictionary that maintains a queue of negative examples. This approach requires a set of encoded examples $k_0, k_1, k_2, \dots$ and a query q. A contrastive loss function is applied when a single key k_+ matches the query q, while all other keys are treated as negative examples. The InfoNCE loss [van den Oord et al., 2019] is used:

$$\mathscr{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)} \tag{2.1}$$

where q is the query vector, k_+ is the positive key, k_i are the K negative key vectors, and τ is a temperature hyperparameter. The required diversity of negative examples is obtained by maintaining a sufficiently large queue of keys. Very large queues, however, can impact performance due to the need for similarity searching.

SimCLR [Chen et al., 2020] adopts a mini-batch approach where each of the N original samples is augmented twice, producing $2N$ distinct views. The model is then trained to encourage similar pairs to align. Rather than explicitly selecting the negative examples, SimCLR treats the remaining $2N-1$ augmented views in the batch as negative for each positive pair [Gui et al., 2024]. Let $\mathrm{sim}(u, v) = \frac{u^\top v}{\|u\|\|v\|}$ represent the cosine similarity between two vectors u and v. The SimCLR NT-Xent loss for a positive pair (i, j) is defined as:

$$\mathscr{L}_{i,j} = -\log \frac{\exp\left(\mathrm{sim}\left(z_i, z_j\right) / \tau\right)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp\left(\mathrm{sim}\left(z_i, z_k\right) / \tau\right)} \tag{2.2}$$

where $1_{[k \neq i]}$ is an indicator function that is 1 if $k \neq i$, and τ is a temperature hyperparameter. The overall loss is computed across all positive pairs in the batch. The method is considerably than previous ones [Chen et al., 2020], but is limited by the need to ensure that each batch contains sufficiently diverse examples. This is typically achieved through randomization, which, however, may be suboptimal for very unbalanced datasets.

Since accessing sufficiently diverse negative examples is one of the main challenges of this class of approaches, recent research has concentrated on avoiding representation collapse only using positive examples [Grill et al., 2020]. Instead, they introduce different strategies like self-distillation and feature decorrelation. Still, the main principle of maintaining positive example consistency remains [Gui et al., 2024].

BYOL (Bootstrap Your Own Latent) [Grill et al., 2020] and SimSiam [Chen and He, 2021] are two prominent examples of self-distillation approaches. These methods use siamese networks, i.e., two identical networks with shared or different weights, to maximize the similarity between two augmented views of the same input (each going through a different network), but break the symmetry between the two branches to mitigate the risk of collapsing [Lee and Lee, 2023; Gui et al., 2024].

Specifically for HAR, Qian et al. [2022] compares SimCLR, BYOL, and SimSiam and other contrastive solutions with and without negative examples. The survey also studies the effect of different data augmentation strategies and variations of backbone network architectures. Haresamudram et al. [2022] evaluates both contrastive and pretext-based methods across different dimensions. They examine the properties of learned representations, such as similarity to supervised learning, linear separability, and implicit dimensionality. They also study the influence of dataset characteristics and the generalizability.

Other methods instead rely on feature decorrelation to avoid representation collapse. Barlow Twins [Zbontar et al., 2021] uses two identical "twin" networks, each processing a different augmented view of the same input (Y^A and Y^B). The method encorourages the cross-correlation matrix of their output embeddings (Z^A and Z^B) to resemble the identity matrix: meaning it pushes the on-diagonal elements (representing the correlation between the same components of the two views) to be close to 1, while the off-diagonal elements (representing the correlation between different components) are pushed to be close to 0.

Following on this idea, VICReg [Bardes et al., 2022; Shwartz-Ziv et al., 2023] introduced a more general approach that combines invariance and covariance terms from Barlow Twins with a new variance term that encourages the embedding vectors of samples within a batch to be different, explicitly preventing a collapse due to a shrinkage of the embedding vectors towards zero.

The overall loss function thus contains three terms:

- *Invariance*: encourages the model to produce similar embeddings for augmented views of the same input. It is formulated as the mean square Euclidean distance between the embedding pairs without any normalization.

$$\mathscr{L}_{\text{inv}}(Z^A, Z^B) = \frac{1}{n} \sum_{b=1}^{n} \left\| Z_b^A - Z_b^B \right\|^2 \tag{2.3}$$

  where n is the batch size, and Z_b^A and Z_b^B are the embeddings of the two augmented views of the same input.

- *Covariance*: encourages the model to produce uncorrelated embeddings for different inputs, and as in Barlow Twins, it is defined as:

$$\mathscr{L}_{\text{cov}}(Z) = \frac{1}{d} \sum_{i \neq j}^{n} [C(Z)]_{ij}^2 \tag{2.4}$$

  where $C(Z)$ represents the cross-correlation matrix of the output embeddings.

- *Variance*: is the main contribution of VICReg, and it encourages the model to produce diverse embeddings for different samples within a batch, thus preventing collapse. It is formally defined as a hinge function on the standard deviation of the embeddings along the batch dimension.

$$\mathscr{L}_{\text{var}}(Z^A) = \frac{1}{d} \sum_{j=1}^{d} \max(0, \gamma - S(Z_j^A, \varepsilon)) \tag{2.5}$$

  where Z_j^A represents the vector of the j-th component of the embedding Z^A, and $S(Z_j^A, \epsilon)$ is the regularized standard deviation. The constant γ (set to 1 in the original paper) controls the standard deviation threshold, and ε is a small parameter to avoid numerical instability.

The overall loss function is then a weighted sum of the three terms:

$$\mathscr{L}_{\text{VICReg}} = \mathscr{L}_{\text{inv}}(Z^A, Z^B) + \alpha \left(\mathscr{L}_{\text{var}}(Z^A) + \mathscr{L}_{\text{var}}(Z^B) \right) + \beta \left(\mathscr{L}_{\text{cov}}(Z^A) + \mathscr{L}_{\text{cov}}(Z^B) \right) \tag{2.6}$$

where α and β are hyperparameters that control the trade-off between the invariance, covariance, and variance terms. It is important to note that both the variance and covariance regularization terms are applied independently to the two branches of the architecture.

VICReg has been shown to be effective and can successfully function without sharing weights between the two branches, even with two completely disjoint branch architectures. This is particularly useful as it allows to contrast embeddings from different input modalities [Bardes et al., 2022; Shwartz-Ziv et al., 2023].

Learning from different modalities has also been studied in other CL contexts. For instance, Radford et al. [2021] applied this concept to images and text, while Nonnenmacher et al. [2022] contrasts expert features with time-series data from different domains, including HAR.

In this thesis, we explore both CL as a standalone method to learn good embeddings for HAR, and specifically VICReg as a general technique for multi-modal self-supervised learning.

2.3.2 Semi-Supervised Learning

While in the previous section we focused on methods that use the vast mass of unlabeled data solely to learn representations and the scarce labeled data to solve downstream tasks, semi-supervised learning leverages both labeled and unlabeled data together during training [Loeffler et al., 2024]. Based on the assumption that labeled and unlabeled data are sampled from the same or a similar distribution, the methods are designed to transfer the sparse information from the labeled data to the unlabeled one, thereby improving the model's performance on the target task.

A vast variety of semi-supervised methods have been proposed, for an extensive overview, we refer the reader to the classic book of Chapelle et al. [2010] for the general principles.

Yang et al. [2023] proposes to classify semi-supervised learning methods into the following categories:

- *Deep generative methods* learn a joint distribution over the input data and the labels, allowing them to generate new samples from the learned distribution.
- *Consistency regularization methods* encourage the model to produce similar predictions when unlabeled data is perturbed or augmented in different ways. By minimizing consistency losses, these models tend to move the decision boundaries away from dense regions of unlabeled data. This helps improve performance, especially on image classification tasks, where points in the same cluster usually belong to the same class [Zhai et al., 2019].
- *Graph-based methods* model the relationships between labeled and unlabeled data as a graph, where nodes represent data points and edges represent similarities. These methods propagate label information from labeled to neighboring unlabeled nodes.
- *Pseudo-labeling methods* involve using a model trained on labeled data to assign predicted labels to unlabeled examples, which are then treated as if they were true labels during further training [Grandvalet and Bengio, 2004; Lee, 2013].
- *Hybrid methods* mix and match the above approaches.

In this thesis, we employ pseudo-labeling techniques to refine the downstream task model after the self-supervised pretraining step.

2.3.3 Complementary Paradigms for Data-Efficient Learning

Beyond SSL and semi-supervised learning, many other paradigms have been proposed to leverage limited labeled data effectively. These approaches often complement SSL and semi-supervised learning, providing additional strategies to improve model performance in data-scarce scenarios.

Transfer Learning

Transfer learning is a general concept that covers techniques where knowledge learned in a specific setting is transferred to a different but related one. In the context of this thesis, we are particularly interested in transferring from a source domain where labeled data is abundant to a target domain where labeled data is scarce or unavailable. The intuition at the basis of many deep learning approaches is that early layers of deep neural networks capture general features that can be useful across different tasks, while later layers are more task-specific. This learning approach can dramatically reduce the amount of labeled data needed for training in the target domain [Ek et al., 2025].

Based on this concept, networks are typically separated into two parts: a deep feature extractor, which is trained on the source domain, and a simple classifier, which is trained on the target domain using the features extracted by the first part. Other approaches, such as fine-tuning, adjust the weights of the complete model, using pretraining as initialization for learning on the target domain data. This is particularly useful when the source and target domains are similar, such as in cases of personalization, where the model is first trained on a large dataset and then adapted to a specific user or context [Guo et al., 2019; Dhekane and Ploetz, 2025].

Few-Shot Learning

Few-shot learning (FSL) addresses scenarios where new classes or tasks must be learned from very few labeled examples, typically ranging from one to a handful of examples per class. In this context, the typical gradient descent update fails to produce meaningful loss improvements, since the number of examples is too low to constrain the error landscape.

At the core of FSL is the idea of meta-learning, where the model learns from a few examples by adapting to related tasks. It works on two levels: a base-level where the model is trained on individual tasks, and a meta-level where it learns to adapt quickly to new tasks by extracting common patterns and principles from its experience [Vinyals et al., 2016; Parnami and Lee, 2022]. In this setting, the meta-learner model is used to update the learner model parameters such that they perform better than before.

In this thesis, we will test our methods in a few-shot learning scenario, where very few labeled examples are available for the target task. We will, however, not focus on specific meta-learning aspects, but rather on leveraging the self-supervised representations learned in the pretraining step and fine-tuning using pseudo-labeling. Extending our work with few-shot learning techniques is an interesting direction for future work.

Active Learning

Active learning, also known as query learning, is a paradigm that allows the model to actively select the most informative examples to be labeled by an oracle (e.g., a human annotator).

In practice, active learning methods rank unlabeled samples by some measure of informativeness and then query the oracle to label the most useful ones. This way, the number of labeled examples is gradually increased, without the need to fully label the entire dataset, thus reducing the labeling effort and cost [Li et al., 2025a]. Common strategies include uncertainty sampling (querying samples where the model is least certain about the prediction), query-by-committee (maintaining multiple models and querying samples where they disagree the most), and expected model change (querying samples that would most change the model if labeled) [Li et al., 2025a].

We do not employ active learning in this thesis, but future work could focus on introducing it in the supervised downstream phase. This would allow to identify classes requiring more labeled examples and to further improve quality with the minimum additional manual labeling effort.

2.3.4 Multitask Learning for better embeddings

Many of the previously discussed methods aim to learn embeddings for capturing essential patterns in the data, but in practice they do so by focusing on a single task. Although strong performance and generalization can be expected when the target task closely resembles the pre-training task, this assumption does not always hold.

Multitask learning (MTL) is a sophisticated machine learning paradigm that aims to enhance the performance of multiple related tasks by leveraging shared information among them [Zhang and Yang, 2017]. This approach takes inspiration from the human cognitive process, where knowledge gained from one task, such as playing squash, can be effectively transferred and applied to improve skills in a related domain, like playing tennis [Zhang and Yang, 2022]. In the context of learning with limited labeled data, MTL can be particularly beneficial as it allows to create generalizable and efficient models that grasp multiple aspects of the data, thus creating more robust representations, moving beyond task-specific models.

Unlike transfer learning, where the primary goal is to improve performance on a specific target task using knowledge from a source task, MTL treats all tasks equally, aiming to learn a shared representation that benefits all tasks simultaneously [Zhang and Yang, 2022].

The general approach is to devise task-specific models and combine them into a single network that can learn from all tasks at once. In practice, MTL models share information in different ways. A common method is hard parameter sharing, where most layers in the model are shared across tasks, while only the final layers are task-specific. This works well when the tasks are strongly related and helps reduce overfitting. When tasks are less similar, soft parameter sharing can be used. In this approach, each task has its own model, but their parameters are encouraged to be similar using regularization [Yu et al., 2024]. Over time, MTL has evolved from simple shared layer models to complex and flexible architectures that can handle multimodal and cross-domain data [Zhang and Yang, 2022].

One of the critical challenges in MTL is how to combine the losses from individual tasks into a single objective for optimization. The most straightforward approach is to simply sum the losses of all tasks:

$$\mathscr{L}_{\text{total}} = \sum_{t=1}^{T} \mathscr{L}_t \tag{2.7}$$

where $\mathscr{L}_t$ is the loss for task t, and T is the total number of tasks. This naive approach is simple, but it does not account for the varying importance or difficulty of different tasks. One task may dominate the learning process due to its loss scale, leading to poor performance on other tasks. To address this, task weighting can be applied, where each task's loss is multiplied by a weight that reflects its importance or difficulty:

$$\mathcal{L}_{\text{total}} = \sum_{t=1}^{T} w_t \mathcal{L}_t \tag{2.8}$$

where w_t is the weight for task t. However, determining the optimal weights can be challenging. In the simplest approaches, weights are fixed, manually set by experts or simple rules, such as inversely proportional to training set size [Perera et al., 2018]. Although this method is conceptually simple, it often requires either an extensive grid search to tune the weights or relies on heuristics [Sener and Koltun, 2018a; Cipolla et al., 2018]. One of the key challenges is that in MTL, there is no universally optimal solution that leads to the best performance across all tasks, as improving one task may lead to a decrease in performance on another [Sener and Koltun, 2018a].

Dynamic weighting algorithms have been proposed to address this issue, where the weights are adjusted during the training process. Cipolla et al. [2018] introduced a method that learns the weights based on the homoscedastic or task-dependent uncertainty. The underlying principle is to train the model to prioritize learning from tasks with lower uncertainty, since their learning signals are more reliable. In the paper, the authors showcase an example of a multi-task model that combines classifications with regression tasks. For that case, the loss function is derived as follows:

$$\mathcal{L}_{\text{total}} = \sum_{k=1}^{K} \left[\frac{1}{2(\sigma_k^{\text{reg}})^2} \mathcal{L}_k^{\text{reg}} + \log \sigma_k^{\text{reg}} \right] + \sum_{j=1}^{J} \left[\frac{1}{(\sigma_j^{\text{class}})^2} \mathcal{L}_j^{\text{class}} + \log \sigma_j^{\text{class}} \right] \tag{2.9}$$

where σ^{reg} and σ^{class} are the learnable weights for the regression and classification tasks, respectively, and $\mathcal{L}_{\text{reg}}$ and $\mathcal{L}_{\text{class}}$ are the corresponding losses. In addition to weighting the losses, they also introduce logarithmic regularization terms. Without these terms, the weights would diverge towards infinity, minimizing the overall loss but rendering the whole process meaningless. This method allows for automatic and optimal learning of the task weights without the need for heuristics or extensive grid search for tuning. We will use it in our experiments to combine multiple self-supervised pretraining tasks with the goal of learning a more generalizable representation.

Other methods have also been proposed to dynamically balance task contributions. Dynamic Task Prioritization (DTP) [Guo et al., 2018] increases the weight of underperforming tasks by tracking their performance with moving averages and integrating this into a focal loss formulation to focus on harder tasks. If some tasks are way more difficult than others, DTP may erroneously over-focus on them, leading to suboptimal convergence. Instead of directly controlling the magnitude of individual loss values, Loss Scale Balancing (LSB) [Lee et al., 2021] periodically balances the "loss scale", defined as the product of a loss value and its corresponding weight. This approach has been shown to be particularly effective for pixel-wise vision tasks. Loss Discrepancy Control for MTL (LDC-MTL) [Xiao et al., 2025] first normalizes the task losses to a guessed common scale (e.g., dividing by the initial loss and applying logarithmic scaling), then minimizes the weighted sum of the normalized losses and optimizes the task weights.

One critical issue in MTL is the interference between gradients from different tasks. When tasks are not aligned, their gradients can conflict, and a simple summation of these gradients can result in an update vector that moves in a direction that is not beneficial for one or more losses, potentially causing a specific loss to increase even if the aggregated total loss decreases.

GradNorm [Chen et al., 2018] addresses the problem through the introduction of a penalty term that discourages the gradients of the tasks from being significantly different from the average. More recently, PCGrad [Yu et al., 2020] has been proposed to mitigate gradient conflicts by projecting the gradients of each task onto the normal plane of the other task gradients, thus reducing interference.

Several authors have, instead, proposed to avoid creating a single objective function. Multi-Objective Optimization (MOO) methods explicitly seek a Pareto-optimal solution, where no task can be improved without degrading another [Sener and Koltun, 2018b]. In this context, methods like the Multiple Gradient Descent algorithm (MGDA) [Désidéri, 2012] are used to determine a common descent direction for the shared parameters by combining individual task gradients. While MGDA is powerful, it struggles with scalability in deep learning due to the high dimensionality of gradients and the computational overhead of computing gradients for each task separately [Désidéri, 2012]. To mitigate this, newer methods optimize efficient upper bounds on the multi-objective loss using a single backward pass. This approach dynamically balances tasks and resolves gradient conflicts more effectively than static or heuristic weightings [Sener and Koltun, 2018b].

In our context, MTL is used with the objective of learning powerful and robust embeddings, rather than achieving optimal performance on each individual pretraining task. We thus employ dynamic weighting, which is flexible enough to define a combined target loss function without requiring manual hyperparameter tuning.

2.4 Federated Learning

Independent of the specific learning technique employed, there is a need to construct models from large sets of examples, labeled or not, possibly originating from various sources (e.g., different users, organizations, or devices).

When working with sensitive data, such as personal sensor data, it is often not possible to share the data with a central server for training. FL is a distributed machine learning approach that allows training models while keeping the data on the local devices.

In the following, we will first introduce the main concepts of FL, and then provide an overview of the specific adaptations for self-supervised and semi-supervised settings.

2.4.1 Standard Federated Learning

FL emerged around 2016 with several foundational works [Shokri and Shmatikov, 2015; Konečný et al., 2016, 2017; McMahan et al., 2017] that originated a subfield of research within machine learning [Aledhari et al., 2020; Zhang et al., 2021; Huang et al., 2024]. The main concept is to keep the data locally while only sharing the model parameters or other derived information with a central server that coordinates the training process to build a global knowledge (see Figure 2.3).

The first and most widely used method, called Federated Averaging (FedAvg) [McMahan et al., 2017], combines local stochastic gradient descent (SGD) performed independently on each client with a central server that periodically averages the resulting model updates.

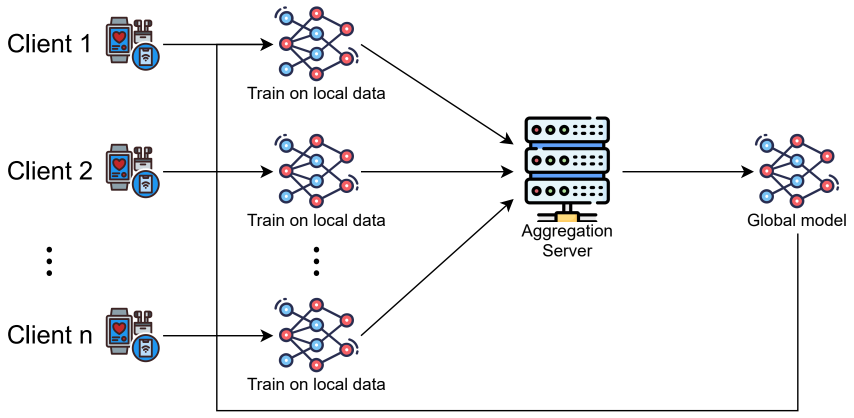


Figure 2.3. Federated Learning pipeline.

The training process is performed in rounds. In each round, the server sends the current global model parameters to all available clients. Each client then performs local training on its own data, traditionally using stochastic gradient descent (SGD) or other optimization algorithms such as Adam [Kingma and Ba, 2014]. After a few local epochs, the clients send the updated local models to the central server. The server then averages the local models to obtain a new global model, which is then sent back to the clients for the next round of training. This process continues until convergence or until a predefined number of rounds is reached. The method still works well even if some clients don't take part in every round, making it flexible and able to handle client dropouts or failures.

Since the introduction of FedAvg, many advances have been made to improve FL in various aspects, in particular tackling communication efficiency [Nariman and Hamarashid, 2025], but also generalization from locally updated heterogeneous clients, robustness to attacks from malicious clients, and fairness in the allocation of resources [Kairouz et al., 2021; Huang et al., 2024].

In this thesis, we will focus on the standard FL setting using Adam [Kingma and Ba, 2014] as a local optimizer.

2.4.2 Federated Learning with Limited Supervision

FL was originally developed with the core motivation to enable collaborative learning while preserving data privacy in supervised settings, where each client has access to correctly labeled data. However, this is common in cross-silo federated settings, where large organizations collaborate to train a model, but is often unrealistic in cross-device FL, where a large number of clients participate in the training and users have to actively interact to properly annotate their local data [Lubana et al., 2022]. For this reason, FL has been extended to work with limited supervision, by designing federated implementations of standard SSL approaches [Jin et al., 2023].

Similar to MTL (see subsection 2.3.4), locally optimizing clients on their own data could lead to diverging gradient updates, especially when the data is very heterogeneous across clients. A typical situation is when each client has only information on only one or few particular classes of data and no information on many others. In HAR, for instance, we could have clients with running and walking examples, and other clients with cycling or sitting, but no client with examples of all classes. To address this, several techniques have been introduced that increase the information exchanged between clients and server. Examples of approaches include sharing embeddings between clients and server [Zhang et al., 2023b], or between clients, using them as negative examples in the local contrastive training [Wu et al., 2022]. These approaches, however, can lead to privacy concerns and potential data leakage. Moreover, from client-specific embeddings, it may be possible to revert the representations to reconstruct original data [Dosovitskiy and Brox, 2016; Nash et al., 2019].

For these reasons, several approaches have tried to achieve federated unsupervised learning for very unbalanced client data distribution while minimizing the information exchanged. One prominent example is Orchestra [Lubana et al., 2022], in which clients send to the server, together with the model updates, local centroids that partition the data into equally sized clusters. The server then uses these centroids to compute a global clustering, which is then sent back to the clients that use them to improve local training with non-local information.

The main subject of this thesis is learning with limited supervision, and in particular SSL and MTL. For this, we also demonstrate the feasibility of an implementation in a federated setting. Our results show that it is possible to achieve reasonable performance using standard FL techniques. This is because, in our case, the clients have sufficient broad information for the local training (see chapter 7). Integrating our solutions with advanced self-supervised FL techniques, such as Orchestra, and testing them for challenging data distributions is an interesting direction for future work.

2.5 Closing Remarks

HAR, which consists of recognizing human activities from sensor data, is a prominent example of learning in scenarios where data is distributed, heterogeneous, and scarcely labeled. The main focus of this work is to evaluate how it is possible to learn effective representations in these settings.

To do that, we will employ and extend state-of-the-art solutions in SSL and MTL. In particular, we will evaluate the performance of different SSL pretraining tasks and investigate how combining multiple pretext tasks and contrastive representations may lead to better inference performance, especially in few-shot learning scenarios. We will also explore the possibility of using pseudo-labeling techniques to further refine the downstream task model after the self-supervised pretraining step. We will test our methods in a federated setting, where the data is distributed across multiple clients. This is particularly relevant in fields like HAR, where collected data is often sensitive and sharing it is not feasible due to privacy concerns. HAR is definitely not the only domain where the need to leverage unlabeled data is crucial, and the methods we will develop can be applied to other domains as well. We will show in particular how to adapt our framework for image recognition tasks.

Chapter 3

Data and Experimental Set-up

This thesis will introduce and compare techniques for leveraging unlabeled data to improve the performance of machine learning models. The methods are general but will be benchmarked on HAR tasks, where dealing with limited labeled data is a common challenge. In this chapter, we will provide details on the data used consistently across all experiments and the testing methods employed.

3.1 Introduction

The main objective of this thesis is to explore how to exploit unlabeled data to improve the performance of machine learning models in scenarios where labeled data is scarce. This is particularly relevant in many real-world applications, where collecting labeled data is expensive, time-consuming, or unfeasible due to the sensitive nature of the data. The techniques discussed and proposed are applicable to any domain, but to provide a concrete example, we focus mostly on HAR, which is a good fit for this purpose (see section 2.2).

To ensure a fair comparison of the proposed methods, we will use the same dataset and testing protocol throughout the thesis, unless specifically indicated. This allows us to isolate the effects of the different techniques and provide a clear evaluation of their performance. Even though the focus is on unlabeled data, we need a fully curated dataset to perform the evaluations. The field of HAR has a rich literature, with many benchmark datasets available. Prominent examples include the PAMAP dataset [Reiss and Stricker, 2012], which collects data from multiple IMU sensors worn on the body and a heart rate monitor, and the WISDM dataset [Kwapisz et al., 2011], which comprises accelerometer data collected from smartphones (and optionally smartwatches) worn by users during various daily activities. Other datasets differ in the type of sensors, wearable devices, and the activities performed. For instance, several datasets contain data collected from non-standard wearable devices, such as smartglasses (e.g., OCOsense [Emteq Labs, 2022]) or earbuds (e.g., USI-HEAR [Laporte et al., 2024]).

For the HAR experiments in this thesis, we have chosen the UCI HAR dataset [Anguita et al., 2013]. It consists of accelerometer and gyroscope data collected from smartphones worn by participants. The dataset is publicly available and has been extensively used in previous research, making it a suitable choice for evaluating the proposed techniques. The characteristics of the dataset are described in section 3.2.

The dataset is typically employed in fully supervised learning settings. Since we are interested in sparse labeling scenarios, we will need to split the data into a sparsely labeled training and validation set, and a fully labeled test set. The splitting technique is presented in section 3.3, which also describes the full testing protocol. To evaluate the generality of the proposed methods, we will also extend the analysis to the image recognition domain. The data and methods used for this purpose are described in chapter 8.

3.2 UCI HAR Dataset

The UCI HAR dataset [Anguita et al., 2013] was collected from 30 participants who performed six different activities (walking, walking upstairs, walking downstairs, sitting, standing, and lying) while wearing a smartphone (Samsung Galaxy S II) on their waist. It includes three-axis accelerometer (linear acceleration) and gyroscope (angular velocity) data, which were recorded at a frequency of 50 Hz.

The raw data was preprocessed to reduce noise using a median filter and a 3rd order low-pass Butterworth filter with a cutoff frequency of 20 Hz. The acceleration signal was separated into body acceleration and gravity components using another Butterworth low-pass filter with a corner frequency of 0.3 Hz, assuming that the gravitational force has only low-frequency components. The 9 signals (3-axis body acceleration, 3-axis gyroscope, and 3-axis gravity) were then sampled in fixed-width sliding windows of 128 samples (2.56 seconds) with a 50% overlap between them, resulting in a total of 10,299 samples.

Two different sample representations are provided: one with the raw sensor (9 channels of 128 samples each) and a second one with a feature vector (561 values) extracted from the raw data. The features were computed using a combination of time-domain and frequency-domain analyses (see Table 3.1), resulting in a rich representation of the participants' movements. They include statistical measures such as mean, standard deviation, and energy, as well as frequency-domain features obtained through Fast Fourier Transform (FFT).

Feature	**Description**
`mean()`	Mean value
`std()`	Standard deviation
`mad()`	Median absolute deviation
`max()`	Largest value in array
`min()`	Smallest value in array
`sma()`	Signal magnitude area
`energy()`	Energy measure. Sum of the squares divided by the number of values.
`iqr()`	Interquartile range
`entropy()`	Signal entropy
`arCoeff()`	Autoregression coefficients with Burg order equal to 4
`correlation()`	Correlation coefficient between two signals
`maxInds()`	Index of the frequency component with largest magnitude
`meanFreq()`	Weighted average of the frequency components to obtain a mean frequency
`skewness()`	Skewness of the frequency domain signal
`kurtosis()`	Kurtosis of the frequency domain signal
`bandsEnergy()`	Energy of a frequency interval within the 64 bins of the FFT of each window.
`angle()`	Angle between two vectors.

Table 3.1. List of measures for computing feature vectors [Anguita et al., 2013].

The dataset is split into a training set and a test set. 70% randomly selected users (21 out

Activity	Training Set		Test Set	
	Samples	Percentage	Samples	Percentage
`Walking`	1226	16.8%	496	16.8%
`Walking Upstairs`	1073	14.6%	471	15.9%
`Walking Downstairs`	986	13.4%	420	14.3%
`Sitting`	1286	17.5%	491	16.7%
`Standing`	1374	18.7%	532	18.2%
`Laying`	1407	19.1%	537	18.2%

Table 3.2. Distribution of activity labels in the UCI HAR dataset for both training and test sets. The dataset shows a relatively balanced distribution across the six activity classes with slight variations in the number of samples per activity. The percentages are calculated based on the total number of samples in each set.

of 30) are included in training, while the remaining 30% (9 users) are used for testing. This results in 7,352 training samples and 2,947 test samples. Table 3.2 illustrates the distribution of activity labels in both sets. As can be seen, the dataset is relatively balanced, with the most represented activity being *Laying* (19.1%) and the least represented being *Walking Downstairs* (13.4%). The distribution of activity labels stays consistent across both sets, ensuring that the model is trained and evaluated on a representative sample of the data.

3.3 Experimental Setup

The UCI HAR dataset offers a solid foundation for evaluating our methods, but needs to be adapted to fit our sparse labeling scenarios. We need, in particular, to define labeled and unlabeled data partitions and validation strategies.

Moreover, to ensure fair and consistent evaluation across all experiments in this thesis, we have established a standardized testing protocol that is applied uniformly to all methods and approaches presented in subsequent chapters.

3.3.1 Creating a Limited Labeled Data Scenario

The original split of the UCI HAR dataset into training and test sets is maintained, but it is insufficient for our purposes. We keep the testing set unchanged, but adapt the training set to simulate scenarios with limited labeled data availability.

As introduced in section 2.2, our techniques are self-supervised, requiring two phases: a pretraining phase on massive amounts of unlabeled data and a downstream training phase on a smaller set of labeled data. To simulate this situation, we create a pretraining dataset by exploiting the entire training set, stripped of the labels. We then create, from the original training set, a series of labeled data regimes for the downstream training phase by systematically reducing the number of labeled samples available. To test realistic scenarios, we randomly select a fixed number of samples per activity per user at each regime, keeping all the users and activities. Since we are interested in evaluating the performance in challenging situations, we start from just one sample per user per activity (126 samples in total) and increase the number up to six with a step of one. As a result, we obtain the six sparsely labeled data regimes shown in Table 3.3. We also include a full dataset regime (7,352 samples) where the entire training set is used to establish an upper bound on performance. Data selection is performed in such a way that the same samples are used across all methods, ensuring a fair comparison.

Regime	**Labeled Samples**	**Training Samples**	**Validation Samples**
1	126	96	30
2	252	192	60
3	378	288	90
4	504	384	120
5	630	480	150
6	756	576	180
Full Dataset Regime	7352	5583	1769

Table 3.3. Summary of labeled data regimes used in the experiments. Each regime specifies the number of labeled samples available for training and validation, simulating scenarios with varying amounts of labeled data. The full dataset regime uses all available labeled training data.

3.3.2 Cross-Validation Strategy

All experiments are conducted using a 5-fold cross-validation approach on the training set, splitting the training set into training and validation sets based on the user IDs and a fixed random seed. For each fold, the 21 users in the training set are randomly divided into two groups: 16 users for training and the remaining 5 users for validation. The original test set, containing data from the other 9 users, remains fixed throughout all experiments and is only used for final evaluation to prevent any form of test set contamination. We apply the same cross-validation strategy to the pretraining and downstream training phases.

3.3.3 Training Configuration

The methods are implemented in PyTorch using a modular approach, allowing for easy integration of different methods and simplifying the configuration of the training parameters (see Appendix A). Unless explicitly discussed, we use the same configuration for all methods.

During the pretraining phase, the SSL training aims to learn robust representations from the unlabeled data. As will be discussed in subsection 5.3.4, we observed that saving the best performing model based on pretraining loss may lead to worse results in the downstream task compared to simply saving the model obtained after a fixed larger number of epochs. Therefore, to set a consistent measure for all methods, we train all SSL encoders for a fixed number of epochs (typically 200) and select the model based on the best validation accuracy on the downstream task.

During downstream training, the pretrained encoder weights are frozen, and only the classifier head parameters are updated. This approach ensures that the learned representations from the pretraining phase are preserved while allowing the classifier to adapt to the specific HAR task. The downstream classifier is trained for a maximum of 100 epochs. After training, the best model is selected based on the validation accuracy and evaluated on the fixed test set.

3.3.4 Evaluation Metrics

Model performance is evaluated using two primary metrics: accuracy (the percentage of correctly classified samples) and macro-averaged F1 score (the unweighted mean of F1 scores across all activity classes). Both metrics are computed on the fixed test set after training is complete. For statistical robustness, all numerical results are reported as mean ± standard deviation over the 5 cross-validation folds. In the plots, we report the mean with a line along with its 90% confidence interval as shaded areas. For simplicity, we will refer to accuracy as the primary metric as, due to the balanced nature of the dataset, F1 mostly reflects the same trends as accuracy.

3.3.5 Hyperparameter Selection

The hyperparameters used throughout this thesis are either adopted from established literature or selected through limited training runs. Given that it is not the main focus of this work, we deliberately avoid extensive hyperparameter optimization to ensure that our comparisons reflect the intrinsic capabilities of different methods rather than the quality of hyperparameter tuning. For SSL methods, we use standard hyperparameter values reported in the original papers when available, for example, trade-off weights between variance, invariance, and covariance regularization in VICReg (see Equation 2.6). For novel combinations and multitask approaches, we perform a limited search over key parameters (e.g., embedding dimensions) as discussed in the specific chapters (see chapter 5). This standardized protocol ensures that all comparisons presented in this thesis are fair, reproducible, and representative of real-world scenarios where labeled data is scarce and computational resources are limited.

3.4 Closing Remarks

The UCI HAR dataset and experimental setup described in this chapter will be used consistently throughout this thesis to evaluate and provide a fair comparison of the proposed and tested methods. The only exception is the preliminary evaluation in the image recognition domain, which is discussed in chapter 8. For reproducibility information and further details, Appendix A provides links to the development repository, implementation details, and notes on the computing infrastructure used.

Chapter 4

Self-Supervised Learning

We analyze the behavior of different SSL techniques under specific conditions in the low-labeled data regime, showing how they outperform traditional supervised training when the amount of labeled data is limited. In the next chapter, we will further improve the performance by combining multiple pretraining tasks together to create better embeddings.

4.1 Introduction

As discussed in chapter 2, SSL has emerged as a powerful approach for learning useful representations from unlabeled data, producing embeddings applicable to a variety of downstream tasks. In this work, the specific problem is classifying human activities from sensor data, which in a self-supervised framework is addressed in two stages: first, pretraining an encoder on large amounts of unlabeled data, and then combining it with a small classifier trained on a small dataset of labeled examples. The embedding produced by the encoder strongly depends on its training techniques.

In this chapter, through extensive experimentation, we explore different SSL pretraining tasks and their impact on the quality of the learned representations. This is done within a unified framework that allows us to easily compare different methods and their performance. In the following, we describe the overall architecture, the selected pretraining tasks, and the results obtained.

4.2 Architecture

SSL pretraining can be realized through various pretraining tasks, each with its own training objective. In their architectures, all tasks include an encoder that learns to extract meaningful features from the input data, creating an embedding. The embedding is then exploited by task-specific heads to solve the pretraining problem. After pretraining, the encoder is frozen and used to generate embeddings for a downstream classifier that solves the actual task of interest. Figure 4.1 illustrates our general architecture for SSL pretraining and downstream classification. The architectural components are detailed in the following sections.

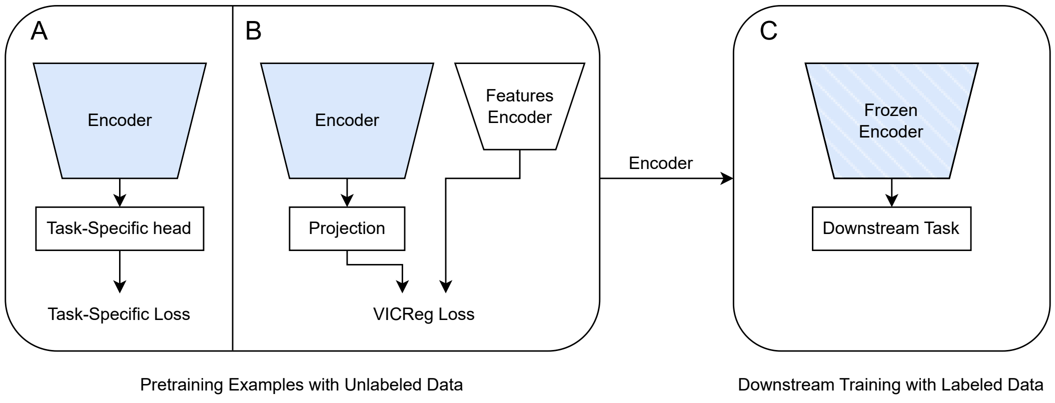


Figure 4.1. SSL setup overview. (A) Example of an architecture for pretraining with task-specific approaches (e.g., augmentation classification, reconstruction) where the encoder learns representations using a task-specific head and loss function on unlabeled data. (B) Pretraining with multimodal contrastive learning (VICReg with features), involving the encoder, a projection head, a designed feature encoder architecture that maps the other input modality to the same space as the main data branch, and a specialized contrastive loss. (C) The encoder from A or B is frozen and used to generate embeddings for a downstream supervised classification task.

4.2.1 Encoder

The goal of this thesis is not to find or develop the best encoder, but rather to use an encoder that works decently well as a common baseline across all the tested and proposed methods. We will thus use the same encoder for all the methods developed and evaluated in this work. Prior work has shown that Fully Convolutional Networks (FCN) are effective for HAR tasks and have demonstrated good performance on the UCI HAR dataset [Qian et al., 2022]. We will then use a similar architecture with the following configuration: the input layer takes 9×128 scalar values, representing the readings of the 9 sensor channels of a temporal window of 128 samples. The encoder consists in 3 convolutional blocks. Each block applies a 1D convolution with kernel size 8 (and increasing channel sizes: 32, 64, and 128), followed by batch normalization, ReLU activation, and temporal downsampling via max-pooling. A dropout rate of 0.35 is applied after the first block to prevent overfitting. After the last block, adaptive average pooling is applied to reduce the output to a desired size (typically 128). We later show in subsection 5.3.6 that increasing the embedding size does not significantly improve the performance.

It should be noted that the specific encoder architecture choice does not significantly change the findings presented in this thesis, as demonstrated by an ablation study (see subsection 5.3.5).

4.2.2 Downstream Classifier

The downstream classifier is responsible for taking the embeddings produced by the encoder and making predictions for the HAR task. We consistently use the same classifier in all the experiments. It consists of a fully connected layer that maps the embedding to a hidden size of 512, followed by ReLU activation, 0.3 dropout rate, and a final fully connected layer that maps the hidden size to the number of classes (6 for the UCI HAR dataset). Since it is out of scope for this thesis to find the best classifier configuration, we did not perform any hyperparameter optimization.

4.2.3 Supervised Baseline

To showcase the effectiveness of the self-supervised methods, we will use a supervised baseline that uses solely the labeled data for training. It consists of the encoder followed by the downstream classifier. Different from all the other SSL methods, the encoder is not pretrained, but it is trained from scratch together with the classifier.

4.2.4 Self-Supervised Variations

Each of the self-supervised methods has its own specific architecture, but they all share the same encoder and downstream classifier. The individual architectures are described in the following sections.

4.3 Pretrain Tasks

As discussed in chapter 2, multiple methods have been proposed to train an encoder in a self-supervised manner. In this section, we briefly describe the different tasks and methods tested in this thesis, which include both pretext tasks and contrastive learning approaches.

Pretext tasks indirectly guide the model to learn useful representations by solving specifically designed problems. Alternatively, contrastive learning paradigms explicitly guide the model to learn representations with certain imposed characteristics by bringing different augmented views of the same input closer together in the embedding space while pushing apart representations of differing samples. These approaches encourage the encoder to capture the underlying structure of the data, resulting in embeddings that are more effective for downstream tasks like HAR.

The goal is to create a good embedding that can be used for the downstream HAR task. Different pretrain tasks can learn different aspects of the data, thus creating different embeddings. In this chapter, we test them individually, while in the next chapter, we will explore their combinations.

4.3.1 Pretext Tasks

Among the many pretext tasks proposed in the literature (see section 2.3.1), we have selected classifying augmentations and reconstruction of a masked input as our primary tasks since they are commonly used and have shown good performance in previous work [Haresamudram et al., 2022; Logacjov, 2024; Zhang et al., 2024].

Classifying Augmentations

One of the simplest pretext tasks is to train a model to identify whether signal transformations are applied to the data or not. Saeed et al. [2019] first used this method to explore SSL on HAR accelerometer data. Their approach involves a multitask learning setup with a separate head for each transformation.

Our implementation is inspired by their work, but we simplify it by using a single head serving as a classifier for all transformations. The transformations used are inspired by the work of Qian et al. [2022] and include both time and frequency domain augmentations:

- **No augmentation**: the original signal, used as a baseline.
- **Negate**: inverts signal sign.
- **Shuffle**: randomly shuffle the sensor channels (per sample) to test channel invariance.
- **Scale**: amplify each channel by a randomly generalized distortion with a mean of 2 and standard deviation of 1.1.
- **Time Flip**: reverses signal along the time axis.
- **Rotation**: applies a 3D rotation to each tri-axial sensor reading by rotating it around a randomly selected axis using an angle drawn uniformly from $-\pi$ to π.
- **Permute**: split the signal into up to 5 random segments along the time axis, and shuffle their order.
- **Resample**: upsample with linear interpolation the signal to 3 times its original frequency, then randomly downsample it to its original shape.
- **Time Warping**: warps the signal in time using smooth cubic splines, randomly stretching it.
- **Permuted Jitter**: applies permutation and jitter to the signal.
- **Scaled Jitter**: applies jitter and scale to the signal.
- **High-Frequency Component (HFC)**: high-frequency component extraction via FFT masking.
- **Low-Frequency Component (LFC)**: low-frequency component extraction via FFT masking.
- **Phase Shift**: randomly shifts the phase of the signal by a random amount uniformly drawn from $-\pi$ to π.
- **Partial Amplitude Phase**: applies Gaussian noise to amplitudes and random shifts to phases within a randomly selected half-length segment of the frequency domain.
- **Full Amplitude Phase**: Perturbs amplitudes and phases across the entire frequency domain, using the same noise and phase shift ranges as in the previous method.

The model is composed of the FCN encoder, followed by two fully connected layers for classification. The hidden layer has the same size as the embedding (typically 128) and uses ReLU activation, while the output layer has a size equal to the number of transformations (16 in total).

Reconstruction

As shown in section 2.3.1, reconstruction-based methods have been widely used in SSL. The idea is to train an encoder-decoder network that learns either the identity function or to reconstruct the original signal from a damaged version of it. The network maps the input signal to a lower-dimensional representation, and then reconstructs the original signal from this representation. This approach encourages the model to include all the relevant information in the learned representation, as it needs it for the reconstruction task.

In this thesis, we connect our encoder to a simple decoder that consists of three fully connected layers mapping back to the original signal shape. The input to the decoder is the output of the encoder (typically 128-dimensional). The following hidden layers have the size of 512 and 1024, while the output layer has the same size as the input (9 channels $\times$ 128 samples for a total of 1152). The decoder uses ReLU activation in the hidden layers and no activation in the output layer.

The masked input is created by randomly selecting a percentage of the input signal (50%) and replacing it with zeros. The encoder-decoder model is trained to minimize the mean squared error between the original signal and the reconstructed version from the masked input.

4.3.2 Contrastive Learning

CL has become one of the most popular approaches to SSL in recent years, as discussed in section 2.3.1. In this thesis, we will use different CL methods, each with its own specific architecture and loss function.

Standard Contrastive Learning

The first method is a standard CL approach: the SimCLR method [Chen et al., 2020]. The model consists of a FCN encoder, followed by a projection head that maps the learned representation to a lower-dimensional space to reduce computational complexity. The projection head is composed of two fully connected layers with a ReLU activation in between. The hidden layer has the same size as the embedding (typically 128) and ReLU activation, while the output layer has a size of 128.

At training time, the model sees two augmented views of the same input signal and learns to maximize the similarity between the two views while minimizing the similarity with all other views in the batch (see section 2.3.1). As in the original SimCLR paper, the contrastive loss is computed using normalized temperature-scaled cross-entropy (NT-Xent) (see Equation 2.2).

VICReg

To test the performance of CL without negative examples, we selected VICReg [Bardes et al., 2022; Shwartz-Ziv et al., 2023] that uses a loss function with variance, invariance, and covariance terms to encourage the model to learn expressive representations (see section 2.3.1).

VICReg has the same architecture as for our SimCLR implementation, with the FCN encoder followed by a projection head that maps the learned representation to a lower-dimensional space of 128 dimensions.

Contrasting with Features

All the previous approaches, based on pretext tasks or CL, take as input the raw sensor data. Since VICReg readily supports different input modalities and architectures, we use it to contrast the embeddings from raw sensor data with those extracted from expert features (see chapter 3), which can be interpreted as a different representation space.

In this contrastive architecture, the first branch is identical to the standard VICReg use case, while the second one needs a different encoder and projection head. Since the feature branch is discarded after training, as it serves only to provide contrastive information, it is implemented as a single network that maps features to the same output dimensionality of the raw data branch. The input size is equal to the feature size (561). The two hidden layers are of size 512 and 256, with ReLU activation, and the output layer has a size of 128, matching the output of the projector of the other branch.

4.4 Results

All the methods described are implemented in the same framework (see Appendix A), and were tested on the same conditions on the UCI HAR dataset (see chapter 3).

Table 4.1 shows the downstream performance of the baseline supervised model and the different SSL approaches. For each method, we report the accuracy and F1 score (macro-averaged) for different amounts of labeled training data, ranging from 1 sample per activity per user (126 labeled samples in total) to the full dataset (7352 labeled samples). Figure 4.2 provides a visual representation of the accuracy for a more intuitive comparison.

The first notable finding is the clear advantage of SSL techniques relative to plain supervised learning in low-data regimes. In the most demanding few-shot setting (1 labeled sample per class per user, 126 labeled samples in total), the supervised baseline (red line in Figure 4.2) achieves an accuracy of approximately 70%. In contrast, most SSL models start with accuracies between 83% and 89%. The significant improvement (13-19 percentage points) demonstrates the effectiveness of the embeddings learned by the SSL methods. The SL baseline lacks the pre-training step and struggles to learn meaningful features from a small labeled dataset, resulting in significantly lower performance.

As the number of labeled training examples increases, the performance gap between the supervised baseline and the SSL methods narrows. When all the dataset is used for training, the supervised baseline obviously aligns with the SSL methods, achieving a competitive accuracy of 90.69%. It should be mentioned that this model is trained from scratch, while the other classifiers freeze the encoder and only train the classifier on top of the learned embeddings. This could make the model overfit to the training data as it has more parameters to train. Bozkurt [2021] studied the performance of different architectures on supervised learning using the fully labeled UCI HAR dataset and reported a maximum accuracy of 96.81% on the test set using a Deep Neural Network (DNN). Without any optimization, the best SSL techniques are not far from such a state-of-the-art performance, and better than several of the other surveyed fully supervised methods [Bozkurt, 2021].

VICReg with identical branches, VICReg contrasting features with raw data, and SimCLR (respectively in blue, cyan, and orange color in Figure 4.2) consistently form the best-performing methods across all training data sizes ranging from 2 to 6 labeled samples per class per user (252 to 756 labeled samples in total). They achieve accuracies of 89.38% to 91.14%. CL obtains the best accuracy of 93.06% when trained with all the labeled data. An important finding is that

Method	Training Labels	Accuracy	F1 Score
Supervised Baseline	1 (126)	69.91 ± 3.80	0.68 ± 0.05
	2 (252)	77.84 ± 7.29	0.77 ± 0.08
	3 (378)	85.56 ± 2.10	0.85 ± 0.02
	4 (504)	85.67 ± 0.93	0.85 ± 0.01
	5 (630)	85.86 ± 1.69	0.86 ± 0.02
	6 (756)	86.39 ± 0.71	0.86 ± 0.01
	All (7352)	90.69 ± 2.08	0.91 ± 0.02
Classifying Augmentations	1 (126)	83.66 ± 1.70	0.83 ± 0.02
	2 (252)	85.95 ± 1.05	0.86 ± 0.01
	3 (378)	86.85 ± 0.72	0.87 ± 0.01
	4 (504)	87.97 ± 1.06	0.88 ± 0.01
	5 (630)	88.35 ± 0.78	0.88 ± 0.01
	6 (756)	88.14 ± 0.77	0.88 ± 0.01
	All (7352)	89.79 ± 1.24	0.90 ± 0.01
Reconstruction	1 (126)	86.18 ± 1.17	0.86 ± 0.01
	2 (252)	88.22 ± 1.86	0.88 ± 0.02
	3 (378)	88.44 ± 1.08	0.88 ± 0.01
	4 (504)	88.75 ± 1.17	0.89 ± 0.01
	5 (630)	89.16 ± 0.76	0.89 ± 0.01
	6 (756)	89.72 ± 0.98	0.90 ± 0.01
	All (7352)	90.02 ± 1.02	0.90 ± 0.01
Contrastive Learning (SimCLR)	1 (126)	84.92 ± 2.47	0.84 ± 0.03
	2 (252)	89.70 ± 1.07	0.90 ± 0.01
	3 (378)	90.38 ± 0.59	0.90 ± 0.01
	4 (504)	89.77 ± 0.83	0.90 ± 0.01
	5 (630)	91.16 ± 1.74	0.91 ± 0.02
	6 (756)	91.56 ± 0.58	0.91 ± 0.01
	All (7352)	93.02 ± 0.85	0.93 ± 0.01
VICReg	1 (126)	85.72 ± 3.76	0.85 ± 0.04
	2 (252)	89.44 ± 1.09	0.89 ± 0.01
	3 (378)	89.69 ± 2.02	0.90 ± 0.02
	4 (504)	89.51 ± 2.23	0.89 ± 0.02
	5 (630)	90.67 ± 2.20	0.91 ± 0.02
	6 (756)	90.86 ± 1.06	0.91 ± 0.01
	All (7352)	91.41 ± 1.07	0.91 ± 0.01
VICReg (features)	1 (126)	88.99 ± 0.93	88.99 ± 0.01
	2 (252)	89.15 ± 1.55	89.15 ± 0.02
	3 (378)	89.66 ± 1.62	89.66 ± 0.02
	4 (504)	89.58 ± 0.78	89.58 ± 0.01
	5 (630)	90.15 ± 1.23	90.15 ± 0.01
	6 (756)	90.19 ± 0.88	90.19 ± 0.01
	All (7352)	89.81 ± 1.32	89.81 ± 0.01

Table 4.1. Results of the different pretext tasks on the UCI HAR dataset. The results (accuracy and macro-averaged F1 score) are reported as mean ± standard deviation over 5 runs. The numbers in parentheses indicate the number of labeled samples used during pretraining. The encoders are pretrained for 200 epochs.

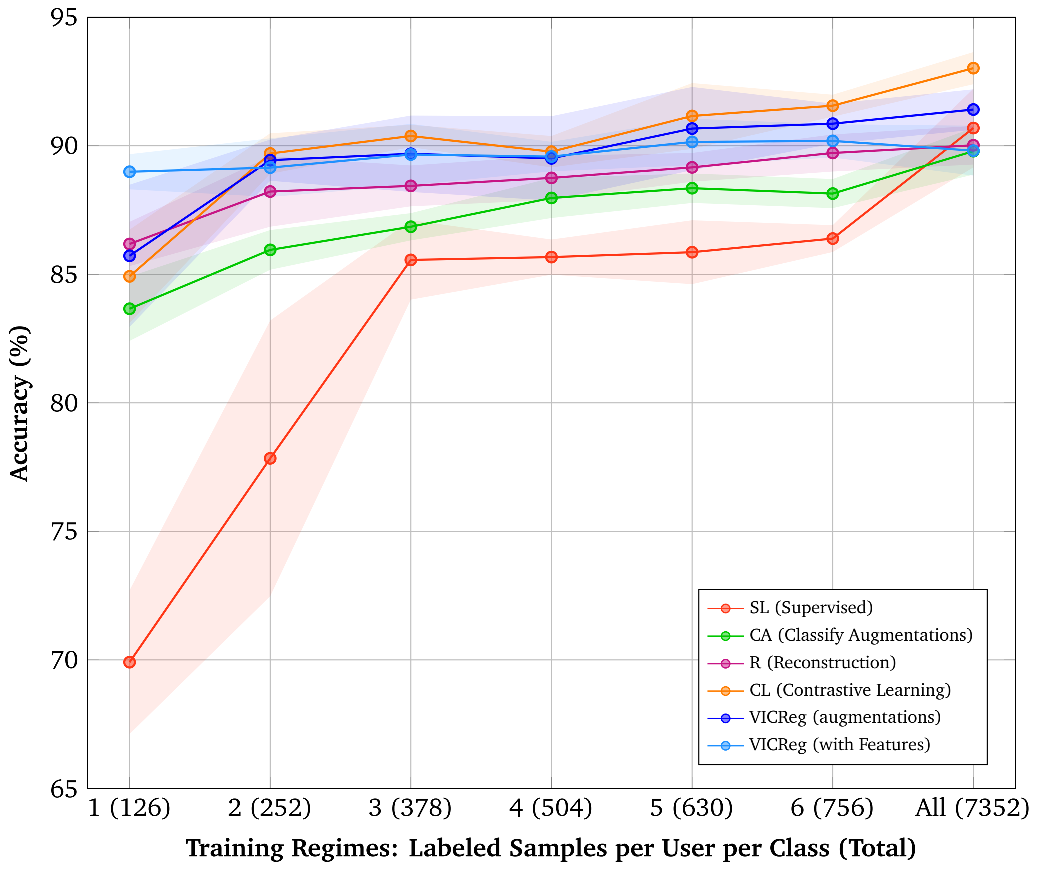


Figure 4.2. Results of the different pretext tasks on the UCI HAR dataset. Reported as mean and its 90% confidence interval over 5 runs. Numbers in parentheses indicate the number of labeled samples used for each experiment.

the VICReg (with features) method, which contrasts the raw sensor data with the handcrafted features, is the most consistent across all training sizes, suggesting that additional engineered features can guide the model to quickly learn useful representations. More importantly, it also suggests that contrasting very different representations may lead to improved performance. The same method also has the best performance in the most extreme low data regime, outperforming the other SSL techniques by a significant margin. However, as more labeled data becomes available, this method is slightly outperformed by the CL and standard VICReg approaches (achieving up to +2.5% accuracy). This suggests that while the guidance from the hand-crafted features is helpful when data is limited, it may also prevent the model from discovering the more refined representations that contrastive methods can extract from the raw data alone.

Classify Augmentations and Reconstruction (green and magenta color in Figure 4.2) deliver solid performance, but they do not reach the peak accuracies of the top-tier methods.

Consistency across the different folds is also an important aspect to consider. The supervised model exhibits a high variance across the low-data regime, as indicated by the wide red shaded area. This means its performance is highly sensitive to the specific small set of labeled samples chosen for training. In contrast, the top-performing SSL methods (VICReg, CL) have much tighter confidence intervals. This demonstrates that their pre-trained representations offer a more robust and stable starting point, making the downstream training process less susceptible to the randomness of small sample sets. This stability is a significant practical advantage, as it implies more reliable performance.

4.5 Closing Remarks

This experiment demonstrated that SSL methods can significantly improve the performance of HAR models, especially when labeled data is expensive or difficult to obtain. By leveraging unlabeled data, SSL methods can successfully train powerful encoders that create useful and robust feature representations. These representations give SSL methods a clear advantage over traditional supervised techniques that are forced to learn with much less data available.

The various SSL techniques achieve similar performances while being driven by different target objectives, which can be as different as pretext tasks or pure CL. This implies that potentially very different internal representations can provide valuable results. In the next chapter, we will see how performance can be further improved through a multi-task approach in which multiple of the discussed techniques are combined together.

Chapter 5

Multitask Self-Supervised Learning

In the previous chapter, we explored how SSL methods can enhance the performance of machine learning models, particularly in scenarios with limited labeled data. Each SSL task grasps a different aspect of the data, leading to diverse and complementary representations. This chapter highlights the benefits of combining multiple SSL tasks to create a more comprehensive and robust feature embedding and presents an extensive ablation study. The next chapters will build on this foundation, exploring how this framework can be further improved, trained while preserving privacy, and adapted to different domains.

5.1 Introduction

The results presented in chapter 4 demonstrate how SSL methods can effectively learn powerful representations from unlabeled data, and use them to successfully solve downstream tasks using only a small amount of labeled samples. The tested techniques are highly diverse, as they range from implicitly determining the encoded knowledge through pretext tasks to explicitly constraining the internal representations through contrastive losses. Moreover, we have also seen that multimodal input data (in particular raw data and expert features) can be successfully leveraged.

The main goal of this chapter is to investigate if and how combining multiple of these representations together during the learning phase may improve performance by capturing different aspects of the data in a single embedding. This will be done in the framework of MTL (see subsection 2.3.4), where multiple tasks are trained simultaneously to optimize a shared objective.

Building on the same architectures presented in chapter 4, we propose a multitask model (described in section 5.2) and extensively evaluate it (see section 5.3). In addition to presenting the results achieved, we perform an extensive ablation study to analyze the impact of each training choice and component on the final performance.

5.2 Architecture

To learn a rich and robust feature embedding, we propose an MTL architecture that combines multiple SSL tasks. This model is designed to simultaneously learn from several diverse objectives. The architecture, illustrated in Figure 5.1, is modular, centered around a shared encoder whose learned representations are fed into multiple task-specific heads. Each head is responsible for a different SSL task, allowing the model to learn complementary features from the data. This modular design enables the addition or removal of tasks without affecting the overall architecture, making it flexible and adaptable to various scenarios.

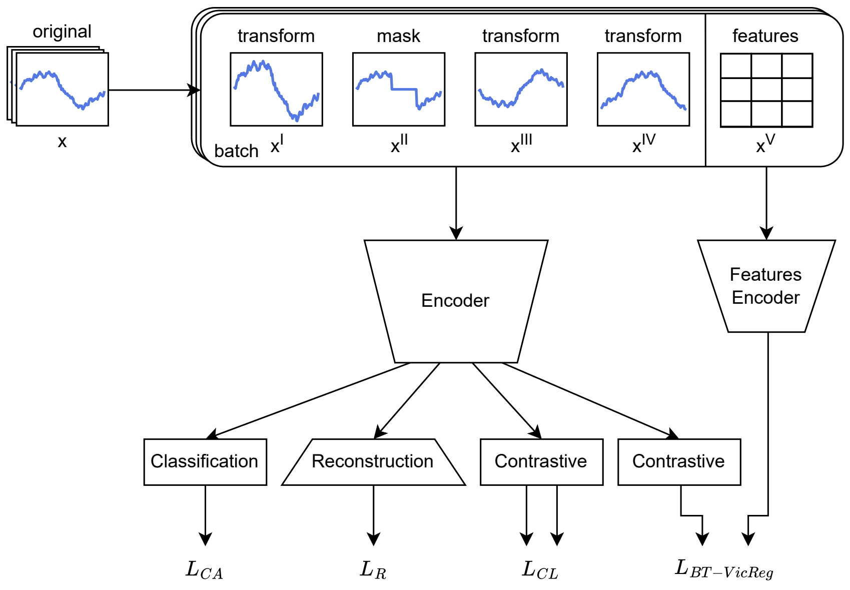


Figure 5.1. Multitask Self-Supervised Learning architecture

At training time, for each batch, the original samples are augmented in different ways. For each task, the corresponding view of the sample is passed through the shared encoder, which generates a feature embedding. This embedding is then processed by the task-specific head, which computes the loss for that particular task. For example, the reconstruction head requires the masked input (x^{II} in the Figure 5.1) to go through the encoder and the decoder, while the contrastive head needs two transformations of the original sample (x^{III} and x^{IV} in Figure 5.1) to be passed through the encoder, the contrastive projection head, and then compared.

To also include expert features in the training (x^{V} in Figure 5.1), we use a different encoder to handle this modality (as discussed in subsection 4.3.2). The output of this encoder is then contrasted with the original sample passed through the main encoder and the contrastive projection head.

Then, a separate loss term is computed for each of the model outputs. In subsection 2.3.4, we discussed several methods for combining the losses of different tasks: from the simple sum of the losses to dynamic weighting or gradient-based approaches. In this work, the actual performance of the model on the pretext tasks is not the main goal, but rather the quality of the learned representations. For this reason, we avoid complex and computationally expensive gradient-based methods and opt for dynamic weighting of the losses [Cipolla et al., 2018]. The method assigns a different learnable weight to each task, which is updated during training, and avoids collapse through a logarithmic regularization term (see Equation 2.9). This approach allows to balance the contribution of each task to the overall loss, while keeping the training process simple and efficient.

As for the single-task SSL case, the encoder is frozen after the pretraining phase, and the embeddings drive the training of the same downstream classifier introduced in subsection 4.2.2.

5.3 Results

One of the main objectives of this thesis is to introduce a simple yet effective multitask technique to improve the pretraining phase of SSL methods. To evaluate its effectiveness, we first compare the performance of the full MTL architecture presented in section 5.2 with the single-task SSL methods discussed in chapter 4. We then analyze the impact of each task on the final performance of the multitask model. This evaluation is simplified by the modularity of the approach, where adding or removing a task translates to simply adding or removing the corresponding task-specific branch and its loss term. As we will see, the experiments highlight the interest of overfitting the encoder during pretraining. We discuss this aspect in subsection 5.3.4. Finally, we perform an ablation study to evaluate the impact of different design choices, including modifying the encoder architecture (see subsection 5.3.5) and the embedding dimensionality (see subsection 5.3.6).

5.3.1 Comparing with Single-Task SSL

In chapter 4, we explored various SSL pretext tasks and their impact on the quality of the learned representations. In this section, we will compare the performance of single-task SSL methods with the complete multitask SSL method proposed.

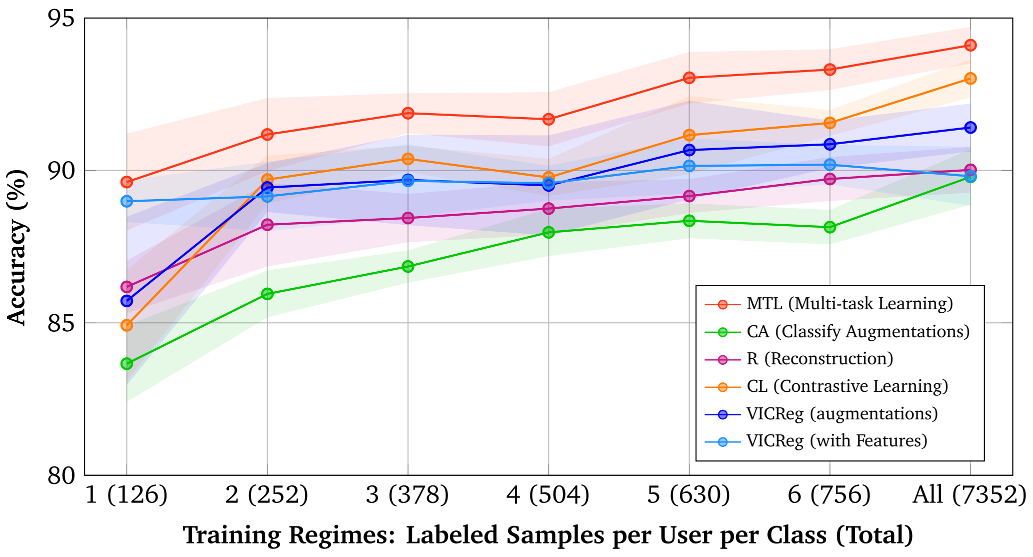


Figure 5.2. Comparison of single-task SSL methods with the multitask SSL method. The plot shows the accuracy of the model as a function of the number of labeled samples used for training, with different colors representing different SSL methods. We report the mean and its 90% confidence interval over 5 runs.

Figure 5.2 visually compares the performance accuracy of the MTL model with the single-task SSL methods. The MTL model consistently outperforms the single-task methods across all labeled data sizes. This comparison highlights that combining multiple SSL tasks not only leads to a model as good as the best single-task SSL method but also provides additional information to the model, allowing it to outperform the best single-task method in all scenarios. This demonstrates that combining multiple SSL tasks can allow the model to learn different aspects of the data, leading to a more comprehensive and robust feature embedding. The tasks are combined with dynamic weighting. Figure 5.3 illustrates the evolution of the learned weights σ_i during training by plotting the factor $f_i = \frac{1}{2(\sigma_i^2)}$ applied to each task loss $\mathscr{L}_i$. For better readability, the factors are reparameterized as percentages, so that the sum of all factors is always 100%. As we can see in the first few epochs, the factors quickly vary to capture the different scales of the losses, and after evolving more smoothly as the training progresses. The plot displays the average value over 5 folds, with the shaded area representing the 90% confidence interval for the mean, which is hardly visible since the weights are very stable across different runs.

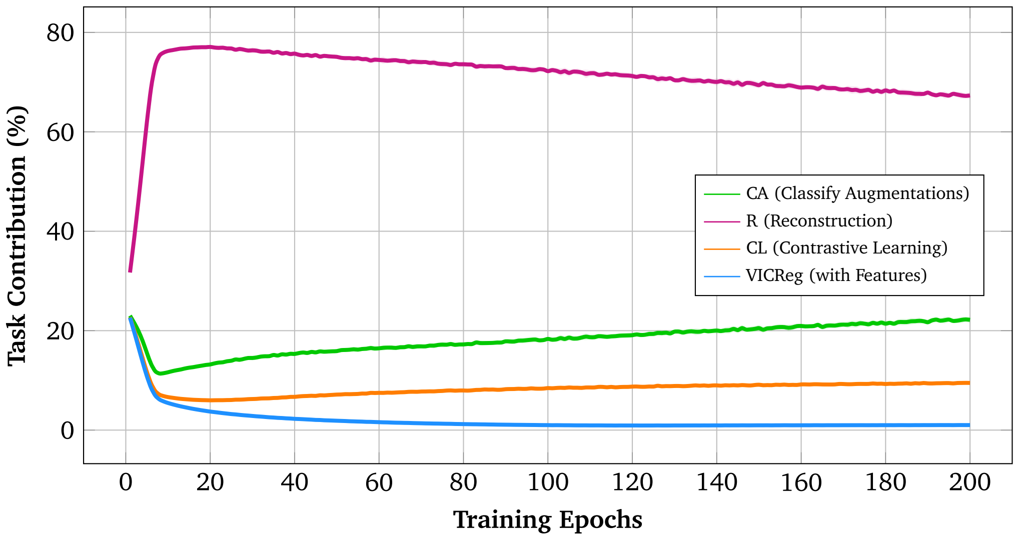


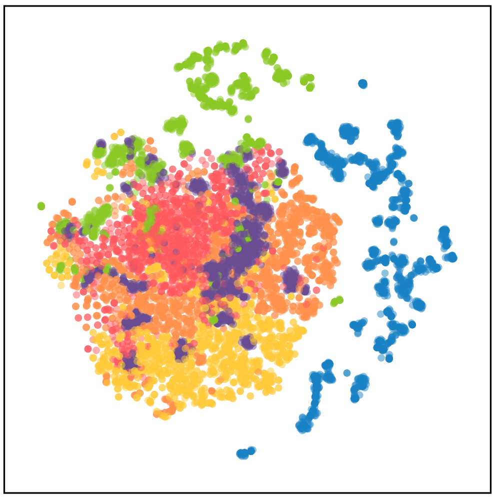
Figure 5.3. Task contribution in percentage during training. The weights are dynamically learned, allowing the model to adapt to the different scales and evolution of the task-specific losses. The plot shows the evolution of the multiplying factors for each task as the training progresses, with different colors representing different tasks.

The MTL model achieves an accuracy of 89.6% with only 126 labeled samples (1 labeled sample per class per user), while the best single-task SSL method (VICReg with features) only reaches 89.0%. The gap widens when more labeled samples are used, with the MTL model proving its superiority even when all labeled samples are available, reaching 94.1% accuracy compared to 93.0% for the best single-task method (CL). Moreover, the MTL is always the best in all the regimes, from few-shot learning (126 labeled samples) to the full dataset (7532 labeled samples).

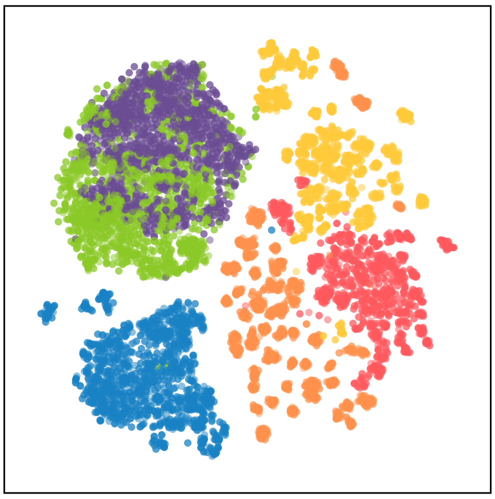
While this finding might look obvious, the situation is much more complex than what may superficially seem. First of all, the large performance increase of MTL relative to the individual techniques shows that each of the techniques extracts fairly different aspects of the data. Otherwise, with very close-by embeddings, we would observe just an averaging effect. Second, it is not granted that combining very diverse solutions leads to a better performance. In particular, as discussed in subsection 2.3.4, directions of improvements from different tasks may cancel out.

5.3.2 Visualizing the Data Representations

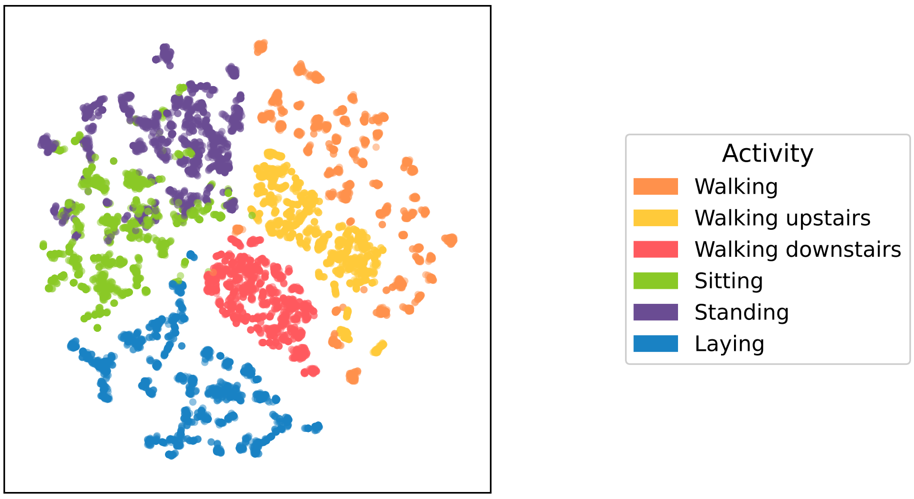
To illustrate the effectiveness of the learned representations, Figure 5.4 visually compares the raw data, expert features, and MTL embeddings using t-Distributed Stochastic Neighbor Embedding (t-SNE) [Maaten and Hinton, 2008].



(a) Raw Data (1,161)



(b) Expert Features (561)



(c) MTL Embeddings (128)

Figure 5.4. t-SNE visualization of the data representations. The first two subfigures show the raw data and expert features, while the last one shows the MTL embeddings. The colors represent different activities.

The t-SNE algorithm is a dimensionality reduction technique that helps visualize high-dimensional data by projecting it into a lower-dimensional space, preserving local structures. We projected the entire dataset into a 2D space, assigning a different color to each activity class.

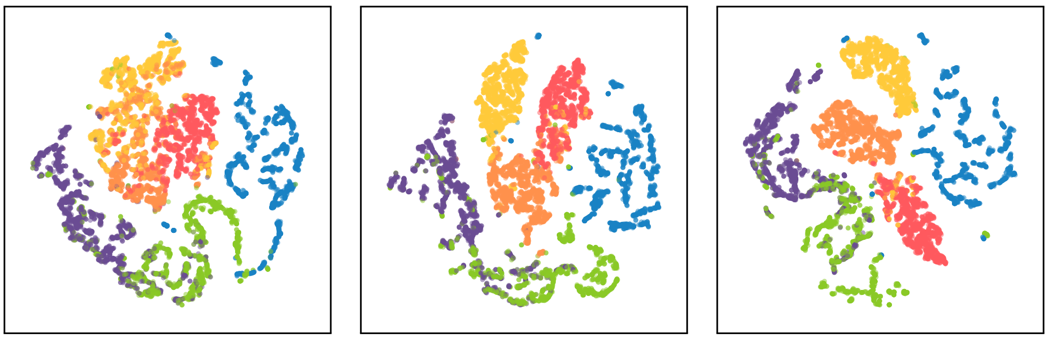
Figure 5.4a displays projections of the raw data. As can be noted, most of the data is cluttered together, and the only visible separation is between Laying and the other activities. Several of the Sitting instances are also slightly separated, but others are very close to other types of activities. This highlights the difficulty of distinguishing activities directly from the raw data.

With expert features (Figure 5.4b), the separation is much improved, and in most cases, we can identify clusters corresponding to each activity. This shows that the domain experts who handcrafted the features were able to select ones relevant to the classification task. However, we can also see that when clusters are well defined, there are situations where there is little or no gap between clusters, making classification ambiguous in boundary areas. Moreover, distinguishing between Sitting and Standing seems very difficult, as the instances of both classes are mixed together.

Figure 5.4c shows the MTL embeddings extracted directly from unlabeled data with self-supervised pretraining. As can be noted, the various classes are well clustered and well separated. It is still challenging to distinguish some instances of Sitting and Standing, but the overall separation is much clearer.

Additionally, it is important to note that the dimensionality of the different representations is very different: the raw data has 1,161 dimensions, the expert features have 561, and the MTL embeddings have only 128. For this reason, Figure 5.4c also looks sparsely populated, as the t-SNE algorithm piles together many instances with similar embeddings.

An important consideration is that, while the experts who defined the features in Figure 5.4b were well aware of the downstream problem, the MTL embeddings naturally arise from self-supervision and are thus not only more compact, but potentially more general.



(a) 1 (126) (b) 4 (504) (c) all (7352)

Figure 5.5. t-SNE visualization of the latent embeddings learned by the supervised model with different amounts of labeled data. The colors represent different activities using the same color map as Figure 5.4.

For reference, we also report the t-SNE visualization of the embeddings learned by the supervised model trained from scratch on different amounts of labeled data (Figure 5.5). As can be noted, the separation between classes increases with the data availability. When all the labeled data is used (Figure 5.5c), the clusters are well defined and separated, as achieved by the MTL embeddings (Figure 5.4c) that were learned without any knowledge of the downstream task. The gap between classes is more pronounced then in the MTL case, but we can also see some mixing in the different walking activities that are not present in the MTL embeddings. Despite the availability of labels, distinguishing between Sitting and Standing is still challenging. For smaller data regimes (Figure 5.5a and Figure 5.5b), the clusters are less defined, reflecting the lower accuracy achieved in these scenarios as reported in Table 4.1 and Figure 4.2.

5.3.3 Different Pretraining Tasks Combinations

After analyzing the performance of the multitask SSL method using the full set of pretraining tasks, we evaluate the impact of the individual ones. This is done by exploiting our modular design, rerunning the same test while activating only the desired task combination.

class rec CL VICReg	**Accuracy (%)**						
	1 (126)	**2 (252)**	**3 (378)**	**4 (504)**	**5 (630)**	**6 (756)**	**all (7352)**
✓✗✗✗	83.66 ± 1.70	85.95 ± 1.05	86.85 ± 0.72	87.97 ± 1.06	88.35 ± 0.78	88.14 ± 0.77	89.79 ± 1.24
✗✓✗✗	86.18 ± 1.17	88.22 ± 1.86	88.44 ± 1.08	88.75 ± 1.17	89.16 ± 0.76	89.72 ± 0.98	90.02 ± 1.02
✗✗✓✗	84.92 ± 2.47	89.70 ± 1.07	90.38 ± 0.59	89.77 ± 0.83	91.16 ± 1.74	91.56 ± 0.58	93.02 ± 0.85
✗✗✗✓	88.99 ± 0.93	89.15 ± 1.55	89.66 ± 1.62	89.58 ± 0.78	90.15 ± 1.23	90.19 ± 0.88	89.81 ± 1.32
✓✓✗✗	83.64 ± 1.90	87.85 ± 3.24	88.46 ± 1.73	88.27 ± 1.85	89.44 ± 1.89	88.94 ± 1.19	91.21 ± 2.02
✓✗✓✗	85.95 ± 2.87	89.93 ± 1.36	90.06 ± 1.36	90.36 ± 0.59	91.73 ± 0.79	91.58 ± 1.02	93.21 ± 0.99
✓✗✗✓	**90.11 ± 1.36**	91.03 ± 2.64	91.16 ± 1.84	90.46 ± 2.36	91.46 ± 1.85	91.17 ± 1.74	92.30 ± 1.29
✗✓✓✗	85.21 ± 3.26	89.74 ± 2.17	90.52 ± 1.77	90.20 ± 1.21	91.70 ± 2.08	92.00 ± 1.09	93.53 ± 1.02
✗✓✗✓	85.59 ± 2.17	87.82 ± 3.58	87.86 ± 3.05	88.86 ± 2.35	89.10 ± 2.25	89.39 ± 2.46	89.54 ± 1.88
✗✗✓✓	89.30 ± 2.80	91.23 ± 1.62	91.59 ± 1.34	91.30 ± 0.99	92.33 ± 1.25	92.50 ± 0.95	93.18 ± 0.60
✓✓✓✗	86.13 ± 2.59	89.72 ± 2.39	91.06 ± 1.35	90.95 ± 0.85	92.20 ± 1.70	92.12 ± 1.24	94.07 ± 0.97
✓✗✓✓	89.69 ± 0.79	**91.80 ± 1.43**	**91.94 ± 0.76**	91.77 ± 0.94	**93.00 ± 0.87**	92.94 ± 0.27	**94.33 ± 0.63**
✓✓✗✓	90.00 ± 1.29	91.53 ± 2.13	90.89 ± 2.02	91.19 ± 1.69	91.61 ± 1.78	91.12 ± 1.18	91.88 ± 0.96
✗✓✓✓	89.11 ± 2.06	91.60 ± 1.57	91.54 ± 0.86	91.69 ± 0.76	92.29 ± 1.19	92.62 ± 0.48	93.33 ± 0.91
✓✓✓✓	89.64 ± 2.02	91.40 ± 1.57	91.76 ± 0.83	**91.97 ± 1.20**	92.69 ± 1.80	**93.28 ± 0.93**	94.12 ± 1.14

Table 5.1. Ablation study on multitask SSL task combinations. Each row shows a different combination of SSL tasks: classification augmentation (class), reconstruction (rec), contrastive learning (CL), and VICReg with features, indicated by checkmarks (✓) and cross marks (✗). The table presents downstream classification accuracy (%) across different numbers of labeled training samples. Bold values indicate the best performance for each data regime. The optimal three-task combination (class + CL + VICReg) consistently outperforms other combinations, while the inclusion of reconstruction often degrades performance when combined with contrastive methods. Results are averaged over 5 folds with encoders trained for 200 epochs.

Different from section 4.4, where each task is trained separately, we focus here on understanding whether the nature of the tasks and their combination can lead to different outcomes. For example, two tasks might perform similarly when trained separately, but they might learn different aspects of the data, and their combination might lead to a better representation. On the other hand, two other tasks might learn similar features, and their combination might not provide any additional information to the model. Finally, it could also be the case that two different representations could be conflicting and cancel each other out, leading to a worse performance.

Table 5.1 presents the results of this ablation study, where we systematically explore the impact of different task combinations on performance. We tested all the possible combinations of two, three, and four tasks, the last one being the full multitask model. The analysis has revealed several key insights into the interactions and importance of the different tasks.

The first observation is that contrasting raw data with expert features (VICReg with features) is crucial for achieving the best performance in scarce data regimes. When this task is excluded, in the most extreme scenario (few-shot learning with 126 labeled samples), the performance drops significantly, losing from 3.7% to 6.4% depending on the task combination. This result further confirms the importance of multimodal CL and exploiting expert features in the pretraining phase. However, as more labeled data is available for downstream training, contrasting with expert features becomes less critical or even unnecessary, as seen in the full dataset regime (7352 labeled samples), where the performance is comparable with and without this task. When the number of labeled samples is high, standard CL between data augmentations becomes the most important task, consistently leading the models to the best performance.

Moreover, removing the reconstruction task (rec) from the combination does not seem to have a significant impact on the performance, as the three other tasks combined achieve performance comparable to the full four-task combination. However, small improvements are observed when reconstruction and standard CL are combined, suggesting that these two tasks can complement each other in certain scenarios. This could suggest that the other two tasks (classification augmentation and VICReg with features) capture similar information to what reconstruction learns, making reconstruction redundant in this context.

The four-task combination (bottom row) or the three-task combination without reconstruction (fourth to last row) always achieves the best accuracy or is very close to the best performing task combination. This finding emphasizes the importance of including diverse tasks in the pretraining phase to capture a wide range of features and improve generalization.

5.3.4 Consequences of Overfitting during Pretraining

While in standard MTL, the performance and generalization ability of the model across different tasks are the ultimate goal, we use it in an SSL setting only to improve the quality of the learned representations. Therefore, only the performance on the downstream task is relevant, and we can afford to overfit the pretraining tasks on the training data. Figure 5.6 shows the downstream task accuracy as a function of the number of epochs used to train the multitask encoder.

It should be noted that the best performance on the downstream task is not always achieved by the best-performing pretraining model. In this case, the best pretraining performance is achieved after around 108.8 epochs (average of the 5 folds). However, even though the model starts overfitting the pretraining tasks, we continue to see a slight improvement in the downstream task accuracy. For example, when training the encoder for 200 epochs, the downstream task accuracy is 89.6% when only the smallest amount of labeled data is used, compared to 88.9% taking the best pretraining epoch model. Similarly, even when the full labeled data is used, the accuracy is 94.1% compared to 93.4%. These might look like small improvements, but they can make a significant difference in real-world applications where every percentage point of accuracy counts.

Another interesting observation is that, even when the number of training epochs is high, a loss in downstream performance is not observed. After around 200 epochs, the improvement curve flattens out. A possible explanation is that the embeddings are latent representations and far from the last task-specific layers. Therefore, the model can still overfit the last layers

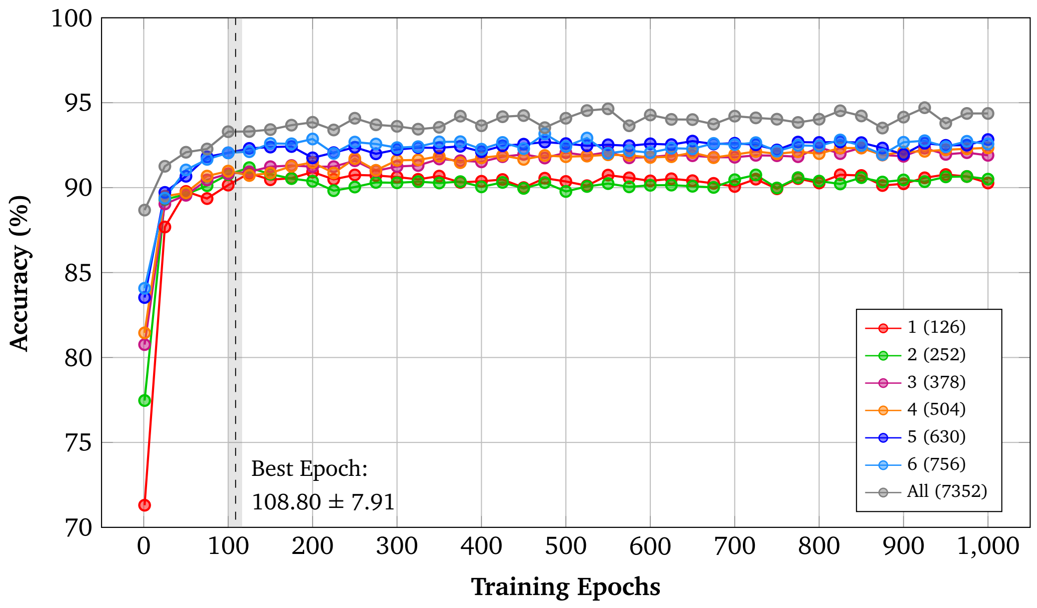


Figure 5.6. Overfitting the pretraining tasks on the UCI HAR dataset. The plot shows the accuracy of the model as a function of the number of training epochs of the multitask encoder, with different colors representing different amounts of labeled data used for training.

without affecting the quality of the deep learned representations.

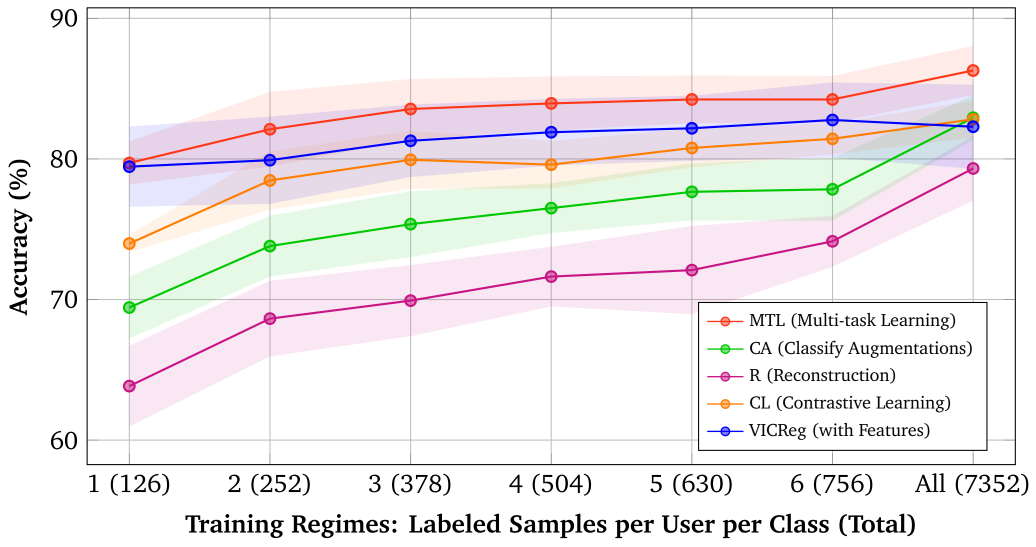
This behavior is not unique to multitask SSL but also occurs in single-task SSL settings. For instance, in section 4.4, we presented the results after 200 epochs of training the encoder. However, when taking the encoder at the epoch with the best pretraining task performance, the downstream accuracy drops. For example, considering the standard VICReg trained with features we observe a consistent decrease of 3-4 percentage points in accuracy across all the labeled data sizes. Similar behavior occurs for the other SSL methods.

For this reason, all the results presented in this thesis are obtained using a fixed number of epochs (200) for training the encoder and saving the last model. Additional research could be done to further investigate this behavior.

5.3.5 Ablation: Different Encoders

To evaluate the robustness of the results obtained on the specific configuration of the encoder, we perform an ablation study replacing the FCN encoder with other architectures commonly used in time series analysis: LSTM and Transformer. The architectures of these encoders are inspired by Qian et al. [2022], which uses them in a CL scenario to experiment with different CL frameworks, backbone architectures, and data augmentations.

The first encoder is an implementation of a Deep Convolutional LSTM (DeepConvLSTM) with 4 convolutional layers with ReLU activation functions, followed by a 0.5 dropout layer and 2 LSTM layers. The output of this encoder is a 128-dimensional vector. The encoder contains roughly 0.55 million parameters, which is significantly more than the FCN encoder used in the previous sections (see subsection 4.2.1), which has only 85K parameters.



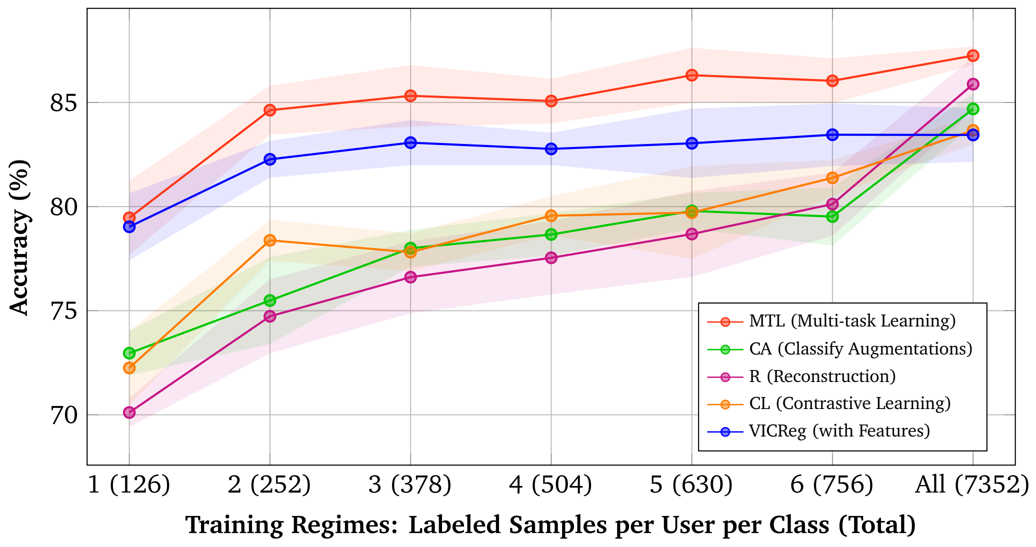


Figure 5.7. Comparison of multitask SSL with single-task SSL methods using a Deep-ConvLSTM encoder (top) or a Transformer-based encoder (bottom). The plots show the accuracy of the model as a function of the number of labeled samples used for downstream training, with different colors representing different SSL methods. We report the mean and its 90% confidence interval over 5 runs. The encoders are trained for 200 epochs.

The second encoder is based on Transformers. It first uses a linear layer to convert the input time series into embeddings of 128. A special 128-dimensional token is added to the embedded sequence as a representation vector. Positional encoding is added to the embedding to give the model information about the order of the tokens. The core of the network consists of a stack of 4 identical blocks, each containing a multi-head self-attention layer and a fully connected feed-forward layer. Residual connections are used around each layer to facilitate stable training. The output of this encoder is also a 128-dimensional vector. The Transformer encoder has around 0.33 million parameters.

Figure 5.7 shows the results of the ablation study using the DeepConvLSTM and Transformer encoders, respectively. The relative performance of the multitask SSL method compared to single-task SSL methods is consistent with the results obtained using the FCN encoder: when combining multiple SSL tasks, the MTL model consistently outperforms the single-task SSL.

However, the absolute performance of the models trained with these encoders is lower than that obtained with the FCN encoder. This might be due to the size of the encoders, which are way larger than the FCN one (0.55M and 0.33M parameters for the DeepConvLSTM and Transformer, respectively, compared to 85K for the FCN encoder). Additionally, architectures like Transformer need a larger amount of data to generalize well, and the UCI HAR dataset is relatively small.

The results obtained confirm at the same time the robustness of the approach when using different encoders and that the selected FCN encoder is a good choice.

5.3.6 Ablation: Testing the Embedding Size

A crucial aspect of SSL is the size of the learned embedding. A larger embedding might capture more information about the data, but it can also lead to overfitting, especially when the amount of labeled data is limited. Smaller embeddings, on the other hand, compress the information, and they force the model to focus on the most relevant features.

To investigate the impact of the embedding size on the downstream performance, we conducted an ablation study where we trained the FCN encoder with different output sizes. To keep the number of parameters constant, the output of the last convolutional block (2304) is reshaped to the desired embedding size using pooling. Studying the impact of the embedding size is not easy, as it heavily impacts the downstream classifier. As explained in subsection 4.2.2, the downstream classifier consists of a fully connected layer that maps the embedding to a hidden size of 512, followed by ReLU activation, 0.3 dropout, and a final fully connected layer that maps the hidden size to the number of classes (6 for the UCI HAR dataset). For a complete comparison, one should study the best architecture for each embedding size, but this is out of the scope of this work and would require a lot of trials and hyperparameter tuning. Therefore, we keep the same architecture for the downstream classifier, and we only change the first layer to match the embedding size.

Size	1 (126)	2 (252)	3 (378)	4 (504)	5 (630)	6 (756)	all (7352)
128	**89.64 ± 2.02**	91.40 ± 1.57	**91.76 ± 0.83**	**91.97 ± 1.20**	**92.69 ± 1.80**	**93.28 ± 0.93**	94.12 ± 1.14
256	88.88 ± 1.21	**91.68 ± 1.44**	91.52 ± 1.48	91.44 ± 0.76	92.43 ± 0.64	92.45 ± 0.74	94.29 ± 0.63
384	87.51 ± 1.90	90.97 ± 1.86	91.31 ± 1.14	91.55 ± 0.62	91.76 ± 0.59	92.28 ± 0.62	94.40 ± 1.39
512	85.81 ± 1.70	90.40 ± 1.98	90.17 ± 2.29	90.70 ± 1.48	91.29 ± 1.48	91.49 ± 0.90	93.86 ± 1.44
640	85.78 ± 0.61	89.87 ± 1.19	91.27 ± 1.18	91.16 ± 0.51	91.58 ± 0.64	92.18 ± 0.60	**94.50 ± 0.65**
768	84.55 ± 0.99	89.71 ± 1.85	90.45 ± 0.89	90.84 ± 0.64	90.81 ± 0.49	90.62 ± 1.15	93.70 ± 0.94
1024	83.94 ± 2.01	89.17 ± 1.09	89.86 ± 1.50	89.94 ± 0.41	90.95 ± 0.81	90.52 ± 0.92	93.35 ± 1.32
1280	84.69 ± 0.99	89.64 ± 1.82	90.30 ± 1.46	90.68 ± 0.64	91.02 ± 0.90	91.40 ± 1.22	94.47 ± 0.99

Table 5.2. Multitask SSL with different embedding sizes. The table shows the accuracy of the model as a function of the number of labeled samples used for downstream training, with different embedding sizes. The FCN encoder is trained for 200 epochs. The values are the mean and standard deviation of the accuracy over 5 runs. The best accuracy for each number of labeled samples is highlighted in bold.

Table 5.2 shows the results of this ablation study, while Figure 5.8 visually compares the results for some of the embedding sizes. The results show that increasing the embedding size does not provide additional knowledge to the model. When the full labeled dataset is used, all the embedding sizes achieve similar performance, with the 640-dimensional embedding slightly registering the best performance (even though the difference is not statistically significant). However, as the number of labeled samples decreases, the lower embedding sizes (128 and 256) start to outperform the larger ones. This suggests that smaller embeddings are more robust to overfitting when the amount of labeled data is limited, while larger embeddings might capture nuances in the data that are very difficult to learn without a sufficient amount of labeled samples.

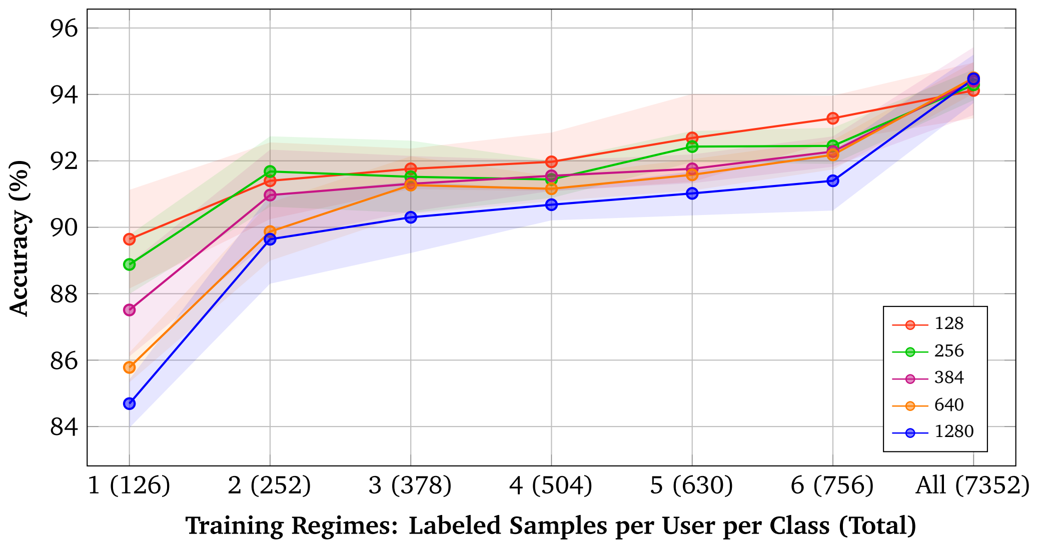


Figure 5.8. Multitask SSL with different embedding sizes. The plot shows the accuracy of the model as a function of the number of labeled samples used for downstream training, with different colors representing different embedding sizes. We report the mean and its 90% confidence interval over 5 runs. The FCN encoder is trained for 200 epochs.

5.4 Closing Remarks

Our extensive experimentation demonstrated that combining multiple SSL tasks in a multitask setting can significantly improve the performance relative to the single-task solutions. In our design, the combination is achieved through a modular architecture where adding or removing tasks is straightforward. By evaluating different task combinations, we found that, especially in the low-labeled data regime, it is important to include in the combination tasks with different characteristics and incorporate, if possible, expert knowledge in engineered features. Our experiments show that in scarce data regimes, the VICReg with features task is crucial, since the performance drops significantly in few-shot learning settings when it is removed from the combination. As labeled data increases, its importance diminishes, and standard CL between augmentations becomes the dominant contributor to performance. When the reconstruction task is removed from the 4-task combination, the performance drop is negligible, suggesting that it captures similar information to the other tasks. Overall, the best or near-best results consistently come from either the full four-task combination or the three-task variant without reconstruction, highlighting the value of diverse pretraining tasks for capturing varied features and enhancing generalization. Another interesting finding is that it is not obvious when to stop pretraining the encoder. This is because the target of pretraining is not to achieve the best performance on pretraining tasks, but to generate embeddings useful for downstream classification that uses labeled data unknown at the moment of pretraining. This might suggest that it could be useful to exploit labeled samples to define stopping criteria for the pretraining phase. Our ablation studies have also demonstrated the robustness of the approach to changes in encoder architecture and embedding size. In particular, we found that smaller embeddings are more robust to overfitting when the amount of labeled data is limited. In the next chapter, we will see how to further improve performance by improving the training of the downstream classifier.

Chapter 6

Improving Accuracy

To exploit the full potential of unlabeled data, semi-supervised methods can be combined with our multitask SSL approach. We previously trained encoders on different pretraining tasks and then used them to train a classifier on small datasets of labeled data. In this chapter, we explore how to improve the accuracy of the classifier even further by leveraging the unlabeled data, also when training the downstream classifier.

6.1 Introduction

The training paradigm used until now is a two-stage process: first, pretrain an encoder on large amounts of unlabeled data using various SSL techniques, and secondly, freeze the encoder and train a classifier on top of the embeddings produced by the encoder using a small dataset of labeled examples. While this approach successfully builds a strong knowledge base from the unlabeled data, it discards the vast amount of information contained in the unlabeled data during the final supervised classification stage. This is particularly problematic when the number of labeled examples is very low.

This chapter investigates how to combine multitask SSL with the strengths of semi-supervised learning. The main idea is to leverage the large set of unlabeled data not only during the pretraining phase but also during the final classifier training phase itself. Given the relatively high performance of the multitask SSL approach, among the many possible solutions (see subsection 2.3.2), we have selected pseudo-labeling since it can exploit the strong generalization capabilities of the encoder that drives the classification task.

6.2 Pseudo-Labeling

Pseudo-labeling is a simple yet powerful technique for leveraging unlabeled data in a semi-supervised learning framework. The main idea is to use the model's predictions on the unlabeled data as "pseudo-labels" and incorporate them into the training process. This assumes that the model's high-confidence predictions on the unlabeled data are likely to be correct and can be used as useful training signals. The error induced by wrong predictions will be surpassed by the benefits of incorporating the more abundant correct ones.

In our implementation, we first train a classifier on the labeled data, then use this classifier to predict labels for the unlabeled data. These pseudo-labels of samples where predictions are confident (above a class probability of 0.35) are then used to augment the training set, and the classifier is finetuned on this increased dataset. The samples with very low confidence predictions are discarded, as they are likely to be noisy and not representative of the true distribution. This process is repeated iteratively for a maximum of 50 epochs. The best epoch model is selected based on the validation accuracy.

6.3 Results

For this experiment, we test the effectiveness of pseudo-labeling on classifiers using two different embedding sizes: 128 and 384. The results are presented in Table 6.1. The table shows the accuracy of the classifier trained on the labeled data only, and the accuracy after applying pseudo-labeling finetuning. As can be seen, pseudo-labeling is particularly effective when the number of labeled examples is small, leading to +3.02% and +0.85% improvement in accuracy for the 128 and 384 embedding sizes, respectively. The improvement is less pronounced when the number of labeled examples is larger, but still significant for the bigger embedding size. It should be noted that finetuning using pseudo-labels should only improve performance, since we select the best model every epoch. However, the model is selected using validation data, which is very scarce, and an improvement in the validation accuracy is not always matched with an improvement in the accuracy on the way larger test set. This explains why there are some negative improvements in the table.

Size	Method	1 (126)	2 (252)	3 (378)	4 (504)	5 (630)	6 (756)
128	Base model	89.62 ± 2.16	92.26 ± 2.15	91.41 ± 0.55	91.82 ± 1.08	92.93 ± 0.80	93.15 ± 0.66
	Pseudo-labels	90.47 ± 1.73	92.36 ± 1.23	91.24 ± 0.80	92.47 ± 1.44	92.62 ± 1.48	92.60 ± 0.90
	Improvement	+0.85	+0.10	-0.17	+0.65	-0.31	-0.55
384	Base model	86.72 ± 2.40	91.31 ± 1.59	91.80 ± 1.00	91.65 ± 0.53	92.20 ± 1.13	92.72 ± 0.46
	Pseudo-labels	89.74 ± 2.62	92.47 ± 1.94	92.32 ± 1.31	92.01 ± 0.59	93.06 ± 1.15	93.16 ± 1.23
	Improvement	+3.02	+1.16	+0.52	+0.36	+0.86	+0.44

Table 6.1. Effect of pseudo-labeling on the accuracy of the model with different embedding sizes. The results are reported as mean ± standard deviation over 5 runs. The numbers in parentheses indicate the number of labeled samples used for each task.

6.4 Closing Remarks

This chapter presents a simple refinement of the multitask SSL approach. Our preliminary results show that incorporating confident pseudo-labels is beneficial at low labeled data regimes. An interesting finding is that the more pronounced improvements are for larger embedding sizes. This could indicate that this technique could be employed for more complex data than what is available in the UCI HAR dataset.

Chapter 7

Multitask Pretraining in a Federated Learning Setting

Previous chapters introduced a multitask self-supervised learning approach to learn powerful representations from unlabeled data, and experimentally demonstrated its effectiveness to improve the performance of downstream tasks. In this chapter, we extend this analysis to federated learning, which is often necessary to protect the privacy of users' data.

7.1 Introduction

This thesis introduced methods to fully exploit the potential of unlabeled data and train models that generalize well even with very few labeled examples. This is very useful in many real-world scenarios, where labeled data availability is difficult to collect. All the experiments conducted so far demonstrated the effectiveness of multitask pretraining as a way to leverage large amounts of unlabeled data to learn robust representations to be used in downstream tasks. However, all the experiments were conducted in a centralized setting, where all the data is available in a single location. This is not always possible, especially when dealing with sensitive data.

HAR is a classical example, where the difficulty of labeling data and privacy concerns meet. For this reason, many of the available datasets are small and tied to individual devices. Collecting large amounts of unlabeled data could be achieved by crowdsourcing, without asking the user to do anything except to carry the sensors. For instance, in the context of a large medical population study, the UK Biobank [Sudlow et al., 2015; Doherty et al., 2017] collected a wide variety of personal information from over 500,000 participants. A subset of 100,000 participants was involved in a study that collected accelerometer data.

It has been demonstrated that using such huge amounts of unlabeled data can significantly improve the quality of the generalizability and accuracy of pretrained models for HAR [Yuan et al., 2024]. However, this was done by collecting all the user data in a centralized location and performing training on the full data. Even though the participants voluntarily joined the experiment, and agreed to share their anonymized data, they gave broad consent to use it for any health-related research [Sudlow et al., 2015], so privacy concerns may still arise.

In this chapter, we evaluate the feasibility of pretraining the multitask SSL model introduced in the previous chapters in a FL setting, where the data is not collected in a centralized location, but rather remains on the users' devices.

7.2 Federated Learning Setting

In the experiments presented in this chapter, we use the standard Federated Averaging (FedAvg) [McMahan et al., 2017] algorithm, which is the most widely used method for FL.

The training process is as follows: the server initializes a global model and sends it to all clients. Each client then trains the model on its local data for a few epochs (two in our experiments), and sends the updated model back to the server. The server then averages the models received from the clients to obtain a new global model, which is sent back to the clients for the next round of training. This process continues until convergence or until a predefined number of rounds is reached.

The implementation used in this thesis is the one provided by the Flower framework [Beutel et al., 2022], which is a flexible FL framework. Similar to the centralized setting, we use Adam as the optimizer, and the same hyperparameters as in the previous chapters.

Since the training and validation split used for centralized training was based on user IDs (see section 3.3), we use the same split for the FL setting and associate a different client to each user. This readily simulates a realistic situation, in which each participant never shares their data with other clients or the server, ideally performing training on the wearable device itself.

7.3 Results

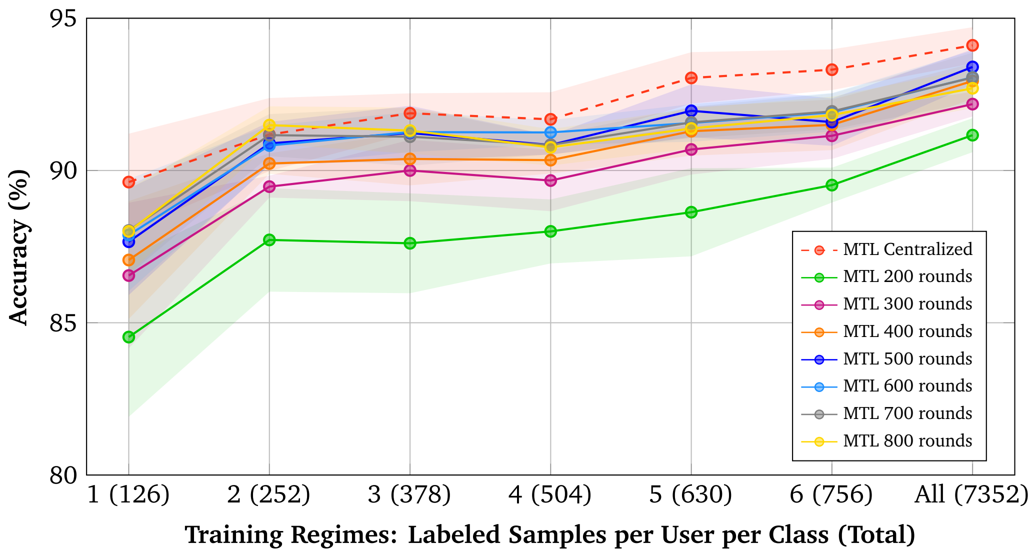


Figure 7.1. Downstream accuracy of MTL SSL in a federated setting. The x-axis represents the number of training labels, while the y-axis shows the accuracy achieved. We report the mean and its 90% confidence interval over 5 runs. All models are trained with 2 local epochs and the indicated number of rounds. The dashed line is the MTL SSL model in a centralized setting, trained with 200 epochs.

Figure 7.1 provides a direct comparison between federated and centralized training approaches. It compares the performance of the multitask SSL federated models after different pretraining rounds with the performance of the multitask SSL model trained in a centralized setting with 200 pretraining epochs. Table 7.1 summarizes the accuracy results across different communication rounds and labeled data regimes. For each number of rounds, ranging from 100 to 800, we report the results for different labeled data regimes, from one to six labeled examples per activity per user. The reported results are the average accuracy and standard deviation across 5 folds. With sufficient communication rounds (500+), the federated approach nearly matches the performance of centralized training (red dashed line in Figure 7.1) across most labeled data regimes. The largest performance gap between federated and centralized approaches, only around 1.6%, occurs in the lowest labeled data regime (126 samples).

Round	**1 (126)**	**2 (252)**	**3 (378)**	**4 (504)**	**5 (630)**	**6 (756)**	**All (7352)**
100	77.90 ± 4.22	83.02 ± 3.75	84.23 ± 2.44	84.98 ± 2.62	85.65 ± 2.66	85.73 ± 3.32	88.56 ± 0.96
200	84.53 ± 3.56	87.72 ± 2.31	87.61 ± 2.23	88.00 ± 1.43	88.63 ± 1.97	89.52 ± 0.78	91.16 ± 0.76
300	86.55 ± 3.25	89.47 ± 0.49	90.00 ± 1.37	89.67 ± 1.37	90.69 ± 1.12	91.14 ± 1.03	92.18 ± 0.55
400	87.06 ± 2.63	90.23 ± 0.46	90.38 ± 1.18	90.34 ± 0.60	91.29 ± 1.10	91.50 ± 1.13	92.94 ± 0.79
500	87.66 ± 2.38	90.89 ± 0.96	91.23 ± 1.22	90.84 ± 0.43	**91.96 ± 1.18**	91.59 ± 1.06	**93.40 ± 0.79**
600	87.88 ± 2.50	90.82 ± 0.90	91.26 ± 0.90	**91.25 ± 0.55**	91.56 ± 0.84	91.91 ± 0.86	93.06 ± 1.19
700	**88.03 ± 1.98**	91.16 ± 0.94	91.11 ± 1.30	90.84 ± 0.42	91.58 ± 0.70	**91.94 ± 0.80**	93.05 ± 1.11
800	88.00 ± 1.89	**91.49 ± 0.84**	**91.31 ± 1.03**	90.76 ± 0.81	91.38 ± 0.90	91.82 ±0.82	92.70 ± 0.89

Table 7.1. Downstream accuracy results for different numbers of training labels and different pretraining rounds in a federated learning setting. The numbers in parentheses indicate the total number of training examples used for each method. The best performing method for each number of training labels is highlighted in bold.

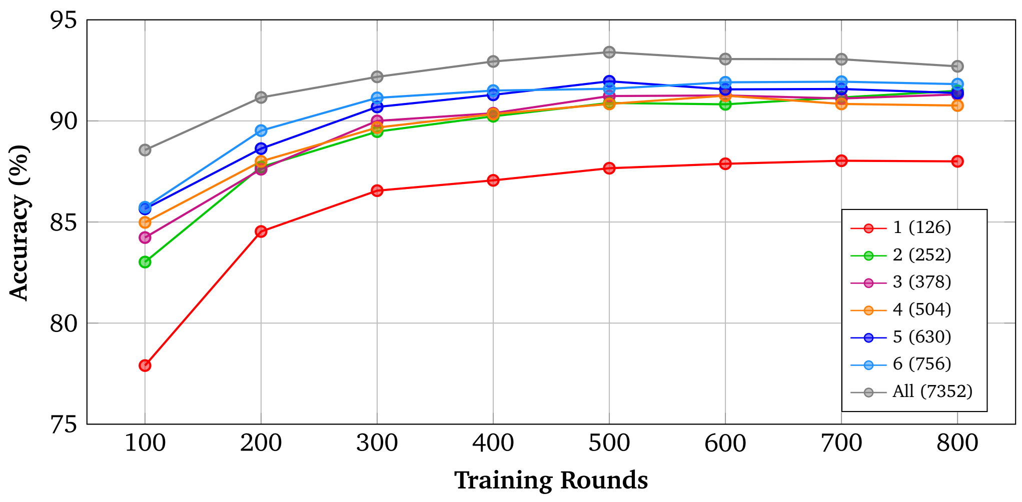


Figure 7.2. Downstream accuracy of MTL SSL in a federated setting with different numbers of training labels and pretraining rounds. The x-axis represents the number of training rounds, while the y-axis shows the accuracy achieved. The different lines represent different numbers of downstream labeled examples used for training.

This is an important finding, as our setup performs federated training under particularly challenging conditions, given the limited data available in the UCI HAR dataset. First of all, each client has a very small amount of data, down to only around 300 samples per client. Based on this little data, each client has to extract multitask information through the optimization of a complex loss function. In addition, each client is person-specific, meaning that, even if all clients participated in all activities, the data gathered by each client may significantly differ due to different movement patterns, placement of the sensors, and other factors. This could increase the complexity of the aggregation performed by the server, which could receive conflicting updates. These conflicts might also not be only task-specific, where some particular task (e.g., contrastive) might be easier for a client and difficult for another, leading to problems in the dynamic weighting of the tasks. These results suggest that, although some of the previously mentioned issues may still exist, they do not have a major impact on overall performance. Still, it would be interesting to explore whether techniques developed in FL for skewed data distributions (see subsection 2.4.2) could be applied to further improve performance and fully close the gap with centralized training.

The increased complexity of the FL setting is also apparent in the need for more optimization steps relative to the centralized scenario to reach the optimum. Figure 7.2 provides a visual representation of downstream accuracy achieved as a function of the number of pretraining federated rounds and the number of downstream labeled examples. Downstream performance improves significantly up to 500 rounds (of two local epochs each), and then plateaus. This behavior is similar to what was observed in the centralized setting (see subsection 5.3.4), which, however, required 200 epochs to plateau.

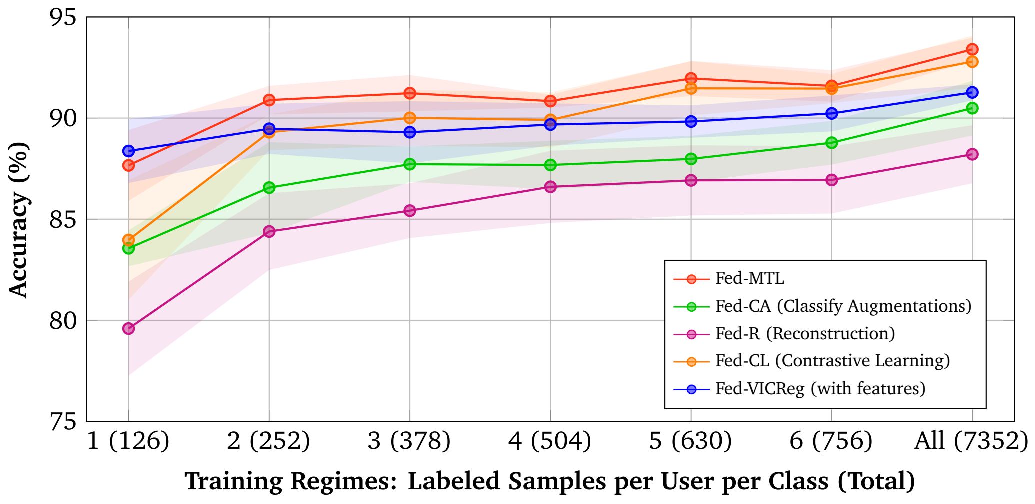


Figure 7.3. Downstream accuracy of federated single-task and multitask SSL models. The x-axis represents the number of training labels, while the y-axis shows the accuracy achieved. We report the mean and its 90% confidence interval over 5 runs. The models are trained with 2 local epochs and 500 pretraining rounds.

Figure 7.3 compares the downstream accuracy achieved by the federated multitask SSL model with that of federated single-task SSL models. Despite the added complexity of multi-tasking, the MTL model surpasses the performance of the simpler models in all labeled data regimes. The performance gap is lower than in the centralized setting due to the mentioned difficulty of optimizing and aggregating local models with personalized data.

7.4 Closing Remarks

The overall results suggest that the multitask SSL model can effectively leverage unlabeled data in a federated setting, achieving performance comparable to centralized training with sufficient communication rounds. This demonstrates the practical applicability of multitask SSL in real-world scenarios where protecting user privacy is crucial.

The experiments have concentrated on the pretraining phase for the most challenging multitask SSL model. It could be readily combined with standard FL techniques for the downstream phase to obtain a fully federated approach. The techniques that should be employed for downstream federated training are the classic ones used in FL, as the downstream task is a standard supervised classification problem. The training with few labels will be facilitated since it will not train the entire network, but only a small classifier. Moreover, and most importantly, the encoder will already likely separate the classes well and filter out client-specific noise. Moreover, the embedding created during pretraining could be used to train personalized models without any need for collaboration.

A recent work [Yuan et al., 2024] has clearly shown that the quality of the embeddings created during SSL pretraining significantly improves with the size and diversity of the data. They compared the downstream performance of models pretrained on data sizes ranging from 36K to 6B samples and from 55 to around 100K users. For comparison, the UCI HAR dataset used in this thesis contains only around 10K samples from 30 users. The demonstration that FL can be applied in this context indicates very promising paths for creating massive amounts of training data without any privacy concerns. Using a federated approach individual users can be enrolled into the pretraining phase without them releasing any personal data. This can solve one of the major problems of large centralized settings. Yuan et al. [2024] also mentions that even the huge 6B samples dataset they used is limited, as it collects only data from a specific region (UK) and specific demographics (adults). They suggest strong interest in training on worldwide unlabeled data, covering larger demographics. FL could offer a viable solution to this problem, either in cross-silo or cross-device settings.

Chapter 8

Generalization to Other Domains

The methods proposed in this thesis are designed to be general and applicable to various domains beyond HAR. To illustrate this, we present a case study on a different domain, specifically image recognition, one of the most prominent fields in computer vision.

8.1 Introduction

While the primary focus of this thesis has been on HAR, the methods proposed are designed to be general and applicable to different tasks and domains. To study the application of the proposed methods in a different setting, we selected image recognition as a case study, since it is a fundamental computer vision problem that can benefit from solutions that exploit large masses of unlabeled data. Due to the widespread use of digital photography and the common practice of sharing images online, photos of almost any object can be rapidly acquired or downloaded from the Internet. Manually labeling such data can obviously be done only at a small scale, and thus SSL can be a powerful tool to leverage the abundance of unlabeled data to boost performance [Schmarje et al., 2021].

In the following, we will first describe the chosen dataset among the many available for image recognition, then we will introduce the architecture and the pretraining tasks that we chose to employ in the experiments. Finally, we will discuss the obtained results, comparing the performance of the proposed multitask SSL approach with single-task SSL methods and fully supervised baselines.

8.2 Data

The field of image recognition has a long history, and many datasets have been proposed to benchmark the performance of different methods. For instance, the ImageNet dataset [Deng et al., 2009] contains over one million labeled images of objects with their bounding boxes. Other popular datasets, most specifically designed for object recognition, include CIFAR-10 [Krizhevsky, 2009], which consists of 60,000 images of 10 different classes, and CIFAR-100 [Krizhevsky, 2009], which collects 60,000 images of 100 different classes.

For our experiments, we selected the STL-10 dataset [Coates et al., 2011], which was created from a subset of the larger ImageNet dataset specifically designed for benchmarking SSL solutions. STL-10 contains RGB images of size 96x96 pixels, labeled with 10 classes: airplane, bird, car, cat, deer, dog, horse, monkey, ship, and truck (see Figure 8.1).

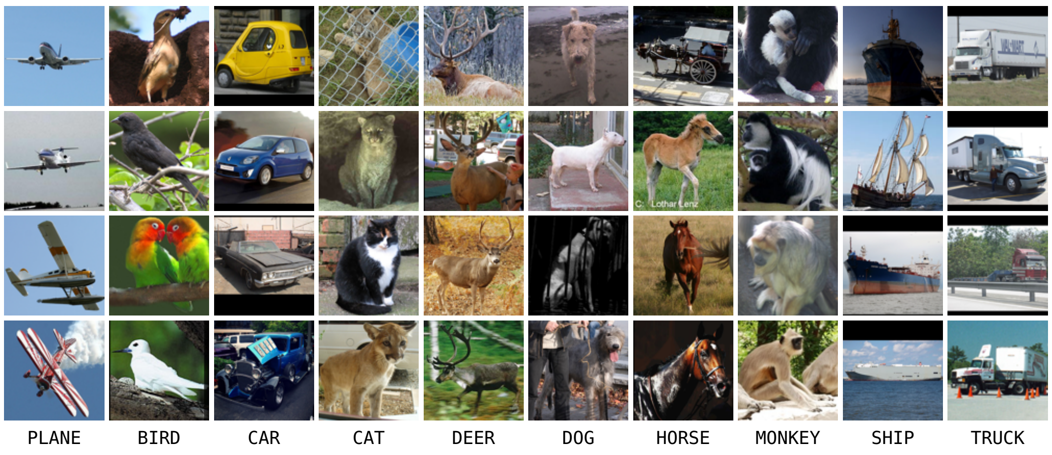


Figure 8.1. Sample labeled images from the STL-10 dataset. The labeled and test dataset contains 10 classes of objects. The unlabeled dataset contains a broader set of object classes, including other types of animals and vehicles. Each class has multiple images.

As mentioned, the dataset is designed for SSL, and thus it contains a large amount of unlabeled data. In particular, it provides 100,000 unlabeled images from a broader set of objects than the labeled set, for instance, it contains other types of animals (bears, rabbits, etc.) and vehicles (trains, buses, etc.). For downstream supervised training, the dataset provides 500 labeled training images and 800 test images per class. In this work, we use the full set of unlabeled images for pretraining, and we test different labeled data regimes for downstream training, from 100 to 500 labeled images per class. Downstream training is performed at each regime on 80% of the available labeled images for training, leaving the remaining 20% for validation. The test set is always kept separate and used only for final evaluation. For processing, the original image channels' range of $[0, 255]$ is remapped to $[-1, 1]$.

8.3 Architecture

To facilitate the comparison of different pretraining methods, we will keep the same encoder and downstream classifier architecture for all the experiments, and use them as components for the task-specific models.

8.3.1 Encoder

We use ResNet-18 as our encoder [He et al., 2016]. It is a widely used architecture for many recognition tasks. The initial image ($96 \times 96 \times 3$) is passed through a 7×7 convolutional layer, followed by a max pooling layer, and then through four groups of residual blocks to arrive. After the last group, a global average pooling layer is applied to obtain a 512-dimensional feature vector.

Residual blocks are the key component of the ResNet architecture, as they allow the model to grow deeper without suffering from vanishing gradients. Each residual block consists of two

3 × 3 successive 3 × 3 convolutional layers, each followed by batch normalization and ReLU activation. The input passes through this sequence to produce an output, which is then added to the original input via an identity skip connection. This allows the model to learn residual mappings, which are easier to optimize than the original unreferenced mappings.

Typically, the ResNet-18 ends with a fully connected layer that outputs the class probabilities, but in our case, we will remove this layer to implement task-specific architectures.

8.3.2 Downstream Classifier

For the downstream classifier, instead of designing a multilayer classifier as for the HAR use case (see subsection 4.2.2), we will use a single fully connected layer that takes the 512-dimensional feature vector from the encoder and maps it to the number of classes.

This is the same as in the original ResNet-18 architecture, with the difference that the final fully connected layer is trained separately during downstream optimization, keeping all the other layers frozen.

8.4 Selected Pretrain Tasks

Similarly to the HAR case, we selected CL and two pretext tasks for the pretraining phase.

8.4.1 Contrastive Learning

Given the success of CL in the HAR domain, we used the standard SimCLR [Chen et al., 2020] approach. For image processing, the model consists of the ResNet-18 encoder, followed by a projection head that maps the 512-dimensional feature vector to a lower-dimensional space. The projection head consists of two fully connected layers with ReLU activation in between, and a final layer that outputs the 128-dimensional representation.

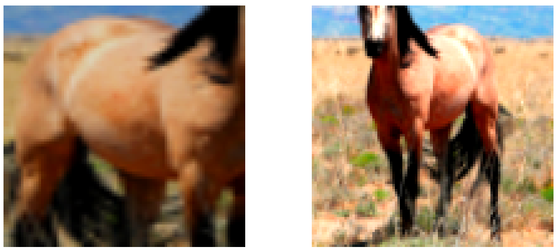

(a) Augmentation 1. (b) Augmentation 2.

Figure 8.2. Contrastive learning pretraining task. The model learns to maximize the similarity between two augmented views of the same image. The figure shows an example of two augmented views of the same image.

At training time, the model contrasts two augmented views of the same input image (see Figure 8.2), learning to maximize the similarity between the two representations while minimizing the similarity with representations of other images in the same batch. The batch size is set to 512, much larger than the 64 used in the HAR case, to allow for more negative samples and thus better CL. This was made possible by the larger number of unlabeled images available in the STL-10 dataset with respect to the small size of the UCI HAR dataset. The augmentation pipeline consists of multiple transformations: random horizontal flip, random rotation, random masking of a rectangular region, random crop and resize to the original size, random color jittering, random grayscale conversion, and random Gaussian blur. These augmentations are common in the literature [Pathak et al., 2016; Gidaris et al., 2018; Chen et al., 2020].

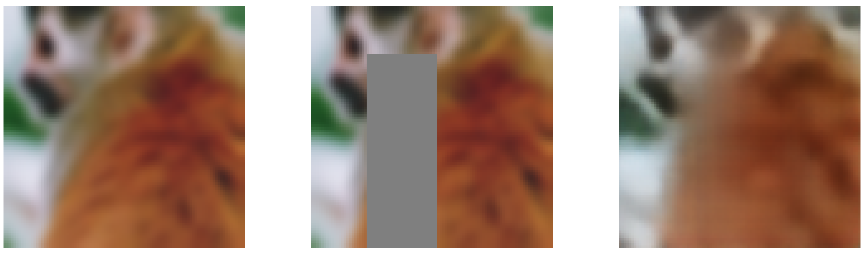
The augmentations are not applied in isolation, but composed together to create a more diverse set of views. They are computed by the training data loader, letting the model see different views of the same image at each training epoch. Similarly to the HAR implementation, the contrastive loss is computed using the NT-Xent loss (see Equation 2.2).

8.4.2 Pretext Tasks

In addition to CL, we selected two pretext tasks with the purpose of evaluating whether they can provide additional information when combined with CL in a multitask setting.

Masked Autoencoder

The first pretext task is a masked autoencoder that reconstructs the original image from a masked version of it. In the implemented version, the mask is a randomly sized and positioned rectangular region that replaces the image pixels with a uniform gray value (see Figure 8.3).



(a) Augmented. (b) Masked. (c) Reconstructed.

Figure 8.3. Masked autoencoder pretext task. The model learns to reconstruct the original image from a masked version of it. Before the masking, the original image is augmented to increase diversity. The figure shows an example of the original image, the masked version, and the reconstructed image.

Masked autoencoders have been widely used for SSL in the literature, and many specialized architectures have been proposed [Zhang et al., 2023a]. In this work, we use a direct implementation that composes the ResNet-18 encoder with a decoder mapping the 512-dimensional feature vector back to the original image size. The decoder consists of 4 up-convolutional blocks (each composed of a convolutional layer, batch normalization, and ReLU activation), followed by a final convolutional layer that outputs the original image size ($96 \times 96 \times 3$) and a tanh activation to ensure the output values are in the range $[-1, 1]$ as the input images. The loss used is a standard mean squared error (MSE) loss, computed between the original image and the reconstructed image. To further improve generalization, the input images are augmented before being masked. The augmentation pipeline consists of random horizontal flip, random crop and resize, random color jittering, random grayscale conversion, and random Gaussian blur.

The selected architecture performs reconstruction without any modification to the ResNet-18 encoder, which is not specifically designed for this task. Specific techniques, such as masked convolutions [Gao et al., 2022], and different architectures have been proposed in the literature to improve masked reconstruction performance [Zhang et al., 2023a]. However, we decided to keep the same encoder architecture for all tasks to facilitate comparison and focus on evaluating multitask performance rather than optimizing individual tasks.

Rotation Prediction

The second pretext task is rotation prediction, where the model learns to predict the rotation angle applied to the input image. This task might seem trivial, but it has been shown to be effective for unsupervised representation learning [Gidaris et al., 2018] and as a regularizer for CL [Kinakh et al., 2021; Addepalli et al., 2022].

During training, the input images are randomly rotated by 0°, 90°, 180°, or 270°, and the model must predict the correct angle (see Figure 8.4).

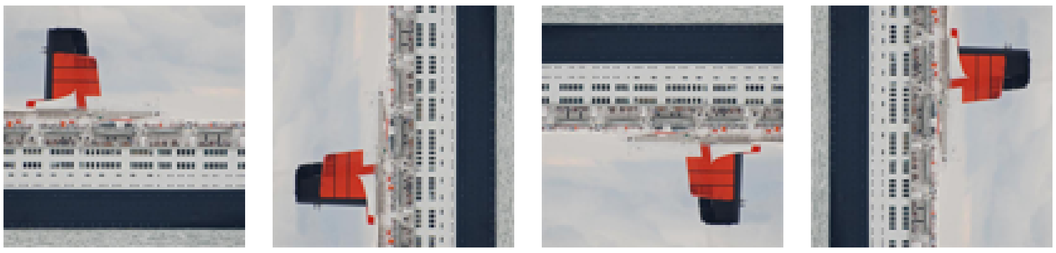

(a) 0 degrees. (b) 90 degrees. (c) 180 degrees. (d) 270 degrees.

Figure 8.4. Rotation prediction pretext task. The model learns to predict the rotation angle applied to an unseen input image. The images are shown with the predicted angle (0, 90, 180, or 270 degrees). The model predicts the correct angle with 100% confidence.

The model consists of the ResNet-18 encoder followed by two fully connected layers with ReLU activation in between. The first layer maps the 512-dimensional feature vector to a 256-dimensional vector, and the second layer outputs a 4-dimensional vector representing the probabilities of each rotation angle (softmax activation). The loss used is a standard cross-entropy loss, computed between the predicted probabilities and the true rotation angle label. To increase input diversity, the input images are augmented with a random horizontal flip.

8.4.3 Multitask SSL Model

The multitask SSL model is implemented as described in section 5.2. For the evaluation on images, we do not test all the possible task combinations, but only the combination of CL with one of the pretext tasks.

Similarly to the HAR case, the ResNet encoder is shared, and we use dynamic task weighting to combine the losses of the different tasks. Since the main task is CL, and the other task is intended as providing additional information, we scale the weight of the secondary pretrain task by a factor λ (set to 0.3 in the experiments), while keeping the dynamic weights adaptation during training. Reducing the contribution of the secondary task with a scale factor was also performed by Kinakh et al. [2021] and Addepalli et al. [2022] that, however, did not use dynamic weighting. Kinakh et al. [2021] found that static scaling was harmful and manually defined a schedule, forcing a zero weight for the first training epochs and 0.3 afterwards. By contrast, our method is fully automatic.

8.5 Results

Table 8.1 summarizes the results obtained by the individual pretraining methods and the two multitask combinations, also comparing them to the full supervised baseline. The results are reported in terms of downstream test accuracy. Due to limitations in the available computational resources, we do not perform k-fold cross-validation, but we report the results of a single run. Additionally, the encoders are pretrained on the full unlabeled dataset for just 450 epochs. Other works typically train for 1000 epochs or more [Kinakh et al., 2021]. By measuring the downstream accuracy using previous checkpoints, we found that the performance was still slowly increasing at the end of the pretraining phase, and thus, our reported results might be underestimated. We, however, do not expect any qualitative difference in the results, especially concerning the relative performance of the different methods.

Method	**100 (80 + 20)**	**250 (200 + 50)**	**500 (400 + 100)**
Supervised (no pretrain)	46.92	58.48	66.59
Masked Autoencoder (MAE)	42.60	46.96	49.79
Rotation Prediction (RP)	52.50	56.19	59.41
Contrastive Learning (CL)	73.65	76.65	78.51
MTL: CL + MAE	74.01	76.75	77.91
MTL: CL + RP	**74.59**	**78.22**	**80.17**

Table 8.1. Downstream test accuracy of the supervised training and of the different pretraining methods on the STL-10 dataset. The supervised baseline trains the model from scratch using the available labeled data at the different regimes. All the other models are pretrained on the full unlabeled dataset, and then downstream training is performed on the labeled data. The table shows the accuracy for different labeled data regimes: 100, 250, and 500 labeled images per class. For each regime, 80% of the labeled data is used for training and 20% for validation. The best results are highlighted in bold.

The supervised baseline is reported here only for reference. The network was trained from scratch on the available labeled data at the different regimes. To increase diversity, the input images were augmented with random horizontal flip and random crop and resize. The results show that the supervised model is not able to learn effectively at the reported labeled data regimes and it is significantly outperformed by our MTL approach.

Before discussing the multitask results, it is interesting to analyze the performance of the individual pretraining tasks. The first important consideration is that there is a very significant difference in the performance of the individual tasks, with CL consistently outperforming the other two pretrain objectives. Between the two pretext tasks, rotation prediction performs better than masked autoencoder, but still with a modest downstream performance. This gap can be explained by the fact that CL is presented with a large variety of augmentations and has a loss that explicitly drives the shape of the learned representations. The other two tasks only implicitly shape the representation space by solving a single problem. Their low performance indicates that the knowledge learned by solving these problems is less general and not directly transferable to the downstream classification task. This effect is evident in the t-SNE visualization of the embeddings obtained by rotation prediction (Figure 8.5a) and CL (Figure 8.5b), where the separation of classes emerges only for the latter.

The situation, however, changes when we consider the multitask combinations. Combining CL with MAE does not lead to a significant improvement. On the other hand, RP provides a large performance boost across the different labeled data regimes (+0.94 percentage points when using only 100 labeled images, +1.57 with 250, and +1.66 with 500). The achieved results are not far from the reported ones on the STL-10 dataset (see Figure 1 in Kinakh et al. [2021]).

Reconstructing the original image from a masked version of it, in addition to underperforming as a single-task pretrainer, seems to be redundant when combined with CL. It should be noted that masking is one of the augmentations used to compute contrastive views, so CL might already have acquired knowledge on reconstruction.

Rotation prediction also has a low performance when used alone, but it definitely provides additional knowledge when used in combination with CL. The effectiveness of rotation prediction when used in combination with CL is consistent with the findings of Kinakh et al. [2021] and Addepalli et al. [2022]. Addepalli et al. [2022] highlights that the CL task is noisy because different augmentations may lead to low similarity between the two generated views of the same image (that should count as positive examples), while at the same time the batch may contain many images of similar content (that should count as negative pairs), especially when the number of classes is much smaller than the batch size (in our case, 10 classes vs. 512 batch size). The rotation prediction task, instead, is totally unambiguous, leading to a much smoother objective. The combination thus adds knowledge but also simplifies the optimization landscape, leading to better performance.

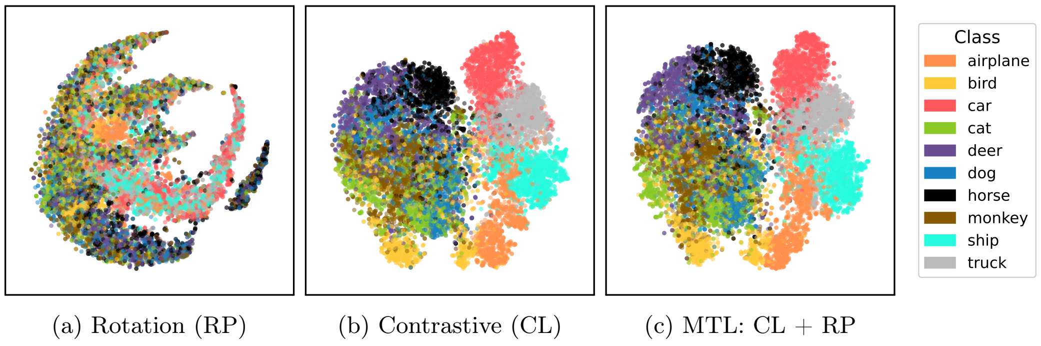


Figure 8.5. t-SNE visualization of the learned representations for the STL-10 dataset. The plotted samples are unseen during pretraining. The first two figures show the representations learned by the individual pretraining tasks: rotation prediction (RP) and contrastive learning (CL). The third figure shows the representations learned by the multitask model combining CL and RP. The colors represent the different classes in the dataset. The t-SNE embeddings are computed on the 512-dimensional feature vectors obtained from the ResNet-18 encoder.

The embedding structures generated from CL (Figure 8.5b) and multitask CL + RP (Figure 8.5c) show a reasonably clear clustering. It is interesting to note that the vehicles (planes, cars, ships, trucks) are close together, and each individual class is well separated. Moreover, the airplane cluster is close to the bird cluster. The other classes are more mixed, even though a differentiation is still visible (e.g., deer and horse are close together and far from birds). The difference between the simple contrastive and the multitask model is not visually very evident in the t-SNE 2D projection, even though the performance of the latter is better.

8.6 Closing Remarks

This chapter illustrated the generality of the proposed methods, showing a preliminary evaluation in a different domain than HAR, which has been used as the motivating example throughout the thesis.

The chosen example was image recognition, and the study focused on combining a strong CL method with complementary pretext tasks. Different from the HAR case, the pretext tasks were not designed for high performance when used alone. The study has shown that combining CL with a complementary and smoother task (rotation prediction) leads to improvements in the downstream performance, using the same techniques discussed in the rest of the thesis.

Chapter 9

Conclusions

As this thesis reaches its conclusion, we summarize in this chapter the main contributions of our work, discuss its limitations, and outline potential directions for future research.

9.1 Summary of Findings and Contributions

The thesis explored the combination of self-supervised learning (SSL) and multitask learning (MTL), taking Human Activity Recognition (HAR) as a case study. The main contributions can be summarized as follows:

- We conducted an analysis of the background and current state-of-the-art in HAR, strategies for learning with limited labeled data (including self-supervised and semi-supervised learning), MTL, and federated learning (FL). This analysis identified, in the combination of SSL and MTL, an interesting direction to explore to create robust encoders from unlabeled and possibly federated data.
- We defined a common framework and evaluation protocol for implementing and comparing different SSL methods and selected HAR as a benchmark domain.
- We identified promising pretraining objectives for SSL (including contrastive approaches and pretext tasks) and evaluated their effectiveness on downstream performance, demonstrating how they heavily improve performance with respect to fully supervised baselines in low-labeled-data regimes. An important finding is that multimodal contrastive learning (CL) between raw sensor data and handcrafted features is a very effective pretraining strategy in scarce labeled data scenarios, but as more labeled data becomes available, the additional advantage of features fades away.
- We proposed a modular multitask SSL model that combines multiple pretraining tasks using dynamic task weighting to create robust representations and evaluated it, showing increased performance with respect to single-task baselines across all tested labeled data regimes. In particular, we showed that dynamic task weighting effectively balances competing pretraining objectives, enabling the combination of tasks with very different loss scales. An extensive ablation study showed that the multitask approach is robust to different design choices, including the number and choice of tasks, the choice of encoder architecture, and latent dimension size.

- We explored how to exploit a pseudo-labeling strategy to leverage unlabeled data during downstream classifier training, and showed that it can further improve accuracy, especially in very low labeled-data regimes.

- We assessed the feasibility of applying the multitask SSL approach in a FL setting, preserving data privacy while maintaining model performance. Despite the added complexity of multitasking, we showed that the proposed approach still surpasses the performance of single-task baselines, even though the gap is smaller than in the centralized setting due to the difficulty of optimizing and aggregating very different local models.

- We showed that the methods are not specific to the HAR use case, but can be adapted to other domains. In particular, we showed how to combine CL with an auxiliary pretext task in the image recognition domain to improve performance.

9.2 Limitations and Future Work

The methods proposed in this thesis have shown promising results, but there are still several limitations and areas for future testing and research.

First of all, for practical reasons, all the experiments were mostly conducted on a single, relatively small-scale dataset (UCI HAR [Anguita et al., 2013]). The dataset, widely used in the literature, was very adapted to the task, and also challenging given the small amount of data available. We also performed a preliminary evaluation for image recognition using the STL-10 dataset [Coates et al., 2011], which is also widely used as a standard benchmark for SSL methods. Performing tests on datasets with different characteristics and eventually performing transfer learning between datasets would strengthen the results and show the generalization of the proposed methods. Moreover, as noted by Yuan et al. [2024], the quality of the embeddings significantly depends on the diversity and size of the unlabeled pretraining data. Thus, scaling to datasets of orders of magnitude larger could lead to very large improvements in performance. Given the tested feasibility of applying the proposed multitask approach in a FL setting, future work could focus on testing the methods on massive distributed datasets, tackling at the same time scaling and privacy issues.

Additionally, this work has mainly concentrated on evaluating all the models in the HAR context and has performed a limited evaluation in the image recognition domain. Verifying the effectiveness of the proposed methods in other domains would be valuable. This could also include modifying the network architecture to fit the specific characteristics of the data and eventually apply it to other tasks, while keeping the overall structure of the proposed multitask approach.

Furthermore, the goal of this thesis was not to optimize and tune the individual models, but rather to evaluate alternatives and provide a fair comparison of the different approaches. Nonetheless, the achieved performance on downstream tasks is not very far from the fully supervised state-of-the-art approaches in the HAR use case [Bozkurt, 2021]. Future work could focus on tuning the models, selecting other pretrain tasks, loss terms, and various hyperparameters to verify the real limits of our approach in various labeled data regimes.

Our ablation studies have shown that the different tasks do not contribute equally to the final downstream performance. An interesting direction for future work would be to refine the analysis to identify the cross-task relationships and the interactions between them that could be as differently specified as CL and classifying augmentations.

The proposed dynamic task weighting strategy successfully achieved the goal of combining multiple pretraining tasks, but more sophisticated approaches could be explored (see subsection 2.3.4). In particular, gradient-based methods, such as PCGrad [Yu et al., 2020], could blend well with the current implementation.

Finally, the current federated implementation has concentrated on the pretraining phase and should be extended to the downstream training phase to obtain an end-to-end privacy-preserving solution. Such an extension could also include the integration of FL methods designed for heterogeneous and unbalanced data distributions, such as Orchestra [Lubana et al., 2022].

Appendix A

Implementation Notes

The methods are implemented in PyTorch using a modular approach, allowing for easy integration of different methods and simplifying the configuration of the training parameters. All the source code is available on GitHub at `https://github.com/Alessandro-Gobbetti/MTSSL_for_Label-Efficient_Learning`. The repository also includes the configuration settings and instructions to replicate our experiments.

In the following, we provide information not included in the main text that helps with reproducing the results presented in this thesis.

Computational Environment. All experiments are conducted on a NVIDIA GeForce RTX 3080 with 10GB of GPU memory and a NVIDIA RTX A5000 with 24GB of GPU memory, along with an Intel Core i9-10900K CPU and 128GB of RAM. The implementation is based on PyTorch 2.7.1. To ensure reproducibility, all random seeds are fixed, and the experiments are run with deterministic algorithms to minimize variability in results.

Training Configuration. The pretraining phase uses the Adam optimizer with a learning rate of 1×10^{-3} and batch size of 64 samples. Unless explicitly discussed, we use the encoder obtained after 200 epochs of pretraining for all methods. The downstream classifiers are trained for a maximum of 100 epochs using the Adam optimizer with a learning rate of 1×10^{-3}. The batch size for downstream training is set to 32 samples. After training, the best model is selected based on the validation accuracy and evaluated on the fixed test set.

Multitask Learning Configuration. For multitask SSL approaches, dynamic loss weighting is employed using the method proposed by Cipolla et al. [2018], where task-specific uncertainty parameters are learned during training to automatically balance the contribution of different pretext tasks. The parameters are initialized to 1 and passed to the optimizer to update them alongside the model parameters.

Bibliography

Sravanti Addepalli, Kaushal Bhogale, Priyam Dey, and R. Venkatesh Babu. Towards efficient and effective self-supervised learning of visual representations. In *European Conference on Computer Vision*, pages 523–538. Springer, 2022. doi: 10.1007/978-3-031-19821-2_30.

Shams Forruque Ahmed, Md. Sakib Bin Alam, Maruf Hassan, Mahtabin Rodela Rozbu, Taoseef Ishtiak, Nazifa Rafa, M. Mofijur, A. B. M. Shawkat Ali, and Amir H. Gandomi. Deep learning modelling techniques: current progress, applications, advantages, and challenges. *Artificial Intelligence Review*, 56(11):13521–13617, 2023. doi: 10.1007/s10462-023-10466-8.

Mohammed Aledhari, Rehma Razzak, Reza M. Parizi, and Fahad Saeed. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8:140699–140725, 2020. doi: 10.1109/ACCESS.2020.3013541.

Hamza Amrani, Daniela Micucci, and Paolo Napoletano. Unsupervised deep learning-based clustering for human activity recognition. In *2022 IEEE 12th International Conference on Consumer Electronics (ICCE-Berlin)*, pages 1–6. IEEE, September 2022. doi: 10.1109/icce-berlin56473.2022.9937141.

D. Anguita, Alessandro Ghio, L. Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *The European Symposium on Artificial Neural Networks*, 2013.

Ong Chin Ann and Lau Bee Theng. Human activity recognition: A review. In *2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2014)*, pages 389–393, 2014. doi: 10.1109/ICCSCE.2014.7072750.

Akin Avci, Stephan Bosch, Mihai Marin-Perianu, Raluca Marin-Perianu, and Paul Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *23th International conference on architecture of computing systems 2010*, pages 1–10. VDE, 2010.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-Invariance-Covariance Regularization for self-supervised learning. *ArXiv preprint*, 2022. doi: 10.48550/arXiv.2105.04906.

Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Flower: A friendly federated learning research framework. *ArXiv preprint*, 2022. doi: 10.48550/arXiv.2007.14390.

Ferhat Bozkurt. A comparative study on classifying human activities using classical machine and deep learning methods. *Arabian Journal for Science and Engineering*, 47, 07 2021. doi: 10.1007/s13369-021-06008-5.

O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010. ISBN 9780262013848.

Kaixuan Chen, Dalin Zhang, Lina Yao, Bin Guo, Zhiwen Yu, and Yunhao Liu. Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities. *ACM Comput. Surv.*, 54(4), May 2021. ISSN 0360-0300. doi: 10.1145/3447744.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15745–15753, 2021. doi: 10.1109/CVPR46437.2021.01549.

Yanbei Chen, Massimiliano Mancini, Xiatian Zhu, and Zeynep Akata. Semi-supervised and unsupervised deep visual learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(3):1327–1347, 2024. doi: 10.1109/TPAMI.2022.3201576.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 794–803. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/chen18a.html`.

Roberto Cipolla, Yarin Gal, and Alex Kendall. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, 2018. doi: 10.1109/CVPR.2018.00781.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

Florenc Demrozi, Graziano Pravadelli, Azra Bihorac, and Parisa Rashidi. Human activity recognition using inertial, physiological and environmental sensors: A comprehensive survey. *IEEE Access*, 8:210816–210836, 2020. doi: 10.1109/ACCESS.2020.3037715.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5):313–318, 2012. ISSN 1631-073X. doi: 10.1016/j.crma.2012.03.014.

Sourish Gunesh Dhekane and Thomas Ploetz. Transfer learning in sensor-based human activity recognition: A survey. *ACM Comput. Surv.*, 57(8), March 2025. ISSN 0360-0300. doi: 10.1145/3717608.

Aiden Doherty, Dan Jackson, Nils Hammerla, Thomas Plötz, Patrick Olivier, Malcolm H Granat, Tom White, Vincent T Van Hees, Michael I Trenell, Christoper G Owen, et al. Large scale population assessment of physical activity using wrist worn accelerometers: the UK biobank study. *PloS one*, 12(2):e0169649, 2017. doi: 10.1371/journal.pone.0169649.

Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4829–4837, 2016. doi: 10.1109/CVPR.2016.522.

Sannara Ek, Riccardo Presotto, Gabriele Civitarese, François Portet, Philippe Lalanda, and Claudio Bettini. Comparing self-supervised learning techniques for wearable human activity recognition. *CCF Transactions on Pervasive Computing and Interaction*, 7(1):1–21, March 2025. ISSN 2524-5228. doi: 10.1007/s42486-024-00182-9.

Frank Emmert-Streib. From the digital data revolution toward a digital society: Pervasiveness of artificial intelligence. *Machine Learning and Knowledge Extraction*, 3(1):284–298, 2021. ISSN 2504-4990. doi: 10.3390/make3010014.

Emteq Labs. OCOsense Smart Glasses HAR Dataset, 2022. URL `https://www.kaggle.com/datasets/emteqlabs/emteq-ocosense-smart-glasses-har-data`.

Peng Gao, Teli Ma, Hongsheng Li, Ziyi Lin, Jifeng Dai, and Yu Qiao. Mcmae: Masked convolution meets masked autoencoders. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 35632–35644. Curran Associates, Inc., 2022.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018. doi: 10.48550/arXiv.1803.07728.

Alessandro Gobbetti, Martin Gjoreski, Hristijan Gjoreski, Nicholas Lane, and Marc Langheinrich. Fedmma-har: Federated learning for human activity recognition with missing modalities using head-worn wearables. *IEEE Pervasive Computing*, 23(4):40–49, 2024. doi: 10.1109/MPRV.2024.3475473.

Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NeurIPS '04, pages 529–536, Cambridge, MA, USA, 2004. MIT Press.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent a new approach to self-supervised learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NeurIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A survey on self-supervised learning: Algorithms, applications, and future trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9052–9071, 2024. doi: 10.1109/TPAMI.2024.3415112.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic task prioritization for multitask learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Yunhui Guo, Honghui Shi, Abhishek Kumar, Kristen Grauman, Tajana Rosing, and Rogerio Feris. Spottune: Transfer learning through adaptive fine-tuning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4800–4809, 2019. doi: 10.1109/CVPR.2019.00494.

Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. Deep, convolutional, and recurrent models for human activity recognition using wearables. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 1533–1540. AAAI Press, 2016. ISBN 9781577357704.

Harish Haresamudram, Irfan Essa, and Thomas Plötz. Assessing the state of self-supervised human activity recognition using wearables. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 6(3), September 2022. doi: 10.1145/3550299.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2020. doi: 10.1109/CVPR42600.2020.00975.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15979–15988, 2022. doi: 10.1109/CVPR52688.2022.01553.

Wenke Huang, Mang Ye, Zekun Shi, Guancheng Wan, He Li, Bo Du, and Qiang Yang. Federated learning for generalization, robustness, fairness: A survey and benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9387–9406, 2024. doi: 10.1109/TPAMI.2024.3418862.

Ayokunle Ige and Mohd Halim Mohd Noor. A survey on unsupervised learning for wearable sensor-based activity recognition. *Applied Soft Computing*, 127:109363, 07 2022. doi: 10.1016/j.asoc.2022.109363.

Wenchao Jiang and Zhaozheng Yin. Human activity recognition using wearable sensors by deep convolutional neural networks. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, page 1307–1310, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450334594. doi: 10.1145/2733373.2806333.

Yilun Jin, Yang Liu, Kai Chen, and Qiang Yang. Federated learning without full labels: A survey. *ArXiv preprint*, 2023. doi: 10.48550/arXiv.2303.14453.

Im Y Jung. A review of privacy-preserving human and human activity recognition. *International Journal on Smart Sensing and Intelligent Systems*, 13(1):1–13, 2020. doi: 10.21307/ijssis-2020-008.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1-2):1–210, June 2021. ISSN 1935-8237. doi: 10.1561/2200000083.

Michail Kaseris, Ioannis Kostavelis, and Sotiris Malassiotis. A comprehensive survey on deep learning methods in human activity recognition. *Machine Learning and Knowledge Extraction*, 6(2):842–876, 2024. ISSN 2504-4990. doi: 10.3390/make6020040.

Vitaliy Kinakh, Olga Taran, and Svyatoslav Voloshynovskiy. Scatsimclr: self-supervised contrastive learning with pretext task regularization for small-scale datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1098–1106, 2021. doi: 10.1109/ICCVW54120.2021.00129.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint*, 2016. doi: 10.48550/arXiv.1610.02527.

Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint*, 2017. doi: 10.48550/arXiv.1610.05492.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.*, 12(2):74–82, March 2011. ISSN 1931-0145. doi: 10.1145/1964897.1964918.

Matías Laporte, Davide Casnici, Martin Gjoreski, Shkurta Gashi, Silvia Santini, and Marc Langheinrich. USI-HEAR dataset. *Zenodo*, March 2024. doi: 10.5281/zenodo.10843791.

Oscar D. Lara and Miguel A. Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys & Tutorials*, 15(3):1192–1209, 2013. doi: 10.1109/SURV.2012.110112.00192.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539.

Byeongchan Lee and Sehyun Lee. Implicit contrastive representation learning with guided stop-gradient. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NeurIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

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

Jae-Han Lee, Chul Lee, and Chang-Su Kim. Learning multiple pixelwise tasks based on loss scale balancing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5087–5096, 2021. doi: 10.1109/ICCV48922.2021.00506.

Clayton Souza Leite, Henry Mauranen, Aziza Zhanabatyrova, and Yu Xiao. Transformer-based approaches for sensor-based human activity recognition: Opportunities and challenges. *arXiv preprint*, 2024. doi: 10.48550/arXiv.2410.13605.

Dongyuan Li, Zhen Wang, Yankai Chen, Renhe Jiang, Weiping Ding, and Manabu Okumura. A survey on deep active learning: Recent advances and new frontiers. *IEEE Transactions on Neural Networks and Learning Systems*, 36(4):5879–5899, 2025a. doi: 10.1109/TNNLS.2024.3396463.

Mohan Li, Martin Gjoreski, Pietro Barbiero, Gašper Slapničar, Mitja Luštrek, Nicholas D Lane, and Marc Langheinrich. A survey on federated learning in human sensing. *arXiv preprint*, 2025b. doi: 10.48550/arXiv.2501.04000.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2021. ISSN 2326-3865. doi: 10.1109/tkde.2021.3090866.

Christoffer Loeffler, Rasmus Hvingelby, and Jann Goschenhofer. *Learning with Limited Labelled Data*, pages 77–94. Springer Nature Switzerland, Cham, 2024. ISBN 978-3-031-64832-8. doi: 10.1007/978-3-031-64832-8_4.

Aleksej Logacjov. Self-supervised learning for accelerometer-based human activity recognition: A survey. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 8(4), November 2024. doi: 10.1145/3699767.

Ekdeep Singh Lubana, Chi Ian Tang, Fahim Kawsar, Robert P. Dick, and Akhil Mathur. Orchestra: Unsupervised federated learning via globally consistent clustering. In *International Conference on Machine Learning*, 2022.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017. URL `https://proceedings.mlr.press/v54/mcmahan17a.html`.

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6706–6716, 2020. doi: 10.1109/CVPR42600.2020.00674.

Goran Saman Nariman and Hozan Khalid Hamarashid. Communication overhead reduction in federated learning: a review. *International Journal of Data Science and Analytics*, 19(2): 185–216, 2025. doi: 10.1007/s41060-024-00691-x.

Charlie Nash, Nate Kushman, and Christopher K.I. Williams. Inverting supervised representations with autoregressive neural density models. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1620–1629. PMLR, 16–18 Apr 2019. URL `https://proceedings.mlr.press/v89/nash19a.html`.

Jianyuan Ni, Hao Tang, Syed Tousiful Haque, Yan Yan, and Anne H. H. Ngu. A survey on multimodal wearable sensor-based human action recognition. *arXiv preprint*, 2024. doi: 10.48550/arXiv.2404.15349.

Manuel T Nonnenmacher, Lukas Oldenburg, Ingo Steinwart, and David Reeb. Utilizing expert features for contrastive learning of time-series representations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16969–16989. PMLR, 17–23 Jul 2022.

Francisco Javier Ordóñez and Daniel Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1), 2016. ISSN 1424-8220. doi: 10.3390/s16010115.

Archit Parnami and Minwoo Lee. Learning from few examples: A summary of approaches to few-shot learning. *arXiv preprint*, 2022. doi: 10.48550/arXiv.2203.04291.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context Encoders: Feature Learning by Inpainting . In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, Los Alamitos, CA, USA, June 2016. IEEE Computer Society. doi: 10.1109/CVPR.2016.278.

Vittorio Perera, Tagyoung Chung, Thomas Kollar, and Emma Strubell. Multi-task learning for parsing the alexa meaning representation language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.12019.

Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010. ISSN 0262-8856. doi: 10.1016/j.imavis.2009.11.014.

Hangwei Qian, Tian Tian, and Chunyan Miao. What makes good contrastive learning on small-scale wearable-based tasks? In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pages 3761–3771, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393850. doi: 10.1145/3534678.3539134.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

E. Ramanujam, Thinagaran Perumal, and S. Padmavathi. Human activity recognition with smartphone and wearable sensors using deep learning techniques: A review. *IEEE Sensors Journal*, 21(12):13029–13040, 2021. doi: 10.1109/JSEN.2021.3069927.

Attila Reiss and Didier Stricker. Introducing a new benchmarked dataset for activity monitoring. In *2012 16th International Symposium on Wearable Computers*, pages 108–109, 2012. doi: 10.1109/ISWC.2012.13.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B. Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Comput. Surv.*, 54(9), October 2021. ISSN 0360-0300. doi: 10.1145/3472291.

Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. Multi-task self-supervised learning for human activity detection. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 3(2), June 2019. doi: 10.1145/3328932.

Lars Schmarje, Monty Santarossa, Simon-Martin Schröder, and Reinhard Koch. A survey on semi-, self- and unsupervised learning for image classification. *IEEE Access*, 9:82146–82168, 2021. doi: 10.1109/ACCESS.2021.3084358.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018a.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS '18, pages 525–536, Red Hook, NY, USA, 2018b. Curran Associates Inc.

Taoran Sheng and Manfred Huber. Reducing label dependency in human activity recognition with wearables: From supervised learning to novel weakly self-supervised approaches. *Sensors*, 25(13), 2025. ISSN 1424-8220. doi: 10.3390/s25134032.

Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1310–1321, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450338325. doi: 10.1145/2810103.2813687.

Ravid Shwartz-Ziv, Randall Balestriero, Kenji Kawaguchi, Tim G. J. Rudner, and Yann LeCun. An information theory perspective on variance-invariance-covariance regularization. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 33965–33998. Curran Associates, Inc., 2023.

Cathie Sudlow, John Gallacher, Naomi Allen, Valerie Beral, Paul Burton, John Danesh, Paul Downey, Paul Elliott, Jane Green, Martin Landray, et al. UK biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age. *PLoS medicine*, 12(3):e1001779, 2015. doi: 10.1371/journal.pmed.1001779.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint*, 2019. doi: 10.48550/arXiv.1807.03748.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390294.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NeurIPS '16, pages 3637–3645, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 119:3–11, 2019. ISSN 0167-8655. doi: 10.1016/j.patrec.2018.02.010.

Christoph Wieland and Victor Pankratius. Tinygraphhar: Enhancing human activity recognition with graph neural networks. In *2023 IEEE World AI IoT Congress (AIIoT)*, pages 0047–0054, 2023. doi: 10.1109/AIIoT58121.2023.10174597.

Sangmin Woo, Sumin Lee, Yeonju Park, Muhammad Adi Nugroho, and Changick Kim. Towards good practices for missing modality robust action recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(3):2776–2784, Jun. 2023. doi: 10.1609/aaai.v37i3.25378.

Yawen Wu, Zhepeng Wang, Dewen Zeng, Meng Li, Yiyu Shi, and Jingtong Hu. Decentralized unsupervised learning of visual representations. In Lud De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 2326–2333. International Joint Conferences on Artificial Intelligence Organization, 7 2022. doi: 10.24963/ijcai.2022/323.

Peiyao Xiao, Chaosheng Dong, Shaofeng Zou, and Kaiyi Ji. Ldc-mtl: Balancing multi-task learning through scalable loss discrepancy control. *ArXiv preprint*, 2025. doi: 10.48550/arXiv.2502.08585.

Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. A survey on deep semi-supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(9):8934–8954, 2023. doi: 10.1109/TKDE.2022.3220219.

Jun Yu, Yutong Dai, Xiaokang Liu, Jin Huang, Yishan Shen, Ke Zhang, Rong Zhou, Eashan Adhikarla, Wenxuan Ye, Yixin Liu, Zhaoming Kong, Kai Zhang, Yilong Yin, Vinod Namboodiri, Brian D. Davison, Jason H. Moore, and Yong Chen. Unleashing the power of multi-task learning: A comprehensive survey spanning traditional, deep, and pretrained foundation model eras. *arXiv preprint*, 2024. doi: 10.48550/arXiv.2404.18961.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NeurIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Hang Yuan, Shing Chan, Andrew P. Creagh, Catherine Tong, Aidan Acquah, David A. Clifton, and Aiden Doherty. Self-supervised learning for human activity recognition using 700,000 person-days of wearable data. *npj Digital Medicine*, 7(1):91, 2024. doi: 10.1038/s41746-024-01062-3.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12310–12320. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/zbontar21a.html`.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1476–1485, 2019. doi: 10.1109/ICCV.2019.00156.

Chaoning Zhang, Chenshuang Zhang, Junha Song, John Seon Keun Yi, and In So Kweon. A survey on masked autoencoder for visual self-supervised learning. In Edith Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 6805–6813. International Joint Conferences on Artificial Intelligence Organization, 8 2023a. doi: 10.24963/ijcai.2023/762.

Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021. ISSN 0950-7051. doi: 10.1016/j.knosys.2021.106775.

Fengda Zhang, Kun Kuang, Long Chen, Zhaoyang You, Tao Shen, Jun Xiao, Yin Zhang, Chao Wu, Fei Wu, Yueting Zhuang, and Xiaolin Li. Federated unsupervised representation learning. *Frontiers of Information Technology & Electronic Engineering*, 24(8):1181–1193, 2023b. doi: 10.1631/FITEE.2200268.

Kexin Zhang, Qingsong Wen, Chaoli Zhang, Rongyao Cai, Ming Jin, Yong Liu, James Y. Zhang, Yuxuan Liang, Guansong Pang, Dongjin Song, and Shirui Pan. Self-Supervised Learning for Time Series Analysis: Taxonomy, Progress, and Prospects. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 46(10):6775–6794, October 2024. ISSN 1939-3539. doi: 10.1109/TPAMI.2024.3387317.

Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 5(1): 30–43, 09 2017. ISSN 2095-5138. doi: 10.1093/nsr/nwx105.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2022. doi: 10.1109/TKDE.2021.3070203.