DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2020

On the effectiveness of β-VAEs for image classification and clustering

Using a disentangled representation for Transfer Learning and Semi-Supervised Learning

VITTORIO MARIA ENRICO DENTI

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

On the effectiveness of β-VAEs for image classification and clustering

Using a disentangled representation for Transfer Learning and Semi-Supervised Learning

VITTORIO MARIA ENRICO DENTI

Master in Computer Science
Date: June 16, 2020
Supervisor: Tianze Wang
Examiner: Prof. Amir H. Payberah
School of Electrical Engineering and Computer Science
Host company: Bontouch AB
Swedish title: Om effektiviteten hos β-VAE er för bildklassificering och klustering

Abstract

Data labeling is a critical and costly process, thus accessing large amounts of labeled data is not always feasible. *Transfer Learning (TL)* and *Semi-Supervised Learning (SSL)* are two promising approaches to leverage both labeled and unlabeled samples. In this work, we first study TL methods based on unsupervised pre-training strategies with *Autoencoder (AE)* networks. Then, we focus on clustering in the Semi-Supervised scenario.
Previous works introduced the β-VAE, an AE that learns a disentangled data representation from the unlabeled samples. We conduct an initial study of unsupervised pre-training with AEs to assess its impact on image classification tasks. We also design a new training method for the β-VAE based on cyclical annealing. The results show that annealing β during pre-training favours the learning of the target task. However, the best results on the target classification problem are obtained with a ResNet architecture with random initialization, trained only on labeled samples. Empirical evidence suggests that a deep network designed to learn complex patterns can achieve better results than a simpler pre-trained one.
It is known that the quality of the data representation also affects the clustering algorithms. Deep Clustering leverages the strengths of Deep Learning to find the representation that better supports clustering. Hence, we introduce the β-VAE with cyclical annealing in the training process of several methods based on Deep Clustering. With respect to a *Denoising Autoencoder (DAE)*, the β-VAE with annealing increases the Clustering Accuracy of the *Deep Embedded Clustering (DEC)* algorithm of 1% in the unsupervised scenario for the CIFAR-10 dataset. A new learning approach is also designed for clustering in the Semi-Supervised setting. We add an auxiliary supervised fine-tuning phase on the labeled samples. If 20% of the available examples are labeled, and the auxiliary task is executed, the Clustering Accuracy improves of 3.5% when the DAE is replaced by the β-VAE on the Fashion-MNIST dataset.
Experiments also show improvements over previous works in the literature.

Sammanfattning

Datamärkning är en kritisk och kostsam process, vilket försvårar möjligheten att komma åt stora mängder av märkt data. *Transfer Learning (TL)* och *Semi-Supervised Learning (SSL)* är två lovande metoder för att utnyttja prover som är både märkta och omärkta. I det här arbetet kommer vi först att studera TL metoder baserade på oövervakade förutbildningsstrategier med *Autoencoder (AE)* nätverk. Vi kommer sedan att fokusera på att samla ihop det semi-övervakade scenariot.

Tidigare arbete har introducerat β-VAE, en AE som lär sig en odelad datarepresentation från de omärkta proverna. Vi genomför en första studie av oövervakad förutbildning med AE för att utvärdera dess påverkan på bildklassificeringsuppgifter. Vi designar även en ny träningsmetod för β-VAE baserat på cyklisk glödgning. Resultatet visar på att glödgning β under förutbildning främjar inlärning av måluppgiften. De bästa resultaten på målklassificeringsproblemet erhålls emellertid med ResNet-arkitektur med slumpmässig initialisering, endast utbildad på märkta prover. Empiriska bevis föreslår att ett djupt nätverk designat för att lära sig komplexa mönster kan erhålla bättre resultat än en enklare förutbildad.

Det är känt att kvaliteten på datarepresentationen också påverkar klusteralgoritmerna. Deep Clustering utnyttjar styrkorna på Deep Learning för att hitta den representation som bättre stöder klustring. Därför introducerar vi β-VAE med cyklisk glödgning i träningsprocessen för flera metoder baserade på Deep Clustering. Med avseende på *Denoising Autoencoder (DAE)*, ökar β-VAE med glödgning klusternoggrannheten av *Deep Embedded Clustering (DEC)* algoritmen på 1% i det oövervakade scenariot för CIFAR-10 datasetet. Ett nytt inlärningssätt är också utformat för att klustra i den semi-övervakade inställningen. Vi lägger till en extra övervakad finjusteringsfas på de märkta proverna. Om 20% på de tillgängliga proverna är märkta och hjälpuppgiften utförs förbättras klusternoggrannheten med 3.5% när DAE ersätts av β-VAE på datasetet Fashion-MNIST. Experimentet visar också på förbättringar jämfört med tidigare verk i litteraturen.

Acknowledgements

I would like to start by thanking all the people that shared a piece of their path with me during my life, both from a professional perspective as well as from a personal one. I am grateful to my academic supervisors and examiners. Prof. Amir Payberah and Tianze Wang from the KTH Royal Institute of Technology, Prof. Marco Brambilla from my home university, Politecnico di Milano. They provided the guidance necessary for the development of this thesis.

I would also like to express my appreciation for all the people working at Bontouch AB. They introduced me to the ritual of the Swedish fika, taught me the first (and hopefully, not last) words in Swedish, and gave me the possibility to enjoy their workplace while working on my thesis. My gratitude goes to my industrial supervisor Carlo Rapisarda for his valuable advice and feedback when required. Also, special thanks go to Sara Blom for the support given in the Swedish translation of the Abstract of this document.

Furthermore, I would like to express my thankfulness to my family and my closest friends. If I had the opportunity to study for many years and live unique experiences in an international environment, it was thanks to the alacrity and the sacrifices made by my grandparents. I am also eternally grateful to my parents and my sister Enrica for growing me and teaching the proactivity necessary to face the real challenges in life.

Finally, I am grateful to the friends that closely shared this journey with me. First, all the amazing warriors and flatmates I found during this year, in particular Francesco Lorenzo, Francesco Staccone and Gabriele Gullì. We visited Scandinavia from the rainy Copenhagen up to the wild fiords of Tromsø, Norway. My thanks also go to Andrea Scotti for the experiences we lived together during spring 2020. *Audentes fortuna iuvat*.

Stockholm, June 16, 2020
Vittorio Maria Enrico Denti

Contents

List of Acronyms

β-VAE β-Variational Autoencoder

AE Autoencoder

AI Artificial Intelligence

ANN Artificial Neural Network

AUC Area Under Core

CNN Convolutional Neural Network

CV Computer Vision

DAE Denoising Autoencoder

DEC Deep Embedded Clustering

DL Deep Learning

GAN Generative Adversarial Network

ILSVRC ImageNet Large Scale Visual Recognition Challenge

InfoGAN Information Maximizing Generative Adversarial Network

KL Kullback-Leibler

ML Machine Learning

MLP Multi Layer Perceptron

MSE Mean Squared Error

NLP Natural Language Processing

NMI Normalized Mutual Information

ROC Receiver Operating Characteristic

SA Simulated Annealing

SSL Semi-Supervised Learning

TL Transfer Learning

VAE Variational Autoencoder

List of Figures

List of Tables

Chapter 1

Introduction

> *Sometimes it seems as though each new step towards AI, rather than producing something which everyone agrees is real intelligence, merely reveals what real intelligence is not.*
>
> Gödel, Escher, Bach: An Eternal Golden Braid

This introductory chapter gives a complete view over the purpose of the research project, introduces the main concepts related to *Transfer Learning (TL)* and *Semi-Supervised Learning (SSL)* and discusses the contributions deriving from the new approaches proposed in this thesis. Section 1.1 and Section 1.2 describe the motivation and the research context respectively. Section 1.3 and Section 1.4 report the problem formulation and the research questions addressed in the project. Section 1.5 offers an overview about the main contributions and the proposed approaches. Section 1.6 discusses the main limitations and possible directions for future investigation. Finally, Section 1.8 serves as outline for the rest of the document.

1.1 Motivation

The term *Artificial Intelligence (AI)* became very popular in the last decades as interest grew in the industry, in the research community, and the society. *Machine Learning (ML)* and *Deep Learning (DL)* are subsets of AI, whose recent advances find application in *Computer Vision (CV)*, *Natural Language Processing (NLP)* and more learning domains. Most of the new discoveries in the area of AI came from companies or universities which could access a large number of computing resources. In fact, these learning paradigms are based on heavy mathematical computations, to find approximated solutions to

optimization problems. One may observe that the most recent advances were favoured by cloud computing services, which offer access to computational and storage resources on a pay per use basis, by removing the need of owning physical computing and storage servers.

In AI another bottleneck is often represented by the amount of data necessary to build a model capable of generating reliable predictions. Several studies in the research area involve well known and publicly available labeled datasets. However, in many real domains, only a few data samples are available and the process of generating labels is often difficult due to high costs and possible human biases [1]. It is possible to argue that a complete AI has not been reached yet but research is going into that direction. In fact, if we consider the main characteristics of human intelligence, we can observe that humans can easily learn after a short experience, they can also generalize and transfer the knowledge from one task to another. Despite the recent successes of the learning approaches based on neural networks, they can still be considered far from the generality and robustness of biological intelligence [2]. Current ML and DL architectures struggle to achieve the aforementioned properties, in particular in the area of CV, as the more examples of experience are given as input, the better the resulting model.

In order to solve these limitations, a large part of the effort in research is about achieving generalization, learning from a few labeled examples, and exploiting available unlabeled data. This is necessary so as to employ learners in real-world scenarios and develop a real AI. Thus, this project studies state-of-the-art methods for learning from labeled and unlabeled data in the domain of image classification and proposes new successful approaches.

1.2 Research context

Traditional Supervised Learning problems can be addressed through ML and DL models. However, solving a Supervised Learning problem requires a large amount of labeled data because more data are accessible during the training phase, the lower the prediction error. In the area of CV, state-of-the-art learners are built upon *Artificial Neural Network (ANN)* architectures. Also, visual data needs to be labeled by humans. One may note that obtaining a large amount of labeled data is costly, the labeling process could suffer from human biases, and it could also raise privacy concerns when it deals with confidential data. The most popular methods to face these challenges are TL and SSL.

TL consists of transferring knowledge from one or more source tasks to

a specific target task in order to reduce the prediction error on the target task [3]. Of course, the more the target and the source learning tasks are related, the better the performance of the TL approach. A simple but very effective way to apply it is to use a model pre-trained on a similar source task, and start a fine-tuning procedure on the labeled data available for the target task.

SSL tries to learn both from labeled and unlabeled data, combining supervised and unsupervised methods to improve the learning behaviour. This approach can exploit the available unlabeled data to improve learning when the labeled data are scarce or expensive to obtain [4]. Clustering could be beneficial for SSL to propagate labels from the labeled to the unlabeled samples. In addition, the clustering assignments could be used as extra features for the enrichment of the available labeled examples. A promising direction of research is about improving the quality of the data representation for clustering in the Semi-Supervised setting.

1.3 Problem definition

It was demonstrated that TL and SSL provide the state-of-the-art performance to learn from small labeled datasets. It is noticeable that previous researches in the literature were focused on studying TL and SSL techniques to find possible ways of improvement for image classification [5, 6, 7, 8, 9].

TL is based on the concepts of domain and task [10]. A model is pre-trained to solve a source task in a source domain, then the knowledge is transferred for solving the target task in the target domain. The model that solves the source task is learnt through a pre-training procedure, then it can be adapted and fine-tuned to solve the target task. For instance, one approach that does not require labeled data in the source domain is the training of AE networks. The encoder network is pre-trained by taking advantage of the training of the AE, then it is fine-tuned on the target task. Erhan et al. [6] discuss the effect of several pre-training strategies on unlabeled examples for supervised problems. It is possible to define an unsupervised pre-training phase as the initial training step of a model (or a portion of it) on unlabeled data. Thus, unsupervised pre-training with AEs consists of training the AE so as to find a good initialization for the encoder network.

As the clustering assignments are used to enrich the labeled samples, SSL methods based on clustering require to build high-quality clusters in order to be successful. It is known that clustering is sensible to the data representation, thus most of the approaches first find a compressed representation of the inputs, then solve the clustering assignment task in the new feature space [11]. Deep

Clustering [12, 13] methods leverage the TL paradigm to learn an informative data representation with AE networks. First an unsupervised pre-training phase is executed with an AE, then the model learnt during pre-training is fine-tuned to extract the features necessary for the clustering algorithms.

The effect of TL through a novel generative method like the β-VAE [14] still needs to be evaluated. It is interesting to understand if the β-VAE can outperform traditional AEs in the context of TL via unsupervised pre-training, as it is expected to learn a disentangled latent representation. In addition, one may observe that the performance of a state-of-the-art architecture that completely ignores the unlabeled training data is often not reported while studying TL and unsupervised pre-training [15]. However, so as to properly evaluate the experimental results, it would be meaningful to consider this scenario to understand if each method really benefits from the unlabeled training samples. Thus, further investigation on unsupervised pre-training with AEs is needed to assess the impact on the final image classification task.

It is possible to note that the quality of the data representation also affects SSL methods based on clustering and label propagation. It is fundamental extracting good disentangled features that describe the raw data, transfer the knowledge from the pre-training task to the final clustering task, and jointly optimize the feature extraction and the clustering processes. Thus, the unsupervised pre-training with β-VAEs should be investigated in the context of clustering for high dimensional data. It is worth considering Deep Clustering since it is based on the "pre-train and fine-tune" paradigm and it uses DL for the clustering assignments.

Given this scenario, the research aims to investigate new approaches for TL built upon the pre-training paradigm with AEs. First, the investigation focuses on the design of pre-training strategies with Convolutional AEs for the image classification problem, then we focus on pre-training for Deep Clustering in the Semi-Supervised setting. In particular, we consider TL scenarios where the source and the target domains correspond, so we study the transfer of knowledge from an unsupervised task to a supervised one, but the data distribution does not change as the samples involved in the two tasks belong to the same dataset.

1.4 Research question

Considering the relevance of the problem, the formal research question addressed in this work can be decomposed into two related questions. We decouple the investigation phase from the improvement phase.

- Does unsupervised pre-training with AE networks, followed by fine-tuning, increase the predictive performance on image classification tasks?
- Is it possible to design pre-training strategies based on AEs to increase the quality of image clustering in a Semi-Supervised setting?

The first research question requires to understand whether the unsupervised pre-training with AEs can be beneficial for the image classification task. In particular, given the same network architecture, we want to understand whether the pre-trained final model is better than the one with random initialization. This is done by analyzing how the classification performance, after the fine-tuning of the network for the target task, changes by varying the amount of available labeled data. On the other hand, the second question is focused on finding possible ways to improve the clustering metrics reported in Section 2.5, by taking advantage of the design of pre-training strategies both for the Unsupervised and the Semi-Supervised setting. This is investigated by leveraging the TL paradigm and the unsupervised pre-training with AEs. During the project, we analyze the β-VAE and evaluate new learning approaches derived from it.

Concerning the first part of the research question, we hypothesize that TL could be beneficial for increasing disentanglement in the latent space and improve the predictive performance. Also, we expect that different pre-training strategies based on AEs give different results in terms of predictive performance on the target task. With respect to the second part of the question, the hypothesis is that by leveraging the TL paradigm it is possible to increase the quality of the predicted clusters thanks to the knowledge gained while solving different tasks.

1.5 Contributions

State-of-the-art frameworks for TL and SSL are investigated on images as data. The research is conducted on high dimensional data, with variable proportions of labeled examples, to test the frameworks in a non-trivial scenario. To the best of our knowledge, we are the first to investigate the effect of the β-VAE on TL and Deep Clustering. We also introduce cyclical annealing during the training process of the β-VAE and design new learning approaches for clustering in the Semi-Supervised scenario.

1.5.1 β-VAE applied to Transfer Learning

The first research question is answered by comparing the β-VAE with standard AE networks for TL. We introduce a new training process based on cyclical annealing and also compare the results with state-of-the-art architectures for image classification. The contributions can be summarized as follows:

- Analysis of unsupervised pre-training strategies with different AEs and benchmarking, in terms of predictive performance, with image classifiers that ignore the unlabeled training data.
- Investigation of the effect on multi-class image classification of unsupervised disentangled feature learning via pre-training. Cyclical annealing is introduced in the training process of the β-VAE.

The results show that annealing β during pre-training improves the performance of the target classification task. However, the best results are obtained by a ResNet architecture with no pre-training. Thus, the empirical evidence suggests that a deep network designed to learn complex patterns can achieve better results than a simpler pre-trained encoder.

1.5.2 Semi-Supervised Deep-Clustering

The second research question is answered by improving image clustering. We demonstrate that the pre-training via a β-VAE with annealing is beneficial for Deep Clustering. Also, we extend Deep Clustering for the Semi-Supervised scenario. The contributions can be summarized as follows:

- Introduction of unsupervised disentangled feature learning for clustering. Deep Clustering is combined with the β-VAE and annealing.
- Design of a novel training approach built on TL for clustering in the Semi-Supervised setting. The new method adds an auxiliary supervised fine-tuning stage to increase the degree of disentanglement.
- Extended experiments are conducted on the MNIST digits dataset to assess how the behaviour of the algorithms changes depending on the complexity of patterns in the inputs.

The new methods show improvements in clustering in terms of Clustering Accuracy, *Normalized Mutual Information (NMI)* score, and Silhouette score.

Therefore, the β-VAE and the new training approach derived from Deep Clustering for the Semi-Supervised setting are valuable methods. In particular, the β-VAE with annealing increases the Clustering Accuracy of the DEC algorithm. In a fully unsupervised scenario, it improves of 1% with respect to a *Denoising Autoencoder (DAE)* on the CIFAR-10 dataset. On the other hand, if 20% of labeled samples are used for the auxiliary task, the Clustering Accuracy improves of 3.5% when the DAE is replaced by the β-VAE on the Fashion-MNIST dataset. In addition, the new Semi-Supervised approach improves the results in the literature up to 7% in terms of final Clustering Accuracy on the CIFAR-10 dataset.

1.6 Limitations and future work

The experimental setting considers different percentages of labeled data to define a Semi-Supervised environment. The first delimitation is due to the percentages of samples considered in the research. As this is a first study, we evaluate the performances of each model considering six different amounts of labeled examples for each dataset. However, for future work, we call for more experiments considering more percentages. In particular, it would be meaningful to focus the analysis on the lowest amounts of examples.

An interesting direction of investigation is about the unsupervised pre-training of low layers in state-of-the-art architectures. We believe that the pre-training of individual residual blocks of ResNet could be a successful approach. Thus, we suggest investigating this topic and focus the study on the β-VAE, as empirical evidence suggests that it benefits from cyclical annealing during pre-training.

We evaluate the new Semi-Supervised training pipeline in terms of clustering metrics. However, we also believe that studying the effect of the auxiliary supervised fine-tuning phase on the data representation in the latent space may find directions for research and further improvement.

Finally, it is worth considering the limitations in terms of computational power. We ran the experiments on the `Google Colab` platform that offers free computing resources. We could access one NVIDIA Tesla K80 GPU, with 25GB of RAM and 68GB of HDD. We decided to avoid the usage of Google Cloud and AWS virtual machines mainly because of the high costs.

1.7 Research methodology

During the project, the research methodology typical of the scientific area [16] is combined with the pragmatism typical of the engineering field. We start with well-known methods and increase the level of complexity while narrowing down the scientific analysis. Therefore, during the project an *empirical research method* is applied so as to run multiple *quantitative experiments* to answer the research question and draw the final conclusions.

The experimental setting is based on a synthetic unlabeling procedure. Starting from the original datasets, variable percentages of labeled samples are retained while the remaining ones are considered as unlabeled. In particular, we define the Semi-Supervised environment by retaining 20%, 40%, 50%, 60%, 80%, and 100% of the original labeled training data.

The experiments involve three different datasets containing visual data: CIFAR-10 [17], Fashion-MNIST [18] and, finally, the MNIST digits [19]. We use Keras [20] with TensorFlow [21] backend and well known Python packages (numpy, scipy, sklearn, matplotlib) for all the proposed architectures.

1.8 Outline

Chapter 2 reports the theory behind ANNs, AEs, clustering, TL and SSL. The goal of this chapter is to describe the theoretical fundamentals behind the main topics studied in this thesis.

Chapter 3 discusses state-of-the-art methods provided in the literature, with a focus on the β-VAE and Deep Clustering. The goal is to study the most relevant works in the areas of TL and SSL, focusing on unsupervised pre-training with AEs. For SSL, the interest is on the clustering techniques that jointly improve feature extraction from images and clustering.

Chapter 4 provides a deep explanation of the investigation conducted in the thesis, explains the main contributions, and describes the newly proposed approaches built upon the β-VAE.

Chapter 5 shows the experimental setting and reports the results coming from the empirical experiments, analyzed with graphs as well as tables.

Finally, Chapter 6 discusses the results coming from the empirical method. Moreover, conclusions are derived from the thesis project and the limitations are commented to indicate possible directions for future work.

Chapter 2

Background

The more you know, the more you realise you know nothing.

Socrates

This chapter explains the theoretical background necessary to approach the research area. Section 2.1 formalises the learning problem, explains the importance of the labeled samples for supervised tasks and presents the main classification metrics. Section 2.2 defines the theory behind ANNs, as they are widely applied for CV problems. Section 2.3 focuses on the *Convolutional Neural Network (CNN)*, a model used for learning patterns from visual data. Section 2.4 serves as introduction to AEs, architectures used for learning efficient data representations. Section 2.5 reports the theory behind clustering, a learning technique that highly depends on the quality of the data representation. Finally, Section 2.6 and Section 2.7 present the main concepts and assumptions related to TL and SSL.

2.1 Preliminary concepts

ML and DL are subfields of the area of AI and they are promising topics of research both for the industry and the academia. The main idea behind learning is to discover patterns and regularities in the data samples, through the use of computer algorithms and optimization methods [22]. A simple example that clarifies what ML is and which are the main difficulties is given by the task of visual digits classification.

The goal of the learner is to take an image as input (described as a matrix of pixels) and generate as output the identity of the digits $0, ..., 9$. Therefore,

starting from the set of the available images (known as the training set) and a target vector that defines the true label corresponding to each digit sample in the training set, the algorithm that implements the training procedure is executed so as to build the model that learns the mapping between image and label. It is possible to define the resulting model as a function $\mathbf{y} = f(\mathbf{x})$ that takes as input a new image $\mathbf{x}$ and predicts the label corresponding to the data sample given as input.

Figure 2.1: Some image samples contained in the MNIST digits dataset [19].

It is noticeable that a model can be defined as good only if it achieves a good *generalization*, so it identifies the most relevant pieces of information in each input sample. This implies making correct predictions even for samples that are not exactly equals to those seen during the training phase. Since this thesis is focused on advanced concepts related to ML and DL, we are not going to describe in depth the basic theory behind these topics because it is widely explained by Bishop [22].

2.1.1 Taxonomy of Machine Learning

It is meaningful to briefly introduce the taxonomy of ML in order to define the terms and the concepts that we refer to in the next chapters of the document. ML can be divided into three main areas of interest: Supervised Learning, Unsupervised Learning, and Reinforcement Learning [22].

Supervised Learning is a learning setting where the input data are made available along with their corresponding target labels. This is the case of the example reported in Figure 2.1, as the goal is to estimate the unknown model that maps the input to the output given both the input samples and the ground truth labels. The most common tasks are *classification* (the output belongs to a discrete category) and *regression* (the output is a continuous value).

Unsupervised Learning is focused on learning an efficient representation of the given input in order to discover high-level patterns. Some of the most common problems are *clustering* (the goal is to find groups of similar samples) and *compression* (the goal is to project data to a lower-dimensional space).

Reinforcement Learning is a family of methods whose final objective is to find the best action to take, given a certain condition of the environment, in order to maximize the cumulative reward [23]. The focus is on learning how to do specific tasks. In most of the cases, the model is learnt through direct experience of the learner, with a process of trial and error.

2.1.2 The classification task

Classification is a supervised task where the model has to predict a discrete output, belonging to a set of predefined classes, given the input sample. In a binary classification problem, the output class can either be 0 or 1, while in case of a multi-class task the prediction can be any label belonging to a set of output classes. Many algorithms were developed for classification, the most known are Support Vector Machines, Logistic Regression, Decision Trees, Gradient Boosting and others explained in [22, 24, 25].

For each learning task, there are suitable metrics to consider to evaluate the predictive performance. Each metric has a specific goal in measuring the predictions. In the case of a binary classification problem, there are only two possible outcomes, positive class or negative class, depending on the true labels. Thus, it is possible to define the following variables to evaluate the quality of the predictions:

- *True Positive (TP)*: the model predicts the sample as belonging to the positive class, the sample really belongs to the positive class.
- *True Negative (TN)*: the model predicts the sample as belonging to the negative class, the sample really belongs to the negative class.
- *False Positive (FP)*: the model predicts the sample as belonging to the positive class, but the sample actually belongs to the negative class. From a statistical point of view, this is known as error of type 1.
- *False Negative (FN)*: the model predicts the sample as belonging to the negative class, but the sample actually belongs to the positive class. From a statistical point of view, this is known as error of type 2.

The values of True Positive, False Positive, True Negative and False Negative can be visually analyzed through a *confusion matrix*, then be used to compute

more advanced classification metrics that better summarize the predictive performance of the classifier.

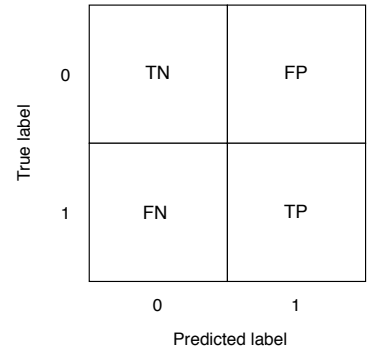


Figure 2.2: Confusion matrix for a binary classifier.

The most meaningful classification metrics are Accuracy, Precision, Recall and F1-score.

Accuracy
Accuracy is defined as the fraction of all the correct predictions over the total number of predictions. This metric is a good general indicator but it is not meaningful in the case of unbalanced datasets. It does not give specific information about the ability of the classifier in predicting positive and negative labels.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

Precision
Precision is defined as the number of samples correctly predicted as positive, divided by the total number of samples predicted as positive. It is possible to note that Precision allows understanding if the model suffers for a large number of False Positive predictions. This metric measures the accuracy in predicting the positive class.

$$Pre = \frac{TP}{TP + FP} \tag{2.2}$$

Recall
Recall is defined as the ratio of positive instances that are correctly detected by the classifier. This metric allows to understand whether there is a high penalizing cost due to the false negatives.

$$Rec = \frac{TP}{TP + FN} \tag{2.3}$$

F1-score
F1-score can be defined as the harmonic mean between Precision and Recall. Since the previous metrics are often in a trade-off (increasing Precision reduces Recall and vice versa), it is a common practice to evaluate the predictive performance through this metric, in particular while dealing with unbalanced datasets. F1 gets a high value only if both Precision and Recall have a high value.

$$F1_{score} = \frac{2 \times Pre \times Rec}{Pre + Rec} \tag{2.4}$$

Another common indicator is given by the *Receiver Operating Characteristic (ROC)* curve. It summarizes the trade-off between the True Positive rate and the False Positive rate in classification problems which consider multiple probability thresholds. The final goal is to build the ROC curve and to maximize the area under it, known as *Area Under Core (AUC)* [25].

In a multi-class learning task, it is necessary to classify each sample into 1 of N different classes, but the meaning of the metrics does not change. If we consider class i, Precision can be defined as the number of samples correctly predicted as i out of all predicted i samples. On the other hand, Recall is defined as the number of samples correctly predicted as i out of all the total number of actual i samples. While working on multi-class problems, it is a common practice to analyze the confusion matrix to understand if the model faces difficulties on a particular subset of the classes.

2.1.3 Labeled data as scarce resource

The example reported in Figure 2.1 gives an intuitive idea to introduce the area of research and highlight the role of data. The samples used as training data play a fundamental role while learning the models. The more data are available, the better the predictive performance that the final model can achieve [24]. It is known that the training data represent a bottleneck in ML, as a large dataset containing multiple examples is fundamental to improve the generalization of the learner, avoid the risk of overfitting and increase the accuracy of the model in making predictions [24].

Building a large labeled dataset of training data is not always feasible because data labeling is an activity that cannot be executed with satisfactory confidence by machines, so humans are needed to solve that task. In the example reported in Figure 2.1, a human agent is needed to assign the label to each image sample. It is easy to understand that the operation cannot scale to large datasets since human labelers require time, the process is costly and it

could generate privacy concerns if the data to label are confidential. In addition, if we consider the application of ML to medical problems if the process of data labeling suffers from human biases, a low final performance of the learning method could have a negative impact on the decision process. One of the most promising approaches for data labeling these days is offered by crowdsourcing methods. However, trying to create label guidelines for the labelers could generate ambiguity that results in differing interpretations of the same concept and favour the generation of inconsistent labels [26]. Moreover, it is worth mentioning that crowdsourcing for labeling is feasible only if the people have enough domain knowledge to solve the task and the data are not strictly confidential. It was shown that annotation in the video domain, as well as technical domains such as predictive maintenance, finance, and medicine, requires specialized skills [1]. Most of the workers are poor annotators, so this approach is not applicable to all kinds of datasets.

It is generally easy to acquire and store large amounts of data, but the bottleneck is represented by the process of data labeling. For this reason, it is relevant studying the areas of TL and SSL as they allow to jointly learn from labeled and unlabeled training data. This is possible because they combine supervised and unsupervised methods to solve the learning task.

2.2 Artificial Neural Networks

An ANN is a learning model widely used to solve Supervised, Unsupervised and Reinforcement Learning tasks as it can model non-linear functions, which describe the relationship between the input sample and the predicted output. In this section we explain the key concepts related to ANNs starting from the theoretical explanations provided by Bishop [22] and Mitchell [24].

2.2.1 Definition and main concepts

According to the *universal approximation theorem*, each ANN with an output layer that applies a linear activation function, and one hidden layer with any activation function can learn and compute any non-linear function. The theorem implies that this learning model is the most advanced since there are no constraints on the kind of mapping that it is possible to learn. ANNs take inspiration from the human brain, where hierarchical networks of neurons are connected by axons and where each neuron is triggered by the signals coming from the other neurons.

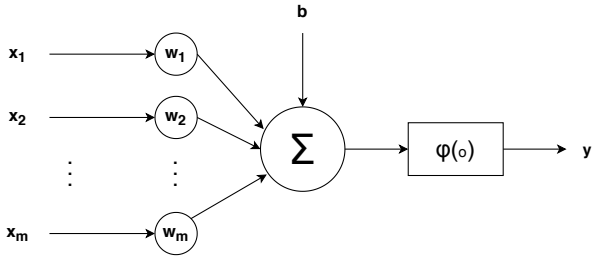


Figure 2.3: Structure of the artificial neuron.

A neuron takes an input vector $\mathbf{x}$ and compute the output $y = f(\mathbf{x})$ through the following steps:

1. Compute the dot product between each input and the corresponding weight: $\mathbf{x}^T\mathbf{w}$;
2. Add the bias parameter b to the result of the dot product: $\mathbf{x}^T\mathbf{w} + b$;
3. Compute the output of the neuron by applying the activation function $\varphi(\cdot)$: $y = \varphi(\mathbf{x}^T\mathbf{w} + b)$.

The structure of the artificial neuron described in Figure 2.3 shows that the parameters which define each neuron are the values of the weights w_i and the bias b. The activation function is part of the architecture and needs to be chosen depending on the task to solve and the structure of the network. A deep neural network is built by connecting multiple layers, each one consisting of several neurons, as shown in Figure 2.4.

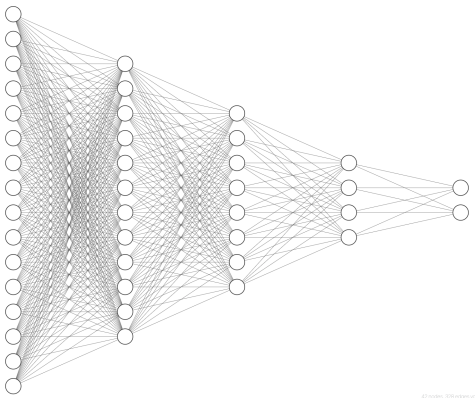


Figure 2.4: The connection of neurons defines a deep neural network.

During training, a network is forced to optimize its parameters so as to minimize a value, known as *loss*, which is computed through a loss function. When labeled data are available, the loss function is used to measure the difference between the value predicted by the network and the true value associated with the sample. The goal is to learn the parameters that make the two values as close as possible. The choice of the loss function depends on the task that we are aiming to solve. In the case of a classification problem, a common choice is the Binary Cross-entropy, while in the case of regression problems the suitable loss function is the *Mean Squared Error (MSE)*. More details about the desirable properties for the loss functions, as well as a description of the most frequently used losses, are provided by Goodfellow et al. [27].

2.2.2 Training procedure

The training of ANNs consists of solving an optimization problem to update the weights associated with each neuron while minimizing the loss. This is achieved by applying the Gradient Descent optimization method to identify the slope of the loss after each iteration of the algorithm and point in the direction of the largest change. Since the goal is to minimize a certain value, we want to follow the gradient downwards and update the parameters according to it. The backpropagation algorithm is used to compute the gradients, while the optimizer defines how to update the network parameters depending on the value of the gradient.

Backpropagation consists of two phases. First, the forward pass is executed to make a prediction given the input, as well as compute the error through the loss function. Second, the backward pass is used to go through each layer in reverse order, to evaluate how much each connection contributed to the final error. Finally, it is possible to update the network weights.

The *optimizer*, also known as optimization algorithm, solves the optimization problem and its goal is to reach the point of global minimum, so as to find the optimal solution. The optimizer updates the weights according to the results of the backpropagation algorithm through the concept of *learning rate*. The general update formula can be described as in equation 2.5. L is the loss function, η is the learning rate and w_{ij} is the j-th weight in the weight vector $\mathbf{w_i}$ of neuron i:

$$w_{ij}^{(next)} = w_{ij} - \eta \frac{\partial L(\mathbf{w}_i)}{\partial w_{ij}} \tag{2.5}$$

Depending on the chosen optimizer, the update formula slightly changes by involving new parameters and derivatives but the main structure remains as

described above. The most used optimizers are Stochastic Gradient Descent and its variants like Momentum, Adaptive Gradient, RMSProp and Adam (it combines RMSProp and AdaGrad) [27]. Adam is a popular optimizer because it handles most of the weaknesses of the other methods. For instance, it handles sparse gradients and does not require stationary targets.

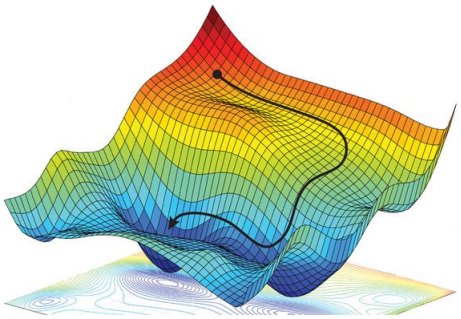


Figure 2.5: An example of optimization surface with the path successfully followed by the optimizer to reach the global optimum [28].

Another relevant engineering choice is the strategy to apply during the training process. In fact, it is possible to compute the error and update the model after each input sample (stochastic gradient descent), after each cycle through the whole training data (batch gradient descent) or after a batch of training data, whose size is a training parameter (mini-batch gradient descent).

Thanks to the successes of ANNs in solving learning tasks, more advanced architectures were developed in order to solve different types of problems. Some of the most popular are: Convolutional Neural Networks [29], Recurrent Neural Networks [30] and Long-Short Term Memory Networks [31].

2.3 Deep Learning in Computer Vision

CV is a field of research focused on studying how machines can extract information from digital images and videos. From an engineering point of view, the goal is to automate operations traditionally done by the visual system of humans [32]. DL methods can be applied to understand the content of images, through the extraction of visual information, to support pattern recognition in traditional learning problems such as regression, classification or policy learning in the case of a Reinforcement Learning scenario.

Since in the context of this thesis the final goal is to evaluate the experimental results on visual data, in the next sections we explain the role of feature extraction and the most promising architectures.

2.3.1 Learning efficient representations

Humans observe the world through their senses and build a simplified model of the environment in order to decide how to behave. During life, people handle a large amount of information, the brain processes all the data so humans are able to access a simplified and abstract representation of the world and take decisions [33]. Each learning task is influenced by the features that bring relevant information as informative features allow to better make predictions. For example, in the case of emotion classification from facial images, it is important to extract the representations that properly describe the shape of the faces of individuals, such as the lines of the eyes and the mouth, to create a good learning setting. For representation learning the best is being able to find disentangled features, features that are independent, and associated with some particular patterns in the input [34].

It is known that the quality of the extracted features has an impact on the final performance of the learner. Therefore, during the years research focused on feature selection methods, ANN architectures to improve the quality of the feature extraction process as well as learning frameworks to find high-quality features with a good degree of disentanglement. One of the first attempts to efficiently learn relevant features is related to sparse coding. It consists of unsupervised methods to learn a sparse representation of the data through a set of defined sparsity constraints [35]. In addition, in recent years DL methods became state-of-the-art for dimensionality reduction and feature extraction from high dimensional input data.

2.3.2 Convolutional Neural Networks

A *Convolutional Neural Network (CNN)* is an architecture often applied while dealing with CV problems, such as image classification, object detection, image captioning, and other related tasks. CNNs are designed to extract complex patterns and features from visual data. The main idea is to find hierarchical patterns in the input samples to build more advanced representations as a combination of simpler features. Each neuron in a CNN only responds to a restricted region of the visual field, called receptive field. The receptive field of different neurons partially overlaps, through a sliding mechanism known as stride, to cover the entire input field at the end. The network is typically built as a sequence of convolutional layers and pooling layers [36].

The *convolutional layers* consist of a set of learnable filters (also known as kernels) that cover a specific receptive field. During the forward pass, the dot product between each filter and the input data is computed in order to

extract higher-level features. In this way, the network learns the filters which are activated each time a certain pattern appears in the input image.

The *pooling layers* are used to execute a down-sampling. This supports the network in learning higher-level features, as well as prevent overfitting over the training data. The most common mechanism is max pooling: it consists in dividing the input image into rectangles of size $m \times n$ and extracting the biggest value for each region.

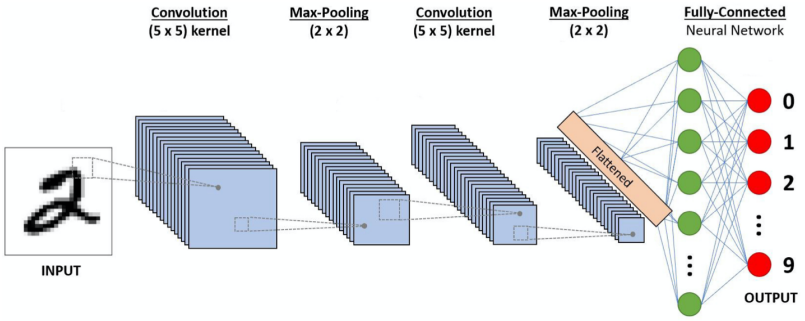


Figure 2.6: A simplified schema of a CNN for image classification.

A CNN architecture needs to be designed and tuned depending on the task to solve. For example, some critical design choices are the number of convolutional layers, the number of filters, the filter size, the stride, and the number of pooling layers. In the last decade, the research effort was dedicated to find advanced architectures for image classification tasks. State-of-the-art CNNs are LeNet-5 [37], VGG-16 [38], Inception-v1 [39], ResNet50 [40] and more complex networks built on top of them.

2.4 Autoencoders

An AE is an architecture used to learn a compressed representation of the input to efficiently extract features. The main goal is to reduce the dimensionality of the samples while retaining the most meaningful information. It may also serve as a pre-training strategy to initialize networks for feature extraction.

AEs are often designed as symmetric architectures. The left side of the network (known as *encoder*) is dedicated to the compression of the input data so as to retain high-level features, while the right side of the network (known as *decoder*) tries to reconstruct the reduced representation as close as possible to the original data. The loss function used by the network forces the reconstructions to be similar to the inputs.

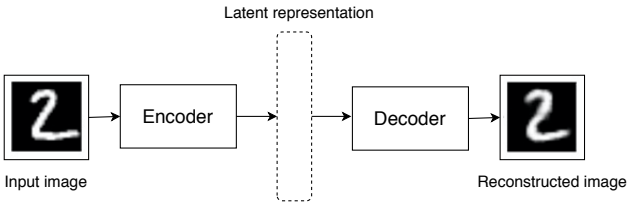


Figure 2.7: The main building blocks of the AE architecture.

AEs are unsupervised models because they are not trained on labeled data. However, in order to define a training framework, the input samples themselves work as pseudo labels: the loss function compares the input example with the reconstructed output to measure the quality of the reconstructions and update the network parameters according to the backpropagation algorithm. The output layer has the same number of neurons as the input layer, the penultimate layer has the same number of neurons as the second layer and so on in order to create a symmetric network, to sequentially execute reductions and reconstructions. One layer is dedicated to the central bottleneck, where the compressed code representations (known as *latent features* or *latent codes*) are learnt.

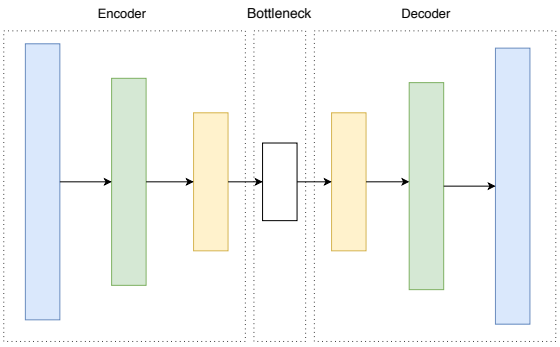


Figure 2.8: The symmetric structure of a deep AE architecture.

As the model is expected to learn how to retain relevant information, the objective is not to learn how to exactly reproduce the input on the output. It would not extract knowledge about the informative patterns in the input. For this reason, the network structure is often restricted to approximately reconstruct the original sample, while preserving only meaningful features from the input.

Many architectures for AEs were proposed during time and different approaches were designed to improve the representation of the data. It is the case of the *Sparse AE* [41], the *Denoising Autoencoder (DAE)* [42] and the *Variational Autoencoder (VAE)* [43].

2.4.1 Sparse Autoencoders

It is known that learning representations by forcing sparsity improves the final predictive performance, both for regression and classification tasks, as the degree of generalization is increased.

Sparse AEs add sparsity constraints during training to discourage the network parameters from reaching large values, as well as increase the level of generalization. Most of the time the constraints are in the form of L1 regularizations and L2 regularizations. This forces the model to respond to the real statistical distribution underlying the training data, as well as learning high-level features [41]. Another advantage is that sparsity constraints encourage the activation of specific regions of the network depending on the input sample while forcing the other areas to keep their neurons inactive to better respond to the relevant patterns. Hence, Sparse AEs prioritize which aspects of the input need to be learnt to extract useful properties from the data, in the form of an efficient feature representation, thanks to the use of regularizers. Regularizers are usually applied to reduce the risk of overfitting as explained by Mitchell [24], however in this context they cause the network to represent each input as a combination of a small number of active neurons. As a consequence, each neuron in the bottleneck (the coding layer) models a meaningful feature. Typical loss functions for evaluating the reconstructions on the output are the MSE and the Binary Cross-entropy.

It is possible to formalize the *encoder* function as $\mathbf{l_i} = f(\mathbf{x_i})$, where $\mathbf{x_i}$ is a generic input vector given to the network, and the *decoder* function as $\mathbf{y_i} = g(\mathbf{l_i})$, where $\mathbf{y_i}$ is the reconstruction produced as output vector. Given m samples as training data, if n is the total number of pixels in each image, x_{ij} the j-th input pixel within the input sample $\mathbf{x_i}$ and y_{ij} the j-th pixel in the reconstructed output $\mathbf{y_i}$, the loss functions are defined as follows.

- The MSE is applied for the unsupervised training of AEs to estimate how much the input sample and the reconstruction differs. This function allows to formulate the reconstruction task as a regression problem. In the case of visual data, it corresponds to the pixel-by-pixel difference.

$$L_{MSE} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (y_{ij} - x_{ij})^2 \tag{2.6}$$

- The Binary Cross-entropy can be applied only if the output layer of the decoder network has a sigmoid activation function. In the case of images, if the pixel values are normalized in the range $[0, 1]$, the Binary

Cross-entropy loss gives a good estimates of the pixel-by-pixel difference.

$$L_{BC} = -\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\log(y_{ij}) + (1 - x_{ij})\log(1 - y_{ij}) \qquad (2.7)$$

2.4.2 Denoising Autoencoders

While in Sparse AEs the internal representation is subject to sparsity constraints, DAEs try to achieve a compressed representation by adding a random noise in the encoding network, to increase the degree of generalization. The idea is to feed a corrupted input so as to train the network to reconstruct the original non corrupted input. The theoretical principle behind this technique is that a high-level representation can be learnt as meaningful features are insensible and robust to random corruption of the input [42]. Moreover, so as to properly remove the initial noise, the network is forced to learn features that represent useful patterns in the data distribution and extract high-level information. The techniques for image reconstruction, image restoration, and image denoising may be applied to several machine vision tasks [44].

DAEs are trained to minimize either the MSE (2.6) or the Binary Cross-entropy (2.7) as loss. Regularization terms can be added to build a Sparse DAE. The training process of a DAE follows these principles:

- The initial input is corrupted either by adding a random Gaussian noise or by randomly deactivating some neurons, both in the input layer as well as in the whole encoder network, according to a *dropout* rate.
- The network is trained as a standard AE to learn the mapping between the input and the latent features.
- The latent features are used to reconstruct the input sample, the loss is computed and the network weights are updated.

As in the case of Sparse AEs, each layer of neurons produces a representation of the input that is more abstract than the one computed by the previous layer, as it is obtained by aggregating more operations.

It is worth to note that the output of the network is not stochastically generated, but that a *stochastic perturbation* is added only during the training process. In fact, once the model has learnt the parameters supposed to be optimal, no corruption is added in the encoder network. AEs can be applied

both on standard multi-dimensional inputs as well as high dimensional data samples with spatial information such as images. In the second case, the network is composed of convolutional and fully connected layers. That defines a Convolutional AE.

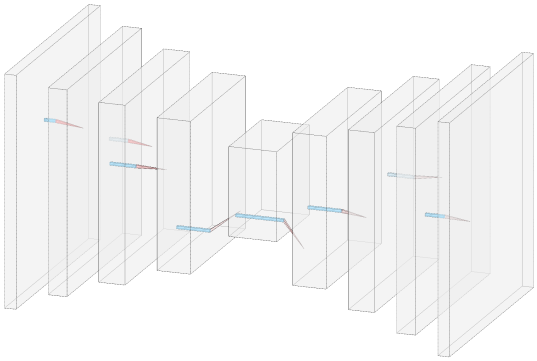


Figure 2.9: A schema of a Convolutional AE for images.

2.4.3 Variational Autoencoders

The VAE is a generative method, thus it is capable of generating new data. A VAE is an architecture that follows the traditional patterns explained before: the encoder network is connected to the decoder network through a bottleneck and the two networks are symmetric. Unlike the Sparse AE and the DAE, the VAE can generate new data instances that look like they are sampled from the training data. It defines a *directed probabilistic graph model* where a posterior distribution is generated through a neural network. This architecture learns a distribution of latent variables by leveraging a variational approach. It requires an additional component in the loss function, so as to force the network to learn the latent statistical distribution that reflects the target distribution [43].

The encoder does not directly generate the features that are used during the reconstruction but a mean coding μ and a standard deviation coding σ. Then, the latent representation is randomly sampled from a Gaussian distribution $N(\mu; \sigma)$ and the decoder starts the reconstruction from those features.
In a VAE the loss function is the sum of two terms:

1. The standard *reconstruction loss*, that measures the quality of the reconstructed images. It can be either the MSE described in 2.6 or the Binary Cross-entropy of equation 2.7.

2. The *latent loss*, which forces the network to have latent features that look like they are sampled from a Gaussian distribution. This can be computed as the *Kullback-Leibler (KL)* divergence between the target Gaussian distribution and the predicted distribution of the internal codings. The KL divergence loss allows to measure how much a given probability distribution differs from a second one. If k is the size of the latent vectors and m the number of samples, it is defined as follows:

$$L_{KL} = -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{k}(1 + \log(\sigma_{ij}^2) - \sigma_{ij}^2 - \mu_{ij}^2) \tag{2.8}$$

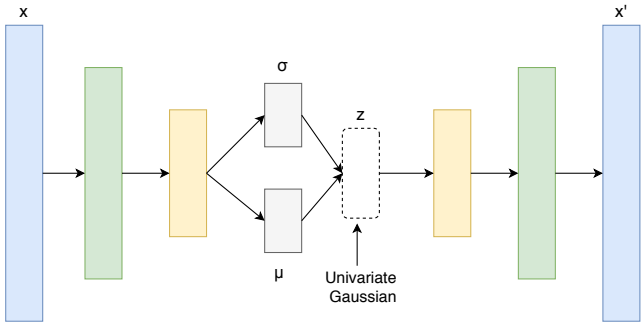


Figure 2.10: Diagram showing the logical structure of the VAE.

Since the VAE is a generative model, it is capable of generating new instances even after the training phase as opposed to DAEs, where randomness is applied only during the training. Generative modeling emulates the process of data generation to discover the casual relations between the given samples, as well as find the features that characterise the data distribution.

2.5 Clustering

Clustering is a technique belonging to the area of Unsupervised Learning. It finds patterns in data without the need for labeled examples, to create groups of similar entities. Clustering is often applied to the samples to find latent structures and define clusters: members of the same cluster are similar to each other while members of different clusters have a high dissimilarity [45]. Solving clustering tasks is not trivial when the input data have high dimensionality, like in the case of images, because it makes it difficult to clearly identify the different clusters if the data representation is not convenient.

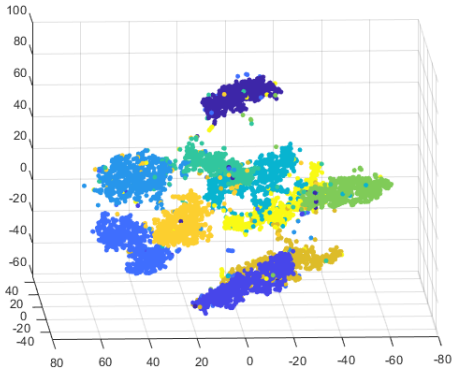


Figure 2.11: Clusters in the 3D space [46]. In a high dimensional space the points tend to have the same distance from each other, thus finding clusters is not trivial. This is a typical issue while working with images.

2.5.1 Traditional techniques

The first clustering methods were based on hierarchical algorithms and they are known as *Hierarchical Clustering*. In the case of *Agglomerative Hierarchical Clustering*, each initial point is considered as a cluster and iteratively the two nearest clusters are combined into one. On the other hand, *Divisive Hierarchical Clustering* starts with one cluster and recursively splits it. These methods are not usually applied because of their high complexity. At each step, there is the need to compute the pairwise distances between the clusters at cost $O(N^2)$, where N is the number of input points, and this is repeated N times, so the final cost can be estimated as $O(N^3)$. Some improvements were proposed to reduce the complexity to $O(N^2 \log(N))$ [45].
From a computational point of view, this is expensive for large datasets so the algorithms belonging to the *point-assignment class* are considered as next topic.

K-means
The K-means algorithm belongs to the point assignment family of clustering algorithms. It assumes a Euclidean space, so it could not achieve excellent clustering results while working with high dimensional data. The critical parameter of the algorithm is K, the final number of clusters, which must be chosen carefully by leveraging either prior knowledge or the heuristic described at the end of this section.

The algorithm after each iteration computes the *centroid* of each cluster, which is the mean of the coordinates of the points belonging to the cluster,

to define the unique position of the cluster as a whole [47]. The centroids are used to determine the clustering assignments. K-means takes as input the parameter K and returns as outputs K clusters of samples. It works as follows:

1. Initialize the centroids by picking one point per cluster. For example, choose K random points.

2. For each input sample:

   (a) Compute the distance between the point and each cluster centroid. Then, assign the point to the nearest cluster.

   (b) Update the cluster centroids according to the new clustering assignments.

3. Keep executing the loop over the input data points until clusters are stable and assignments do not change.

If the number of clusters is not known a priori, like in the case of most of the clustering problems, a common practice is to find K through a trial and error procedure: different values of K are evaluated and the one that gives the best clustering performance is chosen. It is worth to note that the choice of the initial K points affects the final result so a good method is to choose the first initial point at random, then choose the second point such that it is the one whose minimum distance from the previously selected point is the largest and repeat the same procedure for the next points. In this way, we are guaranteed to initially select dispersed points and explore the entire space of the possible clustering assignments [47].

For each round, each point is evaluated to find the nearest centroid. Hence, the computational cost of each round is $O(KN)$: this function is linear in N but the final complexity depends on the total number of iterations needed to reach the convergence of the algorithm and it cannot be known a priori. The spatial complexity of K-means is $O(K + N)$.

Spectral clustering

This algorithm is based on geometrical and mathematical concepts related to the spectra of matrices. Spectral clustering computes the eigenvalues (spectrum) of a matrix derived from the similarities between items, in order to solve the clustering problem in the space of the eigenvectors. This allows reducing the dimensionality before clustering so as to be focused on the most relevant directions of information within the input data. The initial N points in the N dimensional space are transformed into N points in the K dimensional space,

where K is the expected number of clusters. A stable clustering is obtained by using the K that maximizes the gap between consecutive eigenvalues [48].

Given the input samples, it is possible to define the *similarity matrix* (also known as *affinity matrix*) as a symmetric matrix A, where element a_{ij} defines the similarity between data item i and item j. Starting from A it is possible to compute the *Laplacian matrix* L, which is used to extract the eigenvalues. In addition, Spectral Clustering is often applied to graphs to detect communities and structures: in that case, there is no need to compute the affinity matrix because the graph is described by its characteristic matrix, where each similarity defines the weight of the edge connecting the two nodes [48].
The macro steps of the algorithm are as follows:

1. Build the affinity matrix A using a Kernel function suitable for computing the similarities between the input data. A threshold is often defined to keep the matrix as sparse as possible and reduce memory usage.

2. Build the Laplacian matrix $L = D - A$, where D is the diagonal matrix that contains the sums of the affinities of each row in matrix A. These values represent the weights associated with the affinities.

3. Find the k smallest eigenvalues (except the first one, that always equals 0) of matrix L and extract the corresponding k eigenvectors to define a k-dimensional subspace.

4. Apply a clustering algorithm in the new subspace (e.g. K-means) to solve the original clustering task.

The main strength of Spectral clustering is that it can learn more complex patterns in the data than K-means, but the drawback is that it cannot be scaled to large datasets. In fact, calculating the affinity matrix has a computational complexity estimated by $O(N^2)$. Also, defining a good Kernel depending on the data to cluster is not trivial.

2.5.2 Evaluation metrics

It is necessary to define meaningful evaluation criteria to estimate the quality of the predicted clusters. The goal of each clustering algorithm is to achieve a high internal cluster similarity and a low external cluster similarity. Thus, clustering metrics are focused on measuring the results from both an internal and an external perspective [49].

Clustering Accuracy
Clustering is an unsupervised method, it means that it does not learn from labeled data. In the context of SSL, clustering tasks might be executed on labeled data, so it is fundamental to evaluate the Clustering Accuracy. Similarly to the Accuracy metric described for classification tasks in equation 2.1, the Clustering Accuracy compares the clustering assignments (so the corresponding predicted labels) with the ground truth labels. This is an external clustering metric as it ignores the compactness and cohesion of clusters but focuses on the resulting labels. Of course, the higher the clustering accuracy and the better the quality of clustering.

Normalized Mutual Information
The *NMI* score is an external metric for clustering problems based on the concept of entropy. One advantage of using NMI is related to the normalization: it allows to measure and compare this metric between different clusterings that have different numbers of resulting clusters. NMI can also be useful to measure the degree of agreement between two different clustering assignment strategies when ground truth labels are not available. It estimates the reduction in the entropy of class labels that we get if we know the cluster labels.

If Y is the set of class labels, C the one of clustering labels, $H(Y)$ is the entropy function calculated on the ground truth labels and $H(Y|C)$ is the entropy of the class labels within each cluster, then NMI is defined as:

$$NMI(Y, C) = \frac{H(Y) - H(Y|C)}{\frac{H(Y)+H(C)}{2}} \tag{2.9}$$

The value at the numerator gives the reduction of uncertainty in Y when C is observed. The higher the NMI score, the better the clustering quality.

Silhouette score
The Silhouette score is an internal measure often applied in the context of fully unsupervised clustering. It measures how similar an object is to its own cluster (degree of cohesion) compared to the other clusters (degree of separation). Thus, this metric balances the level of separation between the different clusters and the cohesion (compactness) within the same cluster. A high value of Silhouette indicates that the sample is well associated with the cluster and weakly associated with the neighbouring clusters. The goal is to have a high score so as to define the clustering configuration successful. Scores range from -1 to $+1$.

If a_i is the average distance between point i and all the other points in the

cluster, b_i the smallest average distance of i to all the points in any other cluster (average distance from the closest cluster), then the score of item i is defined as follows:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{2.10}$$

In this section, we explained the main concepts related to the clustering evaluation by describing the main metrics that are applied in the experimental setting. More information about clustering, alternative techniques, and their limitations can be found in [49].

2.6 Transfer Learning

TL is a research area that focuses on transferring the knowledge gained while solving a source task to a specific target task. For instance, the knowledge acquired by a classifier that recognizes objects can be transferred to a classifier that recognizes faces. The idea is similar to the learning behaviour of humans, where a concept can be learnt as an extension or a specialisation of a similar concept. In the case of human beings, the transfer is by definition part of the learning behaviour [50].

In the context of DL, the importance of TL is related to the fact that most of the models that learn complex functions need a large number of labeled samples but accessing labeled data is not always feasible.

2.6.1 Definition and main concepts

According to Pan et al. [10], TL can be defined through the concepts of *domain* and *task*. A domain D is defined by a feature space X and a probability distribution $P(\mathbf{x})$, where $\mathbf{x}$ is a learning sample. Hence, given the domain $D = \{X, P(\mathbf{x})\}$, it is possible to define a task as a prediction function $f(\cdot)$ and a label space Y: $T = \{Y, f(\cdot)\}$. Given a source domain D_S, a target domain D_T, a source learning task T_S and a destination learning task T_D, the TL framework improves the learning of the target predictive function $f_T(\cdot)$ by leveraging the knowledge from D_S and T_S.
TL can be divided into three main areas: inductive methods, transductive methods, and unsupervised methods.

- *Inductive TL*: in this context, the source domain and the target domain are the same while the target task differs from the source task. If a

large dataset of labeled data is available in the source domain, the inductive setting becomes similar to multitask learning, even if the two tasks are not learnt simultaneously. On the other hand, if no labeled data are available in the source domain, the setting becomes similar to self-taught learning [51], where the unlabeled data are used to construct higher-level features through sparsity constraints.

- *Transductive TL*: in this scenario, the source and target domains are different but related while the source and the target tasks are the same. No labeled data are available in the target domain while a large amount of labeled data is available in the source domain.

- *Unsupervised TL*: this is a learning setting similar to the inductive scenario but the focus is on unsupervised tasks in the target domain. In this case, the source and the target domains are the same while the source and the target tasks are different but related. This method is usually applied to unsupervised tasks such as clustering and dimensionality reduction, where no labeled data are available.

The most common applications of the TL framework for DL are designed to learn meaningful representations from unlabeled data, or to fine-tune a model pre-trained on the source task to solve the target task. Unsupervised pre-training consists of pre-training a network on unlabeled data and transferring that knowledge to a supervised task [6]. The unsupervised pre-training paradigm can be applied through AE networks, as during the unsupervised training they allow to pre-train the encoder model. The pre-trained encoder network then can be fine-tuned to solve the target supervised task.

2.6.2 Advantages and Disadvantages

The application of TL has both advantages and disadvantages as the final success depends on the relation between the source and the target tasks, as well as the degree of similarity between the two domains. In particular, it finds several applications in learning scenarios where acquiring large datasets of training samples is not feasible and the data labeling process is not sustainable, either from an economical point of view or in terms of data confidentiality.

TL has been shown to be successful while dealing with data with high dimensionality (e.g. images) thanks to the unsupervised pre-training with deep AE networks [52]. The main advantage of these techniques is that they allow learning more robust models, with a higher degree of generalization, which

can achieve better predictive performance. Furthermore, as explained above, it can help in solving the problem of having a small amount of labeled data and reduce the training time of complex models. The final success depends on the hypothesis behind the chosen learning paradigm, as the more the domains and the tasks are related, the better the final results [53].

On the other hand, it may happen that the implementation of TL has a negative impact and it could either not lead to any improvement, or worsen the performance on the target task. A negative transfer may happen if the source and the target tasks are not related enough, the two domains are too different or, from an architectural point of view, the chosen learning network is not suited for connecting the two tasks. A simple example from the area of image classification can help in clarifying the previous statements. For instance, if the image samples in the target and in the source domains have a size that differs of more than one order of magnitude, then it may be difficult to transfer the learnt convolutional filters from one task to the other one because the regions of the input they refer to are too far from each other. In that case, the TL framework could not have a positive impact.

2.7 Semi-Supervised Learning

SSL is a learning approach that combines both labeled and unlabeled data. Because of the known limitations of the data labeling process, SSL methods are defined to jointly learn from a small amount of labeled data and a large amount of unlabeled training data. For this reason, SSL is in the intersection between Supervised Learning and Unsupervised Learning. The key idea behind this framework is that the unlabeled data can improve the predictive performance if used in conjunction with the available labeled data. Like TL, SSL tries to take inspiration from the learning paradigms typical of human beings [54].

2.7.1 Definition and main concepts

It is meaningful to formalize the SSL setting and explain the main assumptions and methods. Given a set on labeled examples $x_1, ..., x_l \in X$, the corresponding labels $y_1, ..., y_l \in Y$ and a set of unlabeled examples $x_{1+1}, ..., x_{l+u} \in X$. In the context of classification problems, if the information in X are combined then it is possible to outperform the classification accuracy that can be obtained by a purely supervised framework that discards the unlabeled data [55].
Semi-Supervised Learning can be applied only if the following assumptions are satisfied by the input data.

- *Smoothness assumption* requires that points that are candidates to share a label are close to each other in the chosen feature space. This is an important property so as to benefit from the unlabeled data and being able to favour simple decision boundaries while training the final classifier. Therefore, it is reasonable to require that if two points are close in the feature space, then the corresponding labels are close in the label space.

- *Cluster assumption* states that if some points are in the same cluster, then the points are candidates to share a label. This does not imply that each class corresponds to a single cluster, but it means that it is not likely to observe objects of two distinct classes in the same cluster. From a decision boundary point of view, this implies that a boundary should lie in a low-density region, so the boundary could not cut a cluster.

- *Manifold assumption* says that high dimensional inputs lie on a low-dimensional manifold and that an efficient feature representation can improve the Semi-Supervised task. If the data can lie on a low dimensional space, then the learning algorithm can work in a feature space with a lower dimensionality and reduce the curse of dimensionality.

SSL methods evolved during the years and a complete overview of the techniques is provided in [55].
We focus on the description of graph-based methods and clustering-based methods, as they are related to the topic of this thesis.

- *Graph-based* SSL describes the data as a graph where each input sample is represented by a node and the similarities between samples are represented by edges. Graph methods formalize the learning problem through a Laplacian matrix as described for Spectral Clustering in Section 2.5.1. The learning phase consists in predicting labels for the unlabeled data as for the information propagation paradigm, so this problem can be seen as an extension of clustering because of the nature of the Spectral Clustering algorithm.

- *Clustering-based* SSL is similar to graph-based methods but it solves the problem through the clustering technique. The fundamental idea is to assign the input samples to clusters and use such information either to propagate the labels between the points belonging to the same cluster or to generate extra features. For example, one common approach is to use the cluster assignments or the cluster centroids as extra information

to solve the target supervised task because they could be informative features. This second case of SSL is similar to feature learning and feature enrichment scenarios.

2.7.2 Advantages and disadvantages

Similarly to the TL framework, SSL cannot be always applied and it may be necessary to consider both its advantages and disadvantages. By considering the techniques summarized in [55], one may observe that if all the hypotheses previously described are satisfied, then the framework can be beneficial in case both labeled and unlabeled data are provided. However, it is necessary to choose supervised and unsupervised tasks that can be associated and successfully work together. For instance, the clustering method is often selected to handle the unlabeled examples, but it is not guaranteed to achieve good results on high dimensional data because the clustering performance could not be satisfactory in case of non-informative feature representation. In that case, neither label propagation based on graphs (or clusters), nor the feature enrichment method could help in increasing the final predictive performance. Hence, if the SSL method is not tailored to the learning setting and all the hypotheses are not verified, the results can be unsatisfactory. In that case, the unlabeled data could even decrease the accuracy of the solution with respect to the results that could be obtained by only considering the labeled samples.

On the other hand, the main advantage of a successful application of SSL is that the predictive performance of the final supervised model can increase. More labeled data would be made available or a better model could be learnt thanks to an efficient feature representation. An interesting direction of research is about combining the two frameworks to reduce the risks related to their limitations and investigate the impact of TL on clustering-based methods for SSL.

Chapter 3

Related work

If you can't explain it simply, you don't understand it well enough.

Albert Einstein

The chapter presents the works in the literature about unsupervised pre-training, disentangled feature learning, and clustering methods based on DL. Section 3.1 focuses on TL and unsupervised pre-training with AEs. Section 3.2 presents the literature about the β-VAE for learning a disentangled data representation from unlabeled samples. Section 3.3 explains previous researches about cyclical annealing for pre-training. Section 3.4 dive deeps into state-of-the-art methods for clustering high dimensional data. In particular, the discussion is focused on the methods that work on the quality of the data representation. Finally, Section 3.4.4 discusses works in the literature related to clustering in the Semi-Supervised setting.

3.1 Transfer Learning through pre-training

The pre-training of deep ANNs is considered a fundamental technique, both in the research field and in the industry, to improve the predictive performance of the learning models. An extension of this state-of-the-art paradigm involves a fine-tuning phase on the target data after the initial pre-training on the source data [56].

3.1.1 Pre-training in Neural Networks

LeCun et al. [57] explain that learning good features through pre-training allows reaching excellent results on several tasks in the area of CV such as im-

age captioning, image segmentation, and image classification. However, it is known that experiments on pre-training may not be successful and results may change depending on the given task. Although improvements were reached for object detection and image classification, they are often small and do not scale widely by changing the size of the pre-training dataset.

He et al. [58] argue that pre-training does not automatically improve the regularization of the learner but that it can speed up the model convergence. However, a standard training from a generic initialization could have a total training time that at the end is comparable to the time required by the "pre-train and fine-tune" paradigm. Of course, pre-training is not a useless step while working on DL projects but it needs to be carefully evaluated and adapted to the learning setting. One key observation is that it requires pre-train the learner on some extra samples which obviously need to be collected, analyzed, and, in the case of a Supervised Learning problem, labeled. Moreover, it is worth remembering that the "pre-train and fine-tune" framework applied to classifiers is the simplest form of TL. It may be beneficial only if the pre-train and the fine-tune tasks are related. Therefore, pre-training could not improve the final predictive performance or it could generate a negligible improvement both in terms of predictive performance and training stability.

Hendrycks et al. [59] demonstrate that the pre-training paradigm may increase the model robustness and that it needs to be carefully designed depending on the learning problem and the given dataset. For instance, it could not increase the final performance on classification tasks but it could improve the overall quality of some model components. In the case of image classification problems that involve multiple classes, pre-training could not increase the average metrics but it could improve them only for a specific class. Thus, it is worth to consider the paradigm and adapt it depending on the learning scenario as it may be beneficial for small or unbalanced datasets.

3.1.2 Unsupervised pre-training with Autoencoders

Pre-training can be beneficial when the learning task to solve is not supported by a large and informative dataset of samples. In this scenario, it is possible to apply TL either through the "pre-train and fine-tune" paradigm as explained in the previous section, or through the unsupervised training of deep AEs as proposed by Erhan et al. [6]. The first paradigm can be successful if the source and the target datasets are related and labeled data are made available also during the pre-training phase.

However, accessing large amounts of labeled samples may not be effortless in CV, so a good solution is the unsupervised pre-training with AEs.

It is possible to train an AE network to reconstruct the input images in an unsupervised way and define this task as the source task of the TL framework. Bengio et al. [7] state that while learning how to retain the information necessary to reconstruct the image, the unsupervised pre-training process forces the network to learn which are the informative patterns in the data, and this may be helpful to support a supervised task. One could pre-train the encoder as part of the AE on the unsupervised task and then fine-tune the convolutional layers of the encoder network on the supervised task [60]. The main advantage of this method is that it does not require another dataset of labeled examples to solve the source task and that the pre-train step may be executed on unlabeled data. Hence, during the pre-training it is possible to learn from the available unlabeled data, then use the labeled samples for the final fine-tuning phase.

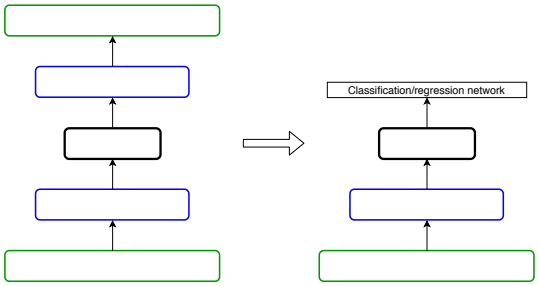


Figure 3.1: The encoder network is fine-tuned on the target task

TL is successful if during the unsupervised pre-training the network is forced to retain the most relevant information in the data to extract disentangled factors of variation. This facilitates learning tasks such as regression and classification. Figure 3.1 shows how TL can be applied through the unsupervised training of deep AEs.

Shu et al. [61] suggest that improvements in the direction of disentanglement of the latent space offer a high degree of separation of the latent data representation into dimensions corresponding to multiple independent factors of variation.

Paine et al. [62] analyze the unsupervised pre-training of CNNs and denote that a successful method is to first train the AE to reconstruct the input images, then fine-tune the encoder network on the target task. A similar approach investigated by Bengio et al. [5] consists of doing a greedy layer-wise pre-training by forcing each layer of the decoder network to reconstruct the features

generated by the corresponding layer in the encoder network. In this scenario the layers are first trained in pairs, so layer M is trained to reconstruct the input of layer 0, layer $M-1$ to reconstruct the input of layer 1, and so on. Then, the layers are composed to form the AE architecture, the whole network is trained and finally the encoder is fine-tuned on the target task. The main advantage of TL through AEs is that it can find high-level features on the source task and transfer them to the target problem [63].

3.2 β-Variational Autoencoder

Lake et al. [2] state that learning a latent representation of disentangled features allows to improve the performance of the existing methods, as well as develop a way of learning which may be more similar to human reasoning.

VAEs are generative methods that create new data instances that look like they are sampled from the original inputs. During the process, the network is forced to learn an efficient representation of the data, so the unsupervised training of VAEs can be used for TL. The main advantage is that the process of generating new images increases the generalization ability of the encoder network and reduces the risk of overfitting. A more powerful AE architecture is the β-VAE [14]. It was proposed by Google DeepMind in 2017 as an extension of the VAE.

3.2.1 A new generative method

It is known that the data representation affects the success of each ML technique and that a complex data representation may increase the difficulty of solving the learning problem. Bengio et al. [64] demonstrate that different data representations can merge and hide different explanatory factors. The more the learnt representative factors are entangled, the more difficult it is detecting patterns in the data and solve the learning task.

The β-VAE is a new generative model designed to learn disentangled representations of the generative factors in the inputs. It is possible to define a fully disentangled representation as a latent representation where each latent factor only responds to changes in a single generative factor. A fully disentangled representation simplifies the final learning task because it allows feeding the learner with meaningful features that bring discriminative information [64]. This can be beneficial in scenarios that require learning from a small dataset, transferring knowledge from one task to another one, or where unlabeled data

can be used for representation learning. The β-VAE is a state-of-the-art framework for the visual learning of meaningful latent representations through an unsupervised training procedure. In the VAE the loss function is defined as the sum of a *reconstruction loss* and a *latent loss*. The new solution introduces a β parameter that balances the sum of the two losses to better manage the trade-off between the independence constraints (controlled by the latent loss), and the reconstruction quality (controlled by the reconstruction loss). Sikka et al. [65] suggest that a properly tuned β-VAE may outperform the traditional VAE, as well as other advanced generative approaches for learning disentangled features.

3.2.2 Definition and main concepts

The β-VAE is a deep generative AE which learns an efficient data representation through an unsupervised training process. It was designed by Higgins et al. [14] to capture the information in the original data generative factors through a set of disentangled latent factors. The goal is to learn the joint distribution of the data $\mathbf{x}$, as well as the set of the generative latent factors $\mathbf{z}$, in order to reconstruct the input data $\mathbf{x}$: $p(\mathbf{x}|\mathbf{z})$. As the objective is to learn a compressed representation, one could desire to reach a latent representation whose dimensionality is as close as possible to the number of generative factors in the data. This would allow to infer a meaning for each latent feature and avoid the rising of features that merge information coming from different patterns in the original input samples.

The traditional reconstruction quality of AEs is measured through a dedicated loss function and it is not directly controlled by any parameters. In the β-VAE, the β parameter directly controls the latent loss: the higher β, the higher the degree of disentanglement since the KL divergence loss has larger importance during the optimization process. In order to favour the rise of disentanglement in the latent distribution $q(\mathbf{z}|\mathbf{x})$, β constraints the training by trying to match the latent distribution to a prior distribution $p(\mathbf{z}) \sim N(\mathbf{0}; \mathbf{I})$, which corresponds to the isotropic unit Gaussian. The idea of matching these distributions through the KL divergence allows to directly influence the latent information bottleneck of the network, and consequently force the learning of independent features as the increased pressure on the bottleneck makes them more factorised. This formulation is similar to the VAE, but the main change is due to the parameter β that allows controlling the weights of the two losses during the optimization process. In fact, a high β increases the degree of learning because extra pressure is added to the bottleneck than in the traditional VAE.

In other words, this encourages disentanglement during learning because the objective is more focused on matching statistical distributions than finding a perfect image reconstruction.
The structure of the loss function of the β-VAE can be summarized as follows:

$$L = L_{Rec} + \beta L_{KL} \tag{3.1}$$

Similarly to VAEs, the most common reconstruction losses are the MSE and the Binary Cross-entropy, while the KL divergence is used to compare the prior Gaussian distribution with the learnt latent distribution. If β is equals to 1, then the framework corresponds to the VAE. If β is greater than 1, then the network is forced to learn a more disentangled representation as asserted in [14]. It is necessary to balance the trade-off between the two objectives so as to control the quality of the reconstruction and the quality of disentanglement. A good reconstruction is achieved when the information retention and the latent channel capacity are balanced through a properly tuned β.

3.2.3 Unsupervised learning of disentanglement

DL methods are still far from the generality of human intelligence and their results strongly depend on the nature of the represented information. The quality of the features impacts the final predictive performance of the learning model and disentangled features facilitate the learning task as stated by Liu et al. [66]. A fully disentangled representation can be defined as a latent representation where each latent factor only responds to changes in a single generative factor. Burgess et al. [67] assert that a highly disentangled representation is a factorised and interpretable representation, where each latent unit encodes a single independent generative source of variation in the data. In other words, a disentangled representation decouples complex directions of variation of the real world and describe them through a set of independent latent units. Chen et al. [68] investigate the effect of a well-tuned β parameter and discover that it simplifies the unsupervised learning of disentanglement while retaining the information needed for the reconstruction of the original images.

Figure 3.2 shows how each latent unit models the generative factors of a dataset containing images of chairs. It is easy to note that the factor described in the first row corresponds to the azimuth. Since the Chairs dataset [69] contains high dimensional data (objects in the 3D space), it is not immediate to understand all the latent directions of variation as the decoupling is not complete in all the cases. However, good overall quality is reached in the reconstructions, and traversal of a latent unit corresponds to isolated changes in one

or few generative features. For example, a good disentanglement is achieved for the azimuth (first row), the size (second row), the back style (fourth row), and the leg style (fifth row).

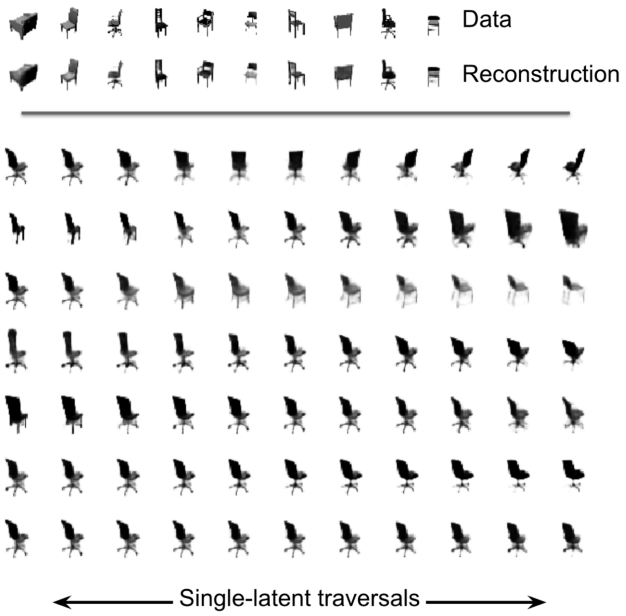


Figure 3.2: The β-VAE controls the latent factors of disentanglement. The images are generated by traversing a latent dimension while keeping the remaining dimensions fixed. Figure from [67].

3.2.4 Disentanglement in β-VAE and InfoGAN

The β-VAE is an unsupervised generative paradigm as it is capable of generating new images that resemble the original unlabeled images. The main drawback of both the VAE and the β-VAE is that they could generate blurry images, whose quality is lower than in the original samples. On the other hand, a *Generative Adversarial Network (GAN)* is capable of generating new images with high resolution. The GAN framework is based on a *generator* network and a *discriminator* network adversarially trained together. The generator produces images that look like the training data so as to trick the discriminator. By contrast, the discriminator tries to distinguish the real images from the fake ones. The main weakness of GANs is that they suffer from mode collapse and the training may be difficult because of instabilities due to the adversarial framework. The complete formulation is provided by Goodfellow et al. [70].

To generate disentangled features the β-VAE is built on top of the VAE, while the *Information Maximizing Generative Adversarial Network (InfoGAN)* framework is designed on top of the GAN paradigm. Chen et al. [71] introduce InfoGAN as a generative adversarial model that learns a disentangled representation through an unsupervised training process. It was demonstrated that it can discover complex concepts such as hairstyle, presence or absence of sunglasses and facial emotions on datasets showing faces. The goal of GANs is learning a generator distribution $p_G(\mathbf{x})$ that corresponds to the original data distribution $p_{data}(\mathbf{x})$ by transforming a random noise vector $\mathbf{z} \sim p_{noise}(\mathbf{z})$ into a new generated image $g(\mathbf{z})$. Then, the generator is trained against the discriminator which tries to distinguish between the samples from the true distribution and the ones coming from the generator. The InfoGAN framework also maximizes the mutual information between a portion of the random noise vector and the observations. GANs use a continuous noise vector $\mathbf{z}$ which could be used by the generator in an entangled way, while the InfoGAN decouples the random noise vector into two parts: a source of incompressible noise $\mathbf{z}'$, similar to the original random noise vector, and a set of latent codes c, that captures the direction of variation in the original data distribution. In order to force the generator to respect the latent factors and increase the disentanglement in the newly generated samples, the training objective includes the maximization of the mutual information between the latent codes c and the generator distribution $g(\mathbf{z}', c)$ [71]. However, a β-VAE with properly tuned β parameter can outperform InfoGAN for disentangled factor learning.

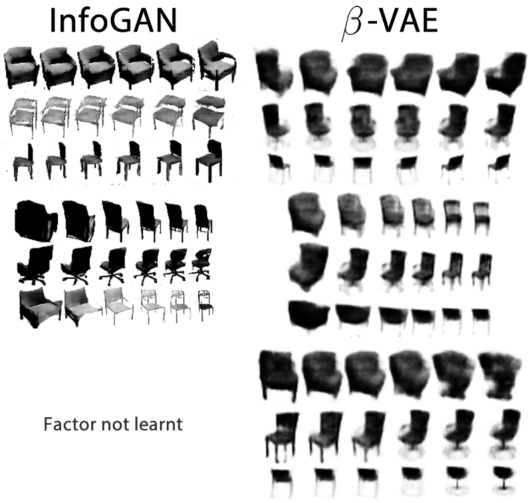


Figure 3.3: Images of chairs generated with the InfoGAN paradigm compared with those generated by the β-VAE. Figure from [14].

Figure 3.3 demonstrates that both β-VAE and InfoGAN learn a disentangled representation. The first block of images shows that the azimuth factor was separated, while the second block shows that the width factor was isolated. However, the InfoGAN is not able to learn the factor related to the leg style. On the other hand, that factor is not completely isolated by the β-VAE but it is highlighted. Another advantage of the β-VAE is that it does not suffer from training instability and the quality of the reconstructions can be indirectly controlled through β, thus reducing the risk of creating blurry images.

3.3 Simulated Annealing and Autoencoders

Simulated Annealing (SA) comes from the field of metallurgy, where it was initially applied to create controlled heating and cooling cycles. It increases the size of the crystals in materials while reducing the frequency of imperfections. In the area of optimization, it allows approximating the global optimum of a specific objective function by reducing the risk of stopping the optimization process in a local optimum. The idea of slow coolings is implemented through a slow decrease in the probability of exploring a worse solution as the number of iterations increases.

The temperature parameter T influences the probability of accepting a temporary bad solution. It decreases from a value greater than 0 to 0. If the new candidate solution improves the current optimal solution, then it is accepted. Otherwise, the algorithm can move to the new (worse) candidate with a probability depending on the T parameter, as well as the loss value associated with the new solution. The annealing cycle forces the temperature T to be higher at the beginning as it is more likely that a local optimum is found, while T is decreased during the time because the convergence towards the global optimum is likely to happen and jumps towards worse solutions are not useful anymore. The idea behind SA is that temporarily accepting a worse solution allows to completely explore the solution space and find the global optimum [72].

3.3.1 An application to NLP for pre-training

Fu et al. [73] demonstrate that SA can be beneficial for the unsupervised training of autoregressive β-VAEs. One could apply annealing to the β parameter that balances the sum of the reconstruction term and the KL regularizer. During training, the process of increasing β multiple times helps in learning more informative latent codes by leveraging the representations learnt during the previous cycles. The authors analyze the benefits of SA on a set of NLP tasks.

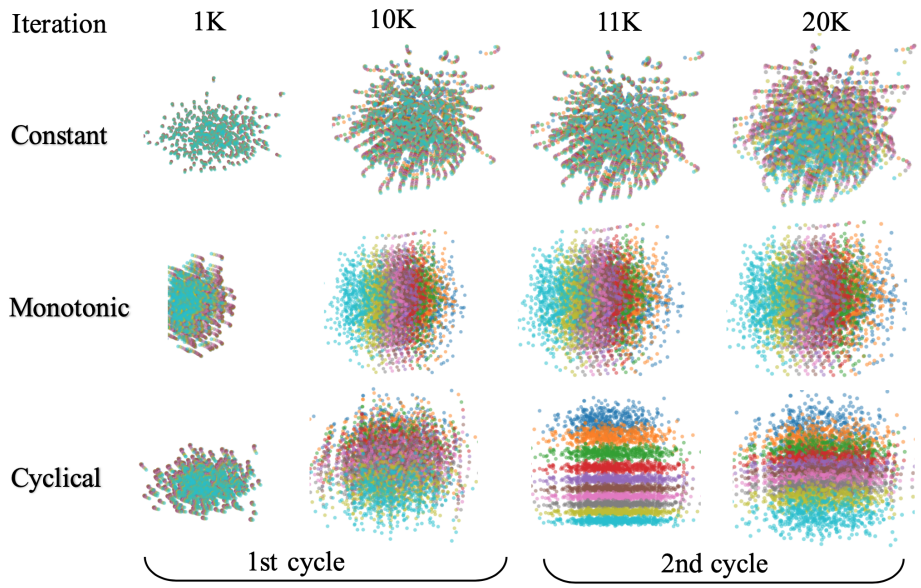


Figure 3.4: The disentangled latent spaces generated during training. Three different annealing methods are compared. Image from [73].

Figure 3.4 shows how disentanglement evolves during training if cyclical annealing is applied. Also, a monotonic annealing schedule increases the degree of disentanglement, but less informative factors are discovered. On the other hand, a constant schedule (no annealing) is not capable of finding independent features because the latent codes are heavily mixed during the whole training process. During the first 10k iterations, the monotonic schedule and the cyclical schedule have similar behaviour but then, as soon as the second cycle starts, better clusters are achieved through cyclical annealing.
Hence, each cycle leverages the latent features learnt before as a good restart for the training in the next cycle.

Fu et al. [73] also discover the main limitation created by a fixed β: the vanishing of the KL regularization. As a consequence, the decoder network could tend to ignore the disentangled latent variables and it could produce a posterior distribution too similar to the given Gaussian prior. By gradually increasing and decreasing the value of the β parameter, it is possible to balance the importance of the reconstruction quality and the disentanglement.

3.4 Autoencoders applied to Clustering

Deep Clustering is a set of clustering techniques built upon ANNs for learning an efficient data representation suitable for the clustering task. Clustering is trivial to solve in low dimensional feature spaces because there are not many directions of variance. On the other hand, if an informative data representation is not learnt for high dimensional data, clustering is difficult to solve.

Deep Clustering was initially proposed to support the clustering of images because they have a high dimensionality, then it was extended to solve other clustering problems such as clustering of biological data, speech separation, and unsupervised feature learning [74, 75, 76].

3.4.1 Deep Clustering

One approach could be to treat feature extraction and clustering separately, but Yang et al. [77] observe that the joint optimization of the two tasks may improve their performance. Therefore, the most recent Deep Clustering methods aim either at optimizing the clustering process starting from the learnt data representation or jointly optimize the feature representation and the clustering assignments. In the second case, the loss function is composed of two terms, known as reconstruction loss L_r and clustering loss L_c. They are combined through a parameter $\lambda \in [0, 1]$. The structure of the loss is as follows:

$$L = \lambda L_c + (1 - \lambda) L_r \tag{3.2}$$

The objective of the reconstruction loss is to learn a meaningful feature representation that encourages the network to avoid trivial solutions (e.g. assigning all the points to one cluster) [11]. The loss L_r is the reconstruction loss of the AE pre-trained with the unsupervised process, and fine-tuned during the joint optimization. The loss L_c measures the quality of the clustering assignments. State-of-the-art frameworks are AE-based Deep Clustering [78] and VAE-based Deep Clustering [79].

AE-based Deep Clustering makes use of non generative AE architectures, such as Sparse AEs or AEs. The network is trained on unlabeled data to learn a sparse feature representation, then it is used to solve the clustering task. It is possible to jointly fine-tune the encoder network on the clustering task or manage the two tasks as different training phases.

VAE-based Deep Clustering makes use of a VAE during the unsupervised training of the AE network to leverage the generative approach. The VAE during training encourages the latent features to follow a probabilistic prior

distribution, usually corresponding to the isotropic unit Gaussian. However, Min et al. [11] state that in the context of clustering it would be better to choose a prior distribution corresponding to the cluster distribution. As it is difficult to know a good prior, the common choice is to use a mixture of Gaussians as priors or to keep the traditional isotropic unit Gaussian and focus the experimental effort in finding a good network architecture.

3.4.2 Deep Embedded Clustering

Deep Embedded Clustering (DEC) is a method proposed by Xie et al. [12] in 2016, which simultaneously learns feature representation and cluster assignments using deep neural networks. It learns a mapping from the input space X to a low dimensional feature space Z in which it solves the clustering task. The idea is to solve the clustering assignment problem while improving the underlying feature representation.

Clusters are iteratively refined through an auxiliary target distribution obtained from the current predicted clustering assignments. This process improves both the clustering as well as the data representation. Minimizing the KL divergence between a given distribution and the feature distribution may be used for dimensionality reduction. In the context of DEC, the KL divergence is used to minimize the difference between the predicted centroid-based probability distribution and an auxiliary target distribution, so as to improve the quality of clustering [12]. DEC is composed of two phases.

1. Initialization of the parameters of the encoder network through the unsupervised training of the AE architecture.

2. Clustering optimization: the KL divergence between the target distribution and the predicted centroids distribution is minimized. During this step, each embedded point is iteratively assigned to a cluster, then the KL loss is computed and the network is updated. This step is repeated until convergence is reached.

The predicted clusters soft-assignments are computed through a probabilistic equation, where z_i is the embedding corresponding to sample x_i, α is the degrees of freedom of the Student's t-distribution ($\alpha = 1$ is default) and μ_j is the centroid of cluster j. The equation is defined as follows:

$$q_{ij} = \frac{(1 + ||z_i - \mu_j||^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1 + ||z_i - \mu_{j'}||^2/\alpha)^{-\frac{\alpha+1}{2}}} \tag{3.3}$$

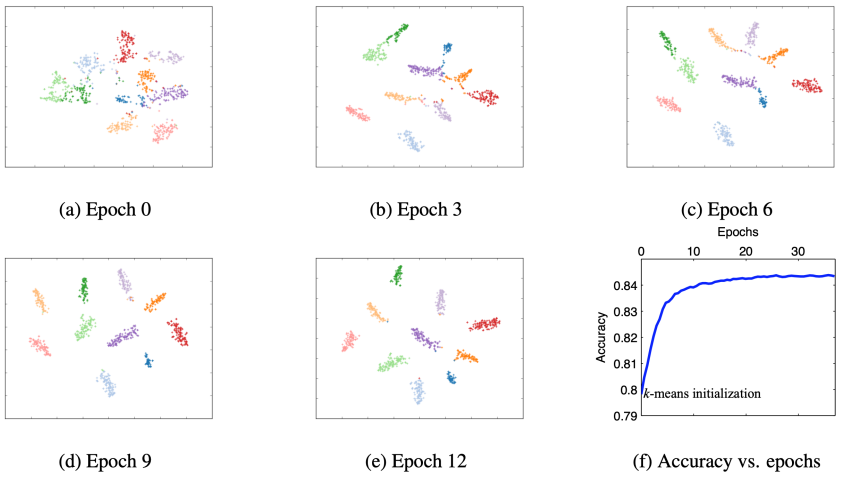
Since q_{ij} is a probabilistic assignment, it is necessary to define probabilistic targets that give a higher importance to points assigned with high confidence. If p_{ij} is the soft target and $f_j = \sum_i q_{ij}$ the cluster frequencies, then the target can be defined as:

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}} \tag{3.4}$$

The final objective of matching the predicted soft assignments with the given target is achieved by minimizing the KL loss between the two distributions:

$$L_{KL} = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{3.5}$$

The authors suggest to initialize the DEC framework via the pre-trained AE. The learnt representation generates informative features, which facilitate the successive clustering task. Guo et al. [80] demonstrate that applying data augmentation during the unsupervised pre-training of the architecture substantially increases the clustering performance.



(a) Epoch 0 (b) Epoch 3 (c) Epoch 6

(d) Epoch 9 (e) Epoch 12 (f) Accuracy vs. epochs

Figure 3.5: The quality of the feature representation, as well as the clustering accuracy, improve on the MNIST digits as the training of DEC proceeds [12].

3.4.3 Jointly optimizing clustering and reconstruction

The weakness of DEC is that after the initial AE training the reconstruction loss is not optimized anymore, as the optimization objective is only defined by the clustering loss described in equation 3.5. Guo et al. [13] propose an

improved version of DEC. The authors suggest to jointly optimize the reconstruction loss and the clustering loss after the initial pre-training of the network. In fact, the risk of minimizing only the clustering loss is that it could lead to the extraction of non-informative features, with a negative impact on the overall clustering performance.

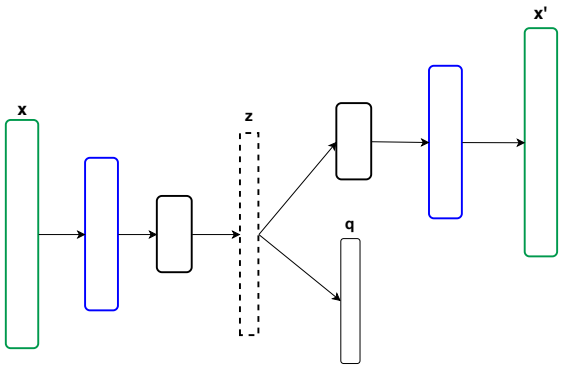


Figure 3.6: A simplified schema which describes the logical structure for the joint optimization of the clustering and the reconstruction tasks.

The fundamental assumption behind this new framework is that both the clustering loss and local pattern preservation are important for the success of Deep Clustering. Thus, it is necessary to include the complete AE in the DEC paradigm.

The most critical hyperparameter associated to the approach is the weight λ used to balance the representation loss and the clustering loss. It needs to be properly tuned so as to maximize the chosen clustering metrics, as well as obtain well-separated clusters. Empirical experiments demonstrate that joint optimization is important for the success of clustering. A promising direction of research is about improving the disentanglement for clustering tasks and embed prior knowledge through TL [13].

3.4.4 Clustering for Semi-Supervised Learning

In the Semi-Supervised setting, it is possible to access both labeled and unlabeled samples. Clustering-based SSL is mainly based either on label propagation or on feature enrichment, by leveraging the information coming from the clustering assignments.

Oliver et al. [15] demonstrate that the Semi-Supervised framework suffers when the unlabeled data and the labeled data come from different distributions. Guo et al. [13] dive deep into the impact of dimensionality and suggest

that the research effort in the area of clustering must be focused on improving the data representation to build high-quality clusters. Zhuang et al. [8] remark the importance of a good latent space when they propose Local Label Propagation, an approach based on the local geometry of the embeddings. The parameters of the neural networks are trained to optimize both the pseudo label categorization as well as the clustering quality. Hu et al. [81] work with neural networks to model the non-linearity of data in the Semi-Supervised setting. The authors propose a data augmentation framework that encourages the representation of the augmented data to be similar to the original data. The solution improves the performance of many unsupervised learning tasks, in particular clustering.

Shukla et al. [82] take inspiration from the work of Xie et al. [12] to define a Semi-Supervised clustering approach. They use neural networks to find suitable clusters. The abundant unlabeled data are augmented with pairwise constraints generated from a few labeled samples. Pairwise constraints specify whether a pair of data samples belong to the same class or not. Ren et al. [83] use pairwise constraints as a form of prior knowledge to improve the original formulation of DEC. The constraints are used during the feature learning process: samples belonging to the same class are forced to be close to each other, while samples from different classes are enforced to be far away in the learnt feature space.

Finally, Peikari et al. [84] propose a cluster-then-label SSL method. It finds clusters of points forming high-density regions. Then, the clustering analysis provides information to a supervised Support Vector Machine. The clusters allow discovering how much the unlabeled points are inclined toward each labeled point.

Chapter 4

Methods

My powers are ordinary. Only my application brings me success.

Isaac Newton

This chapter explains the path followed during the research and the new proposed approaches. Section 4.1 presents the models considered for analyzing the effect of the β-VAE on TL, as well as the main architectural choices. In particular, Section 4.1.3 and Section 4.1.4 explain respectively the architecture of the β-VAE and the novel training strategy based on annealing. Section 4.2 discusses the introduction of the β-VAE in the training process of Deep Clustering. In particular, Section 4.2.4 describes the new learning pipeline designed for clustering in the Semi-Supervised setting.

The first goal of the research is the evaluation of unsupervised pre-training with AEs for image classification tasks. Moreover, since clustering methods for SSL depend on the quality of the data representation, it is worth to investigate whether it is possible to improve it thanks to TL.
The research procedure can be summarized as follows:

- Analysis and benchmarking of unsupervised pre-training via state-of-the-art AEs for image classification tasks.
- Introduction of unsupervised disentangled feature learning through the β-VAE with cyclical annealing for Deep Clustering.
- Design of a new Semi-Supervised training pipeline based on disentangled feature learning for clustering in the Semi-Supervised setting.

4.1 β-VAE applied to Transfer Learning

The simplest case of TL consists in fine-tuning a pre-trained state-of-the-art architecture, like ResNet [40] or Inception [39], on the destination task. ImageNet [85] is a popular dataset often used for pre-training. It contains more than 14 million images belonging to more than 20000 categories of the real world such as animals, vegetables, and objects. For research purposes it is possible to benefit from these pre-trained models because they can achieve excellent results, the classes contained in ImageNet are related to those belonging to most of the public datasets and the training cost is reduced. However, if one considers very technical fields like predictive maintenance and precision farming, the framework could not achieve satisfactory results because of a weak relation between the domains. Moreover, in real use cases, large amounts of unlabeled samples are often accessible and they are expected to further improve the final predictive performance if properly involved in the learning paradigm. For these reasons, the direction of the investigation reported in this section is TL through unsupervised pre-training, with a focus on the β-VAE. In addition, the DAE is considered for benchmarking.

The unsupervised pre-training of a network using an AE implies that the source dataset corresponds to the destination dataset, but the source task does not correspond to the destination task. This approach is still based on pre-training, but an external dataset is not involved. Thus, the framework can be applied by using the available data samples and the corresponding label distribution.

4.1.1 Baseline 1: state-of-the-art architectures

It was demonstrated that the investigation of TL through the unsupervised training of AEs is not trivial [5, 6, 7]. The main difficulty is evaluating the benefits obtained thanks to pre-training. It could speed up the model convergence, but a standard training from a generic initialization could have a training time that at the end is comparable to the time required by the "pre-train and fine-tune" paradigm [58]. Therefore, it is necessary to find proper baselines to consider while defining the experimental setting and evaluating the results. Since our goal is to understand whether it is possible to improve the predictive performance on image classification tasks, we consider as a baseline the predictive accuracy that one may obtain via a state-of-the-art architecture that discards the unlabeled data. In fact, it is not known a priori whether the unlabeled training data could increase the predictive performance, but using them

has a cost equals to the unsupervised pre-training of the network.

As a first relevant baseline, we decide to take inspiration from ResNet, the deep convolutional architecture based on residual connections. The residual connections allow reducing the impact of the vanishing gradient, as the back-propagation algorithm can spread the gradient by skipping some layers. This makes the network suitable for learning complex patterns from images. This architecture is considered state-of-the-art and is one of the most promising in the *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)*. Since we study the predictive performance with variable percentages of labeled data, we propose as an initial baseline a simplified version of ResNet20 v1 with a total of fifteen convolutional layers and six skipping connections for residual learning. This is a valuable benchmark because it is an existing and widely studied approach. Thus, it only needs to be trained on the available labeled data.

4.1.2 Baseline 2: DAE for pre-training

To start the analysis of unsupervised pre-training through AEs for classification tasks, we implement a Convolutional DAE with regularizers. This model is designed to leverage the strengths of CNNs in extracting hierarchical patterns from images. The advantage of an AE that uses convolutional layers is that it can learn a high-level representation of the input images, reduce the dimensionality, and retain the most important patterns in the samples. The random noise is added in the form of dropout layers, each with a specific dropout rate, to randomly deactivate a fraction of the neurons in the layer. In addition, sparsity constraints are added to the network in the form of L1 regularizers and L2 regularizers. This design choice is necessary to increase the degree of generalization and force the network to learn higher-level features.

It is noticeable that a Sparse Convolutional DAE is implemented at the end. Both the level of random noise (defined through the dropout rate) and the sparsity constraints are increased as the layers go deeper. This reduces the risk of overfitting while extracting deep features, as well as forces the network to detect general patterns and locality information. The network bottleneck is not defined through a fully connected layer but with a convolutional layer. This is suggested by cross-validation and helps the network to retain locality patterns from pixels. Figure 4.1 shows the schema of the encoder network. The convention *Conv2D: f-32, k-3, s-1, Reg* indicates a convolutional layer with 32 filters, kernel of size 3, striding of size 1 and the addition of a regularizer.

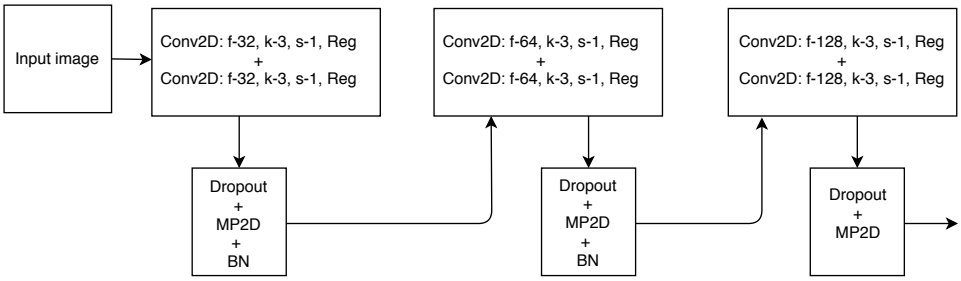


Figure 4.1: Schema of the encoder network used to build the Convolutional DAE with sparsity constraints.

The structure is composed of three macroblocks, each made up of two convolutional layers, connected by intermediate macroblocks containing dropout and max-pooling layers. From an architectural point of view, each convolutional layer applies the elu activation function, images are downsampled using max-pooling and batch normalization is applied to prevent vanishing gradient issues. According to the symmetry principle, the decoder network reflects the structure of the encoder, with a few differences. Thus, in the decoder max pooling is replaced by upsampling and both dropout and regularization are removed, as they are only needed in the encoder network.

During the unsupervised pre-training, the Binary Cross-entropy loss function is optimized, but multiple trials confirmed that there is no difference with the MSE in terms of reconstruction quality. The reconstruction capability of the network is evaluated quantitatively by stopping the training when the loss does not decrease any more. Furthermore, a qualitative analysis is conducted at the end to inspect the quality of the reconstructed images.

4.1.3 β-VAE for unsupervised pre-training

The investigation mainly considers the Convolutional β-VAE in order to asses the impact of a state-of-the-art generative method. It is studied in [14] and [67] to understand its strengths and the main differences with the standard VAE. However, previous studies only involve simple datasets chosen to evaluate disentanglement. To the best of our knowledge, this is the first research investigating the effect of unsupervised pre-training with β-VAEs for TL on image classification tasks.

The Convolutional β-VAE combines the creation of hierarchical features, relevant for the interpretation of images, with the learning of a disentangled latent distribution, which captures the informative directions of variance in the original data. The structure of the network is explained in detail later on. At a

high level, it has a convolutional encoder network, a fully connected bottleneck for the latent representation, and a convolutional decoder network. Since this is a generative method, random noises are not added.

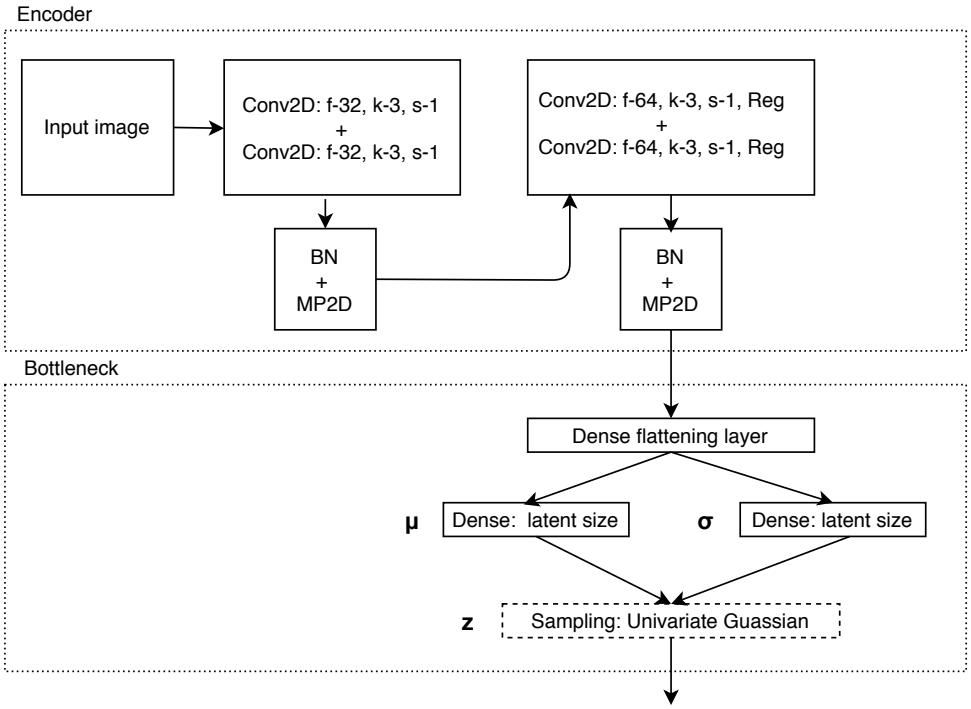


Figure 4.2: The schema shows the Convolutional β-encoder network and the bottleneck for the injection of the univariate Gaussian.

From an architectural point of view, the encoder and the decoder networks are symmetric. Therefore, we explain in detail only the encoder. In the case of a dataset containing RGB images, it takes as input samples of size $32 \times 32 \times 3$. The convolutional network is composed of two macro convolutional blocks with relu as activation function, as shown in Figure 4.2, separated by batch normalization and max pooling. Like in the previous scenario, as the height and width of the images are progressively reduced, the number of filters is incremented. This can be interpreted as a way to mitigate the reduction in information caused by downsampling. Then, the features extracted by the second macroblock are flattened and split into two vectors, each of size equals to the dimension of the latent space, to infer the mean μ and the standard deviation σ of the learnt distribution. These vectors are used to generate new images thanks to the reparameterization trick applied through the univariate Gaussian.

Unlike in the case of the DAE, the convolutional network is simpler. Cross-validation demonstrates that this generative method does not perform better with a deeper network, so the choice is to keep it as simple as possible, to better evaluate the impact of the β parameter on the results. The network is initially trained using the loss function that averages the Binary Cross-entropy (reconstruction loss) and the KL divergence (latent loss) through a fixed β value. The training considers both the reduction of the loss as well as the quality of the reconstructed test images.

4.1.4 Applying annealing to the β-VAE

After studying the effect of the β-VAE with fixed β on TL, we introduce the concept of annealing during training. Although this is not the traditional scenario of SA, we take inspiration from it and design multiple cycles to increase and decrease the β parameter during training.

We think that while training the β-VAE on complex images there is a large amount of information to learn. Hence, annealing could be beneficial to balance the trade-off between reconstruction quality and disentanglement. We propose cycles like the function reported in Figure 4.3.

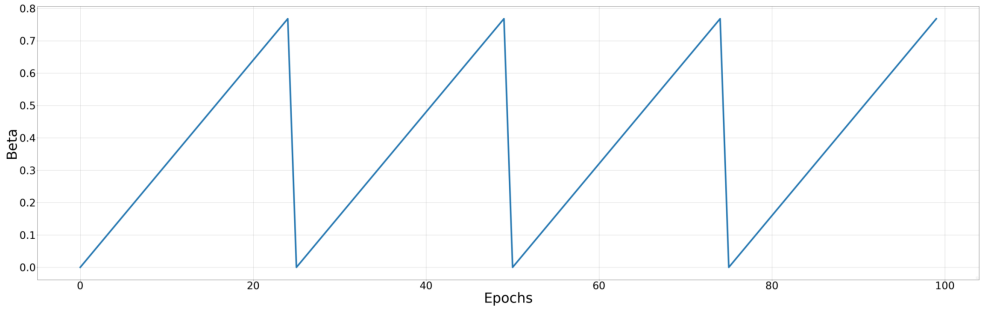


Figure 4.3: Cyclical annealing applied to the β parameter for TL

This is not the traditional application of annealing described in the literature as there is not a concept of worse or better solution during the optimization. Therefore, we take inspiration from the annealing cycles applied in metallurgy. Several cycles of "heating" and "cooling" applied to β may help the network to learn a good disentanglement without the risk of penalizing its reconstruction capabilities. During training the β value is not fixed, it varies depending on the epoch similarly to Figure 4.3. From an architectural point of view, the network is the one reported in Figure 4.2, as we only introduce a new train-

ing paradigm. During training, the focus is also on the reconstruction quality obtained through annealing.

4.1.5 Supervised fine-tuning

The investigation of unsupervised pre-training with AEs requires to apply the "pre-train and fine-tune" paradigm. Hence, after the initial pre-training, it is necessary to execute a supervised fine-tuning phase on the image classification task. The supervised information is injected by removing the decoder network and connecting extra classification layers to the encoder, so as to predict the class corresponding to the input. The new network structure changes depending on the chosen AE:

- *DAE*: the encoder network at the end is connected with three fully connected layers, separated by an intermediate dropout layer. The last layer has ten neurons as we consider ten-classes classification problems.
- β-*VAE*: the initial encoder network, in this case, is simpler, so we connect two convolutional layers followed by a max-pooling layer, each of which has 128 filters, kernel size equals to 3 and stride equals to 1. This allows increasing the ability to extract complex patterns during the supervised fine-tuning. Finally, three fully connected layers are connected like in the aforementioned scenario.

The initialization of the network with the pre-trained weights is expected to improve the quality of the learnt latent space for the classification task.

4.2 Semi-Supervised Deep Clustering

In an experimental environment where a few labeled samples are provided, the learner needs to extract knowledge from the unlabeled data. This may happen through TL or SSL. Clustering-based SSL involves several techniques based on the clustering analysis. The first step always consists of building clusters, then the cluster assignments are used to improve the training on the target supervised problem.

By considering the literature, one may note that the final performance of SSL methods based on clustering depends on the quality of the resulting clusters. Moreover, clustering is affected by the data representation, therefore our direction of research is about improving feature learning for clustering in the Semi-Supervised setting. We focus on Deep Clustering as it is a novel research

field, DL is successful for discovering complex patterns and Deep Clustering is a state-of-the-art technique for clustering high dimensional data such as images. First, we investigate unsupervised disentangled feature learning for clustering through the β-VAE. Then, we study Deep Clustering in the Semi-Supervised scenario and propose a new learning pipeline to improve the quality of clustering.

4.2.1 β-VAE pre-training for clustering

The first step in the process consists of pre-training via the AE network. Similarly to the methodology applied in the TL scenario, we design a β-VAE with only four convolutional layers in the encoder network. As the height and width of the images are progressively reduced, the number of filters is incremented. However, we do not use max pooling for downsampling because we prefer applying strided convolutions. In this way, the network is forced to learn its own spatial downsampling [86].

The β-VAE is built upon a symmetric architecture where the encoder is similar to the one shown in Figure 4.2, except for some minor changes. There are two macroblocks of convolutional layers, separated by one batch normalization layer. Each block consists of two convolutional layers. The first layer in the first macroblock applies 32 filters, the kernel has size equals to 3 and the striding is equals to 1. The second layer is similar, but it applies a striding equals to 2, so this layer downsamples the image. On the other hand, the two layers in the second macroblock apply 64 filters, but the kernel size and the strides reflect those in the first macroblock. Then, the standard structure of the β-VAE is implemented through the fully connected layers used to build the bottleneck, as well as inject the univariate Gaussian. The network is trained with the Binary Cross-entropy as reconstruction loss and the KL divergence as a latent loss. For the unsupervised pre-training, we directly investigate the impact of cyclical annealing, cross-validation demonstrates that a fixed β parameter is less beneficial in this case.

The function reported in Figure 4.4 anneals β between 0 and 1.5. As β reaches values greater than 1, this pre-training forces a strong disentanglement trough the latent bottleneck as expected in [14, 67]. We repeat four annealing cycles and introduce the concepts of period and duty cycle. During a period, corresponding to forty training epochs, the β value is increased for the 75% of the cycle, while for the 25% of the time it is at a fixed value.

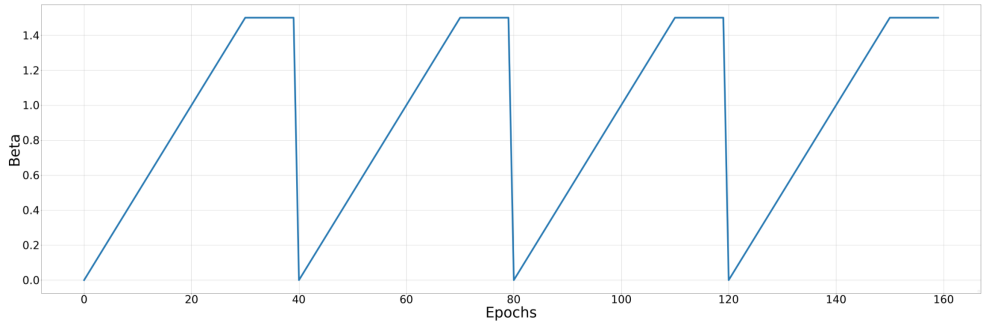


Figure 4.4: Cyclical annealing is applied to the β-VAE during pre-training. The function has a duty cycle corresponding to the 25% of the period.

We define as duty cycle the time during which β is maintained active at value 1.5. This strategy forces the maximum degree of disentanglement for multiple epochs, while leveraging the latent features gradually learnt before.

We also design a DAE to define a proper baseline. It consists of two macroblocks, each containing two convolutional layers, and strided convolutions. This time the convolutional encoder network of the DAE is similar to the one introduced for the β-VAE, the only exception is the introduction of the dropout layers, each one after a convolutional macroblock. Evaluating the impact of the DAE allows to extend our methodology and compare the results with those obtained through the β-VAE.

4.2.2 The clustering algorithms

Three clustering methods are evaluated in the Semi-Supervised setting and all of them are built on top of a pre-trained AE. We combine the AE network with the K-means algorithm, we add the β-VAE to the DEC framework [12] and we improve DEC by adding the joint optimization process as suggested in [13].

We apply the DAE to the original formulation of these methods to define reliable baselines. Then, we integrate the β-VAE to improve the level of disentanglement during pre-training. Finally, Section 4.2.4 describes how we extend the existing methods by proposing a new learning approach.

AE + K-means

The unsupervised training of the AE is used to learn a compressed data representation of the input images. Thus, we do not run the K-means algorithm on the raw images but on the embeddings extracted from them. After pre-training,

we disconnect the decoder network and we directly feed the standard K-means algorithm with the compressed data representation coming from the encoder. The procedure is the same for both the DAE and the annealed β-VAE.

DEC

The first step of the DEC method is the unsupervised pre-training through the AE, then the encoder network is extracted and fine-tuned on the clustering task. We connect the encoder with two fully connected layers, for the flattening of the image sample, and with one customized fully connected layer, used for the clustering assignments. The clustering layer consists of ten neurons as we expect to solve clustering problems that involve ten different clusters. Since we are in the Semi-Supervised setting, according to the principles behind SSL explained in Section 2.7, the number of clusters is assumed equals to the number of classes. Once the training of the AE is completed, the learnt latent space is used to run the K-means algorithm to predict the cluster centroids. Therefore, we use K-means to initialize the clusters' centers before the training of the Deep Clustering model. The initialization strategy through K-means allows starting the optimization from a suboptimal solution.

Once the network is initialized, the training of the clustering method starts. We use the KL loss as derived from the equations 3.3, 3.4 and 3.5. Thus, the clustering loss consists of computing the KL divergence between the learnt distribution of the centroids and the predicted clustering assignments. This improves the representation and facilitates the clustering task. At this stage, we leverage the relationship between Deep Clustering and SSL. In fact, the target distribution P is derived from the soft-assignments Q. The minimization of the KL loss can be seen as a form of self-training, a known paradigm of SSL, where a learner is iteratively retrained using its own predictions.

Improved DEC

The improved version of DEC is the last clustering method that we evaluate, before extending it and proposing our new training pipeline for clustering in the Semi-Supervised setting. It combines the solution described above, with the unsupervised training of the AE. The networks are designed in order to optimize at the same time both the reconstruction loss of the AE and the clustering loss. We decide to work on this framework because it is the most promising for images. The joint optimization forces the network to learn a non-trivial data representation to improve the quality of the clusters, with a high degree of disentanglement.

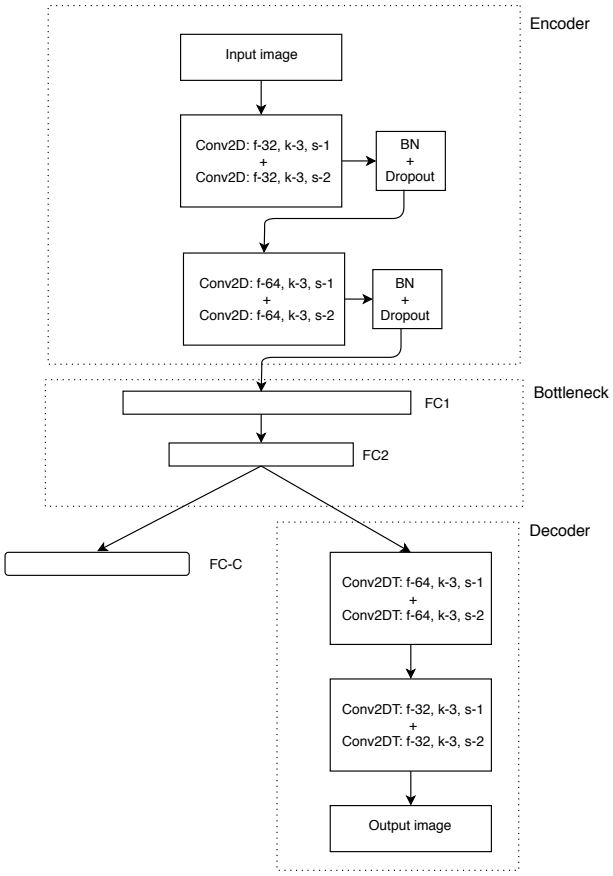


Figure 4.5: The network designed for joint optimization.

During the joint optimization, we use only the DAE since it is not a generative method. By contrast, during the unsupervised pre-training, we consider both the β-VAE and the DAE. The convention *Conv2D: f-32, k-3, s-2* indicates a convolutional layer with 32 filters, kernel of size 3, striding of size 2. We apply strided convolutions to manage the downsampling and transposed convolutions (indicated with *Conv2DT*) for image reconstruction. The clustering layer is directly connected to the bottleneck as it is fed with the representation of the flattened features. The convention *FC* indicates a fully connected layer, thus *FC-C* indicates the clustering layer.

4.2.3 Adapting Deep Clustering to the SSL paradigm

Clustering methods can be applied to SSL to increase the predictive performance of the target supervised model. Our goal, during this part of the research, is not to increase the final performance on the supervised task. We

focus on clustering-based methods in order to improve the quality of the clusters. The idea is that a high Clustering Accuracy implies a better result on the final supervised task. If the clustering algorithm is successful, then the label propagation algorithm, as well as all the other methods presented in Section 2.7 and Section 3.4.4 can be successfully applied. For instance, a low Clustering Accuracy or a low NMI score prevents the application of the framework since it could not be beneficial. There would be the risk of assigning wrong labels to the unlabeled samples or using the cluster assignments as misleading features. Thus, the construction of the clusters is the critical task.

Before introducing the proposed solution, we want to remember the context of SSL. We note that SSL consists in learning a model when both labeled and unlabeled samples are provided. If the knowledge from both the two kinds of samples is combined to solve a classification or a regression problem, then it is possible to improve the predictive performance of the target model.

4.2.4 Proposing a new learning pipeline

The labeled samples are typically used only at the end of the Semi-Supervised framework, or they are considered as unlabeled data for the unsupervised pre-training of the networks. We propose to consider the labeled samples also at the beginning of the learning process. The standard clustering methods are designed to learn patterns and efficient data representations from unlabeled data, as labeled samples are not provided at all. On the other hand, in the Semi-Supervised setting, some labeled samples are available, so we suggest to consider them to support the clustering task.

We develop our solution starting from Deep Clustering, in particular the improved version of DEC, that jointly optimizes the clustering loss and the reconstruction loss. This is state-of-the-art for clustering high dimensional data such as images. However, our new approach can be applied to all the clustering methods that work on the features extracted from images. To the best of our knowledge, we are the first to develop this paradigm in the context of SSL. The most important macro steps are built on top of those proposed by Guo et al. [13]:

1. Unsupervised pre-training through the AE to learn an efficient data representation;

2. Joint optimization of the clustering loss and reconstruction loss.

We focus on the first point because it affects the data representation, thus the success of the clustering algorithm. The pre-training is the initial step of the

method. We propose to train the β-VAE of Section 4.2.1 with annealing applied to the β parameter, as shown in Figure 4.4, to learn a disentangled representation. Then, we design a new way of using the labeled data available in the Semi-Supervised setting. Since labeled samples may be used for the supervised training of ANNs, we introduce a new training step: the supervised fine-tuning of the encoder network on the labeled samples. Finally, we joint optimize the two losses.

This novel training approach for the Semi-Supervised setting can be formulated in three steps:

1. Unsupervised pre-training with the β-VAE. Cyclical annealing is applied to the β parameter during the process.

2. Supervised fine-tuning of the encoder network on the labeled samples. This is an auxiliary classification task used to improve the quality of the data representation.

3. Final training on the clustering task through the joint optimization of the clustering loss and the reconstruction loss.

We introduce a new training strategy for the Semi-Supervised setting, where it is possible to access labeled data. First, choosing a β-VAE for the unsupervised training of the AE allows to learn disentangled features as explained by Higgins et al. [14, 67]. Moreover, "heating" and "cooling" cycles are applied to β as cyclical annealing forces the network to learn a meaningful latent space. Then, we add the new training stage based on the available labeled samples. As labeled data bring relevant information, we design a supervised fine-tuning phase for the encoder network. This is expected to be beneficial since supervised training, by definition, supports the learning of disentangled features. Finally, the network is trained to jointly solve clustering and reconstruction as suggested by Guo et al. [12].

Figure 4.6 shows that the first step is based on pre-training with the β-VAE with cyclical annealing, while the second one is the supervised fine-tuning of the encoder network on the available labeled data. After these two phases, the degree of disentanglement of the latent space is increased. This is expected to simplify the clustering task. Then, the weights of the β-encoder network are transferred to an equivalent DAE.

We use one fully connected layer for the clustering predictions. The optimization procedure, as usual, is initialized with the centroids computed by K-means. All the networks correspond to those described in the previous sections of this chapter.

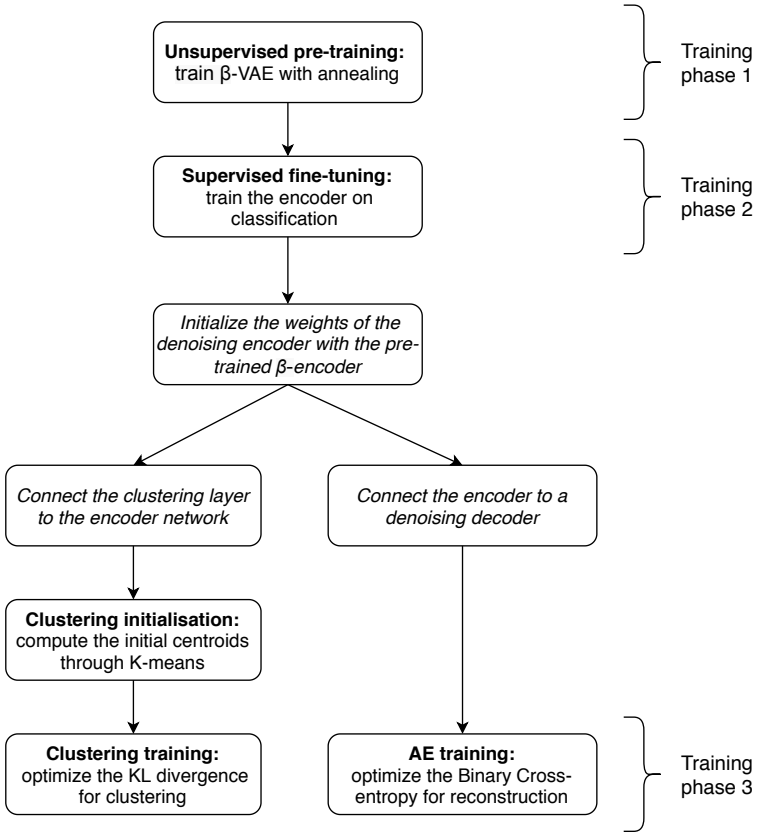


Figure 4.6: The algorithm for Deep Clustering in the Semi-Supervised setting. We design a new disentangled feature learning process. It is built upon unsupervised pre-training through the β-VAE with annealing and the auxiliary supervised fine-tuning phase on the available labeled samples.

Figure 4.7 clarifies how the training pipeline changes with respect to the pure unsupervised approach described in [12, 13]. The unsupervised pre-training with the β-VAE allows reaching a high degree of disentanglement in the learnt features, while balancing the trade-off between reconstruction capability and disentangled representation. Furthermore, the labeled samples provided in the Semi-Supervised scenario enrich the quality of the learnt representation thanks to the auxiliary fine-tuning phase. Finally, the training procedure follows the approach described in the previous sections about DEC.

The auxiliary supervised phase could be applied to each clustering method based on AEs. For instance, after the initial "pre-train and fine-tune" macro step, the encoder could be used as a feature extractor for a standard K-means algorithm instead of Improved DEC.

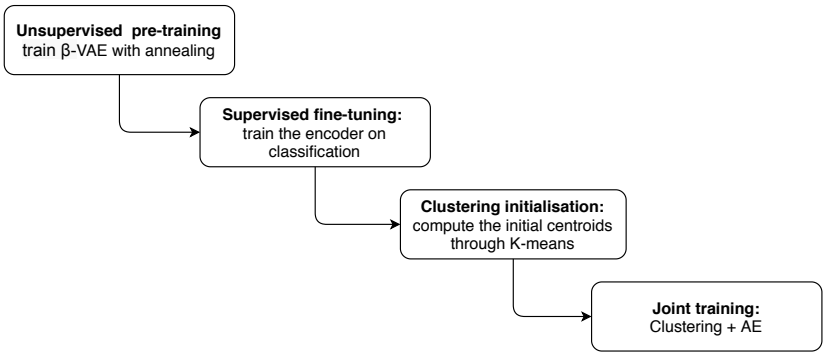


Figure 4.7: The main macro phases in the new training pipeline for the Semi-Supervised setting. There are three training steps. The features learnt at each step are used as initialization for the successive step.

Furthermore, the pre-training approach built upon the β-VAE could also be applied to the typical unsupervised setting of clustering. In that case, there would not be the auxiliary supervised task. However, unsupervised pre-training via the β-VAE with annealing already supports the network in learning a more informative data representation for the final clustering task.

Chapter 5

Experiments and Results

It doesn't matter how beautiful your theory is, it doesn't matter how smart you are, if it doesn't agree with experiment, it's wrong.

R.P. Feynman

This chapter introduces the experimental setup and shows the results obtained from the experiments. Section 5.1 describes the experimental setting, the datasets, and the environments. Section 5.2 reports the empirical results derived from the application of the β-VAE to TL via unsupervised pre-training. Section 5.3 shows the results of the experiments coming from the new Deep Clustering approaches built upon the β-VAE. Finally, Section 5.3.6 extends the experimental procedure by studying how the complexity of patterns impacts Deep Clustering.

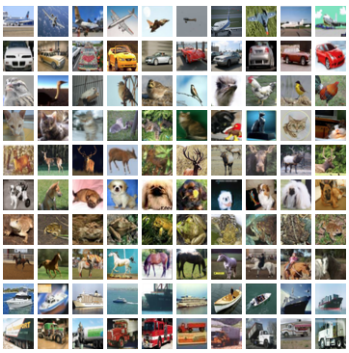
5.1 Experimental setup

The goal of the investigation is to study unsupervised disentangled feature learning, with a focus on the areas of TL and SSL. We analyze the effect of unsupervised pre-training with AEs on image classification tasks. Then, we study Deep Clustering in the Semi-Supervised setting and evaluate our novel learning approaches built upon the β-VAE. Both the phases of the research are based on images as data. We consider RGB images showing subjects in a real environment, as well as grayscale images showing objects. Hence, we study the methods in non-trivial scenarios to get insights on their effect in a hypothetical real use case.

5.1.1 Datasets

We run the experiments on three datasets: CIFAR-10 [17], Fashion-MNIST [18] and, finally, the MNIST digits [19]. For each dataset the pixels are normalized to bring their values in the range $[0, 1]$.

The CIFAR-10 dataset consists of 60000 32×32 RGB images. The samples are divided into 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The Fashion-MNIST dataset contains a training set of 60000 examples and a test set of 10000 examples. Each sample is a 28×28 grayscale image, associated with a label from 10 classes. The examples are equally distributed among the different classes. Fashion-MNIST is often used in research for the benchmarking of ML algorithms.



(a) CIFAR-10

(b) Fashion-MNIST

Figure 5.1: Samples in the two datasets

These datasets contain low resolution images, thus the computational cost for cross-validation and training is acceptable. Furthermore, CIFAR-10 contains objects/animals/vegetables in a real environment, with variable backgrounds and surrounded by minor subjects in some cases. This creates a valuable experimental setting since we can consider RGB images containing complex patterns. On the other hand, the Fashion-MNIST dataset contains grayscale images, mainly showing clothes and accessories. Thus, we consider multiple datasets with different characteristics, for example different colorization and different data domains. This helps to validate the results as well as show the effect of the color information on the methods described in Chapter 4.

Finally, the MNIST digits dataset is considered for a further study of Deep Clustering in the Semi-Supervised setting. This defines an *extended experimental setting* based on simpler images. This allows studying how the complexity of the patterns in the input affects the learning behaviour of Deep Clustering. The dataset contains grayscale handwritten digits, it has a training set

of 60000 examples and a test set of 10000 examples. Unlike the previous datasets, it is not balanced.

5.1.2 Metrics

The first part of the research investigates the impact on image classification of the unsupervised pre-training with AEs. The final image classifier is evaluated according to the metrics discussed in Section 2.1.2. *Precision*, *Recall* and *F1-score* are considered. For each metric, the weighted average over all the 10 classes contained in the datasets is computed. We also decide to analyze the quality of the predictions made for each class, so we plot the confusion matrices after each classification task. It is important to note that both the Fashion MNIST and the CIFAR-10 datasets have a balanced label distribution. The confusion matrices allow to find possible strengths/weaknesses in the predictions at a class level and to evaluate the stability of the models.

The second part of the research is on clustering in the Semi-Supervised setting. As explained in the previous chapters, the goal is to study the quality of the final clusters. Hence, we analyze *Clustering Accuracy*, *NMI* score and *Silhouette* score as clustering metrics. These metrics allow understanding the degree of cohesion/separation between different clusters, as well as the overall clustering quality by considering the ground truth labels available in the original datasets.

5.1.3 Experimental design

During the research project, state-of-the-art methods are implemented as baselines, then new solutions are introduced, and, finally, the results are evaluated. This allows understanding the strengths and weaknesses of the existing methods to find successful ways of improvement.

Conducting research in the areas of TL and SSL requires to create a scenario where both labeled and unlabeled data are provided. We create this by implementing a *synthetic unlabeling* procedure which, starting from a labeled dataset, creates a new version of the same dataset where both labeled and unlabeled samples are available in the train set. Given a *label percentange*, we remove the same percentage of the labeled samples from each class in the training set. This creates a new dataset where the retained labeled samples reflect the label distribution of the original data. Also, not removing the ground truth labels provided in the test set allows comparing the results across different methods. This procedure creates an experimental setting that can easily deal

with variable percentages of labeled data while maintaining the properties of the original dataset.

Transfer Learning and classification

The first part of the research considers TL via the unsupervised training of AEs. We evaluate all the methods for TL described in Chapter 4. Furthermore, we consider a classifier based on a standard architecture, composed of multiple convolutional layers, with no pre-training. The network is composed of three macroblocks, each made up of two convolutional layers, connected by intermediate macroblocks containing dropout and max-pooling layers. From an architectural point of view, each convolutional layer applies the elu activation function. We decide to downsample the examples using max pooling, also we apply batch normalization to prevent vanishing gradient issues. The first two layers apply 32 filters, the successive two layers 64 filters, and the last two layers 128 filters. Each layer has kernel size equals to 3. This network is similar to the encoder network reported in Figure 4.1.

This is a useful benchmark because it reflects the structure of the encoder networks of the designed AEs. It only adds three fully connected layers for classification, separated by dropout. Therefore, this standard classifier corresponds to the networks that we build for the supervised fine-tuning on the classification task, as described in Section 4.1.5. Both the standard classifier and the solutions proposed in Chapter 4 have the same architecture, the only change is in the training process. This new baseline allows understanding whether the methods for the unsupervised pre-training proposed in the thesis have a positive impact. In fact, if the pre-training is beneficial, the proposed networks are expected to outperform the predictive accuracy of the standard classifier described in this section.

In the learning scenario that we design for the experiments, the original train set is decoupled into a labeled train set and an unlabeled train set. During the unsupervised pre-training with the AEs, we train over the entire train set. After removing the decoder and adding the extra layers for classification, we fine-tune the network only over the labeled samples retained from the train set. During fine-tuning, the pre-trained weights are not frozen. Both the adapted ResNet architecture described in Section 4.1.1 and the standard classifier described in this section are trained only on the labeled samples retained from the train set. We always apply for each model the number of epochs necessary for the optimizer to achieve convergence and stabilize the loss. Both during pre-training and fine-tuning we use Adam as optimizer.

Semi-Supervised Learning and clustering

The second part of the research considers clustering in the Semi-Supervised setting. The goal is to evaluate the new methods proposed in Section 4.2.4, compare them with the existing algorithms, and understand how the learning behaviour changes by varying the amount of labeled data provided in the learning setting. By definition, clustering is an unsupervised method used to find patterns in unlabeled data. However, when it is applied in a Semi-Supervised scenario, labeled data are available and they can be beneficial.

The methods studied and proposed in this thesis are based on the pre-training with AEs. Both the AEs and the networks implemented to solve the clustering assignment problem are trained on the entire train set. For the clustering task, the labeled samples are not needed because the techniques proposed in the research are based on a self-training paradigm. The solution we introduce in Section 4.2.4 achieves a significant improvement in terms of clustering quality. This is possible thanks to the β-VAE and the auxiliary supervised fine-tuning on the labeled data. During fine-tuning, the pre-trained weights are not frozen. In this scenario, the encoder network is fine-tuned only on the labeled samples retained from the original train set. All the algorithms use Adam as optimizer both during pre-training and clustering. DEC and Improved DEC run 10000 iterations. Improved DEC assigns $\lambda = 0.9$ as weight for the clustering loss during the joint optimization.

5.1.4 Parameter tuning and results collection

Each architecture reported in Chapter 4, as well as its hyperparameters, are chosen after a cross-validation process. First, we read the literature to find the most promising design choices for the selected networks. Then, we implement our solutions and find the best parameters through a cross-validation process. During the supervised training of the models, we randomly remove the 10% of the labeled samples so as to define a validation set. The results obtained on the validation sets are averaged over all the validation runs and the parameters that optimize the metrics described in Section 5.1.2 are selected. Once we have the best parameters, we fix them and run the final experiments on the test set. Therefore, we rely on the traditional cross-validation process and apply the train-validation-test split to the dataset.

Each final experiment on the test set, according to the experimental procedures found in the literature, is executed five times so as to compute mean and variance for each measured metric. These statistics allow to average the values among multiple runs and reduce the bias in our results.

5.1.5 Hardware and tools

The computational resources used in the project are offered by the `Google Colab` platform. It allows running our experiments in a virtual environment that provides one NVIDIA Tesla K80 GPU, 25GB of RAM, and 68GB of HDD. Also, all the experiments are conducted on this platform to simplify the sharing of code with the supervisors.

We use Keras [20] with TensorFlow [21] backend and well known Python packages (numpy, scipy, sklearn, matplotlib) for all the proposed architectures. The code of the algorithms is implemented in Python 3 and TensorFlow version 1.x. All the experimental results are reported in spreadsheets to share them and compute statistics across all the runs. Then, the results are processed with graphical packages and reported in this chapter.

5.2 β-VAE applied to Transfer Learning

This section reports the results for answering the first part of the research question. Thus, we compare the predictive performance obtained without TL, with the results obtained by applying TL via the pre-training of AEs. In particular, our investigation is focused on the impact of the β-VAE, both in the case a fixed β value is used as well as in case of cyclical annealing on β. The evaluation criteria are based on the final classification performance achieved by the learner after fine-tuning.

5.2.1 Overview and conventions

The research question requires to understand whether the application of TL via the pre-training of AEs is beneficial. We investigate existing techniques, introduce new methods, and evaluate each of them when different amounts of labeled samples are available. Different percentages of labeled data are considered to understand how the performance of each method and baseline changes depending on the amount of labeled and unlabeled training examples. Hence, each method is evaluated on the test set for each percentage of labeled training samples made available in the experimental setting.
For each percentage of labeled data specific experiments are run. During each experiment, we consider five different models.

- *ResNet* is the adapted version of the ResNet architecture as explained in Section 4.1.1. TL is not applied, this solution ignores the unlabeled training samples as it is randomly initialized.

- *Standard Classifier* is an extra baseline for the classification task as explained in Section 5.1.3. This method ignores the unlabeled training samples and the network is randomly initialized.
- *DAE + MLP* is the first method evaluated for unsupervised pre-training with AEs. The encoder network is pre-trained as part of the AE and a *Multi Layer Perceptron (MLP)* is connected to solve the classification task. This is described in Section 4.1.2 and is a state-of-the-art baseline for the unsupervised pre-training.
- *β-VAE (fixed β) + MLP* is the β-VAE trained with a fixed β value to learn disentangled features. Then, the MLP is connected to the encoder for the target image classification task.
- *β-VAE (annealed β) + MLP* is the β-VAE trained with cyclical annealing. This investigates the benefits deriving from annealing in the context of TL for image classification. The MLP is connected for the target classification task.

Table 5.1 shows the results obtained with the 20% of labeled samples. Table 5.2 reports the results obtained with the 40%. In Table 5.3 presents the results in case 50% of labeled examples are available. Table 5.4 considers the scenario with a labels percentage equals to 60%. Finally, Table 5.5 and Table 5.6 report the experiments with the 80% and 100% of labeled samples respectively.

5.2.2 Experimental results

Each table reports the metrics measured both on the CIFAR-10 as well as the Fashion-MNIST datasets to show the performance of each model. This also allows understanding whether the β-VAE can benefit from cyclical annealing during the unsupervised training. We compare the results depending on the number of labeled samples available during the supervised fine-tuning of the target classification network. The higher the value of each measured metric, the better the final predictive performance of the classifier. Finally, the outcome of the experiments is discussed and the performance of the models is presented through a graphical analysis.

CIFAR-10			
Model	Precision	Recall	F1-score
ResNet	0.65904 ± 0.00397	0.66228 ± 0.00483	0.65930 ± 0.00396
Standard Classifier	0.63502 ± 0.00295	0.63898 ± 0.00491	0.63530 ± 0.00310
DAE + MLP	0.65886 ± 0.00725	0.66008 ± 0.00724	0.65926 ± 0.00734
β-VAE (fixed β) + MLP	0.63318 ± 0.00370	0.63676 ± 0.00275	0.63572 ± 0.00306
β-VAE (annealed β) + MLP	0.64778 ± 0.00496	0.65304 ± 0.00418	0.64856 ± 0.00466
Fashion-MNIST			
Model	Precision	Recall	F1-score
ResNet	0.89584 ± 0.00267	0.89608 ± 0.00284	0.89598 ± 0.00288
Standard Classifier	0.88910 ± 0.00366	0.88988 ± 0.00345	0.88962 ± 0.00354
DAE + MLP	0.89378 ± 0.00283	0.89420 ± 0.00305	0.89394 ± 0.00296
β-VAE (fixed β) + MLP	0.89144 ± 0.00318	0.89230 ± 0.00335	0.89174 ± 0.00330
β-VAE (annealed β) + MLP	0.89150 ± 0.00541	0.89202 ± 0.00546	0.89172 ± 0.00542

Table 5.1: Results measured after the supervised fine-tuning on 20% of the original labeled samples. The results are evaluated on the test set. The higher each metric, the better the classification performance.

CIFAR-10			
Model	Precision	Recall	F1-score
ResNet	0.74682 ± 0.00886	0.74786 ± 0.00864	0.74702 ± 0.00883
Standard Classifier	0.69286 ± 0.00552	0.69812 ± 0.00664	0.69470 ± 0.00540
DAE + MLP	0.71108 ± 0.00602	0.71274 ± 0.00590	0.71150 ± 0.00595
β-VAE (fixed β) + MLP	0.69976 ± 0.00483	0.70302 ± 0.00561	0.70030 ± 0.00499
β-VAE (annealed β) + MLP	0.70038 ± 0.01081	0.70324 ± 0.00964	0.70126 ± 0.01035
Fashion-MNIST			
Model	Precision	Recall	F1-score
ResNet	0.91394 ± 0.00233	0.91398 ± 0.00242	0.91392 ± 0.00232
Standard Classifier	0.89948 ± 0.00167	0.90018 ± 0.00171	0.90002 ± 0.00163
DAE + MLP	0.90160 ± 0.00233	0.90272 ± 0.00229	0.90220 ± 0.00221
β-VAE (fixed β) + MLP	0.90054 ± 0.00414	0.90122 ± 0.00430	0.90100 ± 0.00440
β-VAE (annealed β) + MLP	0.90092 ± 0.00514	0.90134 ± 0.00512	0.90110 ± 0.00507

Table 5.2: Results measured after the supervised fine-tuning on 40% of the original labeled samples. The results are evaluated on the test set. The higher each metric, the better the classification performance.

CIFAR-10			
Model	Precision	Recall	F1-score
ResNet	0.77616 ± 0.01043	0.78044 ± 0.00554	0.77668 ± 0.01007
Standard Classifier	0.70594 ± 0.00759	0.70768 ± 0.00506	0.70650 ± 0.00670
DAE + MLP	0.72164 ± 0.00219	0.72290 ± 0.00226	0.72200 ± 0.00227
β-VAE (fixed β) + MLP	0.71402 ± 0.00505	0.71508 ± 0.00422	0.71450 ± 0.00482
β-VAE (annealed β) + MLP	0.71508 ± 0.00691	0.71722 ± 0.00689	0.71574 ± 0.00672
Fashion-MNIST			
Model	Precision	Recall	F1-score
ResNet	0.91724 ± 0.00116	0.91762 ± 0.00077	0.91744 ± 0.00096
Standard Classifier	0.90442 ± 0.00403	0.90462 ± 0.00381	0.90452 ± 0.00393
DAE + MLP	0.90490 ± 0.00318	0.90548 ± 0.00317	0.90518 ± 0.00306
β-VAE (fixed β) + MLP	0.90398 ± 0.00292	0.90458 ± 0.00293	0.90430 ± 0.00283
β-VAE (annealed β) + MLP	0.90466 ± 0.00396	0.90498 ± 0.00397	0.90478 ± 0.00399

Table 5.3: Results measured after the supervised fine-tuning on 50% of the original labeled samples. The results are evaluated on the test set. The higher each metric, the better the classification performance.

CIFAR-10			
Model	Precision	Recall	F1-score
ResNet	0.78534 ± 0.00629	0.78732 ± 0.00547	0.78578 ± 0.00619
Standard Classifier	0.72648 ± 0.00349	0.72774 ± 0.00278	0.72690 ± 0.00320
DAE + MLP	0.73300 ± 0.00674	0.73534 ± 0.00785	0.73400 ± 0.00688
β-VAE (fixed β) + MLP	0.73294 ± 0.01055	0.73456 ± 0.00896	0.73342 ± 0.01012
β-VAE (annealed β) + MLP	0.73302 ± 0.00657	0.73546 ± 0.00754	0.73376 ± 0.00683
Fashion-MNIST			
Model	Precision	Recall	F1-score
ResNet	0.92232 ± 0.00299	0.92282 ± 0.00344	0.92246 ± 0.00318
Standard Classifier	0.90578 ± 0.00147	0.90572 ± 0.00166	0.90566 ± 0.00150
DAE + MLP	0.90594 ± 0.00434	0.90650 ± 0.00425	0.90618 ± 0.00428
β-VAE (fixed β) + MLP	0.90598 ± 0.00133	0.90664 ± 0.00140	0.90630 ± 0.00131
β-VAE (annealed β) + MLP	0.90634 ± 0.00151	0.90674 ± 0.00148	0.90650 ± 0.00157

Table 5.4: Results measured after the supervised fine-tuning on 60% of the original labeled samples. The results are evaluated on the test set. The higher each metric, the better the classification performance.

CIFAR-10			
Model	Precision	Recall	F1-score
ResNet	0.81608 ± 0.00237	0.81704 ± 0.00214	0.81630 ± 0.00243
Standard Classifier	0.75320 ± 0.00401	0.75436 ± 0.00450	0.75370 ± 0.00410
DAE + MLP	0.75934 ± 0.00741	0.76010 ± 0.00709	0.75968 ± 0.00734
β-VAE (fixed β) + MLP	0.75260 ± 0.00928	0.75418 ± 0.00819	0.75302 ± 0.00904
β-VAE (annealed β) + MLP	0.75398 ± 0.00526	0.75504 ± 0.00567	0.75446 ± 0.00554
Fashion-MNIST			
Model	Precision	Recall	F1-score
ResNet	0.92642 ± 0.00173	0.92674 ± 0.00145	0.92652 ± 0.00156
Standard Classifier	0.90892 ± 0.00478	0.90972 ± 0.00460	0.90924 ± 0.00469
DAE + MLP	0.91040 ± 0.00235	0.91082 ± 0.00231	0.91058 ± 0.00240
β-VAE (fixed β) + MLP	0.91052 ± 0.00209	0.91060 ± 0.00156	0.91068 ± 0.00206
β-VAE (annealed β) + MLP	0.91078 ± 0.00386	0.91130 ± 0.00392	0.91094 ± 0.00389

Table 5.5: Results measured after the supervised fine-tuning on 80% of the original labeled samples. The results are evaluated on the test set. The higher each metric, the better the classification performance.

CIFAR-10			
Model	Precision	Recall	F1-score
ResNet	0.82942 ± 0.00284	0.83062 ± 0.00231	0.82976 ± 0.00279
Standard Classifier	0.78100 ± 0.00260	0.78280 ± 0.00277	0.78182 ± 0.00257
DAE + MLP	0.78408 ± 0.00782	0.78732 ± 0.00589	0.78420 ± 0.00793
β-VAE (fixed β) + MLP	0.78240 ± 0.00524	0.78344 ± 0.00499	0.78290 ± 0.00517
β-VAE (annealed β) + MLP	0.78332 ± 0.00466	0.78508 ± 0.00543	0.78372 ± 0.00468
Fashion-MNIST			
Model	Precision	Recall	F1-score
ResNet	0.92902 ± 0.00133	0.92802 ± 0.00439	0.92932 ± 0.00116
Standard Classifier	0.91402 ± 0.00349	0.91482 ± 0.00331	0.91430 ± 0.00344
DAE + MLP	0.91404 ± 0.00416	0.91490 ± 0.00414	0.91440 ± 0.00420
β-VAE (fixed β) + MLP	0.91436 ± 0.00340	0.91508 ± 0.00328	0.91464 ± 0.00325
β-VAE (annealed β) + MLP	0.91562 ± 0.00185	0.91612 ± 0.00187	0.91582 ± 0.00189

Table 5.6: Results measured after the supervised fine-tuning on 100% of the original labeled samples. The results are evaluated on the test set. The higher each metric, the better the classification performance.

5.2.3 Evaluation of pre-training with AEs

The experiments allow understanding the effect of the unsupervised pre-training with AEs. The results need to be studied both by considering the baselines that ignore the unlabeled data, as well as the proposed methods based on pre-training. First, we note that the pre-training with AE networks improves the predictive performance when a few labeled examples are provided. In fact, a simple DAE with random noise applied during the unsupervised training can be beneficial for the pre-training of the network. However, when more labeled samples are provided as input, the benefit deriving from the "pre-train and fine-tune" paradigm decreases. In the case of CIFAR-10, when 80% of the train samples are labeled, the F1-score of *Standard Classifier* is 0.75370, while *DAE + MLP* and β*-VAE (annealed* β*) + MLP* have 0.75968 and 0.75446 respectively. Also, the Precision and Recall metrics follow a similar trend. These three models have the same encoder structure and the only change is due to the pre-training procedure. This means that, given the same architecture, the benefit deriving from the pre-training is not relevant when the labeled training set is large. The predictive performance grows with the number of labeled training samples. The larger the amount of labeled data available during training, the higher Precision, Recall, and F1-score. Furthermore, one may note that the change in the performance of the different AE networks is larger when a few samples are available. This implies that the design of the AE is a task that requires attention. In our case, the best result is given by *DAE + MLP*. It improves the performance of *Standard Classifier* of more than 2% if 20% of labeled examples are retained from CIFAR-10.

Another relevant observation derives from *ResNet*. If only 20% of labeled samples are available, *ResNet* and the other methods achieve similar results on both the datasets. On the other hand, when more labeled data are provided, the predictive performance of *ResNet* significantly improves. This happens thanks to the advanced design of its residual connections. It is noticeable that this state-of-the-art architecture ignores the unlabeled data but at the end achieves a better predictive performance. Therefore, one limitation of the pre-training with AEs in related to the simple architectures of the networks.

In the case of the Fashion-MNIST dataset, we note that TL does not significantly improve the predictive performances. The images in the dataset contain simpler patterns, so the proposed AEs achieve results similar to those of *Standard Classifier*. Also in this case, when more labeled samples are provided, *ResNet* achieves better results.

5.2.4 Evaluation of pre-training with β-VAEs

The experiments demonstrate that an architecture like ResNet achieves good prediction metrics, even if it ignores the unlabeled training samples and there is no pre-training. However, a goal of this first part of the research is about evaluating the effect of cyclical annealing on the β-VAE. AEs often have simple architectures as going deeper could increase the difficulty of training and the risk of overfitting. The models based on the β-VAE often outperform *Standard Classifier*. Thus, the disentangled features learnt during the unsupervised pre-training of the β-VAE may facilitate the final supervised task.

The β-VAE is beneficial for unsupervised pre-training but it does not achieve better results than a DAE. By contrast, the application of cyclical annealing during the training of the β-VAE improves the final classification performance. In fact, *β-VAE (annealed β) + MLP* achieves a better predictive performance when the 20% of labeled samples is made available. In the case of CIFAR-10, the F1-score metric reaches the value of 0.64856 against the 0.63530 of *Standard Classifier*. Hence, the application of cyclical annealing improves the classification metrics for the β-VAE, but a state-of-the-art architecture like *ResNet* may achieve comparable results even on small datasets. In the case of Fashion-MNIST, the application of annealing during pre-training has a positive impact. The *β-VAE (annealed β) + MLP* achieves better results than the corresponding solution without annealing. However, TL does not significantly increase the predictive performance.

5.2.5 Graphical analysis

The graphs in Figure 5.2 and Figure 5.3 allow to understand the behaviour of our models and compare the proposed solutions with the existing ones. When only 20% of the train set is labeled, *ResNet*, *DAE + MLP* and *β-VAE (annealed β) + MLP* have a comparable performance on both the datasets. By contrast, when more labeled data are made available during training, *ResNet* always outperform all the other solutions. One could note that the application of annealing to the β-VAE is beneficial, but learning also from unlabeled data does not allow to achieve a final performance comparable to that of *ResNet* for large datasets. In fact, the β-VAE is designed for learning disentangled features through unsupervised training. Thus, it gives a significant improvement only if few labeled examples are provided. It is possible to note that the benefit deriving from pre-training is not relevant when more than approximately 60% of the train samples are labeled. In that case, all the pre-trained models tend to achieve the same predictive performance.

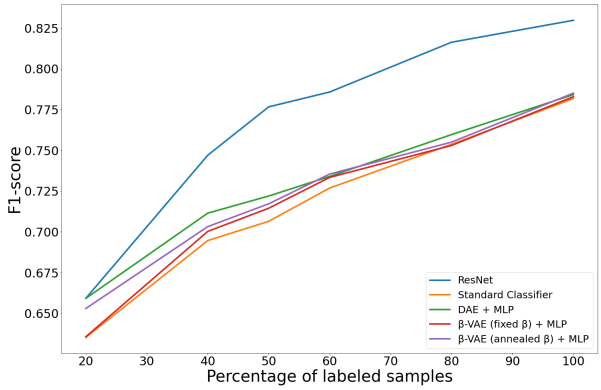


Figure 5.2: F1-score measured on CIFAR-10.

The results of the two datasets follow the same trend. Although the datasets contain images with different properties, *ResNet* achieves the best performance on both of them when more examples are provided. On the other hand, when a few samples are available, the pre-training improves the performance with respect to *Standard Classifier*. Finally, cyclical annealing applied to β is useful in both the cases for the β-VAE if a few labeled examples are accessible.

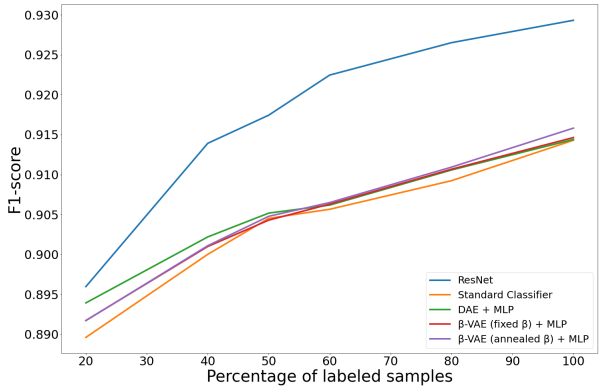


Figure 5.3: F1-score measured on Fashion-MNIST.

One may observe that the complexity of the patterns influences the results of TL. When the 40% of samples are labeled, pre-training may increase the predictive performance on CIFAR-10. By contrast, on Fashion-MNIST there is not a relevant improvement.

5.3 Semi-Supervised Deep Clustering

This section reports the results for answering the second part of the research question. Hence, we evaluate novel methods based on Deep Clustering and investigate the effect of the β-VAE with annealing for pre-training. Furthermore, we evaluate our new proposed solution for Deep Clustering in the Semi-Supervised setting. Finally, we study the proposed methods on the MNIST digits dataset, to better understand how the dimensionality and the patterns in the input affect each technique.

5.3.1 Overview and conventions

We introduce pre-training with the β-VAE for extracting the features to solve the clustering task. Also, we propose a new learning pipeline to leverage the labeled samples provided in the Semi-Supervised scenario. In the first part of Section 5.3.2, we show the results measured for each algorithm in case only unlabeled samples are considered. On the other hand, in the second part of Section 5.3.2, we report the results obtained after the introduction of the new learning pipeline based on the auxiliary supervised task on the available labeled data. All the methods are explained in Section 4.2.2.

- *DAE + K-means* is a standard baseline built upon K-means and the DAE.
- *DAE + DEC* is a technique based on ANNs for clustering. After the initial training of the DAE, the DEC algorithm is applied.
- *DAE + Improved DEC* is the first investigation about the joint optimization of reconstruction and clustering. The DAE is used for pre-training.
- *β-VAE + K-means* investigates the impact of annealing on the pre-training with the β-VAE. This extends the method proposed in 4.2.2.
- *β-VAE + DEC* applies the β-VAE with annealing to the DEC algorithm.
- *β-VAE + Improved DEC* uses the β-VAE with annealing for pre-training. Then, it jointly optimizes clustering and reconstruction.

Table 5.7 provides an evaluation of the β-VAE in the standard unsupervised scenario of Deep Clustering. Table 5.8 shows the results obtained when 20% of samples are labeled. Table 5.9 presents the results obtained with 40% of labels. Table 5.10 contains the results measured in case 50% of examples are labeled. Finally, Table 5.11 and Table 5.12 describe the performances with 60% and 80% of labeled samples respectively.

5.3.2 Experimental results

Each table reports the metrics measured both on the CIFAR-10 and the Fashion-MNIST datasets. This allows to study the performance of each model and understand whether the β-VAE with cyclical annealing can be beneficial for Deep Clustering. Furthermore, the new training process is compared on both the datasets to evaluate the effect of the supervised fine-tuning. We indicate in bold our approaches built upon the β-VAE.

Unsupervised pre-training with AEs

In this section, we show the results obtained after applying the new pre-training process based on the β-VAE. The auxiliary supervised fine-tuning phase is not considered at this stage as we initially focus on a fully unsupervised scenario. During pre-training we introduce the β-VAE and compare it with a DAE.

CIFAR-10			
Model	ACC	NMI	SIL
DAE + K-means	0.21482 ± 0.01229	0.09518 ± 0.00671	0.04072 ± 0.01104
DAE + DEC	0.21850 ± 0.01516	0.09610 ± 0.00632	0.45852 ± 0.02315
DAE + Improved DEC	0.21986 ± 0.01162	0.09632 ± 0.00991	0.48842 ± 0.01516
β-VAE + K-means	0.22434 ± 0.00634	0.10024 ± 0.00570	0.05270 ± 0.01191
β-VAE + DEC	0.22536 ± 0.00881	0.10122 ± 0.01068	0.52868 ± 0.03022
β-VAE + Improved DEC	0.23060 ± 0.00704	0.10710 ± 0.00914	0.53264 ± 0.01197
Fashion-MNIST			
Model	ACC	NMI	SIL
DAE + K-means	0.49360 ± 0.00766	0.54858 ± 0.00884	0.10506 ± 0.01241
DAE + DEC	0.51404 ± 0.01243	0.55334 ± 0.01591	0.83232 ± 0.00416
DAE + Improved DEC	0.51746 ± 0.00954	0.55522 ± 0.01597	0.83988 ± 0.00572
β-VAE + K-means	0.51220 ± 0.00672	0.57014 ± 0.00390	0.16000 ± 0.00362
β-VAE + DEC	0.51944 ± 0.00434	0.58184 ± 0.00997	0.83310 ± 0.00907
β-VAE + Improved DEC	0.52630 ± 0.00544	0.58202 ± 0.00341	0.84250 ± 0.00191

Table 5.7: Results measured in a standard unsupervised setting. Each algorithm runs on the features extracted by the pre-trained encoder networks.

Novel supervised fine-tuning

In this section, we report the results obtained after introducing the new learning pipeline. After the training of the AE, we start a supervised fine-tuning phase of the encoder on the available labeled samples. Therefore, we compare the results depending on the labeled samples provided in the Semi-Supervised setting.

CIFAR-10			
Model	ACC	NMI	SIL
DAE + K-means	0.32302 ± 0.00688	0.22286 ± 0.00811	0.03730 ± 0.00280
DAE + DEC	0.33822 ± 0.00480	0.24306 ± 0.01438	0.46278 ± 0.04729
DAE + Improved DEC	0.34864 ± 0.00853	0.25110 ± 0.02584	0.49002 ± 0.02637
β-VAE + K-means	0.33688 ± 0.02065	0.23610 ± 0.01565	0.05354 ± 0.00496
β-VAE + DEC	0.35504 ± 0.01731	0.24404 ± 0.01293	0.53612 ± 0.02825
β-VAE + Improved DEC	0.35786 ± 0.01481	0.26348 ± 0.02229	0.53820 ± 0.01428
Fashion-MNIST			
Model	ACC	NMI	SIL
DAE + K-means	0.74270 ± 0.02695	0.74256 ± 0.01378	0.21534 ± 0.01553
DAE + DEC	0.74544 ± 0.03202	0.74786 ± 0.01832	0.83570 ± 0.02580
DAE + Improved DEC	0.74768 ± 0.02758	0.74946 ± 0.01797	0.84172 ± 0.01623
β-VAE + K-means	0.77298 ± 0.01541	0.76626 ± 0.01838	0.24018 ± 0.01354
β-VAE + DEC	0.77682 ± 0.00418	0.77490 ± 0.00397	0.83718 ± 0.00442
β-VAE + Improved DEC	0.78556 ± 0.00652	0.77552 ± 0.01183	0.84384 ± 0.00450

Table 5.8: Results measured for the novel Semi-Supervised Deep Clustering framework. After the unsupervised pre-training, the encoder network is fine-tuned on 20% of the original labeled samples.

CIFAR-10			
Model	ACC	NMI	SIL
DAE + K-means	0.39394 ± 0.01273	0.29410 ± 0.01346	0.03772 ± 0.00362
DAE + DEC	0.39870 ± 0.01798	0.30016 ± 0.01406	0.50416 ± 0.02863
DAE + Improved DEC	0.39908 ± 0.02246	0.31286 ± 0.02723	0.50688 ± 0.03473
β-VAE + K-means	0.41292 ± 0.02588	0.31234 ± 0.02169	0.05414 ± 0.00666
β-VAE + DEC	0.41616 ± 0.02663	0.31558 ± 0.01718	0.54070 ± 0.01372
β-VAE + Improved DEC	0.41978 ± 0.02557	0.32142 ± 0.02583	0.54318 ± 0.03373
Fashion-MNIST			
Model	ACC	NMI	SIL
DAE + K-means	0.74804 ± 0.02909	0.74886 ± 0.00702	0.2186 ± 0.01508
DAE + DEC	0.75148 ± 0.01620	0.74930 ± 0.00912	0.83634 ± 0.01166
DAE + Improved DEC	0.75302 ± 0.03155	0.75180 ± 0.01609	0.84504 ± 0.02759
β-VAE + K-means	0.77976 ± 0.01322	0.77144 ± 0.01372	0.24842 ± 0.01082
β-VAE + DEC	0.78176 ± 0.01494	0.78220 ± 0.01512	0.83968 ± 0.01509
β-VAE + Improved DEC	0.78966 ± 0.00411	0.78882 ± 0.00452	0.84788 ± 0.00781

Table 5.9: Results measured for the novel Semi-Supervised Deep Clustering framework. After the unsupervised pre-training, the encoder network is fine-tuned on 40% of the original labeled samples.

CIFAR-10			
Model	ACC	NMI	SIL
DAE + K-means	0.41328 ± 0.01603	0.31328 ± 0.00559	0.03884 ± 0.00558
DAE + DEC	0.42004 ± 0.02073	0.31890 ± 0.00638	0.50680 ± 0.02519
DAE + Improved DEC	0.42316 ± 0.01219	0.33154 ± 0.02195	0.51144 ± 0.02192
β-VAE + K-means	0.42116 ± 0.01951	0.33618 ± 0.01254	0.05430 ± 0.00615
β-VAE + DEC	0.42758 ± 0.01883	0.33894 ± 0.01236	0.54240 ± 0.03630
β-VAE + Improved DEC	0.43106 ± 0.01494	0.34214 ± 0.02384	0.54374 ± 0.01467
Fashion-MNIST			
Model	ACC	NMI	SIL
DAE + K-means	0.75178 ± 0.02573	0.75134 ± 0.02901	0.21936 ± 0.01791
DAE + DEC	0.75420 ± 0.02542	0.75548 ± 0.01529	0.83680 ± 0.02157
DAE + Improved DEC	0.75656 ± 0.01575	0.75646 ± 0.01484	0.84712 ± 0.01230
β-VAE + K-means	0.78232 ± 0.01814	0.78636 ± 0.01349	0.26736 ± 0.01327
β-VAE + DEC	0.78578 ± 0.01413	0.78732 ± 0.00847	0.84124 ± 0.00630
β-VAE + Improved DEC	0.79128 ± 0.00822	0.78976 ± 0.00958	0.84902 ± 0.00797

Table 5.10: Results measured for the novel Semi-Supervised Deep Clustering framework. After the unsupervised pre-training, the encoder network is fine-tuned on 50% of the original labeled samples.

CIFAR-10			
Model	ACC	NMI	SIL
DAE + K-means	0.42296 ± 0.01399	0.32240 ± 0.01324	0.04864 ± 0.00422
DAE + DEC	0.42866 ± 0.01962	0.32726 ± 0.00927	0.52980 ± 0.03360
DAE + Improved DEC	0.43206 ± 0.02037	0.33800 ± 0.01482	0.53148 ± 0.02751
β-VAE + K-means	0.43414 ± 0.02097	0.34504 ± 0.01442	0.05586 ± 0.01547
β-VAE + DEC	0.44634 ± 0.00895	0.34682 ± 0.00831	0.55620 ± 0.02855
β-VAE + Improved DEC	0.44832 ± 0.02765	0.35608 ± 0.02623	0.55712 ± 0.00735
Fashion-MNIST			
Model	ACC	NMI	SIL
DAE + K-means	0.75520 ± 0.01986	0.75654 ± 0.01982	0.22156 ± 0.01009
DAE + DEC	0.75948 ± 0.01834	0.76240 ± 0.01679	0.83794 ± 0.01623
DAE + Improved DEC	0.76682 ± 0.00749	0.76390 ± 0.00739	0.84952 ± 0.00929
β-VAE + K-means	0.78846 ± 0.00738	0.79178 ± 0.01004	0.26944 ± 0.02402
β-VAE + DEC	0.79292 ± 0.00771	0.79538 ± 0.01338	0.84220 ± 0.00821
β-VAE + Improved DEC	0.79512 ± 0.00381	0.79724 ± 0.00646	0.85104 ± 0.00621

Table 5.11: Results measured for the novel Semi-Supervised Deep Clustering framework. After the unsupervised pre-training, the encoder network is fine-tuned on 60% of the original labeled samples.

CIFAR-10			
Model	ACC	NMI	SIL
DAE + K-means	0.45028 ± 0.01476	0.35444 ± 0.02097	0.06044 ± 0.00872
DAE + DEC	0.45986 ± 0.01411	0.35754 ± 0.02325	0.54364 ± 0.03902
DAE + Improved DEC	0.46750 ± 0.01891	0.37226 ± 0.02339	0.54720 ± 0.03366
β-VAE + K-means	0.45204 ± 0.03364	0.37390 ± 0.01591	0.06138 ± 0.00301
β-VAE + DEC	0.46438 ± 0.02552	0.37942 ± 0.01338	0.55880 ± 0.02595
β-VAE + Improved DEC	0.46912 ± 0.01931	0.38518 ± 0.01657	0.56468 ± 0.03274
Fashion-MNIST			
Model	ACC	NMI	SIL
DAE + K-means	0.76414 ± 0.01423	0.76160 ± 0.01235	0.22602 ± 0.01665
DAE + DEC	0.76730 ± 0.00891	0.76574 ± 0.00890	0.84212 ± 0.01392
DAE + Improved DEC	0.77108 ± 0.01522	0.76684 ± 0.00923	0.85114 ± 0.01589
β-VAE + K-means	0.79220 ± 0.01131	0.79408 ± 0.00939	0.27932 ± 0.00349
β-VAE + DEC	0.79696 ± 0.01072	0.79872 ± 0.00766	0.84416 ± 0.01127
β-VAE + Improved DEC	0.81092 ± 0.03506	0.80760 ± 0.01753	0.85312 ± 0.01016

Table 5.12: Results measured for the novel Semi-Supervised Deep Clustering framework. After the unsupervised pre-training, the encoder network is fine-tuned on 80% of the original labeled samples.

5.3.3 Evaluation of pre-training with the β-VAE

We introduce cyclical annealing to balance the reconstruction quality and the degree of disentanglement of the latent features during pre-training. First, we consider the results reported in Table 5.3.2 and note that using the β-VAE during the unsupervised pre-training increases the final clustering metrics. For instance, in the case of the CIFAR-10 dataset, the *Clustering Accuracy (ACC)* increases from 0.21986 of *DAE + Improved DEC* up to 0.23060 of *β-VAE + Improved DEC*. Improvements are also achieved in terms of *Silhouette (SIL)*, it means that a better degree of cohesion within the same cluster is reached, while the separation between different clusters increases.

One may also note that the design of the AE impacts the final clustering performance. If the pre-training strategy does not change, we observe that more advanced algorithms like DEC and Improved DEC allow reaching improvements, but they are not always significant. For example, in the case of CIFAR-10, the Clustering Accuracy of *β-VAE + K-means* is 0.22434, while for *β-VAE + DEC* it is 0.22536. Given the complexity of the patterns in the images, even the methods based on neural networks struggle to improve the final clustering performance. By contrast, Fashion-MNIST contains simpler patterns and the algorithms can achieve larger improvements.

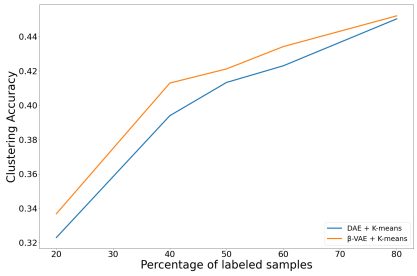
5.3.4 Evaluation of the Semi-Supervised approach

We propose a new learning pipeline to improve the quality of the clusters in the Semi-Supervised setting. The unsupervised training with cyclical annealing of the β-VAE is beneficial. As we expect, disentangled features simplify the clustering task. However, in the Semi-Supervised scenario, it is possible to access a portion of labeled samples, so we use them to fine-tune the encoder network and improve the data representation. This new training step increases the clustering performance. In the case of CIFAR-10, if 20% of the training samples are labeled, the Clustering Accuracy of *DAE + K-means* raises from 0.21482 up to 0.32302. This is possible because the labeled samples force the encoder to learn a better latent representation. This extension of the learning process is more beneficial if applied to the β-VAE. It is noticeable that *DAE + Improved DEC* has an accuracy equals to 0.34864, while *β-VAE + Improved DEC* has a value of 0.35786. Therefore, with only 20% of labeled data, the metric increases of almost 2%. We can observe from CIFAR-10 that the combination of the pre-trained β-VAE and the supervised step is beneficial also when more labeled samples are provided. In fact, using the β-VAE during pre-training regularly outperform the DAE. Our novel approach also increases the Silhouette score, so better clusters are built from a cohesion point of view.

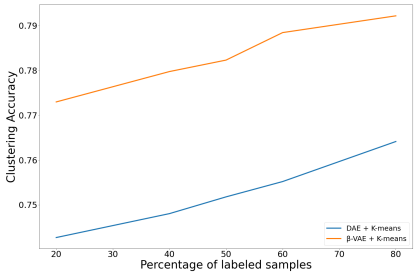
Therefore, in a Semi-Supervised scenario, the labeled samples may be used to improve the data representation. From the analysis of the results obtained on both the datasets, it is possible to note that the auxiliary classification task on the labeled data is more beneficial if the encoder is pre-trained through the β-VAE. The supervised fine-tuning step becomes useful in case complex patterns need to be detected from images. In fact, larger improvements are achieved in the case of CIFAR-10 than Fashion-MNIST.

5.3.5 Graphical analysis

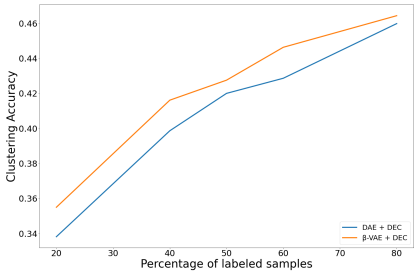
The following graphs allow studying the proposed methods depending on the number of labeled examples provided in the Semi-Supervised scenario. The application of DL to solve the clustering task is beneficial, but the process still depends on the quality of the learnt data representation. We note that the approaches based on the β-VAE achieve better results than those based on the DAE. Also, all the techniques built upon the auxiliary classification task achieve higher results than the corresponding methods without that procedure. The experimental results reported in Figure 5.4 show that a β-VAE with cyclical annealing is a valuable pre-training choice.



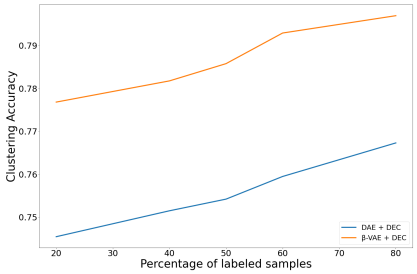
(a) K-means on CIFAR-10.



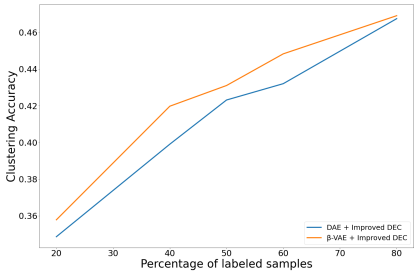
(b) K-means on Fashion-MNIST.



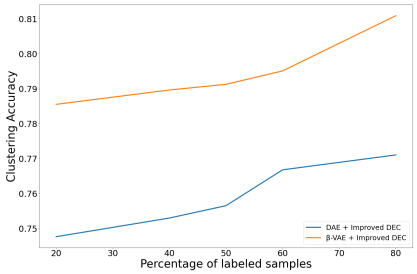
(c) DEC on CIFAR-10.



(d) DEC on Fashion-MNIST.



(e) Improved DEC on CIFAR-10.



(f) Improved DEC on Fashion-MNIST.

Figure 5.4: The new learning approach evaluated on different methods. For each clustering algorithm, the best results are obtained through the β-VAE.

Although the datasets contain images with different properties, the unsupervised pre-training strategy with the β-VAE is beneficial in both cases. One may also note that thanks to the proposed learning process built upon the auxiliary supervised fine-tuning phase, the Clustering Accuracy tends to linearly increase with the amount of labeled samples.

5.3.6 Extended study on MNIST digits

It is possible to observe from the experiments conducted on CIFAR-10 and Fashion-MNIST that a large improvement is related to the choice of the AEs. However, changing the clustering algorithm could not significantly improve the clustering metrics. For example, we can observe this phenomenon in Table 5.3.2. The Clustering Accuracy on CIFAR-10 for *DAE + K-means* is 0.21482, for *DAE + DEC* is 0.21850 while for *DAE + Improved DEC* it is 0.21986. Also, the NMI score follows a similar trend. The same happens for the algorithms based on the β-VAE. Since in the literature study we discuss the effect of dimensionality and the role of the complexity of patterns in the images, we further analyze Deep Clustering on the MNIST digits dataset.

We design an extended experimental setting based on MNIST digits to benchmark our implementation and results with those reported in [12]. In addition, this allows studying the behaviour of each algorithm on a dataset which contains images with low dimensionality (they are grayscale) and showing simpler patterns. Thus, we can evaluate how the dimensionality affects each algorithm. As we already considered the impact of different AEs during pre-training, we now focus on the DAE and evaluate it both in a fully unsupervised scenario as well as in the Semi-Supervised setting.

Evaluation of pre-training with the DAE

This section shows the results obtained after the pre-training with the DAE. We design a network with three convolutional layers and two dropout layers. The first convolutional layer has 32 filters, size of the kernel equals to 5, stride equals to 2, and same padding. The second convolutional layer has the same structure, but it applies 64 filters. The third layer has 64 filters, the size of kernel equals to 4, stride equals to 1, and same padding. Each pair of convolutional layers is separated by one dropout layer to generate randomness during training. Finally, the bottleneck is realized through a fully connected layer with ten neurons. The network is pre-trained using Adam as optimizer and

the MSE as reconstruction loss. We define a simple network since the dataset contains grayscale images with simple patterns.

The new learning approach based on the β-VAE is not considered at this stage of the research, as the focus is on the performance reached by the different algorithms. It is possible to note from Table 5.13 that the approaches based on Deep Clustering, like DEC and Improved DEC, significantly improve all the clustering metrics.

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-means	0.84826 ± 0.00911	0.7854 ± 0.00440	0.22946 ± 0.00623
DAE + DEC	0.87752 ± 0.02047	0.86650 ± 0.01295	0.90112 ± 0.00630
DAE + Improved DEC	0.89434 ± 0.00747	0.88558 ± 0.00488	0.90582 ± 0.00771

Table 5.13: Results measured during the extended clustering study. First, the algorithms are evaluated in an unsupervised scenario.

Xie et al. [12] report a Clustering Accuracy equals to 0.8184 for *AE + K-means* and 0.8430 for *AE + DEC*. Guo et al. [13] obtain 0.8806 as Clustering Accuracy for *AE + Improved DEC*. The authors of the papers consider AEs with no noise added during training. By contrast, we obtain better performances as we use a DAE to better generalize the data representation. In the case of MNIST digits, the algorithms based on neural networks for the clustering assignments achieve significantly better results than a standard K-means. The application of DAE achieves a Clustering Accuracy equals to 0.84826 with *DAE + K-means*. On the other hand, *DAE + DEC* and *DAE + Improved DEC* reach values equal to 0.87752 and 0.89434 respectively. Comparing the results with those reported in the original papers allows to validate our implementation of the proposed solutions.

Therefore, in the case of the MNIST digits dataset, it is possible to significantly improve the results of a standard K-means thanks to Deep Clustering. DEC increases the performance of about 3%. Moreover, the joint optimization procedure of Improved DEC increases the Clustering Accuracy of almost 5%. We can observe that, in the case of data with simple patterns, the reported unsupervised algorithms allow to build better clusters and achieve a performance comparable to a supervised task. On the other hand, in the case of CIFAR-10 and Fashion-MNIST, the results reported in Section 5.3.2 do not show relevant improvements.

Evaluation of the Semi-Supervised approach

In this section, we report the experimental results collected after applying the new learning pipeline designed for the Semi-Supervised scenario. The procedure is the same as described before. We pre-train with the AE network and, before running the clustering algorithm, we fine-tune the encoder on the auxiliary classification task. In Section A.2.1 we show all the results obtained for each amount of labeled samples. Here, we report the results obtained when 20% of the samples are provided with labels in the Semi-Supervised setting, then we plot the experimental values.

One may note from Table 5.14 that the new approach improves the clustering performance also in the case of MNIST digits. The first observation is that it reaches higher values for each metric. For instance, the Clustering Accuracy of *DAE + DEC* raises from 0.87752 up to 0.98298 by taking advantage of the auxiliary supervised step.

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-means	0.94454 ± 0.00652	0.89512 ± 0.01170	0.57710 ± 0.00897
DAE + DEC	0.98298 ± 0.00177	0.95200 ± 0.00598	0.90666 ± 0.00853
DAE + Improved DEC	0.98354 ± 0.00278	0.95484 ± 0.00238	0.90750 ± 0.00436

Table 5.14: Results measured for the novel Semi-Supervised Deep Clustering framework on MNIST digits. After the unsupervised pre-training, the encoder network is fine-tuned on 20% of the original labeled samples

Another observation from Table 5.14 is about the impact of the clustering algorithms. The different algorithms achieve significantly different results. For instance, the NMI score of *DAE + K-means* is 0.89512, while the score from *DAE + DEC* is 0.95200. Therefore, the new learning approach increases the quality of the data representation, and clustering based on neural networks still achieves better results. This means that working on images containing simple patterns facilitates Deep Clustering, even in case the learnt latent space already contains disentangled features.

It is noticeable from Figure 5.5 that the methods change their performance depending on the percentage of labels provided in the Semi-Supervised scenario.

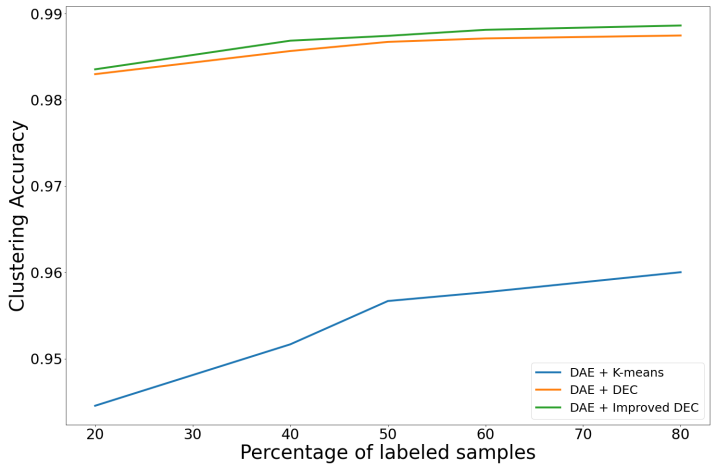


Figure 5.5: Evaluation of the Clustering Accuracy on MNIST digits.

The metrics evaluated for DEC and Improved DEC do not significantly improve when more examples are considered during the auxiliary supervised fine-tuning step. Once again, the simple patterns in the images already facilitate the clustering assignment process, so less labeled data are needed by the Deep Clustering methods.

Chapter 6

Discussion and Conclusions

Before we work on artificial intelligence, why don't we do something about natural stupidity?

Steve Polyak

This chapter discusses the experimental results, the achievements, and promising directions for future work. Section 6.1 comments the results obtained via the pre-training with the β-VAE and evaluates the effect of pre-training for TL. Section 6.2 explains the results obtained by taking advantage of the new learning approaches for Deep Clustering built upon the β-VAE, as well as compare the results with previous works in the literature. Section 6.3 and Section 6.4 identify possible limitations in the research and suggest relevant directions for future work respectively. Finally, Section 6.6 draws the conclusions.

6.1 β-VAE applied to Transfer Learning

The first part of the research question is answered thanks to the evaluation of TL via unsupervised pre-training with AEs. We compare DAEs with the β-VAE. Also, we introduce cyclical annealing during the unsupervised training of the β-VAE. We study the results obtained with the application of TL and compare them with those obtained through the ResNet architecture.

The results reported in Section 5.2 highlight that the application of cyclical annealing to the β parameter during pre-training is beneficial. Both in the case of the CIFAR-10 and Fashion-MNIST datasets, it allows increasing the final predictive performance. For instance, the F1-score for the model built upon the pre-training of the β-VAE raises from 0.63572 up to 0.64856 thanks to annealing on CIFAR-10 when 20% of the labeled samples are retained.

However, ResNet achieves an F1-score equals to 0.65930. We consider ResNet with random initialization trained only on the available labeled samples to follow the suggestions provided by Oliver et al. [15]. This allows understanding whether the models based on unsupervised pre-training studied in the research could be beneficial. Fu et al. [73] show the advantages of cyclical annealing during the training of a β-VAE for NLP tasks. We observe that annealing is a successful strategy also for the TL paradigm.

The results suggest that pre-training a feature extraction network increases the quality of the data representation, in particular when a few labeled samples are provided. Bengio et al. [7] demonstrate the role of pre-training as a form of regularizer to prevent overfitting and that it is more effective for lower layers than for higher layers. To the best of our knowledge, we are the first to evaluate the effect of TL via pre-training with β-VAEs in terms of final predictive performance on image classification. Given the same feature extraction network, an annealed β has a positive impact on the classification task. However, we note that a deeper state-of-the-art architecture designed to learn complex patterns may be better, even with random initialization.

6.2 Semi-Supervised Deep Clustering

The second part of the research is focused on Deep Clustering in the Semi-Supervised setting. We extend the DEC algorithm proposed by Xie et al. [12] and improve the training process by taking inspiration from the joint optimization proposed by Guo et al. [13].

6.2.1 Deep Clustering applied to MNIST digits

We initially test our implementation of DEC on the MNIST digits dataset and introduce a standard Convolutional DAE. We are able to replicate the results reported in the original paper and by taking advantage of the DAE we can improve the clustering metrics. Our implementation of DEC achieves a Clustering Accuracy equals to 0.87752 while Xie et al. [12] report a value equals to 0.84300. This design choice also increases the metric for Improved DEC, which reaches a value of 0.89434. By contrast, Guo et al. [13] report a score equals to 0.88060. The study of the algorithms on the MNIST digits dataset allows understanding how complex patterns impact image clustering. In fact, the simplicity of patterns in this dataset facilitates the training of the networks as already demonstrated in Section 5.3.6.

6.2.2 Gains deriving from the new approach

The main contribution of this work is related to the design of a new learning approach based on the β-VAE. Higgings et al. [14] demonstrate the β-VAE to learn a disentangled feature representation through unsupervised training. We leverage this strength and introduce it in the standard training pipeline of DEC. Our new solutions based on the β-VAE allow to reach a higher Clustering Accuracy than the corresponding algorithms based on a DAE. In the case of a fully unsupervised clustering task on the CIFAR-10 dataset, the β-VAE combined with DEC achieves a value equals to 0.22536, while the DAE combined with DEC has a value equals to 0.21850. A similar improvement is noticeable also for the Fashion-MNIST dataset. However, as it contains images with lower dimensionality and simpler patterns, it is not as large as in the previous case.

The new learning approach for the Semi-Supervised scenario is designed to leverage the potential of the labeled samples. The investigation about Deep Clustering demonstrates the key role played by the data representation. In the case of high dimensional data like images, Deep Clustering is more successful because it is capable of learning an efficient data representation. The introduction of the β-VAE increases the degree of disentanglement in the latent space, and it simplifies the clustering task. For the Semi-Supervised scenario, we also introduce an auxiliary supervised fine-tuning step on the available labeled samples, to further increase disentanglement and, consequently, learn a better data representation for the clustering problem. Hence, we do not change the algorithm for the assignments proposed by Xie et al. [12] as we focus the work on the training strategy.

Shukla et al. [82] propose a clustering method that uses pairwise constraints created from the labeled samples. They augment the unlabeled data with the labeled ones to find a representation suitable for clustering. The authors conduct experiments neither on the CIFAR-10 nor on the Fashion-MNIST dataset. However, the experiments conducted on other datasets containing high dimensional images show promising results. The combination of the KL loss with a traditional K-means loss indicates possible ways of improvement that could be beneficial for our approach as well.

Ren et al. [83] include pairwise constraints in the DEC algorithm to embed the knowledge deriving from the labeled samples. They work on the data representation because the available constraints are added to the latent bottleneck so as to increase the quality of the latent space. When the 20% of labeled samples are retained from the original dataset, our Semi-Supervised pipeline based

on the β-VAE reaches a Clustering Accuracy equals to 0.35504 for CIFAR-10. By contrast, the approach proposed by Ren et. al [83], even when more labeled samples are provided, does not reach a value higher than 0.27260. The main strength of our approach is that we initially learn a disentangled representation, then increase its quality through the auxiliary supervised fine-tuning phase. Therefore, we leverage the labeled data through a supervised training of the encoder network, while the approach based on pairwise constraints faces the problem only from a feature representation point of view.

6.3 Limitations

The thesis considers two related research questions and investigates the TL paradigm from different perspectives. The project focuses on the area of CV, so one first limitation is related to the data domain. Images are complex and high dimensional data that require careful labeling, however, TL and SSL would also require a complete investigation for other relevant domains, for instance, NLP and Recommender Systems.

From a theoretical point of view, the study initially analyzes TL via unsupervised pre-training with AEs. We consider a generative model like the β-VAE and apply linear annealing functions. More studies on advanced generative models derived from the GANs proposed by Goodfellow et al. [70] could open space for further research in the area of TL. In addition, some delimitations should be noted for the second part of the research, where we propose new approaches for Semi-Supervised Deep-Clustering. As the application of clustering to SSL is based on the cluster assumption described in Section 2.7, we measure and evaluate the clustering metrics. However, an extended research about the impact of the clustering information for solving the final supervised task would allow understanding more in-depth the strengths of the framework. We decide to focus the research on pre-training with AEs and on the clustering performance in the Semi-Supervised scenario to create a well-defined research scope, answer the research questions and, according to the outcomes, define directions for future work.

The experimental setting was designed to handle different percentages of labeled samples and define a Semi-Supervised environment. The first delimitation is due to the percentages of samples considered in the research. As this is a first study, we evaluate the performances of each model considering only six different amounts of labeled examples for each dataset. The more percentages are considered, in particular in the case of low values, the more insights on the behaviour of each proposed solution may be collected. The

second limitation is due to the intrinsic focus of the research. We evaluate TL and SSL in terms of predictive performance for classification and clustering. We build the research upon the works in the literature that focuses on similar areas, as well as provide an understanding of disentanglement in terms of data representation in the latent space. We decide not to directly study the latent space because we are interested in the final learning performance. However, an ablation study focused on the latent representation could generate further insights and open space for new research.

Finally, it is worth considering the limitations in terms of computational power. We ran the experiments on the `Google Colab` platform that offers free computing resources. We could access one NVIDIA Tesla K80 GPU, with 25GB of RAM and 68GB of HDD. Also, we decided to avoid the usage of Google Cloud and AWS virtual machines. This choice was taken due to the high costs and the risk of interrupted executions because of limited bandwidth in spring 2020.

6.4 Future work

The research provides several contributions and the outcomes open space for further research about TL and SSL. An interesting direction of investigation is the unsupervised pre-training of the low layers in state-of-the-art architectures. We note that ResNet with no pre-training achieves better results than a simpler encoder network pre-trained with the AE paradigm. We believe that the pre-training of individual residual blocks of ResNet could be beneficial. Thus, we suggest investigating this topic and focus the study on the β-VAE as it benefits from cyclical annealing during training for learning a disentangled representation. Moreover, providing a comparison of different annealing functions could be useful to clarify the impact of the pre-training procedure.

Further research on the pre-training approaches derived from GANs could be beneficial to determine the effect of the source task in the TL paradigm. We believe that generative methods are the most promising approaches for pre-training. However, the difficulty and the instabilities in the training of GANs could limit their applications in real use case scenarios.

Finally, Deep Clustering methods achieve good results on image clustering thanks to the application of the β-VAE. The new Semi-Supervised pipeline significantly improves the results. We believe that studying the effect of the auxiliary supervised fine-tuning phase on the data representation in the latent space may indicate directions for research and further improvement. In addition, more work about the combination of the clustering loss and the re-

construction loss during the joint training of Improved DEC could increase the performance. We think that applying cyclical annealing to the parameter that combines the two losses during this final training step could prevent the risk of finding a suboptimal solution, as well as simplify the tuning process.

6.5 Benefits, ethics, and sustainability

The main benefit deriving from this thesis is that it proposes new ways to learn from unlabeled data, with no need of allocating financial and human resources for data labeling. This may also define new directions of research within the area of AI, in particular for TL and SSL.

From an ethical point of view, the thesis studies techniques to learn from unlabeled data, this could reduce the risk of violating the privacy of individuals and open new opportunities for research in the area of anonymized ML.

A solution that achieves good performance with the need of scarce labeled data will be able to reduce the number of energy resources dedicated to human labeling. Those resources could be allocated for more relevant tasks. In addition, a reduction in energy demand favours the long term sustainability by making companies and startups working in the area of AI more compliant to industry laws in terms of environmental impact.

6.6 Conclusions

The first motivation behind this thesis is about learning from labeled and unlabeled data. Data labeling is a critical process, so accessing large amounts of labeled samples is difficult, while unlabeled data are often abundant and easy to obtain. Therefore, we study how to learn from unlabeled and labeled samples belonging to the same distribution.

An initial investigation of the literature highlights the role of the unsupervised pre-training with AEs. One of the most promising methods is the β-VAE, proposed by Deep Mind in 2017. As it learns a disentangled representation from unlabeled samples, it is expected to find meaningful features during pre-training. We introduce cyclical annealing in the unsupervised training of the β-VAE and compare the results with those of a standard DAE, which achieves good performances for TL. The empirical metrics are also compared with a randomly initialized ResNet, a state-of-the-art architecture widely applied in CV. Surprisingly, ResNet with no pre-training achieves better results than simpler networks pre-trained with AEs on the unsupervised samples. We do not

consider external datasets for pre-training because we investigate the potential of the unlabeled data belonging to the same dataset as the labeled ones. In addition, external datasets could not be beneficial if there was a weak relation with the target dataset. It could happen in the case of very technical domains, like precision farming and predictive maintenance.

We also investigate approaches based on clustering in the Semi-Supervised setting. The goal is to improve the clustering performance when both labeled and unlabeled samples are available. We consider Deep Clustering because it is the most recent approach proposed for dealing with images. The success of clustering depends on the quality of the data representation, hence we define a new learning process based on the β-VAE to increase the disentanglement in the latent space. Moreover, an auxiliary supervised fine-tuning phase is designed to embed knowledge from the labeled samples available in the Semi-Supervised environment. The new Semi-Supervised approach improves the results provided in the literature of more than 7% in terms of final Clustering Accuracy on the CIFAR-10 dataset. In addition, the introduction of the β-VAE with annealed β during pre-training is successful also in a fully unsupervised scenario. With respect to a standard DAE, it increases the Clustering Accuracy of the DEC algorithm of 1% on the CIFAR-10 dataset. Moreover, we further analyze the impact of the complexity of patterns in the samples. It is possible to note that, given the same pre-training strategy, different algorithms achieve significantly different results on the MNIST digits dataset. On the other hand, for the CIFAR-10 as well as the Fashion-MNIST datasets, which both contain more complex images, relevant changes in the final performance depend mainly on the choice of the pre-training procedure. Thus, improving the quality of clustering requires to work on the latent space to find informative representations.

As mentioned in the initial discussion sections, this thesis open spaces for further experiments and studies in the areas of TL and SSL. We think that the first steps should follow the suggestions provided in Section 6.4. In particular, a promising direction is related to the layer-wise pre-training of the residual blocks of ResNet. Finally, we call for more experiments on low percentages of labeled data, so as to get more insights about scenarios with a few labeled samples.

Bibliography

[1] Carl Vondrick, Donald Patterson, and Deva Ramanan. "Efficiently scaling up crowdsourced video annotation". In: *International journal of computer vision* 101.1 (2013).

[2] Brenden M Lake et al. "Building machines that learn and think like people". In: *Behavioral and brain sciences* 40 (2017).

[3] Tadanobu Inoue et al. "Transfer learning from synthetic to real images using variational autoencoders for precise position detection". In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE. 2018.

[4] Xiaojin Zhu and Andrew B Goldberg. "Introduction to semi-supervised learning". In: *Synthesis lectures on artificial intelligence and machine learning* 3.1 (2009).

[5] Yoshua Bengio. "Deep learning of representations for unsupervised and transfer learning". In: *Proceedings of ICML workshop on unsupervised and transfer learning*. 2012.

[6] Dumitru Erhan et al. "Why does unsupervised pre-training help deep learning?" In: *Journal of Machine Learning Research* 11.Feb (2010).

[7] Dumitru Erhan et al. "The difficulty of training deep architectures and the effect of unsupervised pre-training". In: *Artificial Intelligence and Statistics*. 2009.

[8] Chengxu Zhuang et al. "Local Label Propagation for Large-Scale Semi-Supervised Learning". In: *arXiv preprint arXiv:1905.11581* (2019).

[9] Ahmet Iscen et al. "Label propagation for deep semi-supervised learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

[10] Sinno Jialin Pan and Qiang Yang. "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009).

[11] Erxue Min et al. “A survey of clustering with deep learning: From the perspective of network architecture”. In: *IEEE Access* 6 (2018).

[12] Junyuan Xie, Ross Girshick, and Ali Farhadi. “Unsupervised deep embedding for clustering analysis”. In: *International conference on machine learning*. 2016.

[13] Xifeng Guo et al. “Improved deep embedded clustering with local structure preservation.” In: *IJCAI*. 2017.

[14] Irina Higgins et al. “β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework.” In: *Iclr* 2.5 (2017).

[15] Avital Oliver et al. “Realistic evaluation of deep semi-supervised learning algorithms”. In: *Advances in Neural Information Processing Systems*. 2018.

[16] Peter Bock. *Getting it right: R&D methods for science and engineering*. Academic Press, 2001.

[17] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. “CIFAR-10 (Canadian Institute for Advanced Research)”. In: (). URL: `http://www.cs.toronto.edu/~kriz/cifar.html`.

[18] Han Xiao, Kashif Rasul, and Roland Vollgraf. “Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms”. In: *arXiv:1708.07747 [cs, stat]* (2017).

[19] Yann LeCun and Corinna Cortes. “MNIST handwritten digit database”. In: (2010).

[20] François Chollet et al. *Keras*. `https://keras.io`. 2015.

[21] Martin Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `http://tensorflow.org/`.

[22] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[23] Richard S Sutton and Andrew G Barto. “Reinforcement learning: An introduction”. In: (2011).

[24] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[25] Ian Witten et al. *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier, 2011.

[26] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. “Revolt: Collaborative crowdsourcing for labeling machine learning datasets”. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2017, pp. 2334–2346.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[28] Alexander Amini, Daniela Rus, and M Atarod. *Stochastic Gradient Descent for optimization*. URL: https://www.sciencemag.org/news/2018/05/ai-researchers-allege-machine-learning-alchemy.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012.

[30] Danilo P. Mandic and Jonathon Chambers. *Recurrent Neural Networks for Prediction: Learning Algorithms,Architectures and Stability*. John Wiley & Sons, Inc., 2001.

[31] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997).

[32] Thomas Huang. “Computer vision: Evolution and promise”. In: (1996).

[33] David Ha and Jürgen Schmidhuber. “World models”. In: *arXiv preprint arXiv:1803.10122* (2018).

[34] Yu Liu et al. “Exploring disentangled feature representation beyond face identification”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[35] John Wright et al. “Sparse representation for computer vision and pattern recognition”. In: *Proceedings of the IEEE* 98.6 (2010).

[36] Salman Khan et al. “A guide to convolutional neural networks for computer vision”. In: *Synthesis Lectures on Computer Vision* 8.1 (2018).

[37] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998).

[38] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[39] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

[40] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[41] Alireza Makhzani and Brendan Frey. “K-sparse autoencoders”. In: *arXiv preprint arXiv:1312.5663* (2013).

[42] Pascal Vincent et al. “Extracting and composing robust features with denoising autoencoders”. In: *Proceedings of the 25th international conference on Machine learning*. 2008.

[43] Carl Doersch. “Tutorial on variational autoencoders”. In: *arXiv preprint arXiv:1606.05908* (2016).

[44] Stefan Roth and Michael J Black. “Fields of experts: A framework for learning image priors”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 2. IEEE. 2005.

[45] Anand Rajaraman and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

[46] MathWorks. *Visualizing high dimensional data*. URL: `https://www.mathworks.com/help/stats/visualize-high-dimensional-data-using-t-sne.html`.

[47] Bock Hans-Hermann. “Origins and extensions of the k-means algorithm in cluster analysis”. In: *Journal Electronique d’Histoire des Probabilités et de la Statistique Electronic Journal for History of Probability and Statistics* 4.2 (2008).

[48] Ulrike Von Luxburg. “A tutorial on spectral clustering”. In: *Statistics and computing* 17.4 (2007).

[49] Mohammed J Zaki and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

[50] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[51] Rajat Raina et al. “Self-taught learning: transfer learning from unlabeled data”. In: *Proceedings of the 24th international conference on Machine learning*. 2007.

[52] Yoshua Bengio. “Deep learning of representations for unsupervised and transfer learning”. In: *Proceedings of ICML workshop on unsupervised and transfer learning*. 2012.

[53] Stergios Christodoulidis et al. “Multisource transfer learning with convolutional neural networks for lung pattern analysis”. In: *IEEE journal of biomedical and health informatics* 21.1 (2016).

[54] Xiaojin Zhu and Andrew B Goldberg. “Introduction to Semi-Supervised Learning”. In: *Synthesis lectures on artificial intelligence and machine learning* (2009).

[55] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-supervised learning*. MIT Press, 2006.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012.

[57] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. “Convolutional networks and applications in vision”. In: *Proceedings of 2010 IEEE international symposium on circuits and systems*. IEEE. 2010.

[58] Kaiming He, Ross Girshick, and Piotr Dollár. “Rethinking imagenet pre-training”. In: *Proceedings of the IEEE International Conference on Computer Vision*. IEEE. 2019.

[59] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. “Using pre-training can improve model robustness and uncertainty”. In: *arXiv preprint arXiv:1901.09960* (2019).

[60] Pierre Baldi. “Autoencoders, unsupervised learning, and deep architectures”. In: *Proceedings of ICML workshop on unsupervised and transfer learning*. 2012.

[61] Zhixin Shu et al. “Deforming autoencoders: Unsupervised disentangling of shape and appearance”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

[62] Tom Le Paine et al. “An analysis of unsupervised pre-training in light of recent advances”. In: *arXiv preprint arXiv:1412.6597* (2014).

[63] Chetak Kandaswamy et al. “Improving transfer learning accuracy by reusing stacked denoising autoencoders”. In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2014.

[64] Yoshua Bengio, Aaron Courville, and Pascal Vincent. “Representation learning: A review and new perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013).

[65] Harshvardhan Sikka et al. “A Closer Look at Disentangling in β-VAE”. In: *arXiv preprint arXiv:1912.05127* (2019).

[66] Yu Liu et al. “Exploring disentangled feature representation beyond face identification”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[67] Christopher P Burgess et al. “Understanding disentangling in β-VAE”. In: *arXiv preprint arXiv:1804.03599* (2018).

[68] Tian Qi Chen et al. “Isolating sources of disentanglement in variational autoencoders”. In: *Advances in Neural Information Processing Systems*. 2018.

[69] Mathieu Aubry et al. “Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models”. In: *CVPR*. 2014.

[70] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014.

[71] Xi Chen et al. “Infogan: Interpretable representation learning by information maximizing generative adversarial nets”. In: *Advances in neural information processing systems*. 2016.

[72] Joab R Winkler. *Numerical recipes in C: The art of scientific computing*. 1993.

[73] Hao Fu et al. “Cyclical annealing schedule: A simple approach to mitigating kl vanishing”. In: *arXiv preprint arXiv:1903.10145* (2019).

[74] Debsindhu Bhowmik et al. “Deep clustering of protein folding simulations”. In: *BMC bioinformatics* 19.18 (2018).

[75] John R Hershey et al. “Deep clustering: Discriminative embeddings for segmentation and separation”. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016.

[76] Mathilde Caron et al. “Deep clustering for unsupervised learning of visual features”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

[77] Bo Yang et al. “Towards k-means-friendly spaces: Simultaneous deep learning and clustering”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017.

[78] Chunfeng Song et al. “Auto-encoder based data clustering”. In: *Iberoamerican Congress on Pattern Recognition*. Springer. 2013.

[79] Zhuxi Jiang et al. “Variational deep embedding: An unsupervised and generative approach to clustering”. In: *arXiv preprint arXiv:1611.05148* (2016).

[80] Xifeng Guo et al. “Deep embedded clustering with data augmentation”. In: *Asian conference on machine learning*. 2018.

[81] Weihua Hu et al. “Learning discrete representations via information maximizing self-augmented training”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017.

[82] Ankita Shukla, Gullal Singh Cheema, and Saket Anand. “Semi-Supervised Clustering with Neural Networks”. In: *arXiv preprint arXiv:1806.01547* (2018).

[83] Yazhou Ren et al. “Semi-supervised deep embedded clustering”. In: *Neurocomputing* 325 (2019).

[84] Mohammad Peikari et al. “A cluster-then-label semi-supervised learning approach for pathology image classification”. In: *Scientific reports* 8.1 (2018).

[85] J. Deng et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*. 2009.

[86] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015).

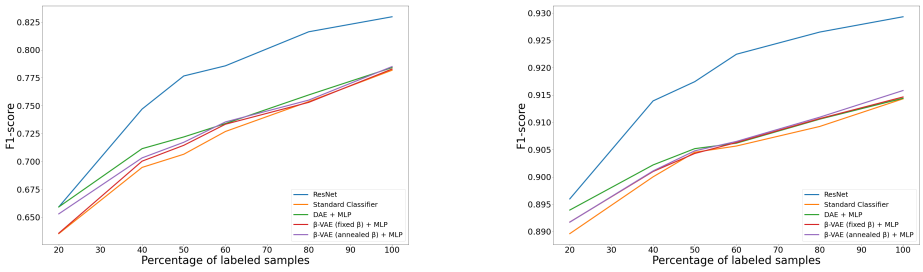
Appendix A

Appendix

Section A.1 shows the complete graphs related to the evaluation of TL with AEs. Section A.1.2 extends the previous results by adding the confusion matrices. In particular, we report those concerning the results when 20% and 100% of labeled samples are provided, as these are the most meaningful. Section A.2.1 and Section A.2.2 complete the results for Deep Clustering.

A.1 β-VAE applied to Transfer Learning

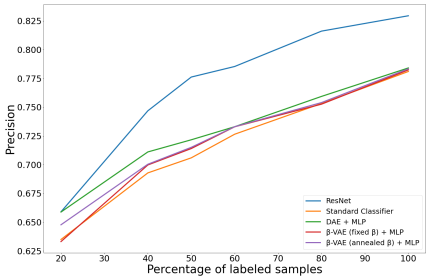
A.1.1 Supplement on experimental graphs



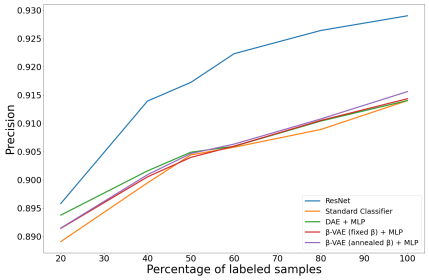
(a) F1-score on CIFAR-10.

(b) F1-score on Fashion-MNIST.

Figure A.1: F1-score results for TL with unsupervised pre-training.

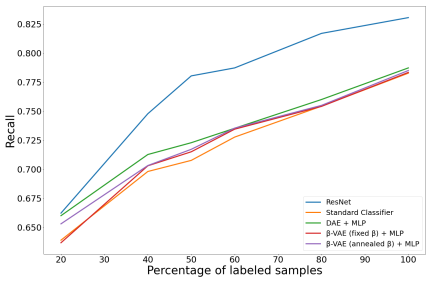


(a) Precision on CIFAR-10.

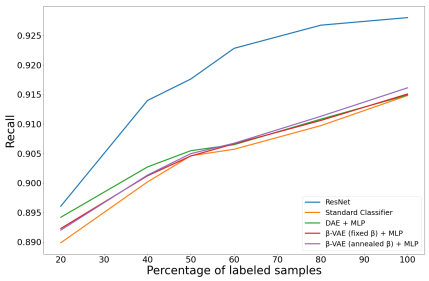


(b) Precision on Fashion-MNIST.

Figure A.2: Precision results for TL with unsupervised pre-training.



(a) Recall on CIFAR-10.

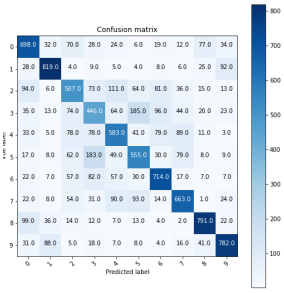


(b) Recall on Fashion-MNIST.

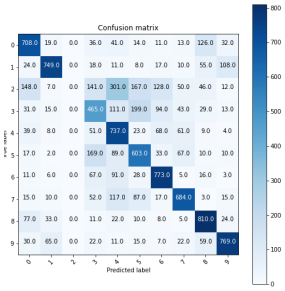
Figure A.3: Recall results for TL with unsupervised pre-training.

A.1.2 Supplement on confusion matrices

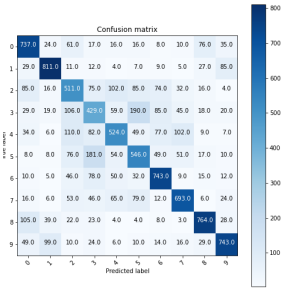
CIFAR-10 dataset with 20% of labeled samples



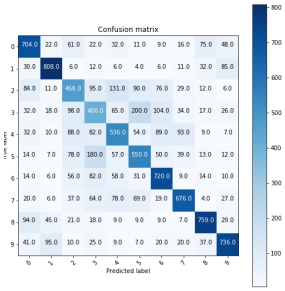
(a) ResNet.



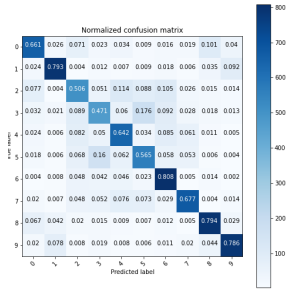
(b) Standard Classifier.



(c) β-VAE (annealed β) + MLP.



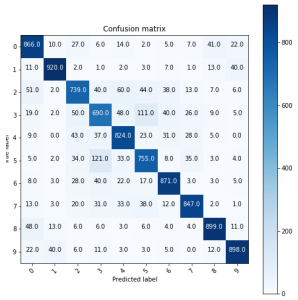
(d) β-VAE (fixed β) + MLP.



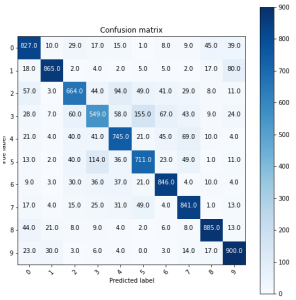
(e) DAE + MLP.

Figure A.4: Empirical results on the CIFAR-10 dataset when the 20% of the original labeled samples is retained.

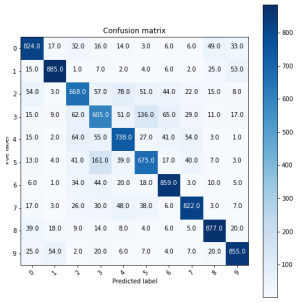
CIFAR-10 dataset with 100% of labeled samples



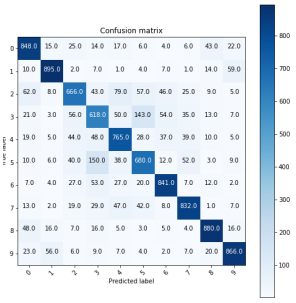
(a) ResNet.



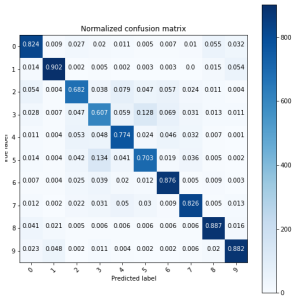
(b) Standard Classifier.



(c) β-VAE (annealed β) + MLP.



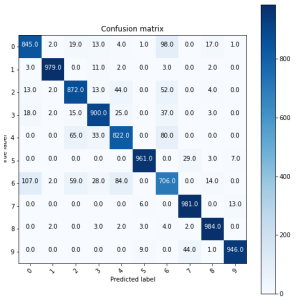
(d) β-VAE (fixed β) + MLP.



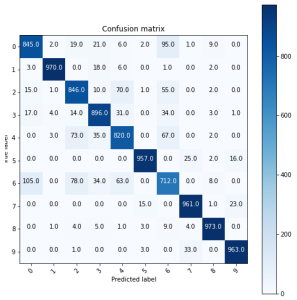
(e) DAE + MLP.

Figure A.5: Empirical results on the CIFAR-10 dataset when 100% of the original labeled samples is retained.

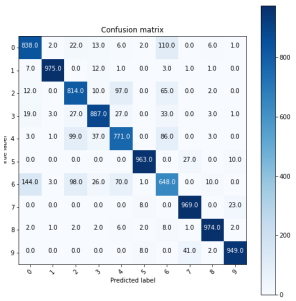
Fashion-MNIST dataset with 20% of labeled samples



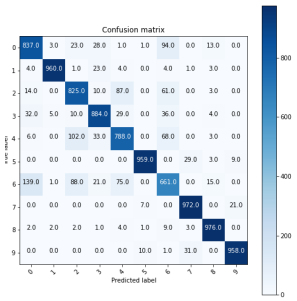
(a) ResNet.



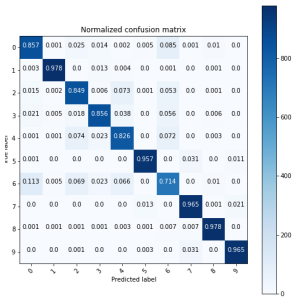
(b) Standard Classifier.



(c) β-VAE (annealed β) + MLP.



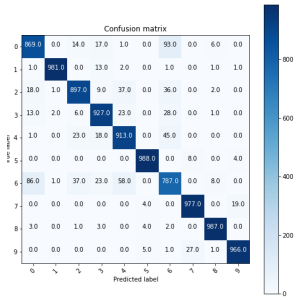
(d) β-VAE (fixed β) + MLP.



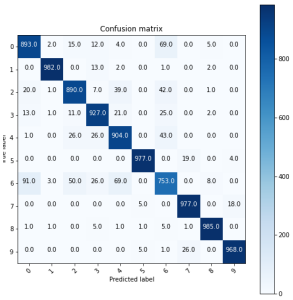
(e) DAE + MLP.

Figure A.6: Empirical results on the Fashion-MNIST dataset when the 20% of the original labeled samples is retained.

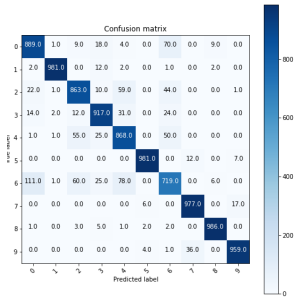
Fashion-MNIST dataset with 100% of labeled samples



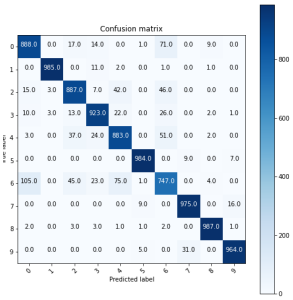
(a) ResNet.



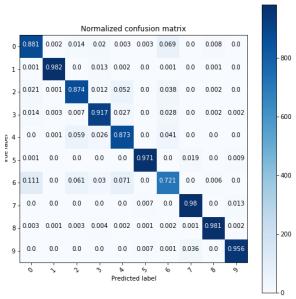
(b) Standard Classifier.



(c) β-VAE (annealed β) + MLP.



(d) β-VAE (fixed β) + MLP.



(e) DAE + MLP.

Figure A.7: Empirical results on the Fashion-MNIST dataset when 100% of the original labeled samples is retained.

A.2 Semi-Supervised Deep Clustering

A.2.1 Supplement on experiments on MNIST digits

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-means	0.94454 ± 0.00652	0.89512 ± 0.01170	0.57710 ± 0.00897
DAE + DEC	0.98298 ± 0.00177	0.95200 ± 0.00598	0.90666 ± 0.00853
DAE + Improved DEC	0.98354 ± 0.00278	0.95484 ± 0.00238	0.90750 ± 0.00436

(a) Experiments on 20% of labeled samples from MNIST digits.

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-means	0.95166 ± 0.00857	0.90842 ± 0.01108	0.58640 ± 0.00485
DAE + DEC	0.98566 ± 0.00084	0.95926 ± 0.00130	0.90966 ± 0.01195
DAE + Improved DEC	0.98686 ± 0.00153	0.96198 ± 0.00173	0.91106 ± 0.00438

(b) Experiments on 40% of labeled samples from MNIST digits.

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-Means	0.95668 ± 0.00919	0.91658 ± 0.00990	0.60068 ± 0.00506
DAE + DEC	0.98672 ± 0.00124	0.96108 ± 0.00227	0.91050 ± 0.01175
DAE + Improved DEC	0.98742 ± 0.00092	0.96284 ± 0.00188	0.91250 ± 0.01039

(c) Experiments on 50% of labeled samples from MNIST digits.

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-Means	0.95770 ± 0.00526	0.91792 ± 0.00617	0.60126 ± 0.01607
DAE + DEC	0.98712 ± 0.00107	0.96298 ± 0.00190	0.91348 ± 0.00526
DAE + Improved DEC	0.98812 ± 0.00040	0.96322 ± 0.00348	0.91518 ± 0.00823

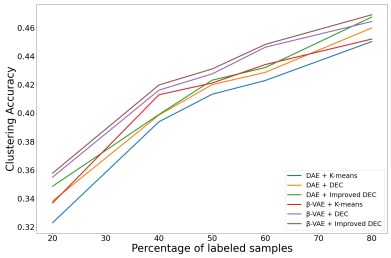
(d) Experiments on 60% of labeled samples from MNIST digits.

MNIST (digits)			
Model	ACC	NMI	SIL
DAE + K-Means	0.96002 ± 0.01127	0.91908 ± 0.01200	0.60538 ± 0.00997
DAE + DEC	0.98746 ± 0.00150	0.96354 ± 0.00303	0.91668 ± 0.01125
DAE + Improved DEC	0.98862 ± 0.00087	0.96604 ± 0.00284	0.91704 ± 0.01255

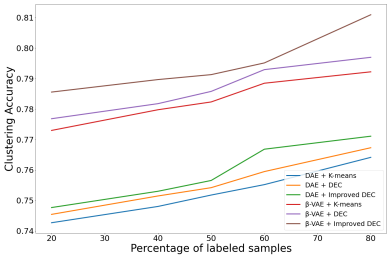
(e) Experiments on 80% of labeled samples from MNIST digits.

Table A.1: Results measured for the novel Semi-Supervised Deep Clustering framework on MNIST digits. After the unsupervised pre-training, the encoder network is fine-tuned on the available labeled samples.

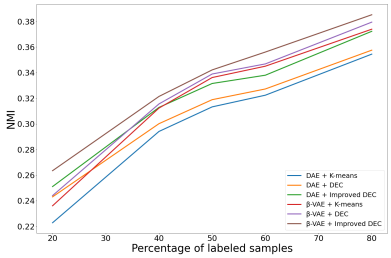
A.2.2 Supplement on experimental graphs



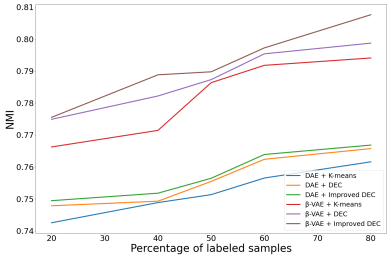
(a) Accuracy on CIFAR-10.



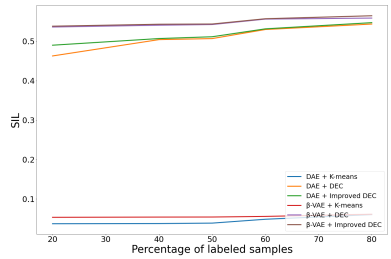
(b) Accuracy on Fashion-MNIST.



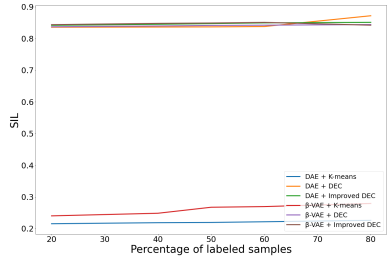
(c) NMI on CIFAR-10.



(d) NMI on Fashion-MNIST.



(e) Silhouette on CIFAR-10.



(f) Silhouette on Fashion-MNIST.

Figure A.8: The new Semi-Supervised approaches evaluated on both the datasets. For each clustering algorithm, the best results are often obtained through the methods built upon the β-VAE.