Inaugural-Dissertation

zur Erlangung der Doktorwürde der

Naturwissenschaftlich-Mathematischen
Gesamtfakultät

der

Ruprecht-Karls-Universität
Heidelberg

vorgelegt von

Timo Milbich

aus Karlsruhe, Baden-Württemberg

Tag der mündlichen Prüfung: _____

Visual Similarity and Representation Learning

by

Timo Milbich

Supervisor: Prof. Dr. Björn Ommer

Acknowledgements

First of all, I would like to thank my supervisor Prof. Dr. Björn Ommer, who gave me the chance to be part of his research group all these years. Thank you for the support, the advice, the patient discussions and everything that helped me to finally reach one of the biggest and probably most difficult milestones of my life. I would also like to thank Prof. Dr. Christoph Schnörr for his interest in my work and for his willingness to review my thesis.

I am deeply grateful for the friendships that I have built and deepened during the time of my PhD and that I hope will last far beyond. Thank you Lisa, Noah, Sabine, Uta, Fabrizio, Robert, Pablo and Biagio for all the moments of joy and laughter, tolerating my bad jokes, but most of all for supporting me in difficult moments and dark times. Thank you, Ameli, for always believing in me.

Finally, I would like to thank all my colleagues and people I worked with, without whom the time of my PhD would have been half the experience and probably would have driven me crazy. Thanks to Miguel, Nikolai, Karsten, Johannes, Patrick, Katja, Boris, Masato, Pamela, Nawid, Ekaterina, Angela, Artsiom, Robin and Barbara. Thanks Tobias for always letting me distract you with coffee breaks and discussions about the world. Special thanks to my office crew, Andreas and Micheal, with whom I probably spent more time than with anyone else in my last year – thanks for all the laughter, solidarity and great collaboration. I hope we will all meet again in one way or another. Thanks to all the people I met in the coffee kitchen for the interesting little chats about and off research – sometimes I did not even know your name, but I really appreciated these moments.

Abstract

Computer Vision aims to artificially mimic the visual reasoning capabilities of humans by using algorithms which, once deployed to mechanical agents and software tools, improve car and traffic safety, enable effective visual search on the World Wide Web, or increase productivity and quality in industrial production processes. Similar to the reasoning processes that are constantly occurring in our brains, such algorithms directly rely on abstract representations of the objects in the visually perceivable environment and beyond. Consequently, learning informative representations that allow to detect and recognize objects, and to evaluate image scenes is of paramount importance to almost all areas of computer vision.

The quality of a learned object representation typically depends on certain properties such as invariance to image noise, e.g. uninformative background, and robustness to object rotation, translation, or occlusion. In addition, many applications require representations that enable comparisons, i.e. to determine how similar or dissimilar two objects are semantically. However, arguably the most challenging aspect of learning object representations is ensuring generalization to unseen objects, object variations, and environments. While on the first aspects a large corpus of similarity learning literature exists, the latter, i.e. the generalization of object representations, is still poorly understood and thus rarely addressed explicitly.

In this thesis, we analyze the current field of similarity learning and identify properties of object representations that correlate well with their generalization performance. We leverage our findings and propose novel methods that improve current approaches to similarity learning, both in terms of data sampling and learning problem formulation. To this end, we introduce several training tasks that complement the prevailing paradigm of standard class-discriminative learning, which are eventually unified under the concept of Diverse Feature Aggregation. To optimally facilitate the optimization of similarity learning approaches, we replace the commonly used heuristic and predefined data sampling strategies with a learnable sampling policy that adapts to the training state of our model.

Typically, similarity learning finds applications in supervised learning problems. However, due to more training data becoming available and annotation processes often being tedious or even infeasible, unsupervised learning settings have been of particular interest in recent years. In the second part of this thesis, we explore the effectiveness of similarity learning for obtaining informative representations without the need for training labels for both static images and video sequences. To enable learning, our approaches alternate between inferring data relations during training and refinement of our visual representations. In doing so, we resort to the classic divide-and-conquer principle: we decompose overall complex learning problems into feasible local subproblems whose solutions are subsequently consolidated to yield concerted, global representations.

Throughout this work, we justify our contributions through rigorous analysis and strong model performance on standard benchmarks sets, often outperforming previous state-of-the-art results.

Zusammenfassung

Computer Vision hat das Ziel das visuelle Denkvermögen des Menschen künstlich mit Algorithmen nachzuahmen, welche eingesetzt in Software Tools und mechanischen Agenten die Sicherheit im Autoverkehr verbessern, effektive visuelle Suche im World Wide Web ermöglichen oder die Produktivität und Qualität in industriellen Produktionsabläufen steigern. Ähnlich wie die Denkprozesse, die ständig in unserem Gehirn ablaufen, stützen sich solche Algorithmen direkt auf abstrakte Repräsentationen der Objekte in der visuell wahrnehmbaren Umgebung und darüber hinaus. Folglich ist das Erlernen informativer Repräsentationen, die es ermöglichen Objekte zu detektieren und erkennen, sowie Bildszenen zu bewerten, von größter Bedeutung für fast alle Bereiche von Computer Vision. Die Qualität einer gelernten Objektrepräsentation hängt typischerweise von bestimmten Eigenschaften ab, wie die Invarianz gegenüber Bildstörungen, z. B. uninformativen Hintergrund, und der Robustheit gegenüber Objektrotation, -translation oder -verdeckung. Darüber hinaus benötigen viele Anwendungen Repräsentationen, die Vergleiche zwischen Objekten erlauben und somit bestimmen lassen, wie ähnlich oder unähnlich zwei Objekte semantisch sind. Der wohl schwierigste Aspekt beim Lernen von Objektrepräsentationen ist jedoch die Gewährleistung der Generalisierung auf ungesehene Objekte, Objektvariationen und Umgebungen. Während zu den erstgenannten Eigenschaften ein großer Korpus an Literatur zum Ähnlichkeitslernen vorliegt, ist der letzte Aspekt, d.h. die Generalisierung von Objektrepräsentationen, noch wenig verstanden und wird daher selten explizit behandelt.

In dieser Arbeit analysieren wir das aktuelle Feld des Ähnlichkeitslernens und identifizieren Eigenschaften von Objektrepräsentationen, die gut mit ihrer Generalisierungsleistung korrelieren. Wir nutzen unsere Erkenntnisse und schlagen neuartige Methoden vor, die aktuelle Ansätze des Ähnlichkeitslernens sowohl in Bezug auf die Datenauswahl als auch auf die Formulierung des Lernproblems verbessern. Zu diesem Zweck führen wir mehrere Trainingsaufgaben ein, die das vorherrschende Paradigma des standardmäßigen klassendiskriminierenden Lernens ergänzen und die schließlich unter dem Konzept der vielfältigen Merkmalsaggregation vereinheitlicht werden. Um die Optimierung von Methoden des Ähnlichkeitslernens bestmöglich zu unterstützen, ersetzen wir die üblicherweise verwendeten heuristischen und vordefinierten Daten-Sampling-Strategien durch eine lernfähige Sampling-Strategie, die sich an den Trainingszustand unseres Modells anpasst.

Typischerweise findet das Ähnlichkeitslernen Anwendung in überwachten Lernproblemen. Da jedoch immer mehr Trainingsdaten zur Verfügung stehen und Annotationsprozesse oft mühsam oder sogar undurchführbar sind, sind unüberwachte Lernsettings in den letzten Jahren immer mehr in den Fokus gerückt. Im zweiten Teil dieser Arbeit untersuchen wir daher die Nützlichkeit des Ähnlichkeitslernens zur Gewinnung informativer Repräsentationen ohne die Notwendigkeit von Trainingslabeln sowohl im Bereich der statischen Bilder als auch der Videosequenzen. Um das Lernen zu ermöglichen, alternieren unsere Ansätze zwischen der Inferenz von Trainings-Datenbeziehungen und die Verfeinerung unserer visuellen Repräsentationen. Dabei greifen wir auf das klassische Divide-and-Conquer-Prinzip zurück: Wir zerlegen komplexe Gesamtlernprobleme in einfachere lokale Teilprobleme, deren Lösungen anschließend zu globalen Repräsenta-

tionen vereinigt werden.
In dieser Arbeit rechtfertigen wir unsere Beiträge durch rigorose Analysen und eine starke Modellleistung bei Standard-Benchmarks, die oft die bisherigen State-of-the-Art-Ergebnisse übertreffen.

Contents

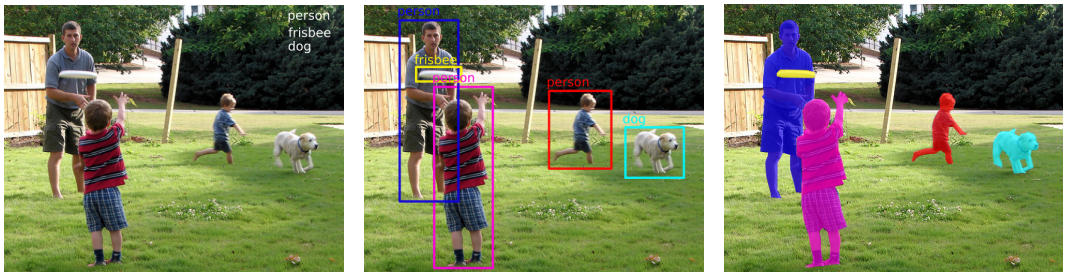
1 Introduction

1.1 Visual Reasoning and Computer Vision

Every day we navigate the world as we perceive and interact with our immediate environment. We recognize and discriminate among tens of thousands of objects [20, 78], read the faces and emotions of other people [104] and identify various types of dangerous situations on our quest for survival. To master these and many other essential tasks of visual reasoning, our brain is continually trained on a steady stream of visual input while simultaneously applying the already built-up knowledge. Beginning with the reception of visual sensory signals on our retina, we process and aggregate the information in our visual cortex [104], the basis of further interpretation and reasoning in various other areas of our brain [67]. Amazingly, despite the enormous complexity and vast amount of information to be processed, we perform these tasks effortlessly and usually even unconsciously [113] at almost every moment of our lives.

The goal of Computer Vision is to artificially mimic this process in mechanical agents and software tools, e.g. to improve safety in automobile traffic, enable effective visual search on the World Wide Web, increase productivity and quality in industrial production processes and aid in many more applications. To this end, algorithms must be developed that can solve the many tasks that constitute visual reasoning. Typical tasks are for instance object detection [64, 187, 190], classification [45, 91, 131] and object segmentation [75, 89], which are illustrated in Fig 1.1. An initial effort to construct a visual system solving such tasks and, thus, put Computer Vision on the map of academic research was launched in the early 1960s with the famous MIT 'Summer Vision Project' and, among others, the seminal work 'Machine Perception Of Three-Dimensional Solids' [192]. After failing to meet initially high expectations, Computer Vision found its first commercial application in the late 1970s with Optical Character Recognition (OCR). Thereafter, research gained momentum in the 1980s and set milestones, for instance with the Lucas-Kanade method for optical flow [10] (1981), the Canny Edge Detector [29] (1986) and the first Face Recognition system [230] (1991).

Nowadays, driven by breakthroughs in machine learning, Computer Vision lives up to early expectations by achieving superhuman performance in many recognition tasks, enabling applications in autonomous driving [106], medical imaging [39, 82] and robotics [35, 52]. The success and growing importance of Computer Vision for these and many other applications is ultimately enabled by increasingly more complex tasks coming within reach and eventually being solved. Ongoing advances in generative modeling, holistic scene understanding, temporal reasoning, and extensions to the 3D world are bringing



(a) Object classification (b) Object detection (c) Instance segmentation

Figure 1.1: *Examples of three main Computer Vision tasks.* (a) Image classification aims at predicting labels reflecting the objects depicted in an image. (b) Object detection additionally localizes the objects by means of bounding boxes. (c) Instance segmentation extends object detection to a pixel-level localization of objects, i.e. assigning pixel-wise class labels for each object detected.

Computer Vision ever closer to the goal of developing systems that exhibit a complete understanding of the visually perceivable environment and beyond.

1.2 The role and importance of object representations

A key factor of making sense of the massive amount and complexity of visual input is our ability to develop abstract *representations* in our brain [78, 135, 152] that describe the objects we encounter in the visual world. Typical visual cues that allow us to identify and distinguish between objects are for instance color and texture. However, since many objects, such as cars, exhibit large variations in these features, also geometrical properties describing object shape must be captured. Eventually, the closer distinct object categories are, the more complex and fine-grained representations are needed to enable effective recognition. Hence, object representations should describe objects on different levels of granularity and detail to ultimately serve as a basis to the various higher-level (visual) reasoning processes [67]. Moreover, to successfully and reliably enable these processes, object representations are required to exhibit the following characteristics [44, 53, 92, 231, 279]:
(i) Our world is dynamic and objects may be perceived in infinite variations. Therefore, consistent and accurate object recognition requires representations to be invariant to variations such as size, rotation, position, and background.
(ii) We are living in a continuous world and reasoning requires us to make situational choices and comparisons. Since these tasks typically involve the evaluation of objects, representations must allow for a notion of *how similar or dissimilar* objects are to others and also account for their own variations. Examples are the condition and value of an object to be purchased, whether or not foods are still consumable, estimating a person's mood or attitude towards us, and the evaluation of skills or actions.
(iii) Finally, the world is vast and ever-changing. The objects we encounter in our lives are not predefined from the start. We will always be discovering and learning new things and are constantly faced with situations we have never experienced before. Rapid evaluation

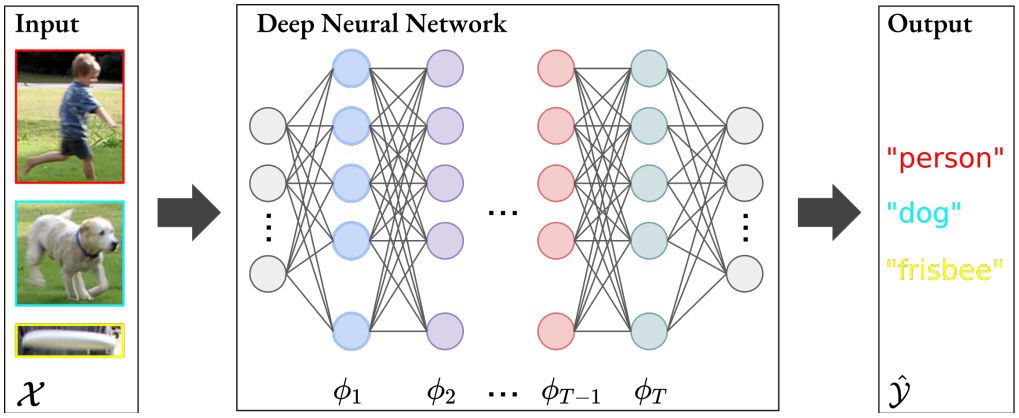


Figure 1.2: *Prototypical depiction of a (deep) neural network model.* Each layer $\phi_1, \ldots, \phi_T$ can be considered as an (intermediate) representation of the inputs $\mathcal{X}$, while the low-level representations are also often referred to as *features*. In essence, deep neural networks are a composition of simple, parametrized functions, e.g. convolutions or inner products, at large scale. Trained to yield outputs $\hat{\mathcal{Y}}$ by optimizing a given task such as object classification, the parameters of the network are updated via the backprogation algorithm [48, 199]. Due to the immediate connection between $\mathcal{X}$ and $\hat{\mathcal{Y}}$, the representations ϕ_t are *learned* to support the task at hand and this optimization procedure is called *end-to-end training*. For a detailed introduction to DNNs and Deep Learning in general, we refer the reader to the 'Deep Learning' book by Goodfellow et al. [79].

of and adaptation to the unknown is thus an important part of our interaction with the world, and was particularly crucial for survival in the earlier days of human existence. Consequently, the representations we rely on must also capture and *generalize* to novel variations of already known objects and their evaluations within new circumstances [11, 53] or even unknown objects.

1.2.1 Object representations in Computer Vision

Considering the human brain as a paragon for effective visual reasoning, almost any Computer Vision model today operates on abstract object representations. Designed to warrant the aforementioned properties, a large corpus of research on representation and similarity learning has emerged in recent decades. Early works proposed representations based on shallow, hand-crafted image features like pixel gradients [13, 16, 49, 147], color histograms [202] and local texture changes [2], thus predefining and fixing the visual patterns and features used to describe and compare objects. With the advent of Deep Learning and the powerful concept of backpropagation-based end-to-end training [48, 79, 199], the paradigm of finding object representations has changed. Nowadays, object representations are *learned*, i.e. explicitly optimized to yield flexible, deep hierarchies of features describing objects on multiple levels of granularity, *directly* solving a particular task. Typically, starting from low-level features that activate on similar patterns as the hand-crafted predecessors, features are gradually aggregated to more high-level and eventually holis-

tic descriptions of objects. Fig. 1.2 illustrates a prototypical depiction of a deep neural network (DNN) consisting of a sequence of fully connected layers. Similar sequential architectures which may consist of various, conceptually different layers, in particular convolution-based[1], are the backbone of almost every modern Computer Vision model. Today, there exist many different network architectures, consisting of increasingly more complex and more densely inter-connected functions, which are specialized to given tasks. Among the most widely used network families are Inception networks [222], residual neural networks (ResNets) [91], UNets [193], autoregressive and recurrent neural networks (RNNs) [40, 97] and generative networks [80, 123]. Equipped with millions of trainable parameters, these models can be trained on huge amounts of data, learn powerful representations, and thus contribute greatly to recent breakthroughs in Computer Vision.

1.2.2 Applications and challenges

Well-generalizing representations that effectively capture similarities and dissimilarities between objects have a strong impact on a broad range of Computer Vision applications. The most immediate application are image retrieval based tasks [145, 161, 241, 256]. For a given query image, these problems aim to select the most similar instances from large collections of images, thus they can also be considered as an equivalent formulation of image classification. In a similar vein, there are numerous related tasks such as face verification[41, 143, 205, 224, 230, 245], person (re-)identification [37, 211], human pose estimation[12, 24, 46, 154, 220] or image style transfer [128]. Moreover, while these tasks are often formulated to operate within a given data distribution, typically a fixed, predefined set of object classes, arguably the most important and challenging application is *transfer learning*. Transfer learning strives for finding representations that not only allow to describe and compare objects within the training distribution but in particular also generalize to novel data (cf. *(iii)* above), such as novel object variations, entirely unseen categories, or even different tasks. Depending on the exact training and evaluation setting, we distinguish between many flavors of transfer learning, e.g. (general) transfer of learned representations [36, 127, 131, 177, 253], few-shot learning [66, 125, 186, 235], domain adaptation [47, 69, 246] and zero-shot learning [170, 171]. While each setting has its own dedicated methods proposed over the past years, the most influential and general class of algorithms today for learning image representations is *Similarity or (Distance) Metric Learning*.

[1] Typically low-level features are learned by convolutional layers. These layers [79] exploit sets of shared parameterized convolution masks to circumvent the heavy computational burden when processing large input images. Consequently, the more widely used expression *convolutional neural networks (CNN)* emerged and is often used synonymously for deep neural networks in general.

1.3 (Deep) Visual Similarity and Representation Learning

Under the paradigm of end-to-end training, in deep neural networks a representation ϕ of an object depicted in image x is composed of a sequence of differentiable, parameterized functions $\phi_1, \dots, \phi_T$,

$$\phi(x;\theta) = (\phi_T \circ \phi_{T-1} \circ \cdots \phi_1)(x;\theta) = \phi_T(\phi_{T-1}(\cdots \phi_1(x;\theta_1)\cdots;\theta_{T-1});\theta_T) \ , \quad (1.1)$$

with trainable parameters $\theta = (\theta_1, \dots, \theta_T)$. Hence, the representation ϕ is adjustable to optimally support the solution to a given task, respectively the optimization of its corresponding learning problem (cf. Fig. 1.2). To this end, a representation learns to capture the information about input images and the depicted objects within in sufficient detail, as well as invariances to potentially unimportant factors such as background clutter, viewpoint, or occlusions. Consequently, the quality of an object representation, i.e. its ability to express similarities between objects, its robustness to noise and clutter, and also its generalization capabilities, directly depend on the mathematical formulation and optimization of the targeted learning problem.

1.3.1 Learning representations by optimizing visual tasks

Let $\{x_1, x_2, \dots x_N\} = \mathcal{X}$ be a set of D' dimensional training images. The goal is to learn a D-dimensional representation $\phi : \mathbb{R}^{D'} \rightarrow \Phi \subseteq \mathbb{R}^D$. Finding a visual representation ϕ is typically framed as a learning problem of a visual task in terms of a task-specific function $\mathcal{G}(\phi(x,\theta),\zeta)$, with $\mathcal{G}$ directly depending on the representation ϕ and additional task-specific parameter ζ. Such objectives are naturally formulated in DNN based learning frameworks (1.1), where θ denotes the trainable parameters of the (deep) representation ϕ and (if required) ζ the trainable parameters specialized to $\mathcal{G}$.
Learning the visual task, thus optimizing the model parameters (θ, ζ), is typically performed by matching its predicted output $\hat{y} = \mathcal{G}(\phi(x;\theta),\zeta)$ to an expected output $y \in \mathcal{Y}$ for the training images $\mathcal{X}$. Depending on the task at hand, y may be discrete object labels, coordinate locations in an image, pixel-wise scores, or any suitable space for modeling the learning problem. Thus, $\mathcal{G}$ expresses a relationship between the representation of images x and the output $\hat{y}$. A typical example for a task $\mathcal{G}$ is object classification, typically formulated to predict probabilities $\hat{y} = P(j|x)$ of an image x depicting a certain object class j by means of the softmax function[2] σ over K classes with parameters $\zeta = (\zeta_1, \dots, \zeta_K)$, with $\zeta_i \in \mathbb{R}^D$ for $i = 1, \dots, K$, i.e.

$$\mathcal{G}(\phi(x;\theta);\zeta) \triangleq \sigma(\phi(x;\theta);\zeta) = \frac{\exp(\zeta_j^\top \phi(x;\theta))}{\sum_{k=1}^{K} \exp(\zeta_k^\top \phi(x;\theta))} \ . \quad (1.2)$$

[2] In this case we assume the bias term, which is typically considered for classification problems, to be equal to zero. Consequently we can omit these terms.

where ζ_k can be interpreted as class representations. Formally, to train our model on the task $\mathcal{G}$, we optimize an objective function $\mathcal{L}$ suitable to match the predictions $\hat{y}$ and expected outputs y, i.e.

$$\min_{\theta,\zeta} \mathcal{L}(\hat{y}, y) \ . \tag{1.3}$$

Note, that (1.3) is in particular optimized for parameters θ. Hence, to optimally perform task $\mathcal{G}$, the representation ϕ needs to extract and capture informative features from the images $\mathcal{X}$. In most cases $\mathcal{Y}$ is provided by human annotated ground-truth information which constitutes the realm of *supervised learning*. However, $\mathcal{Y}$ may also be formulated to solely exploit inductive biases in the training data itself, which gives rise to other research areas such as *unsupervised learning*[3] [30, 36, 90, 165, 257] also see chapter 6 and 7) and *semi-supervised learning* [32, 121, 136, 172].

1.3.2 Deep similarity learning

Optimizing image classification tasks such as (1.2) on large and diverse datasets, such as ImageNet [50] ($\sim$ 1.2 million images), YFCC100M dataset [226] ($\sim$ 100 million flicker images and videos), etc., results in highly discriminative representations. Due to their particularly well-generalizing low-level features, they are the most widely used representations today for initializing Computer Vision models which accelerates and more easily enables further training on various kinds of tasks [127]. However, despite yielding universal low-level features, the actual set of objects to be recognized by these models is pre-defined and fixed. Thus, the naive application of such representations for directly recognizing and retrieving objects outside the training distribution is limited [51], diminishing desired generalization capabilities of the learned object representation ϕ. Moreover, training classification models are governed by the availability of discrete class labels. Consequently, their applicability to weaker or more abstract forms of supervision is restricted, e.g. in cases where only relative image relations can be induced or more continuous notions of similarity are to be learned [119].
Similarity learning addresses these shortcomings by directly optimizing the target representation space Φ to reflect similarity, respectively dissimilarity constraints between images – i.e., constraints specifying if samples are supposed to be similar or not. Given a distance function $d : \Phi \times \Phi \to \mathbb{R}_{\geq 0}$ (e.g. Euclidean or Mahalanobis distance), the mapping ϕ is optimized to embed similar images close together in Φ under d and dissimilar images far apart, reflecting some semantic concept of similarity, cf. Fig. 1.3. Typically Φ is also referred to as the *embedding space*. Hence, we optimize tasks $\mathcal{G}$ expressed as distance functions d between embedded training images $\phi(\mathcal{X}; \theta)$, i.e.

$$\mathcal{G}(\phi(x_1;\theta), \phi(x_2;\theta); \zeta) \triangleq d(\phi(x_1;\theta), \phi(x_2;\theta)) \ , \tag{1.4}$$

to implement the similarity constraints on Φ. Hence, in Eq. (1.4) we define the learning task $\mathcal{G}$ on relations between two or potentially even more samples. One of the most effective and arguably the most widely used category of objective functions $\mathcal{L}$ to optimize

[3] Often equivalently referred to as self-supervised learning.

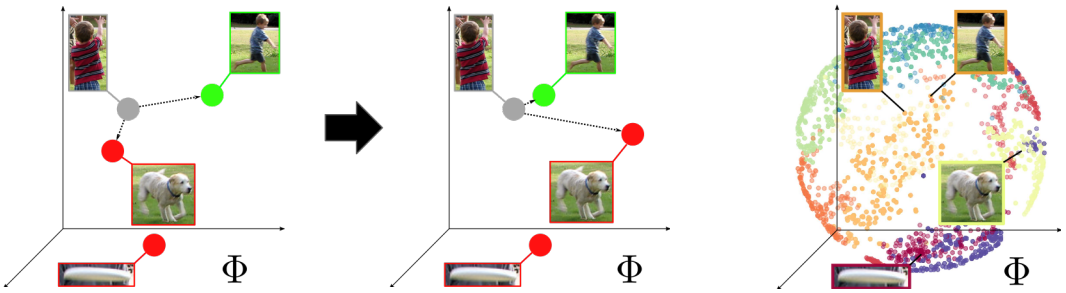


Figure 1.3: *The basic principle of similarity learning.* (a) The representation ϕ is optimized to map similar images close together, i.e. minimizing the distance between some anchor image (grey) and similar images (green) while maximizing its distance to negative images (red). (b) An example of a representation optimized using class membership as binary similarity constraints. Each class is represented by a different color. Similar classes should be close, dissimilar classes far from each other. Moreover, ϕ is regularized by restricting it to the surface of a three-dimensional hypersphere.

our distances are summarized as ranking-based approaches [171, 205, 217, 251, 256]. These methods directly optimize relative ordering constraints between tupels of images based on similarities defined by $\mathcal{Y}$. For instance, suppose $\mathcal{Y}$ allows to induce pairwise similarity scores $y_{ij} \in \mathbb{R}$ which quantifies how similar images x_i, x_j are, with higher scores indicating larger similarity. Using y_{ij} we can construct triplets of images $\{(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} \mid y_{ij} > y_{ik}\}$ to represent our ordering constraints. Choosing d to be the squared euclidean distance, the well-known *triplet loss* [34, 205] is now formulated as

$$\begin{aligned}\mathcal{L}(d(\phi(x_i;\theta),\phi(x_j;\theta)), d(\phi(x_i;\theta),\phi(x_k;\theta)); \mathcal{Y}) = \\ [||\phi(x_i;\theta) - \phi(x_j;\theta)||_2^2 - ||\phi(x_i;\theta) - \phi(x_k;\theta)||_2^2 + \beta]_+ \,.\end{aligned} \tag{1.5}$$

Here, the supervision provided by $\mathcal{Y}$ determines which distances to minimize and and to maximize and $[\cdot]_+ \triangleq \max(0, \cdot)$ denotes the hinge function. Typically, x_i is called anchor, x_j positive and x_k negative. The parameter β denotes a fixed, scalar margin which ensures that (1.5) only enforces constraints $y_{ij} > y_{ik}$ up to a certain degree in order to stabilize optimization. In the literature, similarity learning is naturally addressed by the field of Metric Learning, respectively, in the era of deep neural networks, by *Deep Metric Learning (DML)*. The subsequent chapter 2 provides a dedicated introduction to (Deep) Metric Learning by summarizing and comparing the most important objectives, as well the overall training pipeline of DML models.
Concluding, formulating $\mathcal{G}$ directly based on distance relations offers several benefits for both learning and inference. Among the most important ones are *(i)* similarity constraints are very general and, thus, can be derived from and combined with different kinds of supervision signals; *(ii)* depending on the granularity of $\mathcal{Y}$, we are able to exercise immediate control over the learned embedding ϕ which, for instance, ranges from implementing a clustered structure separating between object categories to total orderings on the training

samples $\mathcal{X}$; *(iii)* the learned embedding ϕ is naturally applicable to classification and image retrieval applications and is also defined for instances and on object classes outside the training data distribution. Moreover, due to the general nature of the contrastive learning signal, similarity learning can be used to formulate many other tasks, such as learning hashing functions [73, 169, 254], recommender systems [142, 203] and density estimation [84]. Finally, also the research area of unsupervised (low-level) representation learning is today heavily influenced by breakthroughs and techniques of similarity learning [36, 174].

1.4 Thesis Objective and Challenges

Benefiting from the advent of deep neural networks, representation and similarity learning have made remarkable progress with supervised, class-discriminative training being the leading learning paradigm. Training ever larger models on vasts amounts of data while still being able to effectively capture the training distribution, clearly demonstrates their excellent scaling and predictive capabilities. However, despite the tremendous success, such models are still subject to serious limitations, such as out-of-distribution generalization and strong label dependency. Motivated by these shortcomings the following research objectives are addressed in the scope of this thesis.

Standard discriminative training by design yields representations which are highly specialized to the training data. As a result, performance deteriorates with a growing gap between the distribution of training data and test data. However, as discussed in Sec. 1.2.2, representations should ideally equally well generalize to data outside the training distribution. Unfortunately, in the limit case, such out-of-distribution samples and classes may be completely unknown when learning our object representation. Therefore it is particularly challenging to formulate learning objectives that yield sufficiently expressive representations that also capture features of these instances. As a results, out-of-distribution generalization remains an open research problem to this day. Addressing this problem leads to the the main research question of this thesis: with only a limited amount of available training data, can we find training signals beside the current paradigm of class-discriminative learning to extend the generalization of object representations beyond the training distribution ? To answer this question, we analyze the current field of similarity learning, respectively Deep Metric Learning, to identify potential factors that drive out-of-distribution generalization. Based on this analysis, we aim to develop new approaches for similarity learning that advance the capabilities and performance of its current state. Since the lack of prior knowledge about the potential objects to be captured seriously hinders the provision of informative training supervision, we particularly aim to explore learning signals which do not rely on specialized annotation information.

In the second part of this thesis, we want to go further and explore the application of similarity learning to the realm of completely unsupervised training settings. Although the number of available training images is growing larger, providing models also with the typically required supervision information is tedious, costly and, thus, ultimately restricts the learning of stronger representations. Especially for complex data domains such as video sequences, providing annotations is often prohibitively expensive, preventing to tackle

certain tasks altogether. The research area of unsupervised representation learning is dedicated to address this problem. While specialized surrogate tasks are typically formulated as substitutes for labeling information, the adaptation of the more general framework of similarity learning to unsupervised representation learning is poorly explored. In this thesis we investigate this question and the applicability of similarity learning for unsupervised representation learning both for the domains of static images and video sequences. Without having access to labels or side information about data relations to use for training, we must carefully infer them ourselves. Given the large amount of potential relations to be exploited, the main challenge is to estimate which relations actually refer to true object similarities and dissimilarities and which are likely noise. Since only a small set of relations can typically be identified at the outset, our goal is to develop training curriculums that progressively refine and improve our object representations during training by carefully inferring and leveraging more and more reliable learning constraints.

1.5 Contributions

In collaboration with my great colleagues, the following publications emerged within the scope of this thesis:

- **Unsupervised Representation Learning by Discovering Reliable Image Relations**
  T.Milbich, O.Ghori*, B.Ommer*
  Pattern Recognition Journal (PR), Volume 102, June 2020

- **DiVA: Diverse Visual Feature Aggregation for Deep Metric Learning**
  T. Milbich, K. Roth*, H. Bharadhwaj, S. Sinha, Y. Bengio, B. Ommer†, J. Paul Cohen†*
  European Conference on Computer Vision (ECCV) 2020

- **Sharing Matters for Generalization in Deep Metric Learning**
  T.Milbich, K.Roth*, B.Brattoli, B.Ommer*
  Transactions on Pattern Analysis and Machine Intelligence (TPAMI), only online, has yet to appear in print.

- **Revisiting Training Strategies and Generalization Performance in Deep Metric Learning**
  K.Roth, T.Milbich*, S.Sinha, P.Gupta, B.Ommer, J.P.Cohen*
  International Conference on Machine Learning (ICML) 2020

- **PADS: Policy-adapted Sampling for Visual Similarity Learning**
  K.Roth, T.Milbich*, B.Ommer*
  Conference on Computer Vision and Pattern Recognition (CVPR) 2020

- **Unsupervised Video Understanding by Reconciliation of Posture Similarities**
  T.Milbich, M.Bautista, E.Sutter and B.Ommer
  International Conference on Computer Vision (ICCV) 2017

As indicated by (*), in particular Karsten Roth contributed under my supervision equally to various works, in particular helping with the implementation in large parts. In summary, this thesis comprises the following main contributions published in these works:

- Based on a study and comparison of the current state of Deep Metric Learning, we analyze driving factors of generalization of object representations. As a result, we identify correlations between generalization and certain structural properties of the learned embedding space. In particular, we uncover a strong relationship to the concept of representation compression.
- In agreement with the insights derived from the generalization analysis, the concept of shared features is introduced to similarity learning. These features target and capture patterns that are shared between training classes, thus increasing the expressiveness of representations. Moreover, effective ways to learn shared features and how to jointly optimize them with discriminative features are proposed and evaluated.
- An analysis of the distribution gap between training and test data and its influence on the performance of ranking-based similarity learning. Results considering both discriminative and shared features indicate a significant benefit of the latter to alleviate the overfitting problem of zero-shot classification.
- The idea of learning features that are complementary to the standard, discriminative learning paradigm is extended to capture various semantic concepts of similarity to further bridge the generalization gap. In particular, learning tasks targeting features across different object classes, features within object classes, and features independent of class assignments are designed and jointly optimized.
- Many deep ranking-based similarity learning approaches strongly rely on effective training data sampling strategies. To optimally support this class of algorithms, a novel data sampling policy is proposed which adapts to the learning state of DML models. Resorting to Reinforcement Learning (RL), the sampling policy is trained to directly enhance generalization performance. (Karsten Roth contributed equally to the development of the sampling strategy).
- New state-of-the-art results are achieved by the proposed novel DML models and model extensions across standard benchmark sets of varying scale and sample-to-class ratios.
- A novel approach to unsupervised representation learning to yield representations that transfer to down-stream tasks such as detection, segmentation, and recognition. The proposed method is based on techniques from similarity learning.
- An effective strategy to mine reliable, most likely correct, similarity constraints for learning without any need for supervision. In order to find such reliable relations, the overall learning problem is partitioned into and optimized on sub-problems. Subsequently, their solution is consolidated into a single, global solution. The

proposed model achieves competitive and new state-of-the-art results on standard benchmark datasets.

- Learning representations that are able to capture and compare video sequences are difficult to train due to the very expensive annotation process for temporal data. This thesis proposes an unsupervised approach to learn such representations for understanding human activity, based on similarity learning.

- In order to derive similarity constraints for temporal visual data, an integer linear problem for pairwise sequence matching of human postures is formulated. For efficient optimization, training sequences are split into sub-sequences. A deep neural network is then trained to consolidate the local correspondences into a consistent global representation.

1.6 Thesis Organization

Chapter 2 first reviews and empirically evaluates the most important aspects of common strategies for training DML models, including objective functions and crucial hyperparameter choices. Moreover, inconsistencies in training protocols in the literature are discussed and best practices for a fair comparison between different models are presented. Based on this analysis, metrics reflecting both the arrangement of samples projected into the embedding space and its captured information content are introduced. We show correlations between our metrics and the generalization performance of DML models which gives rise to driving factors for improving the transfer learning capabilities of such approaches. Finally, exploiting our observations, we present a simple regularization technique that improves the performance of ranking-based DML models.
Chapter 3 further makes use of the insights from chapter 2 to improve generalization in similarity learning. To this end, the concept of shared features is introduced which aims at capturing commonalities across training classes to improve the expressiveness of learned embedding representations. Moreover, we show how to efficiently learn such features next to the classical discriminative training paradigm. A detailed analysis and evaluation of the presented approach verifies the benefit of incorporating shared features for generalization in similarity learning.
Chapter 4 extends the idea of chapter 3 and proposes a multi-task similarity learning framework. We introduce and jointly optimize several novel tasks that yield mutually complementary features to further increase the expressiveness and, thus, the generalization of representations. In particular, learning tasks targeting discriminative, inter-class, intra-class, and sample-specific features are examined. For the latter, we review and analyze the utility of recent self-supervised learning approaches. Experiments on standard benchmark datasets prove the effectiveness of the proposed framework.
Chapter 5 addresses the data sampling aspect of ranking-based DML methods which is crucial for optimizing such models. In contrast to prevailing static sampling heuristics, a dynamic and learned sampling policy is presented. Using Reinforcement Learning the policy is continuously updated to adapt to the learning state of the DML model during

training. Moreover, by optimizing the policy on a dedicated validation set, its effectiveness is further increased. Experiments show the general applicability and utility of our method across different DML approaches, backbone architectures, and datasets.
Chapter 6 applies similarity learning to the task of unsupervised representation learning. To compensate for lacking ground-truth data relations typically required for training, a strategy for mining reliable estimates for such similarity relations is developed. To this end, the overall learning problem is partitioned into subproblems defined on subsets of reliable relations. First, each of them is solved independently by an alternating optimization procedure. Subsequently, these solutions are consolidated to a single, overall representation exploiting inter-relations between the subproblems. We evaluate the utility of this representation for downstream tasks such as classification, detection, and segmentation and demonstrate strong results on each task.
Chapter 7 presents an approach for learning similarities between video sequences at the example of human activities. Contrary to holistic video representations operating on action labels, e.g. learned by standard action recognition, frames are encoded independently and we represent video sequences as trajectories in a fine-grained activity space. Due to the lack of labels, the approach is formulated as an unsupervised learning problem and solved by alternating optimization between identifying frame-wise similarity relations and capturing them in the activity space. Sequence matching is formulated as a novel Integer Linear Program (ILP) and representation learning is optimized using techniques from DML. Activity is finally captured and understand by means of a recurrent neural network operating on activity representations. Experiments including zero-shot pose retrieval, temporal super-resolution, and action synthesis verify the capabilities of the presented method.
Chapter 8 concludes the thesis with a final discussion.

2 Deep Metric Learning: Training Strategies and Generalization

Learning visual similarity is important for a wide range of vision tasks, such as image clustering [22], face detection [205] or image retrieval [256]. Measuring similarity requires learning an embedding which captures images and reasonably reflects their similarities by means of a predefined distance metric. One of the most adopted classes of algorithms for this task is *Deep Metric Learning (DML)* which leverages deep neural networks to learn such a distance metric preserving embedding.
Due to the growing interest in and influence of DML on representation learning in general, a large corpus of literature has been proposed contributing to its success. However, as recent DML approaches explore more diverse research directions such as architectures [105, 261], objectives functions [250, 266] and additional training tasks [141, 194], an unbiased comparison of the impact of such factors becomes more and more difficult. Furthermore, undisclosed technical details of published methods (e.g. data augmentations or training regularization) pose a challenge to the reproducibility of these models, which is of great concern in the machine learning community in general [23].
The first part of this chapter aims at establishing a transparent training and evaluation protocol for fair comparison of DML approaches. We provide an introduction to deep metric learning while briefly summarizing the transition of early linear metric learning approaches to the deep variants mostly employed today. Subsequently, we identify and discuss crucial design choices and hyperparameters of common DML training strategies, which greatly influence the success and performance of these models. To this end, we conduct a comprehensive study examining their impact on model performance and compare current DML baselines under identical training conditions on standard benchmark datasets.
On that basis, we extend our analysis in the second part of this chapter to examine generalization in DML more closely and analyze its connection to the structure of the embeddings learned by these models. In particular, we examine typically applied concepts when learning representations such as *(i)* enforcing large inter-class margins [45, 51, 143], *(ii)* maintaining intra-class variance [141] and *(iii)* compression of the learned representations. In summary, our most important contributions in this chapter can be described as follows:

- We provide an exhaustive analysis of recent DML objective functions, their training strategies, the influence of data-sampling, and model design choices to set a standard benchmark. To this end, we made our code publicly available.

- We provide new insights into DML generalization by analyzing its correlation to the embedding compression (as measured by its spectral decay), inter-class margins and intra-class variance.

- Based on the result above, we propose a simple technique to regularize the embedding compression which we find to boost generalization performance of ranking-based DML approaches.

This chapter is based on our publication *'Revisiting Training Strategies and Generalization Performance in Deep Metric Learning'* [196].

2.1 Preliminaries: From Linear to Deep Metric Learning

Subsequently, we briefly introduce the idea of metric learning and how to learn a similarity preserving distance function.

2.1.1 Learning a distance metric

Learning a metric space reflecting similarity between D'-dimensional training data points $x_i \in \mathcal{X}$ requires to find a corresponding distance function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. For any $x_1, x_2, x_3 \in \mathcal{X}$, a distance function d fulfills

$$\begin{aligned} &d(x_1, x_2) \geq 0 \wedge d(x_1, x_2) = 0 \Leftrightarrow x_1 = x_2 && \text{(identity of indiscernibles)} \\ &d(x_1, x_2) = d(x_2, x_1) && \text{(symmetry)} \\ &d(x_1, x_2) \leq d(x_1, x_3) + d(x_3, x_2) && \text{(triangle inequality)} \end{aligned} \tag{2.1}$$

To learn such a distance function d, we resort to parametrized functions, whose parameters θ are optimized during training (1.3). In practice, enforcing the conditions (2.1) during optimization is tedious, in particular when using deep neural networks to represent d. Instead, we typically choose d to be a predefined metric function, e.g. the Euclidean or Cosine distance, and learn a suitable transformation of our data points x_i such that d reflects provided similarty relations $\mathcal{Y}$, as already mentioned in 1.3.2. Let us know discuss the basic linear metric learning problem formulations, their non-linear extension and the transition to Deep Metric Learning.

2.1.2 (Linear) Mahalanobis metric learning

For a chosen metric d on a target space $\mathbb{R}^D$, the general goal of metric learning is to learn a transformation $\phi \colon \mathbb{R}^{D'} \to \Phi \subset \mathbb{R}^D$ of our data points $x_i \in \mathcal{X}$ such that the available semantic relations, implicitly defined by some supervision information $\mathcal{Y}$, are captured in the pairwise distances $d_\phi(x_i, x_j) = d(\phi(x_i), \phi(x_j))$.
The simplest class of models for the metric learning problem is given by requiring ϕ to be a linear transformation. The most prominent and arguably earliest representative is

Mahalanobis metric learning [252] which is often used equivalently for the general class of linear metric learning in literature. The Mahalanobis distance is defined as

$$d_{\text{mahal}}(x_i, x_j) = \sqrt{(x_i - x_j)^\top \Sigma^{-1}(x_i - x_j)} \tag{2.2}$$

with Σ being the sample covariance matrix (assumed to be positive definite) estimated from the given data $\mathcal{X}$. While originally introduced to quantify the distance between a data point and some distribution[1], (2.2) measures the pairwise distance between datapoints while taking the covariance into account. Equivalently, the Mahalanobis distance can be considered as the Euclidean distance between whitened datapoints $\tilde{x} = \Sigma^{-1/2}(x - \mu)$. General linear metric learning now extends the distance (2.2) allowing an arbitrary positive semi-definite matrix $\boldsymbol{A}$,

$$d_{\boldsymbol{A}}(x_i, x_j) = \sqrt{(x_i - x_j)^\top \boldsymbol{A}(x_i - x_j)}\ , \tag{2.3}$$

Thus we can factorize $\mathbf{A} = \mathbf{G}^\top\mathbf{G}$, resulting in the linear distance metric $d_{\mathbf{A}}(x_i, x_j) = ||\mathbf{G}x_i - \mathbf{G}x_j||_2$ with the linear data transformation $\phi(x;\theta) = \mathbf{G}x$, where the learnable model parameter θ represent the entries of the matrix $\mathbf{G}$.
Considering the supervised learning setting, we typically learn (2.3) using a set of pairwise constraints which, depending on the supervision signal $\mathcal{Y}$, may be given by scores directly inducing quantitative pairwise similarity/dissimilarity constraints or relative ranking constraints between data points $\mathcal{X}$. As already introduced in the introduction, the latter is often represented by means of triplets $t = (x_i, x_j, x_k) \in \mathcal{T}_{\mathcal{X}} \subset \mathcal{X} \times \mathcal{X} \times \mathcal{X}$, where $\mathcal{T}_{\mathcal{X}}$ consists of those triplets t for which x_j is supposed to be more similar to x_i than x_k. Thus, we can formulate learning (2.3) as a regularized optimization problem

$$\begin{aligned} &\min_{\boldsymbol{A}\succcurlyeq 0} \quad r(\boldsymbol{A}) \\ &\text{subject to} \quad d_{\boldsymbol{A}}(x_i, x_j) < d_{\boldsymbol{A}}(x_i, x_k) - \beta \quad \forall t \in \mathcal{T}_{\mathcal{X}}\ , \end{aligned} \tag{2.4}$$

where $\boldsymbol{A} \succcurlyeq 0$ is a shorthand notation for the requirenment of $\boldsymbol{A}$ to be positive-semi definite. Similar to (1.5) β denotes the a predefined margin and r is a regularization on $\boldsymbol{A}$, such as the Frobenius norm $g(\boldsymbol{A})$, i.e.

$$g(\boldsymbol{A}) = \frac{1}{2}\|\boldsymbol{A}\|_F^2 = \frac{1}{2}\sum_{i,j=1}^{D'} (\boldsymbol{A}_{ij})^2\ . \tag{2.5}$$

This norm is commonly used in other machine learning algorithms like ridge regression [98] or support vector machines [45]. The selected regularization $r(\boldsymbol{A})$ has important implications for both the optimization process and properties of the resulting distance metric. Thus, depending on the choice of r, various different algorithms for learning

[1] In this case the distribution is assumed to be gaussian $\mathcal{N}(\mu, \Sigma)$ and we replace the second datapoint x_j in (2.2) with the distribution mean μ.

$d_{\mathbf{A}}(x_i, x_j)$ have been proposed [134].
A popular example of metric learning algorithms for problem (2.4) is the Large Margin Nearest Neighbour classification (LMNN) model proposed by Weinberger et al. [252]. This approach is learned from both direct similarity constraints $(x_i, x_j) \in \mathcal{S} \subset \mathcal{X} \times \mathcal{X}$ given as pairs of data points which are supposed to be similar and relative triplet ranking constraints $\mathcal{T}_{\mathcal{X}}$ and is formulated as

$$\min_{\boldsymbol{A} \succcurlyeq 0} \sum_{(x_i, x_j) \in \mathcal{S}} d_{\boldsymbol{A}}(x_i, x_j) + \lambda \sum_{t \in \mathcal{T}_{\mathcal{X}}} [1 + d_{\boldsymbol{A}}(x_i, x_j) - d_{\boldsymbol{A}}(x_i, x_k)]_+ \tag{2.6}$$

with $[\cdot]_+ = \max(0, \cdot)$ being the hinge function.

2.1.3 Non-linear metric learning

Kernelization. Linear transformations represent weighted combinations of the data points x, which are consequently required to already provide a sufficient data description for $d_{\boldsymbol{A}}$ to be able to reflect similarity induced by $\mathcal{Y}$. However, in practice this is generally not the case and linear models quickly fail to apply to complex data, such as high-dimensional images, as the famous XOR toy problem [26] demonstrates. Consequently, non-linear data transformations $\phi(x)$ are required to tackle complex data domains of real-world applications.
Since the Euclidean distance in its squared form can be formulated by means of inner products, i.e. $||x_i - x_j||_2^2 = (x_i - x_j)^\top (x_i - x_j) = x_i^\top x_i + 2x_i^\top x_j - x_j^\top x_j$ a straight-forward extension of linear models to yield non-linear transformations, is the *kernelization of linear models*. The key idea of kernel methods [209] is to replace linear inner products $x^\top y$ with non-linear kernel functions $\kappa(x, y) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with

$$\kappa(x, y) = \phi(x)^\top \phi(y) \ , \tag{2.7}$$

thus assuming Φ to be an inner product space. It follows that we do not explicitly need to evaluate or even to know the mapping $\phi(x)$, but only implicitly by evaluating the kernel function $\kappa(x, y)$. In order for a kernel function $\kappa(x, y)$ to be valid, it has to satisfy the Mercer condition [209]: Any kernel matrix $\boldsymbol{K} = (\kappa(x_i, x_j)))_{ij}$ of kernel function values defined over $\mathcal{X}$, must always be positive semi-definite. In particular, ϕ can be a complicated, non-linear mapping into a arbitrarily high-dimensional embedding space Φ while we are still able to compute $\kappa(x, y)$ efficiently. The substitution of inner products $x^\top y$ with $\kappa(x, y)$ is known as the *kernel trick* and often also applied to extend other linear methods such as support vector machines [45] or principle component analysis [112].
As a result, we can solve linear metric learning problems such as (2.4) in a kernel space Φ using the now implicit, non-linear transformation ϕ, thus obtaining a distance function $d_{\boldsymbol{A}}(x_i, x_j)$ of form $||G\phi(x_i) - G\phi(x_j)||_2^2$. The distance between novel query samples can in general be computed by $\mathcal{O}(n)$ kernel evaluations over the training samples $\mathcal{X}$. For a detailed introduction to kernelized metric learning and its optimization, we refer the reader to [134].

(Deep) Neural Representations. Recent advances in training deep neural networks show that complex, highly non-linear data transformations can be learned effectively for a large range of machine learning tasks and applications. Thus, representing the data transformation ϕ by means of a deep learning model is a natural extension for metric learning problems of the form (2.4). Analogously, this deep metric learning problem (DML) is then formulated as

$$\begin{aligned} &\min_{\theta} \quad r(\theta) \\ &\text{subject to} \quad d_\phi(x_i, x_j) < d_\phi(x_i, x_k) - \beta \quad \forall t \in \mathcal{T}_\mathcal{X} \ . \end{aligned} \tag{2.8}$$

where θ are now the parameters associated with the network architecture of choice. Moreover, we now denote the distance function to be learned as $d_\phi(x_i, x_j) = ||\phi(x_i) - \phi(x_j)||_2$[2] to indicate the direct dependence on the deep representation. Note that similarity and ranking constraints are basically identical and unchanged. The learning problem (2.8) is referred to as triplet loss learning in DML literature [205] (cf. (1.5) in chapter 1) and has since been subject to several extensions [37, 256].

Discussion. For visual similarity learning, implementing ϕ as a deep representation has several advantages over the kernelized linear variant. In practice kernel methods restrict the choice of the embedding ϕ due to the Mercer condition, which may not optimally capture the data to be represented. Moreover, due to the high dimensionality of the image domain, we typically need to resort to predefined intermediate representations of x. In contrast, deep models allow for learning flexible representations which can be trained to directly support the target metric learning problem. Moreover, evaluating the kernelized distance function $d_{\boldsymbol{A}}$ for arbitrary pairs of data points requires to evaluate kernel functions over the full training set. Thus, given the scale of problem sizes nowadays, including up to millions of images, evaluation of $d_{\boldsymbol{A}}$ may be computationally expensive. Although, deep models can also exhibit costly inference times due to their vast amount of parameters, it does not scale with the training set size. Finally, the massive amount of training parameters are arguably also their strongest advantage, as it allows to learn extremely powerful representations.
Effective learning of such well-generalizing neural distance functions d_ϕ, however, requires carefully designed training strategies, which in particular includes tuning and regularization of the deep learning model. Next, we identify and introduce crucial parts of their training pipeline followed by an analysis of their impact on performance by comparing the recent publications of deep metric learning baseline methods.

2.2 Training a Deep Metric Learning Model

In this section, we summarize key components, hyperparamters and their options for training DML models, which we subsequently analyze in an empirical study. We can

[2] While in chapter 1 we use $\phi(x; \theta)$ to denote the deep embedding function, we now drop the explicit dependence on θ for simplicity reasons.

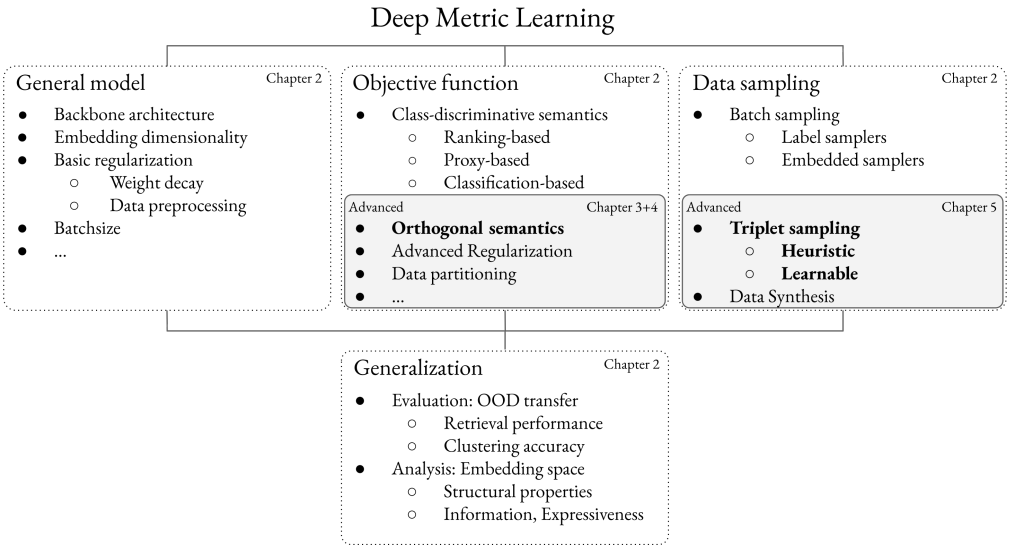


Figure 2.1: *Design taxonomy of Deep Metric Learning models.* In this thesis, we decompose and categorize models on similarity learning, respectively Deep Metric Learning, using the depicted taxonomy. Throughout this work, we address several of those model building blocks (i.e. general model, objective function and data sampling) in various chapters. The remainder of this chapter introduces, summarizes and evaluates most of the basic components of DML models. Advanced components highlighted in the grey boxes (**bold**) are addressed in subsequent chapters. Tuning and improving the different building blocks aims at increasing the generalization capabilities of DML models. To this end, in this chapter, we both thoroughly evaluate the impact of different basic model components based on a transparent training and testing protocol, and subsequently empirically analyze desirable and beneficial properties of well-generalizing embeddings.

roughly group these elements into *(i) training objective*, i.e. the main objective function addressing the core metric learning problem and potential extensions to it, which, for instance, influence the features to be learned or regularizations; *(ii) data sampling strategies* including batch sampling strategies and negative sampling strategies crucial for learning many ranking-based approaches. *(iii) general Deep Metric Learning training components* such as the backbone architecture of a DML model, its general regularization or data preprocessing. Fig. 2.1 provides a taxonomy of these building blocks. In the following, we discuss basic DML model components, which constitute and affect most DML training pipelines, significantly impact their performance and exhibit an increased divergence in the field, thus impairing objective comparisons between proposed approaches.

2.2.1 The objective function

In Deep Metric Learning we learn an embedding function ϕ which allows to measure the similarity between datapoints $x_i, x_j \in \mathbb{R}^{D'}$ typically as the Euclidean distance $d_\phi(x_i, x_j) = ||\phi(x_i) - \phi(x_j)||_2$ between their projections into Φ. The corresponding embedding func-

tion ϕ is formulated as a deep neural network parametrised by θ. To stabilize training [100, 256], the embedding space Φ is often regularized to the real hypersphere, i.e. we choose

$$\Phi = \mathbb{S}^D = \{z \in \mathbb{R}^D : \|z\|_2^2 = 1\} \,. \tag{2.9}$$

Moreover, we assume a supervised training setting where our distance function d_ϕ is trained to reflect semantic similarity defined by given sample-wise class labels $y_i \in \mathcal{Y}$, thus we want samples of the same class to be similar while being dissimilar for other classes. Objective functions formulated to learn such an embedding function can be roughly categorized into ranking-based, proxy-based and classification-based approaches.

Ranking-based DML. The most widely used family of DML are ranking-based loss functions operating on pairs [86], triplets [205, 256, 265] or larger sets of datapoints [37, 171, 217, 250]. Learning ϕ is defined as an ordering task, such that the distances $d_\phi(x_i, x_j)$ between an anchor x_i and positive x_j of the same class, $y_i = y_j$, is minimized and the distances $d_\phi(x_i, x_k)$ to negative samples x_k with different class labels, $y_i \neq y_k$, is maximized. For example, triplet-based formulations (cf. (1.5)) typically optimize their relative distances as long as a margin β is violated, i.e. as long as $d_\phi(x_i, x_k) - d_\phi(x_i, x_j) < \beta$. Another prominent example of instance-based deep metric learning is the margin loss [256], which introduces an additional dynamic, learnable margin α,

$$\mathcal{L}_{margin} = [(-1)^{\mathbf{1}_{y_i \neq y_j}} (d_\phi(x_i, x_j) - \alpha) + \beta]_+ \,. \tag{2.10}$$

Here $\mathbf{1}_a$ denotes the indicator function given the condition a. Moreover, a state-of-the-art representative of ranking-losses considering multiple negatives and positives at once for a given anchor x_i is the multi-similarity loss [251]. Suppose $\mathcal{P}$ is a set of indices denoting positive training instances for x_i and $\mathcal{N}$ a set of indices denoting negative training instances. The multisimilarity loss $\mathcal{L}_{\text{multisim}}$ is then formulated as

$$\begin{aligned} \mathcal{L}_{multisim} = \; & \frac{1}{\alpha_1} \log[1 + \sum_{j \in \mathcal{P}} \exp(\alpha_1 (d_\phi(x_i, x_j) + \beta))] \\ & + \frac{1}{\alpha_2} \log[1 + \sum_{k \in \mathcal{N}} \exp(-\alpha_2 (d_\phi(x_i, x_k) + \beta))] \,, \end{aligned} \tag{2.11}$$

where each negative and positive sample is exponentially weighted depending on its distance to x_i. Further, the constants $\{\alpha_1, \alpha_2, \beta\}$ are predefined, fixed hyperparameters controlling the optimization process.

The performance of ranking-based approaches is particularly sensitive to sampling strategies for negative data points in, e.g. triplets $\mathcal{T}_\mathcal{X}$ or the negative set $\mathcal{N}$, as these sets can only represent small parts of the data distribution[3], thus determining the features to be learned. Consequently, finding the most effective strategies is an active field of research [71, 88, 195, 205, 256, 262]. Chapter 5 is dedicated to this problem and provides an overview

[3] The larger the training set, the more sparsely is the (negative) data distribution representated for a given anchor sample x_i.

over commonly used strategies and proposes a novel approach for learning such sampling strategies.

Proxy-based DML. Proxy-based DML approaches circumvent the sampling issue of ranking-based approaches by approximating the data distribution using one [161] or more [118, 183] learned representatives (proxies) for each class. In the former case, each representative basically acts as a estimated centroid of its class samples in Φ. Replacing the individually sampled, (true) negative and positive data points in standard ranking-based DML with the class representatives dramatically reduces the complexity of the learning problem. Consequently, computing the loss function over the full set of proxy negatives (i.e. class approximations) is now computationally feasible, which leads to more stable and faster training convergence [161].
The first proposed proxy-based DML model is the ProxyNCA loss [161] which is based on the Neighbourhood Component Analysis [77]. Denoting $\psi_{y_i} \in \mathbb{R}^D$ as the proxy representative for class $y_i \in \mathcal{Y}$, ProxyNCA is then formulated as

$$\mathcal{L}_{proxyNCA} = -\log\left(\frac{\exp(-d(\phi_i, \psi_{y_i}))}{\sum_{k=1, k\neq y_i}^{M} \exp(-d(\phi_i, \psi_k))}\right) \tag{2.12}$$

Moreover, comparing equation (2.12) with (1.2), we see that softmax-based classification objectives can also be interpreted as proxy-based approaches. In this case, the model parameters ζ_k in (1.2) represent the learned proxy representations for each class k with the inner product used as distance measure.

Classification-based DML. The interpretation of softmax weights $\zeta = (\zeta_1, \ldots, \zeta_K)$ (cf. Sec. 2.2.1) acting as a class representation (cf. (1.2)) is exploited by classification-based DML approaches by proposing dedicated adaptations. Making use of the equality $\zeta_k^\top \phi(x_i) = \|\zeta_k\|_2 \|\phi(x_i)\|_2 \cos\varphi_{i,k}$ with $\varphi_{i,k} (0 \leq \varphi_{i,k} \leq \pi)$ being the angle between $\phi(x_i)$ and ζ_k, we can reformulate the softmax-logits as

$$\begin{aligned} \sigma(\phi(x_i); \zeta) = &-\log\left[\frac{\exp(\zeta_{y_i}^\top \phi(x_i))}{\sum_{k=1}^{M} \exp(\zeta_k^\top \phi(x_i))}\right] \\ &-\log\left[\frac{\exp(\|\zeta_{y_i}\| \|\phi(x_i)\| \cos(\varphi_{i,y_i}))}{\sum_{k=1}^{M} \exp(\|\zeta_k\| \|\phi(x_i)\| \cos(\varphi_{i,k}))}\right]. \end{aligned} \tag{2.13}$$

By constraining $||\zeta_k||_2 = 1$ and the regularization of Φ to the hypersphere, $||\phi(x_i)||_2 = s$, we get

$$\sigma(\phi(x_i); \zeta) = -\log\left[\frac{\exp(s\cos(\varphi_{i,y_i}))}{\sum_{k=1}^{M} \exp(s\cos(\varphi_{i,k}))}\right]. \tag{2.14}$$

Optimizing (2.14) now only relies on the angle between samples x_i and the class proxies ζ_k, thus directly targeting the margin between them. Consequently, similar to ranking-

based objectives, we can now introduce a margin parameter β to explicitly enforce large decision boundaries between classes, i.e.

$$\sigma(\phi(x_i);\zeta,\beta) = -\log\left[\frac{\exp(s\cos(\beta+\varphi_{i,y_i}))}{\exp(s\cos(\beta+\varphi_{i,y_i}))+\sum_{k=1,k\neq y_i}^{M}\exp(s\cos(\varphi_{i,k}))}\right]. \quad (2.15)$$

Various DML formulations based on (2.15) emerged [51, 143, 144, 243] and have proven to be particularly beneficial when dealing with millions of classes such as in face verification problems. In these applications, each person constitutes an individual class where the corresponding images can be regarded as natural data augmentations. Hence, the class proxy-based learning problem is closely related to the explicit instance-based objectives resulting in fine-grained distance functions d_ϕ.

Advanced DML approaches. Additionally, more involved research extending the above objectives has been proposed as explicitly considered in our overview Fig. 2.1. Examples are the work of Sanakoyeu et al. [201] follow a divide-and-conquer strategy by partitioning and subsequently merging both the data and embeddings; BIER [175, 261] employs an ensemble of specialized learners to represent diverse data features and [156, 157, 194] extends this idea by combining DML with explicit diverse feature learning tasks (cf. chapter 3 and 4). In an orthogonal way, approaches like [141] and [277] generate artificial samples to effectively augment the training data, thus learning more complex ranking relations. The majority of these methods are essentially based on the objective functions introduced above and further hinge on the training parameters discussed in the following study, thus directly benefiting from our findings. We will discuss several of such advanced DML models in more detail and in relation to concepts introduced over the course of the following chapters.

2.2.2 Data sampling

Data sampling plays a crucial role in training deep networks in general [111, 117, 215], as the gradients during optimization can only be computed on small subsets of the training data. While the synergy between tuple mining strategies and ranking losses has been widely studied [71, 205, 256] and is subject of chapter5, this section analyzes the impact of mining informative mini-batches $\mathcal{B}$. This process is independent of the specific training objective and so far has been commonly neglected in DML research. Following we present batch mining strategies operating on both labels and the data itself: *label samplers*, which are sampling heuristics that follow selection rules based on label information only, and *embedded samplers*, which operate on data embeddings themselves to create batches $\mathcal{B}$ exhibiting diverse data statistics.

Label Samplers. To control the class distribution within $\mathcal{B}$ of size $b = |\mathcal{B}|$, we examine two different heuristics based on the number, n, of 'Samples Per Class' (SPC-n) heuristic: *SPC-n:* Given bach-size b, we randomly select b/n unique classes from which we select n samples randomly. In practice, typical values for n are 2, 4 or 8.

SPC-R: We randomly select $b-1$ samples from the dataset and choose the last sample to have the same label as one of the other $b-1$ samples to ensure that at least one triplet can be mined from $\mathcal{B}$. Thus, we effectively vary the number of unique classes within mini-batches.

Embedded Samplers. Increasing the batch-size b has proven to be beneficial for stabilizing optimization due to an effectively larger data diversity and richer training information [25, 158]. As the DML training is commonly performed on a single GPU (limited especially due to tuple mining process on the mini-batch), the batch-size b is bounded by memory. Nevertheless, in order to 'virtually' maximize the data diversity, we distill the information content of a large set of samples $\mathcal{B}^*, b^* = |\mathcal{B}^*| > b$ into a mini-batch B by matching the statistics of $\mathcal{B}$ and $\mathcal{B}^*$ under the embedding ϕ. To avoid computational overhead, we sample $\mathcal{B}^*$ from a continuously updated memory bank $\mathcal{M}$ of embedded training samples. Similar to [159], $\mathcal{M}$ is generated by iteratively updating its elements based on the steady stream of training batches $\mathcal{B}$. Using $\mathcal{M}$, we mine mini-batches by first randomly sampling $\mathcal{B}^*$ from $\mathcal{M}$ with $b^* = 1024$ and subsequently find a mini-batch $\mathcal{B}$ to match its data statistics by using one of the following criteria:
Greedy Coreset Distillation (GC): Greedy Coreset [1] finds a batch $\mathcal{B}$ by iteratively adding samples $x^* \in \mathcal{B}^*$ which maximize the distance from the samples that have already been selected $x \in \mathcal{B}$, thereby maximizing the covered space within Φ by solving

$$\min_{\mathcal{B}:|\mathcal{B}|=b} \max_{x*\in\mathcal{B}^*} \min_{x\in\mathcal{B}} d_\phi(x, x^*) \; . \tag{2.16}$$

Matching of distance distributions (DDM): DDM aims to preserve the distance distribution of $\mathcal{B}^*$. We randomly select m candidate mini-batches and choose the batch $\mathcal{B}$ with smallest Wasserstein distance between normalized distance histograms of $\mathcal{B}$ and $\mathcal{B}^*$ [198].
FRD-Score Matching (FRD): Similar to the recent GAN evaluation setting, we compute the frechet distance [96]) between $\mathcal{B}$ and $\mathcal{B}^*$ to measure the similarity between their distributions using

$$FRD(\mathcal{B}, \mathcal{B}^*) \triangleq \|\mu_\mathcal{B} - \mu_{\mathcal{B}^*}\|_2^2 + \text{Tr}(\Sigma_\mathcal{B} + \Sigma_{\mathcal{B}^*} - 2(\Sigma_\mathcal{B}\Sigma_{\mathcal{B}^*})^{1/2}) \; , \tag{2.17}$$

with $\mu_\bullet, \Sigma_\bullet$ being the mean and covariance of the embedded set of samples. Like in DDM, we select the closest batch $\mathcal{B}$ to $\mathcal{B}^*$ among m randomly sampled candidates.

2.2.3 General model training parameters and architecture

Next to the objectives and data sampling process, successful learning hinges on a reasonable choice of the training environment. While there is a multitude of hyperparameters to be set, we identify several factors which both influence performance and recently exhibit an divergence in proposed approaches.

Architectures. Predominantly three basis network architectures are used in recent DML literature: GoogLeNet [223] (GN, typically with embedding dimensionality 512),

Network	GN	IBN	R50
CUB200, R@1	45.41	48.78	43.77
CARS196, R@1	35.31	43.36	36.39
SOP, R@1	44.28	49.05	48.65

Table 2.1: *Recall performance of commonly used network architectures after ImageNet pretraining.* Final linear layer is randomly initialized and normalized to the unit-hypersphere.

Inception-BN [103] (IBN, 512) and ResNet50 [91] (R50, 128) (with optionally frozen Batch-Normalization layers for improved convergence and stability across varying batch sizes[4], see e.g. [28, 194]). Due to the varying number of parameters and configuration of layers, each architecture exhibits a different starting point for learning, based on its initialization by ImageNet pretraining [50]. Table 2.1 compares their initial DML performance measured in Recall@1 (R@1). The reference to differences in architecture is one of the main arguments used by individual works not to compare themselves to competing approaches. Disconcertingly, even when reporting additional results using adjusted networks is feasible, typically only the performance using a single architecture are reported. Consequently, a fair comparison between approaches is heavily impaired.

Weight Decay. Training deep networks commonly involves a regularization of the parameters to be optimized. Similar to (2.4), $r(\theta)$ in problems like (2.8) is typically implemented as a L2-regularization $||\theta||_2$, which is also referred to as weight decay [132]. In similarity learning, particularly on small datasets, its careful adjustment is crucial to maximize generalization performance. Nevertheless, many works do not report this.

Embedding dimensionality. Choosing a dimensionality D of the embedding ϕ influences the embedding structure and consequently generalization performance. While each architecture typically uses an individual, standardized dimensionality D in DML, recent works differ without reporting proper baselines using an adjusted dimensionality. Again, comparison to existing works and the assessment of the actual contribution is impaired.

Data Preprocessing. Preprocessing training images significantly influences both the learned features and model regularization and is widely adopted when training deep networks. Nevertheless, many works fail to report the applied augmentation protocols. Thus, the value of their proposed approaches is difficult to assess, as potential performance gains are difficult to link to the approaches contributions with sufficient certainty. Moreover, comparisons of results is also suffering from the trend of operating on increasing training and test image sizes, thus increasing the level of details to be captured by the DML model.

[4]Note that Batch-Normalization is still performed, but no parameters are learned.

Batchsize. Deep networks are typically trained using stochastic gradient decent, i.e. gradient updates to the network parameters are computed on small mini-batches $\mathcal{B}$ of the training data only. Consequently, larger batch sizes result in better approximations of the training data distribution, stabilizing optimization and often result in better local minima. Especially large datasets involving many different training classes greatly benefit from increased batch sizes. However, it is commonly not taken into account as an influential factor of variation and thus is not considered for establishing fair evaluation protocols for DML.

2.3 Analyzing DML training strategies

In this section we now first evaluate the previously discussed design choices and hyperparameters. Subsequently, we conduct a large comparison between the most common DML baselines. All our experiments are based on a fair evaluation protocol for Deep Metric Learning which we fix throughout this study.

2.3.1 Datasets

Our experiments are conducted on the following standard benchmark datasets:
CUB200-2011: Contains 11,788 images in 200 classes of birds. Train/Test sets are made up of the first/last 100 classes (5,864/5,924 images respectively) [239]. Samples are distributed evenly across classes.
CARS196: Has 16,185 images/196 car classes with even sample distribution. Train/Test sets use the first/last 98 classes (8054/8131 images) [130].
Stanford Online Products (SOP): Contains 120,053 product images divided into 22,634 classes. Train/Test sets are provided, contain 11,318 classes/59,551 images in the Train and 11,316 classes/60,502 images in the Test set [171]. In SOP, unlike the other benchmarks, most classes have few instances, leading to a significantly different data distribution compared to CUB200-2011 and CARS196.

2.3.2 Experimental protocol

Our training protocol follows parts of [256], which utilize a ResNet50 architecture with frozen Batch-Normalization layers and embedding dimensionality 128 to be comparable with already proposed results with this architecture. While both GoogLeNet [223] and Inception-BN [103] are also often employed in the DML literature, we choose ResNet50 due to its success in recent state-of-the-art approaches [194, 201]. In line with standard practices we randomly resize and crop images to 224×224 for training and center crop to the same size for evaluation. During training, random horizontal flipping ($p = 0.5$) is used. Optimization is performed using Adam [122] with learning rate fixed to 10^{-5} and *no* learning rate scheduling for unbiased comparison. Weight decay is set to a constant value of $4 \cdot 10^{-4}$, as motivated in section 2.3.3. We implemented all models in PyTorch [180], and experiments are performed on individual Nvidia Titan X, V100 and T4 GPUs with memory usage limited to 12GB. Each training is run over 150 epochs

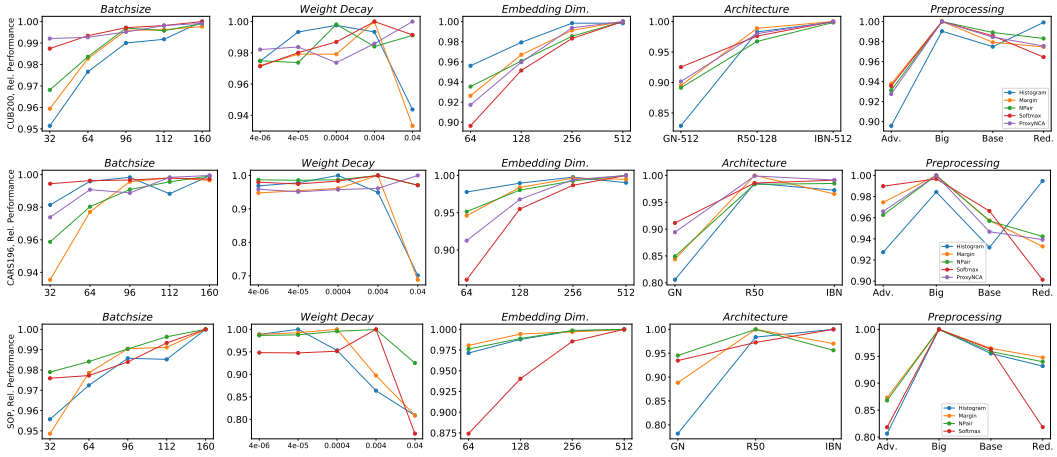
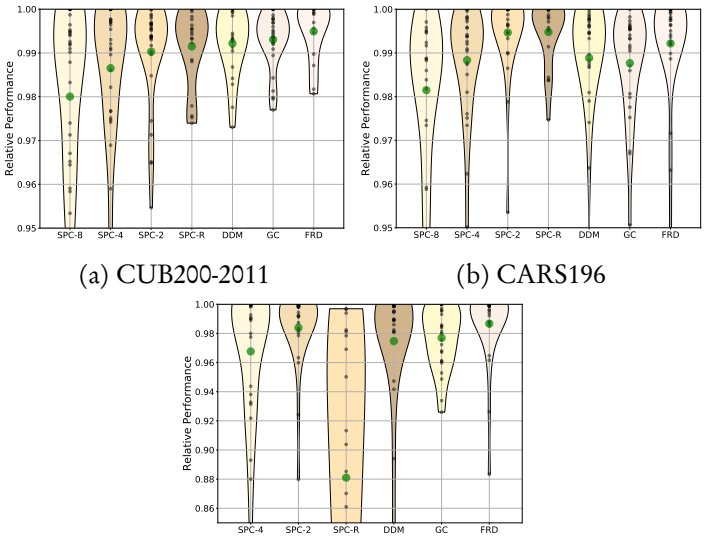


Figure 2.2: *Evaluation of DML pipeline parameters and architectures* on all benchmark datasets and their influence on relative improvement across different training criteria.

for CUB200-2011/CARS196 and 100 epochs for Stanford Online Products, if not stated otherwise. For batch sampling we utilize the the SPC-2 strategy, as motivated in section 2.3.4. Finally, each result is averaged over multiple seeds to avoid seed-based performance fluctuations. For our study, we examine the following evaluation metrics: Recall@1 and Recall@2 [107], Normalized Mutual Information (NMI) [151], F1 score [217] and (class-averaged) mean average precision measured on recall (mAP).

2.3.3 Studying DML parameters and architectures

Now we study the influence of parameters & architectures discussed in Sec. 2.2.3 using five different objectives. For each experiment, all metrics noted at the end of Sec. 2.3.2 are measured. For each loss, every metric is normalized by the maximum across the evaluated value range. This enables an comparable summary of performance across all metrics, where differences correspond to relative improvement. Fig. 2.2 analyzes each factor by evaluating a range of potential setups with the other parameters fixed to values from Sec. 2.3.2: Increasing the batchsize generally improves results with gains varying among criteria, with particularly high relevance on the SOP dataset. For weight decay, we observe loss and dataset dependent behavior up to a relative performance change of 5%. Varying the data preprocessing protocol, e.g. augmentations and input image size, leads to large performance differences as well. *Base* follows our protocol described in Sec. 2.3.2. *Red.* refers to resizing of the smallest image side to 256 and cropping to 224x224 with horizontal flipping. *Big* uses *Base* but crops images to 256x256. Finally, we extend *Base* to *Adv.* with color jittering, changes in brightness and hue. We find that larger images provide better performance regardless of the chosen objective or dataset. Using the *Adv.* processing on the other hand is dependent on the dataset. Moreover, we see that random resized cropping is a generally stronger operation than basic resizing and cropping.
All these factors underline the importance of a complete declaration of the training pro-



(a) CUB200-2011 (b) CARS196

(c) SOP

Figure 2.3: *Comparison of mini-batch mining strategies* on three different datasets. Performance measures Recall@1 and 2, NMI, mAP and F1 are normalized across metrics and loss function. We plot the distributions of relative performances for each strategy.

tocol to facilitate reproducibility and comparability. Similar results are observed for the choice of architecture and embedding dimensionality D. At the example of R50, our analysis shows that training objectives perform differently for a given D but seem to converge at $D = 512$. However, for R50 D is typically fixed to 128, thus disadvantaging some training objectives over others. Finally, comparing common DML architectures reveals their strong impact on performance with varying variance between loss functions. Highest consistencies seem to be achievable with R50 and IBN-based setups.

2.3.4 Batch sampling impacts DML training

We now analyze how the data sampling process for mini-batches impacts the performance of DML models using the sampling strategies presented in Sec. 2.2.2. To conduct an unbiased study, we experiment with six conceptually different objective functions: Margin-loss with Distance-Weighted Sampling, Triplet Loss with Random Sampling, ProxyNCA, Multi-Similarity Loss, Histogram loss and Normalized Softmax loss. To compare our evaluation metrics (cf. 2.3.2), we utilize the same normalization procedure discussed in Sec. 2.3.3. Fig. 2.3 summarizes the results for each sampling strategy by reporting the distributions of normalized scores of all pairwise combinations of training loss and evaluation metrics. Our analysis reveals that the batch sampling process indeed effects DML training with a difference in *mean* performance up to 1.5%. While there is no clear winner across all datasets, we observe that the SPC-2 and FRD samplers perform very well and, in particular, consistently outperform the SPC-4 strategy which is commonly reported to be used in the literature [205, 256].

Benchmarks→	CUB200-2011		CARS196		SOP	
Approaches ↓	R@1	NMI	R@1	NMI	R@1	NMI
Imagenet [50]	43.77	57.56	36.39	37.96	48.65	58.64
Angular [245]	62.10 ± 0.27	67.59 ± 0.26	78.00 ± 0.32	66.48 ± 0.44	73.22 ± 0.07	89.53 ± 0.01
ArcFace [51]	62.67 ± 0.67	67.66 ± 0.38	79.16 ± 0.97	66.99 ± 1.08	77.71 ± 0.15	90.09 ± 0.03
Contrast. [86] (Dist.)	61.50 ± 0.17	66.45 ± 0.27	75.78 ± 0.39	64.04 ± 0.13	73.21 ± 0.04	89.78 ± 0.02
GenLifted [94]	59.59 ± 0.60	65.63 ± 0.14	72.17 ± 0.38	63.75 ± 0.35	75.21 ± 0.12	89.84 ± 0.01
Hist. [232]	60.55 ± 0.26	65.26 ± 0.23	76.47 ± 0.38	64.15 ± 0.36	71.30 ± 0.10	88.93 ± 0.02
Margin ($\beta = 0.6$) [256]	62.50 ± 0.24	67.02 ± 0.37	77.70 ± 0.32	65.29 ± 0.32	77.38 ± 0.11	$\mathbf{90.45 \pm 0.03}$
Margin ($\beta = 1.2$) [256]	$\mathbf{63.09 \pm 0.46}$	68.21 ± 0.33	79.86 ± 0.33	67.36 ± 0.34	$\mathbf{78.43 \pm 0.07}$	90.40 ± 0.03
Multisimilarity [251]	62.80 ± 0.70	$\mathbf{68.55 \pm 0.38}$	$\mathbf{81.68 \pm 0.19}$	$\mathbf{69.43 \pm 0.38}$	77.99 ± 0.09	90.00 ± 0.02
Npair [217]	61.63 ± 0.58	67.64 ± 0.37	77.48 ± 0.28	66.55 ± 0.19	75.86 ± 0.08	89.79 ± 0.03
Pnca [161]	62.80 ± 0.48	66.93 ± 0.38	78.48 ± 0.58	65.76 ± 0.22	–	–
Quadruplet (Dist.) [37]	61.71 ± 0.63	66.60 ± 0.41	76.34 ± 0.27	64.79 ± 0.50	76.95 ± 0.10	90.14 ± 0.02
SNR (Dist.) [266]	62.88 ± 0.18	67.16 ± 0.25	78.69 ± 0.19	65.84 ± 0.52	77.61 ± 0.34	90.10 ± 0.08
SoftTriple [183]	60.83 ± 0.47	64.27 ± 0.36	75.66 ± 0.46	62.66 ± 0.16	–	–
Softmax [268]	61.66 ± 0.33	66.77 ± 0.36	78.91 ± 0.27	66.35 ± 0.30	76.92 ± 0.64	89.82 ± 0.15
Triplet (Dist.) [256]	62.87 ± 0.35	67.53 ± 0.14	79.13 ± 0.27	65.90 ± 0.18	77.39 ± 0.15	90.06 ± 0.02
Triplet (Hard) [205]	61.61 ± 0.21	65.98 ± 0.41	77.60 ± 0.33	65.37 ± 0.26	73.50 ± 0.09	89.25 ± 0.03
Triplet (Rand.) [205]	58.48 ± 0.31	63.84 ± 0.30	70.63 ± 0.43	61.09 ± 0.27	67.86 ± 0.14	88.35 ± 0.04
Triplet (Semi) [205]	60.09 ± 0.49	65.59 ± 0.29	72.51 ± 0.47	62.84 ± 0.41	73.61 ± 0.14	89.35 ± 0.02
R-Contrast. (Dist.)	63.57 ± 0.66	67.63 ± 0.31	81.06 ± 0.41	67.27 ± 0.46	74.36 ± 0.11	89.94 ± 0.02
R-Margin ($\beta = 0.6$)	$\mathbf{64.93 \pm 0.42}$	68.36 ± 0.32	$\mathbf{82.37 \pm 0.13}$	68.66 ± 0.47	77.58 ± 0.11	90.42 ± 0.03
R-Margin ($\beta = 1.2$)	63.32 ± 0.33	67.91 ± 0.66	81.11 ± 0.49	67.72 ± 0.79	$\mathbf{78.52 \pm 0.10}$	90.33 ± 0.02
R-SNR (Dist.)	62.97 ± 0.32	68.04 ± 0.34	80.38 ± 0.35	67.60 ± 0.20	77.69 ± 0.25	90.02 ± 0.06
R-Triplet (Dist.)	63.28 ± 0.18	67.86 ± 0.51	81.17 ± 0.11	67.79 ± 0.23	77.33 ± 0.14	89.98 ± 0.04

Table 2.2: *Comparison of Recall@1 and NMI performances for all objectives* averaged over 5 runs. Each model is trained using the same training setting over 150 epochs for CUB/CARS and 100 epochs for SOP. 'R-' denotes model trained with ρ-regularization. (Dist.) denotes distance-weighted negative sampling [256], (Rand.) denotes random negative sampling, (Hard) denotes hard-negative sampling [205] and (Semi) denotes semi-hard negative sampling [205] being used for constructing triplets. **Bold** denotes best results excluding regularization. **Boldblue** marks overall best results.

In summary, our study indicates that DML benefits from data diversity in mini-batches, independent of the chosen training objective. This coincides with the general benefit of larger batchsizes as noted in section 2.3.3. While complex mining strategies may perform better, simple heuristics like SPC-2 are sufficient.

2.3.5 Comparing DML models

Based on our training parameter and batch-sampling evaluations we compare a large selection of 14 different DML objectives and 4 mining methods under fixed training conditions (see 2.3.2 & 2.3.3), most of which claim state-of-the-art by a notable margin. For ranking-based models, we employ distance-based tuple mining (D) [256] which proved most effective. We also include random, semihard sampling [205] for our tuple mining study using the classic triplet loss. Loss-specific hyperparameters are determined via small cross-validation gridsearches around originally proposed values to adjust for our training setup. Table 2.2 summarizes our evaluation results on all benchmarks. We observe particularly on CUB200-2011 and CARS196 a higher performance saturation between methods

Approach	Architecture	Dim	R@1	R@10	R@100	NMI
DVML[141]	GoogLeNet	512	70.2	85.2	93.8	90.8
HTL[71]	Inception-BN	512	74.8	88.3	94.8	-
MIC[194]	ResNet50	128	77.2	89.4	95.6	90.0
D&C[201]	ResNet50	128	75.9	88.4	94.9	90.2
Rank[250]	Inception-BN	1536	**79.8**	**91.3**	**96.3**	90.4
ABE[120]	GoogLeNet	512	76.3	88.4	94.8	-
Margin (ours)[256]	ResNet50	128	78.4	-	-	90.4

Table 2.3: Comparison to the state-of-the-art DML methods on SOP[171]. *Dim* denotes the dimensionality of ϕ_θ.

as compared to SOP due to the strong difference in data distribution. Generally, performance between criteria is much more similar than indicated by the literature, as also has been noted in concurrent work by [162]. We find that representatives of ranking based objectives in general slightly outperform their classification/NCE-based counterparts. On average, margin loss [256] and multisimilarity loss [251] offer the best performance across datasets. Remarkably, under our carefully chosen training setting, a multitude of losses compete or even *outperform* more involved state-of-the-art DML approaches on the SOP dataset. To this end, Tab. 2.3 provides a detailed comparison between current state-of-the-art DML approaches and our strongest baseline model, margin loss (D, $\beta = 1.2$) [256], on the SOP dataset. The results for these approaches are taken from their publicly available manuscripts. We observe that our baseline model outperforms each of the models using varying architectures, but especially other ResNet50-based implementations. While R50 proves to be a stronger base network than GoogLeNet based model, improvements over MIC and D&C using the same backbone by at least 0.9% and methods based on the similarly strong Inception-BN showcase the relevance of a well-defined baseline. Additionally, even though Rank [250] and ABE [120] employ considerable more powerful network ensembles, our carefully motivated baseline exhibits competitive performance.
In summary, we demonstrated that, under equal training conditions, performance saturates across different methods, contrasting results reported in the recent literature, as carefully trained baseline models even outperform state-of-the-art approaches which use considerable stronger architectures. Thus, to evaluate the true benefit of proposed contributions, baseline models need to be competitive and implemented under comparable settings.

2.4 Generalization in Deep Metric Learning

The previous section showed how different model and training parameter choices result in vastly different performances. However, how such differences can be best be explained on basis of the learned embedding is an open question and, for instance, studied under the concept of *compression* [228]: the information theoretic tradeoff between describing the input data with as little information as possible, while only retaining sufficient infor-

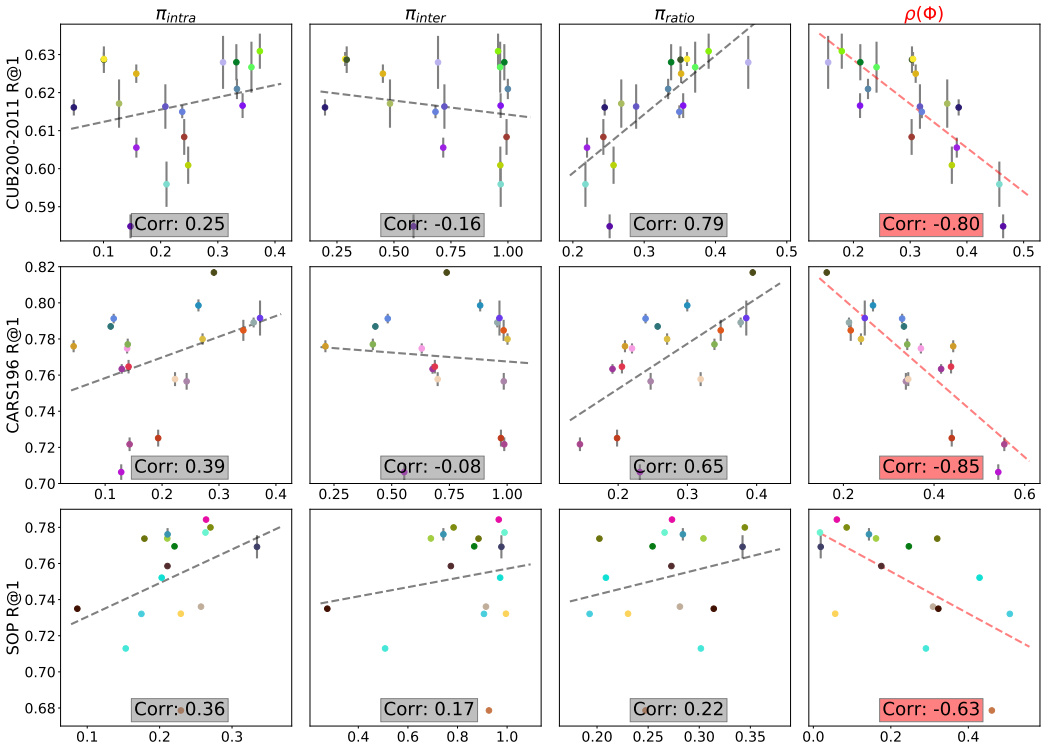


Figure 2.4: *Correlation between generalization and structural properties* derived from $\Phi_{\mathcal{X}}$ using different DML objectives on each dataset. *Left-to-Right*: Mean intra-class distances π_{intra} & inter-class distances π_{inter}, the ratio $\pi_{\text{intra}}/\pi_{\text{inter}}$ and spectral decay ρ.

mation to infer an output variable corresponding to a given task, thus enabling efficient supervised learning. Recent work [233] links compression to class-conditioned flattening of a representation, indicated by an increased decay of singular values obtained by Singular Value Decomposition (SVD) on the data representations. Thus, class representations occupy a more compact volume, thereby reducing the number of directions with significant variance. The subsequent strong focus on the most discriminative directions is shown to be beneficial for classic classification scenarios with i.i.d. train and test distributions. However, this overly discards features which could capture data characteristics outside the training distribution. Hence, generalization in transfer problems like DML is hindered due to the shift in training and testing distribution [14]. We thus hypothesize that actually retaining a considerable amount of directions of significant variance (DoV) is crucial to learn a well generalizing embedding function ϕ.

To verify this assumption, we analyze the spectral decay of the embedded training data $\Phi_{\mathcal{X}} \triangleq \{\phi(x)|x \in \mathcal{X}\}$ via SVD. We then normalize the sorted spectrum of singular values (SV) $\mathcal{S}_{\Phi_{\mathcal{X}}}$ and compute the KL-divergence to a D-dim. discrete uniform distribution $\mathcal{U}_D$, i.e.

$$\rho(\Phi) = \text{KL}(\mathcal{U}_D \,||\, \mathcal{S}_{\Phi_{\mathcal{X}}}) \,, \tag{2.18}$$

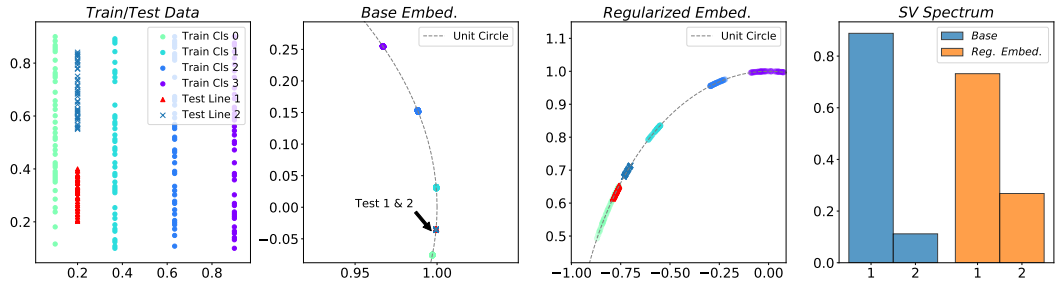


Figure 2.5: Toy example based on horizontally discriminative training data, where to goal is to generalize to vertically discriminative test data. (Leftmost) training and test data. (Mid-left) A small, normalized two-layer fully-connected network trained with standard contrastive loss fails to separate both test classes as it never has to utilize vertical discrimination. (Mid-right) The regularized embedding successfully separates the test classes by introducing additional features and decreasing the spectral decay. (Rightmost) Singular value spectra of training embeddings learned with and without regularization.

where the divergence between distributions p and q is given by $\mathrm{KL}(p,q) = \sum_i p_i log(p_i/q_i)$. For simplicity we use the notation $\rho(\Phi)$ instead of $\rho(\Phi_{\mathcal{X}})$. We do not consider individual training class representations, as testing and training distribution are shifted. Lower values of $\rho(\Phi)$ indicate more directions of significant variance. Using this measure, we analyze a large selection of DML objectives in Fig. 2.4 (rightmost) on CUB200-2011, CARS196 and SOP. Comparing R@1 and $\rho(\Phi)$ reveals significant inverse correlation (≤ -0.63) between generalization and the spectral decay of embedding spaces Φ, which highlights the benefit of more directions of variance in the presence of train-test distribution shifts.
We now compare our finding to commonly exploited concepts for training. For this, let $\Phi_{y_l} = \{\phi_i \triangleq \phi(x_i) | x_i \in \mathcal{X}, y_i = y_l\}$ denote the set of embedded samples of a class y_l, $\mu(\Phi_{y_l})$ their mean embedding and Z_{inter}, Z_{intra} normalization constants. Our comparison includes concepts such as *(i)* Larger margins between class samples in Φ [51, 143], i.e. an increase in average inter-class distances

$$\pi_{\text{inter}}(\Phi) = \frac{1}{Z_{\text{inter}}} \sum_{y_l, y_k, l \neq k} d(\mu(\Phi_{y_l}), \mu(\Phi_{y_k})) \; ; \tag{2.19}$$

(ii) explicitly introducing intra-class variance [141], which is indicated by an increase in average intra-class distance

$$\pi_{\text{intra}}(\Phi) = \frac{1}{Z_{\text{intra}}} \sum_{y_l \in \mathcal{Y}} \sum_{\phi_i, \phi_j \in \Phi_{y_l}, i \neq j} d(\phi_i, \phi_j) \; . \tag{2.20}$$

We also investigate *(iii)* their relation by using the ratio

$$\pi_{\text{ratio}}(\Phi) = \pi_{\text{intra}}(\Phi) / \pi_{\text{inter}}(\Phi) \; , \tag{2.21}$$

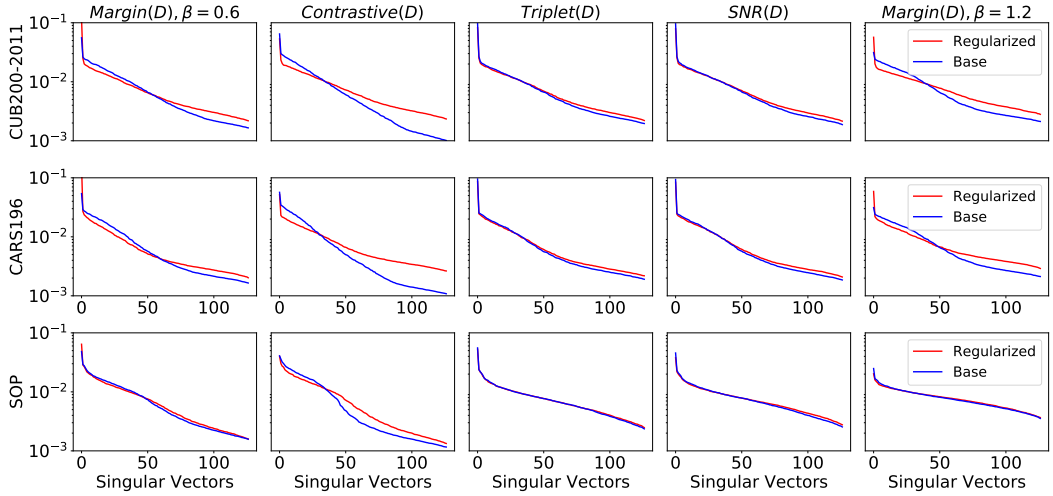


Figure 2.6: *Singular Value Spectrum* for models trained with (red) and without (blue) ρ-regularization for various ranking-based criteria.

which can be regarded as an embedding space density. Fig. 2.4 compares these measures with $\rho(\Phi)$. It is evident that neither of the distance related measures $\pi_\bullet(\Phi)$ consistently exhibits significant correlation with generalization performance similar to $\rho(\Phi)$, especially when taking all three datasets into account. For CUB200-2011 and CARS196, we however we also find that an increased embedding density (π_{ratio}) can be linked to stronger generalization. For large, diverse datasets like SOP, the distance-related measures do not provide sufficient explanation.

2.4.1 ρ-REGULARIZATION FOR IMPROVED GENERALIZATION

We now exploit our findings to propose a simple ρ-regularization for ranking-based approaches by counteracting the compression of representations. We randomly alter tuples by switching negative samples x_n with the positive x_p in a given ranking-loss formulation (cf. Sec. 2.2.1) with probability p_{switch}. This pushes samples of the same class apart, enabling a DML model to capture extra non-label-discriminative features while dampening the compression induced by strong discriminative training signals.
Fig. 2.5 depicts a 2D toy example which illustrates the effect of our proposed regularization and further highlights the issue of overly compressed data representations. Even though the training distribution exhibits features needed to separate all test classes, these features are disregarded by the strong discriminative training signal. Regularizing the compression by attenuating the spectral decay $\rho(\Phi)$ enables the model to capture more information and as a result exhibits stronger generalization to the unseen test classes. In addition, Fig. 2.6 verifies that the ρ-regularization also leads to a decreased spectral decay on DML benchmark datasets, resulting in improved recall performance (cf. Tab. 2.2 (bottom)), i.e. R-constrastive, R-margin, R-SNR and R-Triplet loss.

Fig. 2.7 provides a more detailed illustration comparing the performance of DML training objectives and their corresponding spectral decay $\rho(\Phi)$. For ranking losses, we further include the results using ρ-regularization while training, which further shows that in each case a gain in performance is related to a decrease of $\rho(\Phi)$. Especially the contrastive loss [86] greatly profits from our proposed regularization, as also indicated by the analysis of the singular value spectra in Fig. 2.6. Its large gains, more then 5% on the CARS196 dataset, is well explained by comparison of its training objective with those of triplet-based formulations. The latter optimizes over relative positive ($d_\phi(x_i, x_j)$)) and negative distances ($d_\phi(x_i, x_k)$) up to a fixed margin β, which counteracts a compression of the embedding to a certain extend. On the other hand, the constrastive loss, while controlling only the negative distances by β, is able to perform an unconstrained contraction of entire classes, which facilitates overly compressed embeddings ϕ. Finally, Fig. 2.8 studies the influence of p_{switch} on generalization performance at the example of Margin loss [256].

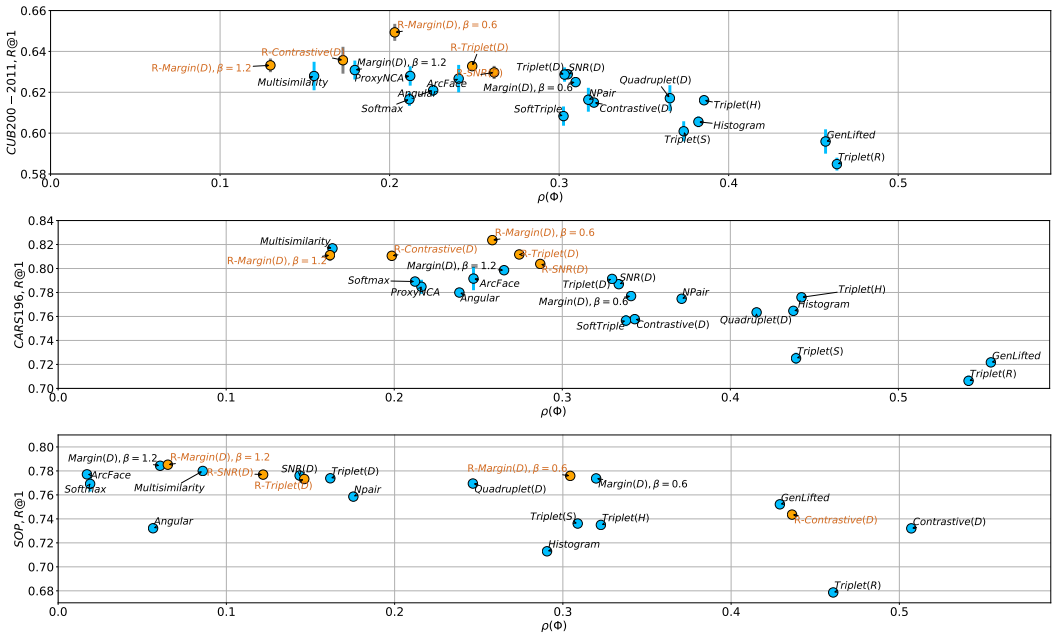


Figure 2.7: Relation between $\rho(\Phi)$ and generalization performance on Recall@1 for models trained with (orange) and without (blue) ρ-regularization. We report mean results and errorbars (gray). When error is small, bars are covered.

2.5 Discussion

In this chapter, we conduct a large, comprehensive study of the most influential components for training a DML model which, further, contribute severely to the impaired comparability of recent approaches. On this basis, we analyze the correlation between DML generalization and the compression of the learned data representation. Our findings reveal that highly compressed representations disregard helpful features for capturing

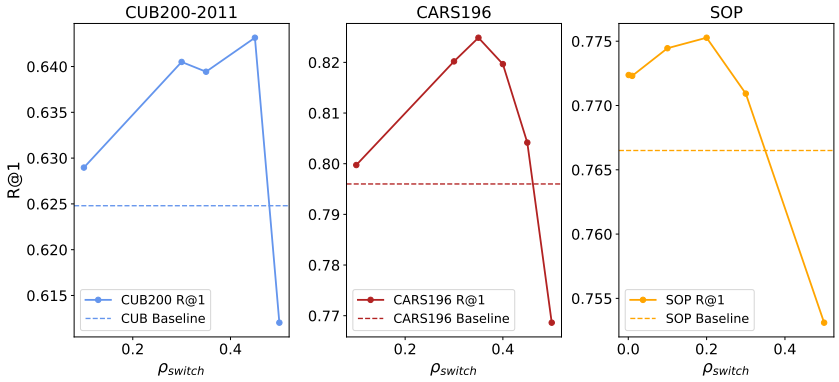


Figure 2.8: *Analysis of the influence of* p_{switch} on Recall@1 using margin loss with $\beta = 0.6$. Dashed lines denote performance without ρ-regularization.

data characteristics that transfer to unknown test distributions. To this end, we propose a simple technique for ranking-based methods to regularize the compression of the learned embedding, which results in boosted performance across all benchmark datasets.

3 Shared Features for Improved Generalization

Deep metric learning (DML)[205, 217, 256] is currently the main paradigm for learning similarities between images. The main challenge is then not just maximizing generalization performance from a training to an independent and identically distributed test set. Rather, DML typically aims at transfer learning, i.e., discovering an embedding that is also applicable for differently distributed test data. A typical example is training and test data that exhibits entirely different classes. This degree of generalization is significantly more demanding. It requires to learn visual characteristics that generalize well and likely transfer to unknown object classes.
Current DML approaches are mostly optimized using purely class-discriminative training objectives [205, 217, 256]. Hence, the task is typically framed as learning only the characteristics which *separate* the classes while being invariant to all those *shared across* classes. The underlying assumption is that features that discriminate between training classes will also help to separate between arbitrary other test classes. As discussed in the previous chapter, such learning results in overly compressed image representations. Intuitively, while the underlying learned features accurately circumscribe each training class, it is unlikely that they will generalize to samples and classes outside the training distribution, cf. Fig. 3.1.
Taking up the idea of ρ-regularization introduced in Sec. 2.4.1, subsequently we analyze how to incorporate a complementary source of features learning more profoundly into DML models. Additionally to the class-specific discriminative features, we now want to explicitly learn those so far neglected characteristics that are shared across samples of various training classes. Since such features are of more general nature, they are more likely to generalize to unseen test classes. Therefore, we develop a strategy to learn such shared features without the need for extra supervision or additional training data. Our approach is formulated as a novel triplet sampling strategy which explicitly leverages triplets connecting images from mutually different classes. Consequently, it can be easily employed by any ranking loss framework. In summary, this chapter

- introduces the concept of shared characteristics into DML on a more general level and analyze its importance for successful DML generalization.

- examines how standard discriminative approaches suffer from impaired generalization capabilities and shows how shared features help to alleviate these issues while providing a complementary training signal.

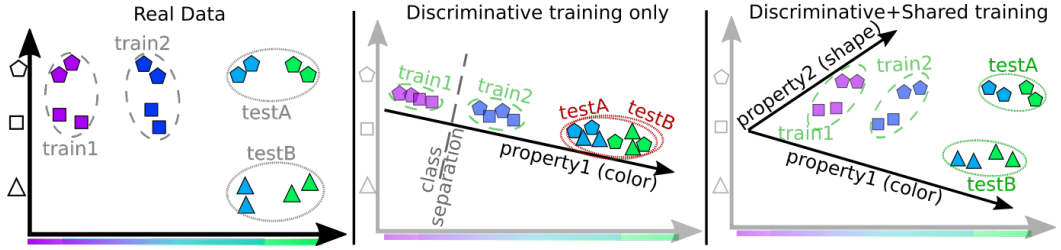


Figure 3.1: *Motivation*. (left) Real data is described by several latent characteristics (color and shape). (center) Only the discriminative characteristics (color) which separate classes are learned, while the others are ignored, thus failing to generalize to unseen classes. (right) Including characteristics shared across classes (shape) leads to a better representation of the data. Also see Sec. 2.4.1 for a toy example illustrating this idea.

- proposes a novel and simple method to effectively learn shared characteristics without the need for extra data or annotations and, further, overcome the shortcomings of our previous heuristic-based approach.
- presents an effective strategy for incorporating the learning of shared characteristics into classic discriminative ranking loss frameworks, thereby strongly boosting generalization performance.

Experiments using different ranking losses and architectures on standard DML benchmarks show consistent improvements over previous DML approaches. We further investigate our model and its generalization ability by conducting thorough ablation studies. This chapter is based on our publication *'Sharing Matters for Generalization in Deep Metric Learning'* [157].

3.1 Efficient Learning of Shared Features in DML

In this section, we propose a method to improve the generalization ability of existing metric learning frameworks by incorporating features that are shared across classes. After defining classical discriminative metric learning, we discuss the rationale behind shared features and how to learn them. Finally, we present how to best incorporate discriminative and shared features in one model.

3.1.1 Recall: Discriminative metric learning

Let us briefly adjust the general formulation of discriminative deep similarity learning models from the previous section to explicitly consider an intermediate feature representation f. Assume $f_i \triangleq f(x_i;\omega)$ to be a D_f-dimensional feature representation $f : \mathbb{R}^{D'} \rightarrow \mathbb{R}^{D_f}$ of a datapoint $x_i \in \mathbb{R}^{D'}$ parametrized by ω. Then, we seek to learn the consecutive mapping $\phi : \mathbb{R}^{D_f} \rightarrow \Phi \subset \mathbb{R}^{D}$ with $\phi_i \triangleq \phi(f_i;\theta) = \phi(f(x_i;\omega);\theta)$, such that similar datapoints x_i, x_j are close in the embedding space Φ under a predefined distance function $d = d(\phi_i, \phi_j) = d_\phi(x_i, x_j)$ and far from each other if they are dissimilar.

Figure 3.2: *tSNE[150] projections of encoding spaces.* (left) Discriminative training of the embedding on $\mathcal{T}_{\mathcal{X}}$ only, (right) Shared training of the embedding on $\mathcal{T}^*_{\mathcal{X}}$ only. The same random image subset from the CARS196[130] training set is visualized. Image contour color indicates ground-truth class. (left) the embedding groups images into compact class-based clusters, (right) the embedding aligns images based on characteristics shared across classes, e.g. view point (green) and color (orange), which are likely to generalize. See supplementary for larger version.

Here, θ parametrizes the mapping ϕ and d is chosen to be the Euclidean distance with $d_\phi(x_i, x_j) = \|\phi_i - \phi_j\|_2$. Typically, $f(x_i, \omega)$ is represented as the output of a feature extractor network and $\phi(f_i; \theta)$ is realized as an encoder network (typically a subsequent fully connected layer) with its output normalized to a unit hypersphere $\mathbb{S}^D$ for training stability [205].
Let us consider the widely used family of ranking-based DML training objectives (cf. 2.2.1) for learning ϕ at the example of the prominent triplet loss [205]. Operating on triplets of datapoints $t = (x_i, x_j, x_k)$ it is formulated as

$$\mathcal{L}(t; \phi) = \mathcal{L}(x_i, x_j, x_k; \phi) = \max(0, d_\phi(x_i, x_j) - d_\phi(x_i, x_k) + \beta) \,, \tag{3.1}$$

where x_i is called the anchor, x_j the positive and x_k the negative sample. Following the paradigm of supervised class-discriminative metric learning with given class labels $y_i \in \mathcal{Y}$, x_i and x_j are sampled from the same class (i.e. $y_i = y_j$) and x_k from another class (i.e. $y_i \neq y_k$). Thus, by optimization on the set of triplets

$$\mathcal{T}_{\mathcal{X}} \triangleq \left\{(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : \; y_i = y_j \wedge y_i \neq y_k\right\} \,, \tag{3.2}$$

metric learning is framed as a discriminative task. Intuitively $\mathcal{L}(t, \phi)$ imposes a relative ordering on the distances within t, pushing x_i closer to x_j than x_k by at least a fixed margin β.

3.1.2 Designing a learning task for shared features

In DML, ranking losses, such as Eq. 3.1, enforce mutual similarity of samples from the same training class and dissimilarity to others. Provided a large number of training classes, each class will be accurately separated from all others. This contracts training categories in the embedding space and separates them from one another, as can be seen in Fig. 3.1. While such accurate models are beneficial for generalizing from training to i.i.d. test data,

Figure 3.3: *Nearest neighbour retrieval using ϕ and ϕ^*.* Based on the class- (ϕ) and shared (ϕ^*) embedding, we show nearest neighbor retrievals limited to images with different class label than the query image. The neighbors obtained based on the embedding ϕ^* trained for shared characteristics exhibit common visual properties. Most prominent: *(a)* red color, *(b)* and *(c)* pose and car type, *(d)* roundish shape, *(e)* back view and color. The embedding ϕ trained for class discrimination fails to consistently retrieve meaningful neighbours outside of the query's class.

transfer to entirely novel classes is significantly more challenging and asks for additional information.

Now, which complementary, so far unused information can we exploit without reverting to additional training data or extra annotations? The class-discriminative task ideally seeks commonalities shared by all samples in a class that separate them from the other classes. A representation that generalizes to different, unknown classes calls for features that are not just discriminative. They should also be shared by different training classes so they are likely to transfer to and reoccur in unknown classes. However, simply merging classes and subsequently learning to separate these super-classes would not be complementary to the existing representation from the class-discriminative task. Moreover, individual classes already have a large intra-class variability. Learning a common representation for even more heterogeneous super-classes would, therefore, be only more difficult and prone to noise. In contrast, we could learn complementary features that are shared across *subsets of different classes* while still separating from other subsets. Such a representation would be *(i)* complementary to the class-discriminative one, since discrimination is between subsets of the original classes and *(ii)* more likely to transfer, since its features are already shared across classes. However, this directly raises the question of finding such subsets without supervision on what is shared between which subsets. We will now first discuss a grouping-based approach for learning shared features between the training samples in $\mathcal{X}$ before presenting a more direct and simple solution.

Learning shared features by grouping The prevailing learning strategy in absence of supervision is that of clustering-based methods[12, 30, 155]. Given a feature representation f, the training data $\mathcal{X}$ is partitioned into L groups $\mathcal{G}_l$, $l \in 1, \ldots, L$. Mutual closeness of group members due to similar feature representations indicates that these members share certain characteristics. Consequently, discriminating groups $\mathcal{G}_l$ from another encourages the model to learn about these shared group characteristics and the corresponding features that distinguishes different groups. Thus, we can formulate the learning objec-

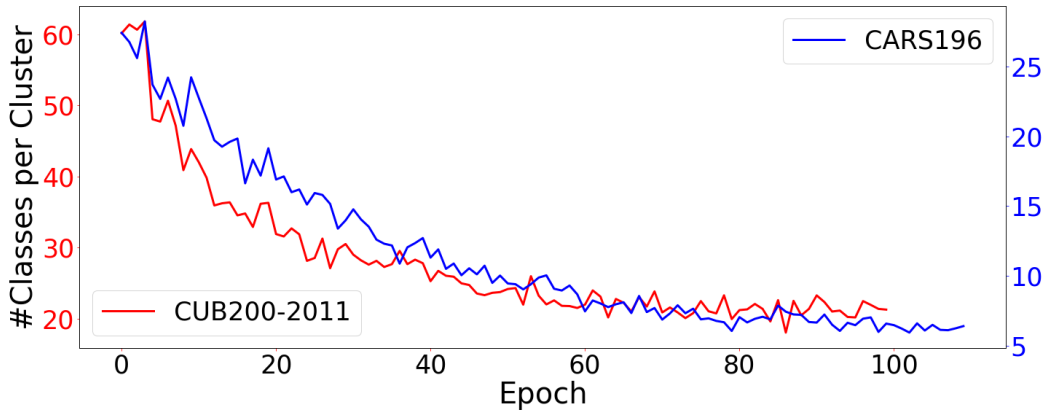


Figure 3.4: *Unique classes per group.* The average number of unique classes per group decreases during training on CARS196[130] and CUB200-2011[239] dataset.

tive as minimization of a standard discriminative ranking loss, such as Eq. 3.1, on triplets defined based on the assignment of samples x_i to groups $\mathcal{G}_l$ [194], i.e. triplets $t^{\mathcal{G}_l}$ sampled from

$$\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l} \triangleq \left\{(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : \; y_i^{\mathcal{G}_l} = y_j^{\mathcal{G}_l} \wedge y_i^{\mathcal{G}_l} \neq y_k^{\mathcal{G}_l}\right\}. \tag{3.3}$$

Here $y_i^{\mathcal{G}_l} = y_j^{\mathcal{G}_l}$ denotes samples x_i, x_j belonging to the same group $\mathcal{G}_l$ and $y_i^{\mathcal{G}_l} \neq y_k^{\mathcal{G}_l}$ indicates x_i, x_j to come from different groups $\mathcal{G}_l, \mathcal{G}_m, l \neq m$.
Unfortunately, clustering-based models are typically strongly biased to learn class-specific structures [30], since images from the same class share many common (class-)properties and thus are likely to be assigned to similar groups. Consequently, in order to learn features which are complementary to the class-discriminative task, we first have to reduce the influence of class characteristics. For this purpose, we perform a per-class feature standardization before grouping our data $\mathcal{X}$: For each ground-truth class $c \in \mathcal{C}$ we compute the mean μ_c and the diagonal of the covariance matrix σ_c based on the features f_i of samples $x_i \in \mathcal{X}$ belonging to class c. To obtain a grouping $\mathcal{G}_l$, we next apply a clustering algorithm like K-Means [146] on the standardized features $f_i = \frac{f_i - \mu_c}{\sigma_c}$, thus reducing the impact of class-specfic information on the feature representations before grouping as presented in our earlier work [194].

Explicit inter-class triplet constraints. The just described procedure which follows the work of MIC [194] enables learning of characteristics shared within a group $\mathcal{G}_l$. However, even though applying feature standardization the impact of class-specific information on these characteristics is still significant as Fig. 3.4 reveals. During training, each group G_l is gradually dominated by only few classes. Consequently, by sampling anchors x_i and positives x_j of triplets $t^{\mathcal{G}_l}$ from the same group ($y_i^{\mathcal{G}_l} = y_j^{\mathcal{G}_l}$), x_i, x_j are increasingly likely to also have the same class label ($y_i = y_j$). As a result, due to a growing intersection between $\mathcal{T}_{\mathcal{X}}$ and $\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$, lots of class-discriminative, thus redundant features are learned. Concluding, only those triplets $t^{\mathcal{G}_l} \in \mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$ will actually provide for new, complementary

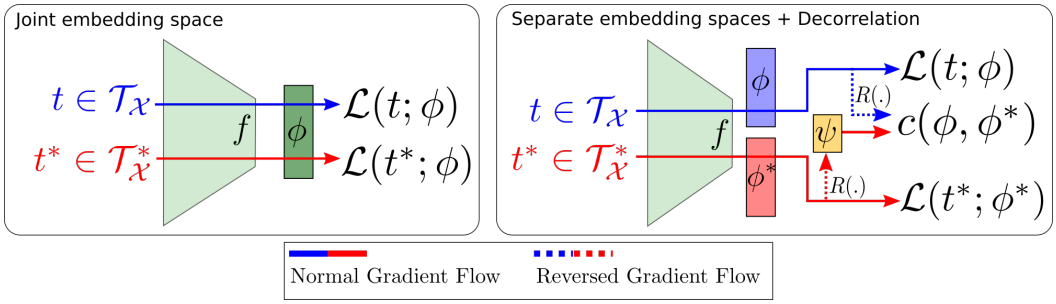


Figure 3.5: *Architecture and gradient flow.* (left) A single encoder ϕ is alternately trained on both tasks. (right) Each task is trained on a dedicated encoder ϕ and ϕ^* based on a shared feature extractor f. Loss is computed per encoder and back-propagated through the shared feature representation f. Using a projection network ψ, we map ϕ^* to the embedding ϕ and compute the decorrelation loss (eq. 3.6) with gradient reversal $R(.)$.

features, where *each* constituent comes from a different ground-truth class so that x_i and x_j are unlikely to share class-specific properties. Following this intuition, we hypothesize that for almost any arbitrarily formed triplet t^* from mutually different classes, the anchor x_i and positive x_j share *some* common pattern when compared to a third, negative image x_k.
Let t^* be such a triplet of images from the set

$$\mathcal{T}^*_{\mathcal{X}} \triangleq \left\{(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : y_i \neq y_j \neq y_k\right\} . \tag{3.4}$$

For each t^*, the commonality between x_i and x_j either represents *(i)* actual shared characteristics across classes which are repetitively supported by many other triplets t^* or *(ii)* some unique or rarely occurring pattern which is typically referred to as noise. Learning such informative characteristics while discarding noisy patterns on $\mathcal{T}^*_{\mathcal{X}}$ constitutes the classical task of DML: Due to the nature of stochastic gradient decent training, deep neural networks only learn by being repetitively exposed to similar training signals. Thus, only the most frequently occurring patterns, i.e. shared characteristics, corroborate their signals and are captured during training, e.g. when learning on imbalanced training classes [33]. Moreover, the learned features are guaranteed to be complementary and also discriminate between different shared characteristics since x_i, x_j, x_k are forced to come from different classes. Fig 3.2 verifies this by comparing the learned embedding by DML trained on $\mathcal{T}_{\mathcal{X}}$ and $\mathcal{T}^*_{\mathcal{X}}$.

Online sampling of inter-class triplets t^*. Shared features can basically be learned between any given training classes from corresponding triplets t^*. However, due to the common regularization of Φ to a unit hypersphere with large dimensionality D (cf. Sec. 3.1.1), distances in Φ are strongly biased towards the analytical mean distance [225]. Thus, to learn shared features between classes from the whole range of distances in ϕ and, in turn, increase their diversity, we employ distance-based sampling [256], which is discussed in detail in Sec. 5.1.

3.1.3 Deep metric learning by combining shared and discriminative characteristics

The features learned in the previous section represent a complementary source of information to the discriminative features from Sec. 3.1.1. Thus, to maximize generalization performance both should be combined. The following discusses strategies for integrating both characteristics, which are compared in the experiments of Sec. 3.3.3.
As we formulated learning shared features as a novel triplet sampling strategy, it can easily be incorporated into any standard ranking loss framework (in the following we use $\mathcal{T}^*_{\mathcal{X}}$). The most natural way to combine both kinds of features is to alternately optimize $\mathcal{L}$ on $t \in \mathcal{T}_{\mathcal{X}}$ and $t^* \in \mathcal{T}^*_{\mathcal{X}}$ using the same encoder ϕ. The combined loss is then formulated as

$$\mathcal{L}^c = \mathcal{L}(t;\phi) + \mathcal{L}(t^*;\phi) \; . \tag{3.5}$$

However, as similarity learned from t and t^* is based on different semantic concepts (class-discriminative vs. class-shared characteristics), simultaneous optimization of a joint encoder ϕ may diminish the individual training signals of both tasks. Hence, to fully exploit the overall training signal, we optimize a dedicated D^* dimensional embedding ϕ^* on t^* to capture the shared characteristics. This requires a second encoder $\phi^* : \mathbb{R}^{D_f} \rightarrow \Phi^* \subset \mathbb{R}^{D^*}$ with parameters θ^*, such that $\phi^*_i \triangleq \phi^*(f_i;\theta^*) = \phi^*(f(x_i;\omega);\theta^*)$. Note that both encoders $\phi(f_i)$ and $\phi^*(f_i)$ act on the same feature representation f_i. Thus, the training signals from $\mathcal{L}(t;\phi)$ and $\mathcal{L}(t^*;\phi^*)$ are merged into the same feature extractor network by backpropagation. Consequently, even though ϕ and ϕ^* optimize different embeddings, both benefit from learning to represent shared *and* discriminative characteristics in f_i. Fig. 3.3 shows that shared characteristics are prominent in ϕ^*, while ϕ is almost random when searching for nearest neighbors across different classes.
While either learning task contributes distinct information to the joint feature extractor f, there may still be redundant overlap in the their training signals. In order to maximize the diversity and information of the overall training signal, we will subsequently decorrelate the embeddings ϕ and ϕ^*. In a first step, we need to make them comparable by learning a projection $\psi : \mathbb{R}^{D^*} \rightarrow \mathbb{R}^D$ from ϕ^*_i to ϕ_i. This is implemented as a regressor network that is trained by maximizing the correlation c between the projection $\psi(\phi^*_i)$ and ϕ_i,

$$c(\phi_i, \phi^*_i) = \|(R(\phi_i) \odot \psi(R(\phi^*_i)))\|^2_2 \; . \tag{3.6}$$

Here, $\odot$ denotes the point-wise product. While the regressor seeks to maximize the correlation c, we invert its gradients (and thus the corresponding training signal) to the embedding representations during backpropagation using a gradient reversal $R(.)$ which flips the gradient sign. As a result, we actually learn to minimize the correlation c and, hence, de-correlate ϕ and ϕ^*. This procedure is inspired by [175], which optimizes an ensemble of learners on the same discriminative DML task so each learner is active on different training classes. In contrast, we aim at learning separate embeddings using different training

tasks to capture shared and discriminative characteristics with minimum overlap. This yields our final training objective

$$\mathcal{L}_{\text{final}} = \mathcal{L}(t;\phi) + \mathcal{L}(t^*;\phi^*) - \gamma \cdot c(\phi,\phi^*) \ , \tag{3.7}$$

where $c(\phi,\phi^*)$ denotes the de-correlation between ϕ,ϕ^* by applying Eq. 3.6 on the individual embedded triplet constituents and the subsequent gradient inversion $R(.)$. The parameter γ balances the metric learning tasks with the de-correlation. Fig. 3.5 visualizes gradient flow and training signal of each embedding ϕ,ϕ^* and Algorithm 1 summarizes the training procedure.
After training, we can now combine the information captured in both encodings by concatenating ϕ_i and ϕ_i^* and obtain distances $d([\phi_i,\phi_i^*],[\phi_j,\phi_j^*])$. However, the experimental analysis in Sec. 3.3.2 shows that both encoders individually already improve over standard DML. By effectively exploiting the shared and discriminative information captured by the feature extractor f and reducing the bias towards the training classes, they exhibit increased generalization onto the test distribution.

3.2 Related Works

Ensemble-based DML. Combining multiple image representations to ensembles is a well-studied line of research to improve the capabilities of DML models [175, 201, 267]. Opitz et al. [175] train many encoding spaces using the same discriminative task while reducing their mutual information. DREML[261] partitions the label space into subsets by means of whole classes and learns independent encoders optimizing the same standard label-based DML task. In contrast, our approach does not represent an ensemble method, as we optimize dedicated encoders for inherently different tasks: standard discriminative DML *and* learning complementary shared characteristics. A novel line of research has been introduced by Lin et al. [141] which proposes to explicitly learn the intra-class variance by training a generative model in parallel to the embedding function. Differently from [141], we directly learn characteristics shared across classes and not only the distribution over the classes. Further, we do not have to revert to a costly generative model, but use standard triplet formulations which can be naturally integrated into standard ranking loss frameworks.

Multi-task learning. The general topic of multi-task learning[19, 182] aims at simultaneously learning multiple classifiers on different semantic concepts and/or datasets, thus sharing a similar motivation with our approach. However, compared to Battarai et al.[19] we require no additional training data and costly extra annotations as we learn shared features solely from the training data already used for the discriminative task. Pu et al.[182] train individual classifiers for groups of whole categories. Our concept of shared features operates on individual samples. Hence, we are able to learn features which are only shared between *some samples* of different classes which is much more flexible.

Algorithm 1: Joint training of ϕ and ϕ^*

Input:
Images $\mathcal{X}$, Feature extractor network f, Embedding network ϕ, Embedding network ϕ^*, Regressor network ψ, Ranking loss $\mathcal{L}$, Batchsize b, Decorrelation weight γ.

$epoch \leftarrow 0$
while *Not Converged* **do**
  repeat
    Get batches of batchsize b
    $\mathcal{B} \leftarrow$ `GetBatch(`$\mathcal{X}$, b`)`
    $\mathcal{B}^* \leftarrow$ `GetBatch(`$\mathcal{X}$, b`)`

    Sample discriminative triplet for each anchor x_i in $\mathcal{B}$ (cf. Sec. 3.1) based on $q^{-1}(d)$
    $\mathcal{T}_{\mathcal{X}} \leftarrow \{x_i, x_j, x_k \overset{q^{-1}}{\sim} \mathcal{B} \mid y_{ij} = 1 \wedge y_{ik} = 0\}$

    Sample explicit inter-class triplet for each anchor x_i in $\mathcal{B}^*$ (cf. Sec. 3.2) based on $q^{-1}(d)$
    $\mathcal{T}^*_{\mathcal{X}} \leftarrow \{x_i, x_j, x_k \overset{q^{-1}}{\sim} \mathcal{B}^* \mid y_{ij} = y_{ik} = y_{jk} = 0\}$

    Alternate optimization (cf. Sec. 3.3)
    $\ell_{\text{discr}} \leftarrow \mathcal{L}(\mathcal{T}_{\mathcal{X}}, \phi) - \gamma \cdot c(R(\phi), R(\phi^*))$
    $\phi, \phi^*, f \leftarrow$ `Backward(`ℓ_{discr}`)`

    $\ell_{\text{inter}} \leftarrow \mathcal{L}(\mathcal{T}^*_{\mathcal{X}}, \phi^*) - \gamma \cdot c(R(\phi), R(\phi^*))$
    $\phi, \phi^*, f \leftarrow$ `Backward(`ℓ_{inter}`)`

  until *end of epoch*;
  $epoch \leftarrow epoch + 1$
end

3.3 Experiments and Analysis

This section presents technical details of our implementation followed by an evaluation of our proposed model on standard Deep Metric Learning benchmarks. Furthermore, to analyze the impact of shared features on embedding generalization, we additionally present ablation studies of our proposed model.

Implementation details. We follow the training protocol of [256] for ResNet50 and [217] for GoogLeNet utilizing the original images without object crops. During training, each image is resized to 256×256, followed by a random crop to 224×224 for

ResNet50 and 227×227 for GoogLeNet, as well as random horizontal flipping. For all experiments, learning rates are set to 10^{-5} for ResNet50 and 10^{-4} for GoogLeNet. We choose the triplet parameters according to [256], with $\alpha = 0.2$. For margin and triplet loss with semihard sampling[205], $m = 4$ images are sampled per class until the batch size is reached[256]. For ProxyNCA[161] $m = 1$, and for n-pair loss[217] $m = 2$. The regressor network ψ (Sec. 3.1.3) is implemented as a two-layer fully-connected network with ReLU-nonlinearity inbetween. We obtain the weighting parameter γ for decorrelation by cross-validation within a range of $[500, 2000]$, depending on the dataset. For implementation we use the PyTorch framework[180]. All experiments are performed on a single NVIDIA Titan X. While training, we use the same ranking loss for optimizing ϕ^* and ϕ, except for ProxyNCA[161]. In this case we train ϕ^* using triplet loss with semi-hard negative mining[205], as $\mathcal{T}^*_{\mathcal{X}}$ operates on individual triplets and thus the concept of proxies is not applicable. For n-pair loss[217] we utilize our described method extended to multiple random negatives.

Benchmark datasets. We evaluated on three standard benchmarks for DML reporting image retrieval performances using Recall@k[107] and the normalized mutual information score (NMI)[151]. Our evaluation protocol follows [256]. For each dataset, we use the first half of the classes for training and the second half for testing. *CARS196*[130]: 16,185 car images divided in 196 classes. *Stanford Online Products (SOP)*[171]: 120,053 images in 22,634 classes from 12 product categories. *CUB200-2011*[239]: 200 classes of bird species for a total of 11,788 images.

3.3.1 Results and comparison with previous works

In Tab. 3.1, 3.2 and 3.3 we compare our approach to state-of-the-art DML methods based on the standard image retrieval task (Recall@K). For our methods, we report the results of the concatenated embedding with a dimensionality of 256. If not stated otherwise, we optimize both the discriminative and shared DML tasks using Margin loss[256]. Thus we also provide its baseline results using 256 dimensions based on our re-implementation for fair comparison. Further, we evaluate our approach for both options to learn shared features, i.e. leveraging $\mathcal{T}^{\mathcal{G}_l}_{\mathcal{X}}$ or $\mathcal{T}^*_{\mathcal{X}}$, respectively. While both options clearly improve over purely discriminative DML methods, using $\mathcal{T}^*_{\mathcal{X}}$ leads to stronger results in general. We conclude that by using triplet constraints which explicitly link different classes, we learn shared features more effectively due to less class-specific information affecting the optimization of the complementary shared feature task. Consequently, in the remainder of the experimental section, we focus on $\mathcal{T}^*_{\mathcal{X}}$.

We outperform other approaches with comparable embedding capacities by at least 2.4% on CARS196[130], 2.5% on CUB200[239] and 1.0% on SOP[171]. Moreover, our model outperforms DREML[261], a large ensemble method, by $\sim 5\%$ on CUB200-2011 (Tab. 3.2) and $\sim 1\%$ on CARS196 (Tab. 3.1). Similar behavior is observed for the clustering task (NMI). The results reported by Ranked list[250] are computed using the InceptionV2[103] architecture and a concatenation of three features layers, totaling 1536

Approach	Dim	R@1	R@2	R@4	NMI
Rank[250][Iv2]	512	74.0	83.6	90.1	65.4
HTG[275]	-	76.5	84.7	90.4	-
HDML[277]	512	79.1	87.1	92.1	69.7
Margin[256]	128	79.6	86.5	90.1	69.1
HTL[71]	512	81.4	88.0	92.7	-
DVML[141]	512	82.0	88.4	93.3	67.6
MIC[194]	128	82.6	89.1	93.2	68.4
D&C[201]	128	84.6	90.7	94.1	70.3
A-BIER[175]	512	82.0	89.0	93.2	-
Rank[250][Iv2]	1536	82.1	89.3	93.7	71.8
DREML[261]	9216	86.0	91.7	95.0	76.4
Margin(ReImp)	256	81.5	88.1	92.8	67.4
Ours ($\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$)	256	85.8	91.3	95.1	**70.3**
Ours ($\mathcal{T}_{\mathcal{X}}^{*}$)[194]	256	**87.0**	**92.1**	**95.4**	69.8

Table 3.1: Evaluation on CARS196[130].

Approach	Dim	R@1	R@2	R@4	NMI
DVML[141]	512	52.7	65.1	75.5	61.4
HDML[277]	512	53.7	65.7	76.7	62.6
HTL[71]	512	57.1	68.8	78.7	-
Rank[250](Iv2)	512	57.4	69.7	79.2	62.6
HTG[275]	-	59.5	71.8	81.3	-
Margin[256]	128	63.6	74.4	83.1	69.0
D&C[201]	128	65.9	76.6	84.4	69.6
MIC[194]	128	66.1	76.8	85.6	69.7
A-BIER[175]	512	57.5	68.7	78.3	-
Rank[250][Iv2]	1536	61.3	72.7	82.7	66.1
DREML[261]	9216	63.9	75.0	83.1	67.8
Margin(ReImp)	256	65.2	75.9	84.5	68.1
Ours ($\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$)[194]	256	67.0	77.3	85.8	69.3
Ours ($\mathcal{T}_{\mathcal{X}}^{*}$)	256	**68.6**	**79.4**	**86.8**	**71.0**

Table 3.2: Evaluation on CUB200-2011[239].

dimensions. Similarly to DREML[261], this significantly increases the capacity of the underlying model, making it not directly comparable.

Detailed evaluation using ResNet50 architecture. In Tab. 3.4 we evaluate our approach based on different ranking losses for optimization: triplet loss with semihard negative sampling[205], n-pair loss[217], ProxyNCA[161] and margin loss with distance

Approach	Dim	R@1	R@10	R@100	NMI
HDML[277]	512	68.7	83.2	92.4	89.3
DVML[141]	512	70.2	85.2	93.8	**90.8**
Margin[256]	128	72.7	86.2	93.8	90.7
A-BIER[175]	512	74.2	86.9	94.0	-
HTL[71]	512	74.8	88.3	94.8	-
D&C[201]	128	75.9	88.4	94.9	90.2
Rank[250][Iv2]	512	76.1	89.1	95.4	89.7
MIC[194]	128	77.2	89.4	95.6	90.0
Rank[250][Iv2]	1536	79.8	91.3	96.3	90.4
Margin(ReImp)	256	76.1	88.1	94.9	89.5
Ours ($\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$)[194]	256	77.7	89.8	95.9	90.0
Ours ($\mathcal{T}_{\mathcal{X}}^{*}$)	256	**78.2**	**90.1**	**96.1**	90.3

Table 3.3: Evaluation on SOP[171].

sampling[256]. We compare the re-implemented baselines with and without learning complementary shared features. For completeness we present for our approach results based on the individual embeddings ϕ, ϕ^* and their concatenation after training our model as described in Sec. 3.1.3. We show the results using ResNet50[91] and present additional results using GoogLeNet[223] in our ablation studies. Our method consistently improves upon the state-of-the-art across all datasets, irrespective of architecture and ranking loss. This clearly indicates the universal benefit of shared feature learning.

Detailed evaluation using GoogLeNet architecture. Similar to Tab. 3.4, we now present evaluations using the GoogLeNet architecture in Tab. 3.5. We provide the results of our re-implementations of the baseline models[161, 205, 217, 256] without and in combination with our proposed approach. The experiment shows that, similar to our ResNet50[91] results, our approach consistently boosts the baseline models in both image retrieval (Recall@1) and clustering (NMI). In particular, we outperform the best baseline recall by 7% on CARS196, 3% on CUB200-2011 and 3.5% on SOP.

3.3.2 Generalization and analysis of the embeddings

This section analyzes the performance of our embeddings ϕ and ϕ^*. It further compares the generalization capabilities of *(i)* pure class-discriminative Deep Metric Learning (only a single embedding ϕ trained on triplets $\mathcal{T}_{\mathcal{X}}$) using margin loss [256] with that of *(ii)* also exploiting shared characteristics (by leveraging $\mathcal{T}_{\mathcal{X}}^{*}$) as suggested in Sec. 3.1.2 and 3.1.3. For the analysis we evaluate different encodings on both train- and testset in Tab. 3.6.

Performance of embeddings. Tab. 3.6 summarizes the performance of the discriminatively trained embedding ϕ, the embedding ϕ^* trained for shared characteristics (only

Approach		Original	Baseline128	Baseline256	ϕ(Ours)	ϕ^*(Ours)	ϕ+ϕ^*(Ours)
	Dataset: CARS196[130]						
Semihard[205]	R@1	51.5	71.9 ± 0.3	72.7 ± 0.3	72.7 ± 0.6	76.5 ± 0.4	$\mathbf{79.4 \pm 0.3}$
	NMI	53.4	64.1 ± 0.3	64.5 ± 0.4	64.1 ± 0.3	63.2 ± 0.3	$\mathbf{66.0 \pm 0.2}$
N-pairs[217]	R@1	71.1	70.2 ± 0.3	70.6 ± 0.2	70.0 ± 0.5	74.8 ± 0.5	$\mathbf{77.2 \pm 0.4}$
	NMI	64.0	62.5 ± 0.3	62.3 ± 0.2	63.2 ± 0.3	62.7 ± 0.1	$\mathbf{64.3 \pm 0.1}$
PNCA[161]	R@1	73.2	79.8 ± 0.1	80.8 ± 0.3	80.2 ± 0.1	81.6 ± 0.1	$\mathbf{82.7 \pm 0.1}$
	NMI	64.9	65.9 ± 0.2	$\mathbf{66.9 \pm 0.3}$	66.1 ± 0.1	64.5 ± 0.3	66.3 ± 0.3
Margin[256]	R@1	79.6	80.1 ± 0.2	81.5 ± 0.3	82.1 ± 0.2	86.2 ± 0.2	$\underline{\mathbf{87.0 \pm 0.1}}$
	NMI	69.1	66.6 ± 0.3	67.4 ± 0.1	68.3 ± 0.3	67.3 ± 0.2	$\underline{\mathbf{69.8 \pm 0.1}}$
	Dataset: CUB200-2011[239]						
Semihard[205]	R@1	42.6	60.6 ± 0.2	61.7 ± 0.3	60.2 ± 0.1	62.5 ± 0.2	$\mathbf{64.6 \pm 0.1}$
	NMI	55.4	65.5 ± 0.3	66.1 ± 0.2	65.8 ± 0.3	67.5 ± 0.2	$\mathbf{68.5 \pm 0.1}$
N-pairs[217]	R@1	51.0	60.4 ± 0.4	60.2 ± 0.3	58.8 ± 0.3	61.1 ± 0.3	$\mathbf{62.9 \pm 0.2}$
	NMI	60.4	66.1 ± 0.4	65.0 ± 0.2	65.1 ± 0.4	66.0 ± 0.2	$\mathbf{67.8 \pm 0.2}$
PNCA[161]	R@1	61.9	64.0 ± 0.1	64.6 ± 0.2	63.6 ± 0.2	65.7 ± 0.2	$\mathbf{66.4 \pm 0.2}$
	NMI	59.5	$\mathbf{68.1 \pm 0.2}$	68.0 ± 0.2	67.5 ± 0.2	67.5 ± 0.3	$\mathbf{68.1 \pm 0.2}$
Margin[256]	R@1	63.6	63.6 ± 0.3	65.2 ± 0.3	66.2 ± 0.3	67.4 ± 0.4	$\underline{\mathbf{68.6 \pm 0.2}}$
	NMI	69.0	68.5 ± 0.3	68.1 ± 0.3	69.2 ± 0.5	69.7 ± 0.6	$\underline{\mathbf{71.0 \pm 0.6}}$
	Dataset: SOP[171]						
Semihard[205]	R@1	-	73.5 ± 0.2	74.7 ± 0.3	75.3 ± 0.2	65.8 ± 0.3	$\mathbf{75.5 \pm 0.2}$
	NMI	-	89.2 ± 0.2	89.4 ± 0.2	89.7 ± 0.1	86.9 ± 0.2	$\mathbf{89.8 \pm 0.1}$
N-Pairs[217]	R@1	67.7	71.3 ± 0.3	72.8 ± 0.2	74.1 ± 0.2	67.8 ± 0.2	$\mathbf{74.6 \pm 0.1}$
	NMI	88.1	89.2 ± 0.2	89.2 ± 0.3	89.8 ± 0.2	87.3 ± 0.1	$\mathbf{89.9 \pm 0.1}$
Margin[256]	R@1	72.7	74.4 ± 0.2	76.1 ± 0.3	77.7 ± 0.2	72.2 ± 0.2	$\underline{\mathbf{78.2 \pm 0.1}}$
	NMI	90.7	89.6 ± 0.2	89.5 ± 0.2	90.1 ± 0.2	88.8 ± 0.2	$\underline{\mathbf{90.3 \pm 0.1}}$

Table 3.4: *Evaluation using different ranking losses using ResNet50 [91] backbone architecture.* Original: results reported in the original paper. Baseline<dim>: our implementation with ResNet50 and <dim> embedding dimensions. ϕ,ϕ^*: class and shared embedding with 128 dimensions each. ϕ+ϕ^*: embedding concatenation resulting in 256 dimensions. Our model is trained using Eq.3.7. **bold**: best result for the given loss. **underlined**: best result on the dataset. We report 5-run average and standard deviation.

for our approach), ϕ with random weight re-initialization after training ($\phi(\mathcal{N})$)[1], and the feature extractor f on the train and test set of CARS196[130] dataset. Comparing the test set results of ϕ and f (for both *(i)* and *(ii)*) shows that the embedding ϕ performs worse than the feature encoding f. Consequently, ϕ is not able to effectively use the information captured in the features f. Comparing ϕ to $\phi(\mathcal{N})$ confirms this, since the randomly re-initialized embedding actually performs better in testing. Moreover, learning shared characteristics improves the features f, which clearly demonstrates them being complementary to the standard discriminative training signal. Note that in our approach, *both* embeddings (ϕ and ϕ^*) have equal access to the discriminative *and* shared features captured in f. However, we observe that ϕ^* performs 4.0% better than ϕ on the test set. This indicates that ϕ is overfitting to the train classes while ϕ^* is able to successfully use both,

[1]Note that the weights of f remain trained and are not re-initialized.

Approach		Original	Baseline512	Baseline1024	ϕ(Ours)	ϕ^*(Ours)	ϕ+ϕ^*(Ours)
	Dataset: CARS196[130]						
Semihard[205]	R@1	51.5	67.3 ± 0.3	66.3 ± 0.4	64.4 ± 0.6	71.4 ± 0.2	$\mathbf{72.7 \pm 0.2}$
	NMI	53.4	60.3 ± 0.3	58.9 ± 0.2	56.9 ± 0.2	58.2 ± 0.5	$\mathbf{60.5 \pm 0.2}$
N-pairs[217]	R@1	71.1	67.4 ± 0.4	64.5 ± 1.2	60.4 ± 0.8	67.8 ± 0.3	$\mathbf{67.0 \pm 0.2}$
	NMI	64.0	60.0 ± 0.3	58.6 ± 0.4	56.1 ± 0.2	59.5 ± 0.3	$\mathbf{60.1 \pm 0.2}$
PNCA[161]	R@1	73.2	71.1 ± 0.2	71.2 ± 0.2	73.8 ± 0.2	76.7 ± 0.2	$\underline{\mathbf{78.3 \pm 0.1}}$
	NMI	64.9	60.2 ± 0.2	58.9 ± 0.2	61.7 ± 0.2	62.3 ± 0.4	$\mathbf{63.8 \pm 0.3}$
Margin[256]	R@1	79.6	72.6 ± 0.3	73.3 ± 0.2	74.3 ± 0.3	75.5 ± 0.3	$\mathbf{77.4 \pm 0.3}$
	NMI	69.1	63.2 ± 0.2	62.2 ± 0.2	63.3 ± 0.5	59.3 ± 0.4	$\underline{\mathbf{64.1 \pm 0.3}}$
	Dataset: CUB200-2011[239]						
Semihard[205]	R@1	42.6	54.8 ± 0.3	56.6 ± 0.4	57.4 ± 0.3	60.2 ± 0.3	$\mathbf{60.9 \pm 0.2}$
	NMI	55.4	61.9 ± 0.2	62.5 ± 0.5	63.5 ± 0.4	65.3 ± 0.3	$\mathbf{66.0 \pm 0.2}$
N-pairs[217]	R@1	51.0	52.2 ± 0.3	50.9 ± 1.4	50.8 ± 0.1	52.5 ± 0.2	$\mathbf{54.8 \pm 0.2}$
	NMI	60.4	60.7 ± 0.3	59.2 ± 1.4	59.5 ± 0.3	61.1 ± 0.3	$\mathbf{61.6 \pm 0.2}$
PNCA[161]	R@1	61.9	55.9 ± 0.2	56.4 ± 0.3	58.6 ± 0.2	60.8 ± 0.1	$\mathbf{61.5 \pm 0.1}$
	NMI	59.5	62.9 ± 0.1	62.0 ± 0.2	64.5 ± 0.2	65.5 ± 0.2	$\mathbf{65.8 \pm 0.2}$
Margin[256]	R@1	63.6	58.3 ± 0.3	59.3 ± 0.3	60.9 ± 0.3	61.9 ± 0.3	$\underline{\mathbf{62.6 \pm 0.2}}$
	NMI	69.0	64.8 ± 0.2	64.4 ± 0.3	65.0 ± 0.3	66.2 ± 0.2	$\underline{\mathbf{66.7 \pm 0.2}}$
	Dataset: SOP[171]						
Semihard[205]	R@1	-	67.3 ± 0.1	67.4 ± 0.3	70.7 ± 0.2	67.5 ± 0.3	$\mathbf{71.1 \pm 0.2}$
	NMI	-	88.4 ± 0.1	88.4 ± 0.3	88.6 ± 0.1	86.8 ± 0.1	$\mathbf{89.2 \pm 0.1}$
N-Pairs[217]	R@1	67.7	67.2 ± 0.3	63.4 ± 0.4	68.3 ± 0.3	66.6 ± 0.1	$\mathbf{68.9 \pm 0.1}$
	NMI	88.1	88.3 ± 0.2	87.2 ± 0.3	88.5 ± 0.2	86.4 ± 0.3	$\mathbf{88.8 \pm 0.1}$
Margin[256]	R@1	72.7	68.5 ± 0.2	67.1 ± 0.3	71.0 ± 0.3	69.2 ± 0.3	$\underline{\mathbf{72.0 \pm 0.3}}$
	NMI	90.7	88.6 ± 0.2	87.6 ± 0.3	88.4 ± 0.1	87.5 ± 0.2	$\underline{\mathbf{89.1 \pm 0.1}}$

Table 3.5: *Evaluation using different ranking losses with GoogLeNet[223] backbone architecture.* Original: results reported in the original paper. Note that the original works of Margin[256] uses ResNet50 and ProxyNCA (PNCA)[161] uses Inception-BN. All other use GoogLeNet. Baseline<dim>: our re-implementation with GoogLeNet architecture and <dim> embedding dimensions. ϕ,ϕ^*: class and shared embedding with 512 dimensions each. ϕ+ϕ^*: embedding concatenation resulting in 1024 dimensions. **bold**: best result for the given loss. **underlined**: best result on the dataset. We report 5-run average and standard deviation.

the strong discriminative features and the more general shared features. Thus, ϕ^* generalizes more effectively to *unseen* test classes. The next paragraph now further analyzes this observation.

Generalization analysis. Additionally to the performance of the individual embeddings on train and test set, Tab. 3.6 further shows their difference, the generalization gap. For *(i)* (Tab. 3.6 top) we observe a large gap of -10.8% compared to -5.8% of ϕ in our approach, thus indicating strong overfitting. Indeed, simply randomly re-initializing the weights of the encoder ϕ before testing already improves the gap to -6.1% and increases test performance by 1.5%. Computing distances based on f further reduces the

Model	Representation	Dim.	Trainset	Testset	Generalization gap
Margin[256]	ϕ	128	90.7	79.9	-10.8
	$\phi\,(\mathcal{N})$	128	87.5	81.4	-6.1
	f	2048	87.0	82.8	-4.2
Ours ($\mathcal{T}^*_{\mathcal{X}}$)	ϕ	128	87.9	82.1	-5.8
	ϕ^*	128	84.8	**86.2**	**1.4**
	$\phi\,(\mathcal{N})$	128	84.2	82.9	-1.3
	f	2048	83.7	83.5	-0.2

Table 3.6: *Generalization gap study*. The generalization gap is measured as the difference of performances on the train and test set of CARS196[130] dateset. Performance measured in Recall@1. $\mathcal{N}(.)$ indicates random weight reset of the embedding layer connected to the feature extractor f. *Dim.* denotes the dimensionalitay of a representation.

Noise	Margin[256]	+ dropout	+ noise output	+ noise input	Ours
Rec@1	79.9	80.5	79.8	80.4	**83.2**

Table 3.7: *Comparison of our approach against standard generalization techniques*. ResNet50[91] is trained on CARS196[130] and Recall@1[107] is reported. Embedding dimensionality is 128 for all cases.

generalization gap. For *(ii)*, Tab. 3.6 bottom not only shows a significant increase in test performance as discussed above but also an improvement in generalization compared to *(i)* due to the additional shared characteristics. The generalization gap is significantly smaller for each encoding. Thus, the benefit of learning shared characteristics for improved generalization in the transfer learning problem addressed by DML seems to be twofold: it not only adds complementary information to the features f but also regularizes training to reduce overfitting to the training classes. Note, that overfitting in transfer learning is of different nature and significantly more challenging to counteract than in standard learning settings with i.i.d. training and testing data. In the latter case this issue can be addressed by training on more data from the underlying training distribution or regularizing the adaptation of a model to the available training samples. However, such techniques can only have small impact in the presence of train-test distribution shifts, as even an ideal representation of the training data may not transfer equally well to a unknown testing distribution. Further, as we are measuring the test performance on *unknown test classes*, we do not know if the overfitting effect would also be as severe when evaluating on an i.i.d. test set (as class-discriminative training is actually the standard way to learn on such data). Therefore, to support our hypothesis, we apply standard regularization techniques to a discriminative DML baseline model.

Comparison with standard generalization techniques. In Tab.3.7 we compare our approach with techniques typically used to improve generalization in deep neural networks such as dropout [218] and noise injection [164], which proved to be effective

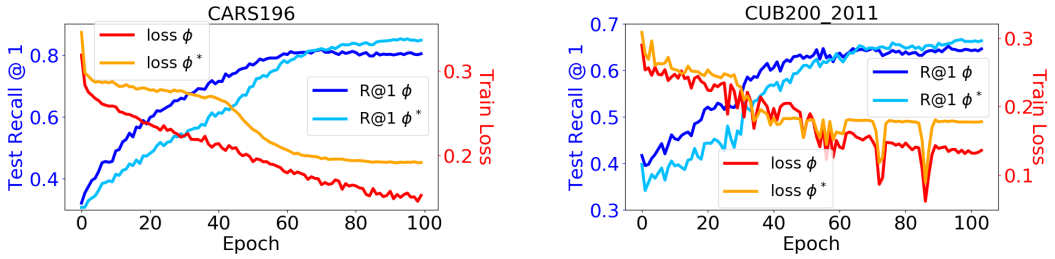


Figure 3.6: *Training curves.* Train loss and test recall@1 for the class ϕ and shared ϕ^* encoders. The model is trained using margin loss[256] with ResNet50[91] on CARS[130] and CUB[239].

for classification and representation learning. As baseline we use margin loss[256] on ResNet50[91]. For fair comparison, we use for our approach the same encoder for both the discriminative and shared task (as also discussed in Tab. 3.8 denoted as 'both') and thus the same architecture as for the analyzed techniques. The evaluation is performed on CARS196[130] reporting Recall@1 on the test set. In particular, we apply dropout between the features f and the embedding ϕ which gives a little boost of 0.6% over the baseline, which is, however, minor respect to the 3.3% gain obtained by our approach. Applying Gaussian noise to the input provides little improvement, while Gaussian noise on the network output reduces performance. Concluding, our approach adds actual complementary information to the discriminative training signal in form of complementary features, and does not only act as a regularization by inducing noise.

Analysis of training progress Fig. 3.6 analyzes the learning behavior of the encoder ϕ and ϕ^* in our model based on training loss and Recall@1 on the test set during training. Evidently, learning shared characteristics across classes that still separate from others is initially a harder task than only discriminating between classes. Thus, we would expect a weaker performance in earlier training epochs. Further, since shared characteristics are less specialized to the training classes, they yield higher overall performance (cf. Tab. 3.6). And indeed, while ϕ^* is initially weaker than ϕ, it continues to increase when ϕ is already saturated.

3.3.3 Ablation studies

Subsequently, we conduct ablation studies for different aspects of our approach. We first analyze our proposed architecture and then examine different triplet assembling and sampling strategies for learning shared characteristics. Further ablations are shown in the supplementary.

Architecture. Sec. 3.1.3 proposes two options for jointly learning discriminative and shared characteristics during training: Alternately optimizing the same single embedding and learning dedicated embeddings for ϕ and ϕ^* with and without decorrelation. Tab. 3.8

Arch.	discr. only[256]	shared only	both	both+sep	both+sep+decor
Rec@1	81.5	36.1	83.2	84.1	**87.0**

Table 3.8: *Architecture study* for our approach by computing Recall@1 on CARS196[130] dataset. We compare the purely discriminative baseline leveraging only $\mathcal{T}_{\mathcal{X}}$ (*discr. only*), a model trained only on shared characteristics, i.e. using only $\mathcal{T}_{\mathcal{X}}^{*}$ (*shared only*), as well as learning both tasks simultaneously using the same encoder (*both*), separate encoders (*both+sep*) and separate encoders with decorrelation (*both+sep+decor*).

compares these options against baselines trained only on either $\mathcal{T}_{\mathcal{X}}$ or $\mathcal{T}_{\mathcal{X}}^{*}$. Firstly, we observe that only learning the shared task (*shared only*) results in a huge drop in performance to 36.1% compared to 81.5% of the discriminative baseline (*discr. only*). This result is to be expected and easily explained: By neglecting the strong discriminative learning signal obtained from optimizing on $\mathcal{T}_{\mathcal{X}}$, no class concept is learned, which stands in contrast to the *class-based* evaluation protocol of nearest neighbor retrieval accuracy. This highlights the importance of of the discriminative task for learning a reasonable distance metric. However, *adding* the shared task to the discriminative baseline consistently improves performance independent of our proposed options for joint optimization. Even the simplest option, i.e. using a single encoder for learning from both $\mathcal{T}_{\mathcal{X}}$ and $\mathcal{T}_{\mathcal{X}}^{*}$ (*both*) improves over the baseline by 1.7%. Further, we see an additional gain of 0.9% when optimizing separate encoders, i.e. ϕ for $\mathcal{T}_{\mathcal{X}}$ and ϕ^{*} for $\mathcal{T}_{\mathcal{X}}^{*}$ (*both+sep*) which is explained by the reduced interference between both tasks during training. Finally, using separate encoders allows for explicit de-correlation *(both+sep+decor)* to minimize the overlap in the captured characteristics between ϕ and ϕ^{*}, yielding another significant gain of 3%.

Strategies for assembling shared triplets. Tab. 3.10 evaluates different strategies to assemble triplets for learning shared characteristics: $\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$*-std.*: Sampling from $\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$ without using feature standardization before grouping. This strategy adds 0.8% to the purely discriminative baseline. However, since these surrogate classes tend to resemble the ground-truth classes in the train set, the performance is significantly worse than our best strategy. Thus, redundant and mostly discriminative signals are added. $\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$*+std.*: Sampling from $\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$ with feature standardization as proposed in [194]. Even though the effect of class-specific information is strongly reduced and thus performance is boosted by 3.5%, this strategy is still inferior to sampling triplets from $\mathcal{T}_{\mathcal{X}}^{*}$. *min d(a,p)*: Sampling from $\mathcal{T}_{\mathcal{X}}^{*}$, but for a given anchor, we constrain sampling the positive by always choosing the closest sample in a batch based on distances defined by ϕ^{*}. This follows the intuition

γ	0	5	50	200	500	1000	2500
Rec@1	84.1	84.3	85.2	86.0	**87.0**	85.5	82.1

Table 3.9: *Controlling the influence of decorrelation by varying the weighting γ in the both+sep+decor setup.*

Setup	Base[256]	$\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$ - std.	$\mathcal{T}_{\mathcal{X}}^{\mathcal{G}_l}$ + std.	min $d(a,p)$	NoConstr	$\mathcal{T}_{\mathcal{X}}^{*}$
Rec@1	81.5	82.3	85.8	82.1	86.0	**87.0**

Table 3.10: *Comparison of different sampling strategies.* Evaluation based on Recall@1 on the CARS196[130] dataset.

that mutually close samples are more likely to share some characteristic. Applying this strategy loses 5% compared to our best result. We conclude that shared characteristics can be learned from almost all image pairs. Thus, restricting the sample range, neglects very important information. *No constraint*: we report numbers for unconstrained assembling of shared triplets as a proxy for $\mathcal{T}_{\mathcal{X}}^{*}$, i.e. anchors, positives and negatives are randomly sampled. With 86.0% this simplest strategy works very well. We reason that the performance drop of 1.0% is explained by direct disagreement between some of the triplets sampled from $\mathcal{T}_{\mathcal{X}}^{*}$ and $\mathcal{T}_{\mathcal{X}}$, distorting the feature extractor f. $\mathcal{T}_{\mathcal{X}}^{*}$: Our proposed strategy of sampling each constituent of a triplet from a different class, i.e. sampling triplets from $\mathcal{T}_{\mathcal{X}}^{*}$.
Note that all proposed strategies show improvement over the baseline (margin loss [256]), clearly proving that learning complementary features is crucial for improved generalization of deep metric learning.

Influence of de-correlation on generalization. To evaluate the relevance of de-correlation between our embeddings ϕ and ϕ^*, Tab. 3.9 examines generalization performance for different values of γ. As we see, increasing the de-correlation boosts performance over the non-decorrelated baseline (with $\gamma = 0$) for a robust interval. However, if γ becomes to large, performance drops below the baseline as enforcing de-correlation strongly dominates the actual DML training signal.

Strategies for sampling shared triplets. In Tab. 3.11 we investigate different procedures for online sampling of shared triplets $\mathcal{T}_{\mathcal{X}}^{*}$. We fix the training procedure for ϕ (Margin [256]) and vary the shared triplet sampling procedure. We see that sampling triplets t^* from a broad range of distances and thus diverse classes (cf. Sec. 3.1.2) is essential for effectively learning inter-class features. This holds especially for distance-based sampling [256] which encourages triplets with anchor-negative pairs sampled uniformly over distances.

Sampling	$\mathcal{T}_{\mathcal{X}}^{*}$ + random	$\mathcal{T}_{\mathcal{X}}^{*}$ + semihard[205]	$\mathcal{T}_{\mathcal{X}}^{*}$ (distance [256])
Rec@1	82.0	85.3	**87.0**

Table 3.11: *Relevance of sampling methods for shared triplets.* $\mathcal{T}_{\mathcal{X}}^{*}$ uses distance-based sampling [256] by default. ϕ is trained with [256] for each test. Only sampling for ϕ^* is changed.

tSNE projection of image embeddings Fig. 3.7 and 3.8 show 1000 images sampled randomly from the CARS196[130] training set and projected in 2D by applying t-SNE[150] on the image encodings. For Fig. 3.7 the underlying model is trained solely on the discriminative task, while in Fig 3.8 the model is trained solely on the shared feature tasks. In Fig. 3.7 the model learns to group classes very compactly and far from each other, which indicates strong adaptation to the training classes. In Fig. 3.8 images which share visual properties are grouped closer together, independently from their ground-truth labels. This results in complementary features which generalize better to new data.

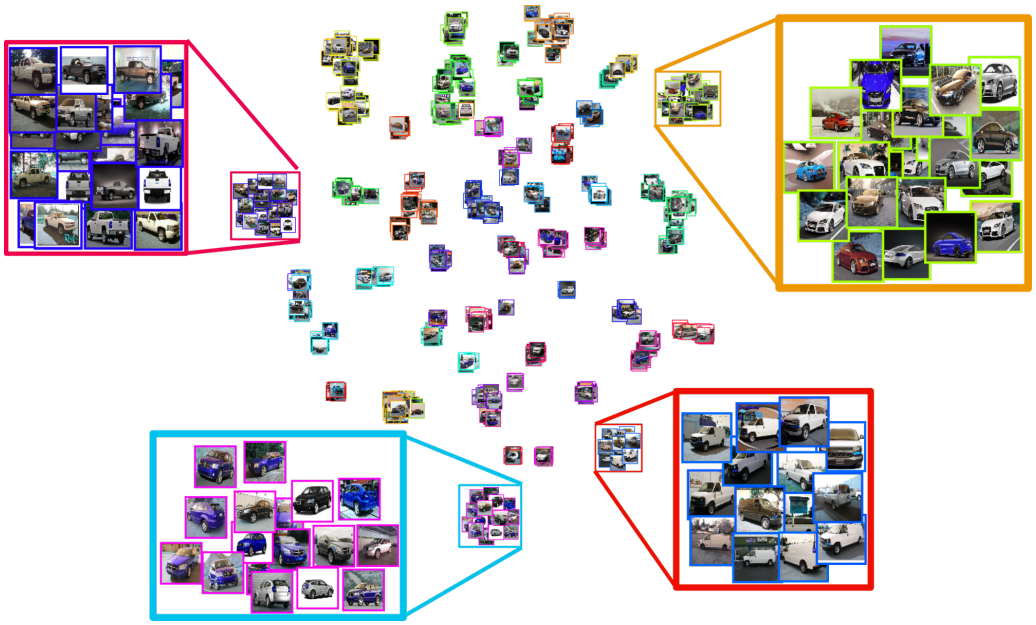


Figure 3.7: *Image projections resulting from a model trained on discriminative task.* The contour color of the individual images indicate the ground-truth class.

3.4 Discussion

This chapter addressed and analyzed the generalization issues of standard deep metric learning approaches arising from their purely discriminative nature. As a remedy, we propose to additionally learn characteristics shared across different classes, which are more likely to transfer to unseen test data. To this end, we additionally train a dedicated encoder on a novel ranking task, explicitly linking samples across classes. Moreover, we show how to combine both discriminative and shared feature learning during training. Evaluations on standard metric learning datasets show that our simple method provides a strong, loss- and architecture independent boost, achieving new state-of-the-art results.

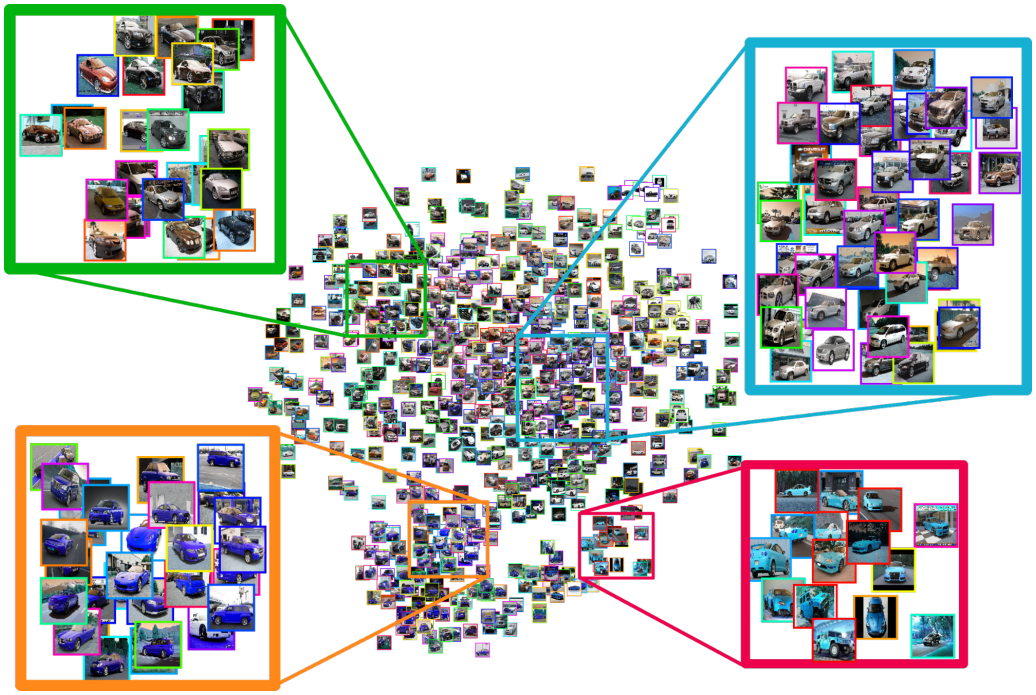


Figure 3.8: *Image projections resulting from a model trained on shared task.* The contour color of the individual images indicate the ground-truth class.

4 Diverse Visual Feature Aggregation

Learning about features which are complementary to those learned by standard, class-discriminative training, proved to help to alleviate the overfitting tendencies of DML models (cf. Ch. 3). As a result, these enhanced models are more likely to capture and describe samples and even classes outside the training distribution. Inspired by this insight, we ideally find additional sources of complementary features to further increase out-of-distribution generalization. Thus the question at hand is, which other training signals can we leverage to learn such features and how to formulate the corresponding learning tasks by only resorting to the available training images and class labels?
Recent breakthroughs in self-supervised learning have shown that contrastive image relations inferred from images themselves yield rich feature representations which even surpass the transfer learning capabilities of supervised features to novel downstream tasks [36, 90, 174]. However, although DML typically also learns from contrasting data relations in the form of pairs [86], triplets [205, 256] or more general image tuples [37, 171], the complementary benefit of self-supervision in DML is largely unstudied. Moreover, the commonly available class assignments give rise to image relations aside from the standard, supervised learning task of 'pulling' samples with identical class labels together while 'pushing' away samples with different labels. As such ranking-based learning is not limited to class-discrimination only, other relations can be exploited to learn beneficial data characteristics which so far have seen little coverage in DML literature.
In this chapter, we extend to notion of learning diverse features for improved generalization in DML by designing learning tasks complementing standard supervised training and representing different relationships between our training classes and samples: *(i)* features discriminating among classes, *(ii)* features shared across different classes, *(iii)* features capturing variations within classes and *(iv)* class-independent features contrasting between individual images. Finally, we present how to effectively incorporate them in a unified learning framework. In our experiments we the study mutual benefits of these tasks and show that joint optimization with each task added further improves generalization performance as shown in Fig. 7.1 (left), outperforming the state-of-the-art in DML. In summary, in this chapter we

- design novel triplet learning tasks resulting in a diverse set of features and study their complementary impact on supervised DML.

- adopt recent contrastive self-supervised learning to the problem of DML and extend it to effectively support supervised learning, since direct incorporation of self-supervised learning does not benefit DML (cf. Fig. 7.1) (right).

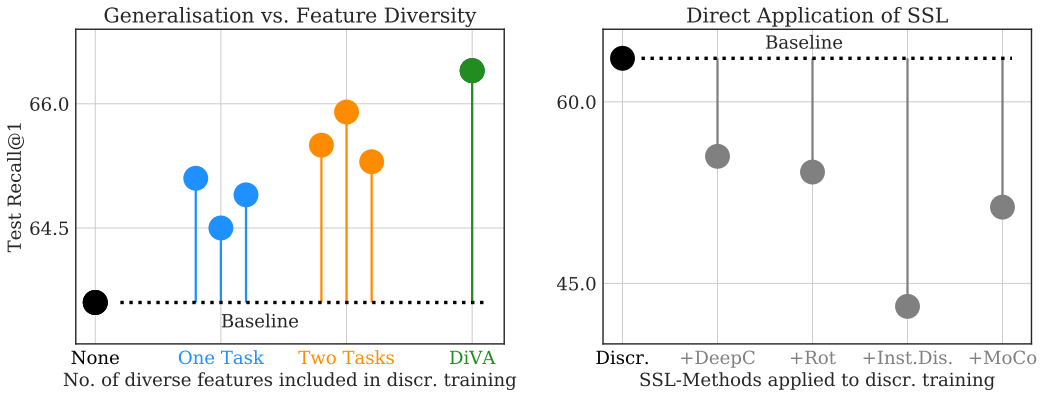


Figure 4.1: *DML using diverse learning tasks vs. naive incorporation of self-supervision only.* (Left) Generalization performance increases with each task added to training, independent of the exact combination of our proposed tasks (blue: one extra task, orange: two extra tasks, green: all tasks). (Right) Directly combining supervised learning with self-supervised learning techniques such as DeepC(luster) [30], Rot(Net) [72], Inst.Dis(crimination) [258] or Mo(mentum)Co(ntrast) [90] actually hurts DML generalization.

- show how to effectively incorporate these learning tasks in a single model, resulting in state-of-the-art performance on standard DML benchmark sets.

This chapter is based on our publication *'DiVA: Diverse Visual Feature Aggregation for Deep Metric Learning'* [156].

4.1 Diverse Complementary Learning Tasks for Similarity Learning

As already discussed previously, in supervised DML we typically learn a single embedding representation ϕ of our images which discriminates between given training classes. Thus, its underlying feature representation is trained to predominantly capture highly discriminative features while being invariant to image characteristics which do not facilitate training class separation. However, as we are interested in generalizing to unknown test distributions, we should rather aim at maximizing the amount of features captured from the training set $\mathcal{X}$, thereby increasing the expressiveness of the embedding ϕ to capture and accurately represent out-of-distribution images and potentially unseen classes. In chapter 3 we show how we can find an additional training signal yielding complementary shared features. Following, we now extend this idea and present a multi-task learning framework for similarity learning to even further increase the expressiveness our learned distance function. Since our model is based on the results of Sec. 3.1, we use the same definitions and notations introduced in Sec. 3.1.1.

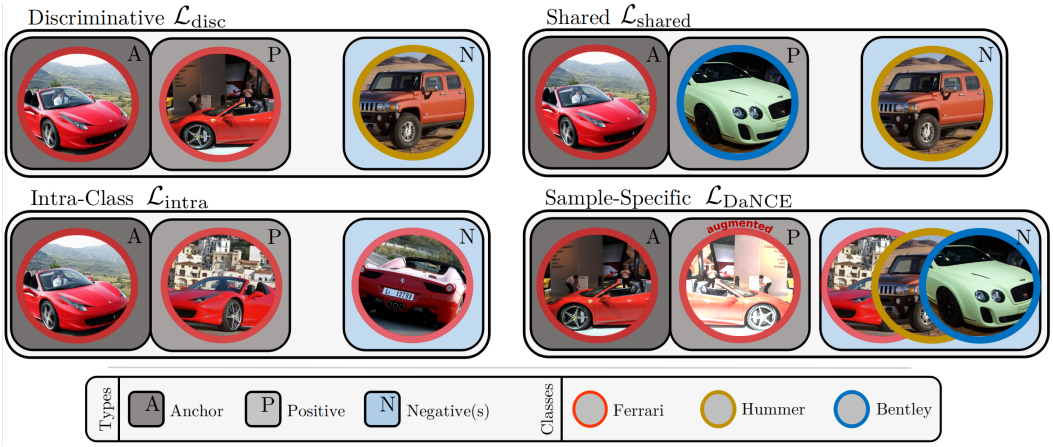


Figure 4.2: *Schematic description of each task.* We learn four complementary tasks to capture features focusing on different data characteristics. The standard *class-discriminative* task which learning features separating between samples of different classes, the *shared* task which captures features relating samples across different classes, a *sample-specific* task to enforce image representations invariant to transformations and finally the *intra-class* task modelling data variations within classes.

4.1.1 Diverse learning tasks for DML

We now discuss the our tasks for learning a diverse set of features, resorting only to the standard training information provided in a DML problem, thus not requiring any additional annotations or side information. Each of these tasks is designed to learn features which are conceptually neglected by the others to be mutually complementary. First, we introduce the intuition behind each feature type, before describing how to learn it at the example of triplet-based objectives $\mathcal{L}(t;\phi)$ such as (3.1) in Sec. 3.1.1 with t representing a triplet of images.

Class-discriminative features. These features are learned by standard class-discriminative optimization of ϕ and focus on data characteristics which allow to accurately separate one class from all others. It is the prevailing training signal of common classification-based [51, 245, 268], proxy-based [161, 183] or ranking-based [175, 194] approaches. For the latter, we can perform training by means of triplets

$$\mathcal{T}_{\mathcal{X}}^{\text{disc}} \triangleq \left\{(x_i, x_j, x_k) \in \mathcal{X}\times\mathcal{X}\times\mathcal{X} : y_i = y_j \neq y_k\right\}, \tag{4.1}$$

and optimization of

$$\mathcal{L}_{\text{disc}} = \frac{1}{Z}\sum_{t\in\mathcal{T}_{\mathcal{X}}^{\text{disc}}} \mathcal{L}(t;\phi)\,, \tag{4.2}$$

thus minimizing embedding distances between samples of the same class while maximizing it for samples of different classes. Again Z is a normalizing constant. Moreover, the

discriminative signal is important to learn how to aggregate features into classes, following the intuition of “the whole is more than the sum of its parts" analyzed in Gestalt theory [238].

Class-shared features. In contrast to discriminative features which look for characteristics separating classes, class-shared features capture commonalities shared across classes as discussed in the previous chapter 3. For instance, birds and cars share a similar variety in colors, which are of little help when separating between them. However, to learn about this characteristic is actually beneficial, when describing other colorful object classes like flowers or fish. Given suitable label information, learning such features would naturally follow the standard discriminative training setup. However, having only class labels available, we must resort to approximations. To this end, we exploit the hypothesis that for most arbitrarily sampled triplets

$$\mathcal{T}_{\mathcal{X}}^{\text{shared}} \triangleq \left\{(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : y_i \neq y_j \neq y_k\right\}, \tag{4.3}$$

i.e. each constituent coming from mutually different classes, the anchor x_i and positive x_j share some common pattern when compared the negative image x_k. Commonalities which are frequently observed between classes y_i, y_j, will occur more often than noisy patterns which are unique to few $t \in \mathcal{T}_{\mathcal{X}}^{\text{shared}}$, which is commonly observed when learning on imbalanced data [33, 70, 126]. Learning class-shared features is then performed by

$$\mathcal{L}_{\text{shared}} = \frac{1}{Z} \sum_{t \in \mathcal{T}_{\mathcal{X}}^{\text{shared}}} \mathcal{L}(t, \phi) \,. \tag{4.4}$$

As deep networks learn from frequent backpropagation of similar learning signals resulting in informative gradients, only prominent shared features are captured. Further, since shared features can be learned between any classes, we need to warrant diverse combinations of classes in our triplets $\mathcal{T}_{\text{shared}}$. Thus, enabling triplets to be sampled from the whole range of difficulty using distance-based sampling [256] is crucial to avoid any bias towards samples which are mostly far (random sampling) or close (hard-negative sampling) to a given anchor x_i. See 5.1 for a detailed discussion on triplet sampling strategies.

Intra-class features. The tasks defined so far model image different relations between classes. In contrast, intra-class features describe variations within a given class. While these variations may also apply to other classes [141, 194] (thus exhibiting a certain overlap with class-shared features) more class-specific details are targeted, thus explicitly retaining intra-class variance when learning ϕ. As shown by Lin et at. [141], employing generative models allows to add additional intra-class signal to a certain extend. However, this greatly increases the computational cost and, thus, is not considered in this work. Instead, to capture such data characteristics we again resort to triplet constraints which can be naturally incorporated into DML frameworks as shown in Sec. 3.1, *'Explicit inter-class triplet*

constraints'. Consequently, to train this task we follow a similar intuition as for learning class-shared features: We define triplets

$$\mathcal{T}_{\mathcal{X}}^{\text{intra}} \triangleq \left\{(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : y_i = y_j = y_k\right\} \tag{4.5}$$

thus using only triplets of images with all constituents coming from the same class when minimizing

$$\mathcal{L}_{\text{intra}} = \frac{1}{Z} \sum_{t \in \mathcal{T}_{\mathcal{X}}^{\text{intra}}} \mathcal{L}(t; \phi) \ . \tag{4.6}$$

Sample-specific features. Recent approaches for self-supervised learning [9, 90, 174] proved that informative features exhibiting strong generalization for transfer learning can be learned only from images themselves, without resorting to a supervised learning task. Looking at the learning tasks we defined so far which are based on the concept of classes, self-supervised approaches offer another complementary source of features.

Overview: Self-supervised representation learning literature: Commonly, self-supervised representation learning aims to learn transferable feature representations from unlabelled data, and is typically applied as pre-training for downstream tasks [90, 155, 165]. Early methods on unsupervised representation learning are based on sample reconstructions [123, 234] which have been further extended by interpolation constraints [18] and generative adversarial networks [38, 55, 59]. Further, introducing manually designed surrogate objectives encourage self-supervised models to learn about data-related properties. Such tasks range from predicting image rotations [72] to solving visual jigsaw puzzles by spatial image decomposition and shuffling [165, 168] or leveraging inductive biases of neural networks using clustering algorithms [30]. Recently, self-supervision approaches based on *contrastive learning* result in strong features performing close to or even stronger than supervised pretraining [36, 90, 159, 227] by leveraging invariance to realistic image augmentations. As the these approaches are essentially defined on pairwise image relations, they share common ground with ranking-based DML. Therefore, contrastive learning constitutes an attractive approach to include instance-specific features into our multi-task similarity learning framework.

Noise contrastive estimation (NCE) [84] is the driving mechanism behind the aforementioned contrastive learning frameworks. As NCE learns to increase the correlation between embeddings of an anchor sample and a similar positive sample by constrasting against a set of negative samples, it naturally translates to DML. He et al. [90] proposed an efficient self-supervised framework which first applies data augmentation to generate positive surrogates $\tilde{x}_i$ for a given image x_i. Next, using NCE we contrast their embeddings $\phi(x_i; \theta), \phi(\tilde{x}_i; \theta)$ against randomly sampled negatives $x_k \in \mathcal{N} \subset \mathcal{X}$ by minimizing

$$\mathcal{L}_{\text{NCE}} = \frac{1}{Z} \sum_{x_i \in \mathcal{X}} -\log \frac{\exp(\phi(x_i; \theta)^\top \phi(\tilde{x}_i; \theta)/\tau)}{\sum_{x_k \in \mathcal{N}} \exp(\phi(x_i; \theta)^\top \phi(x_k; \theta)/\tau)} \tag{4.7}$$

where the temperature parameter τ is adjusted during optimization to control the training signal, especially during earlier stages of training. Note, that in NCE formulations typically inner products are used as a distance function, in contrast to the euclidean distance used for general DML formulations. Moreover, (4.7) closely resembles the standard softmax formulation of classification tasks and the formulations underlying proxy-based learning (see Sec. 2.2.1). While the latter contrasts samples against learned class proxies, NCE typically operates fully between actual training samples. By contrasting each sample against many negatives, i.e. large sets $\mathcal{N}$, this task effectively yields a general, class-agnostic feature description of our data. Moreover, as the contrastive objective explicitly increases the similarity of an anchor image with its augmentations, invariance against data transformations and scaling are learned.
Fig. 4.2 summarizes and visually explains the different training objectives of each task.

4.2 Improved generalization by multi feature learning

Following we show how to efficiently incorporate the learning tasks introduced in the previous section into a single DML model. We first extend the objective Eq. 4.7 using established triplet sampling strategies for improved adjustment to DML, before we jointly train our learning tasks for maximal feature diversity.

Adapting noise contrastive estimation to DML. Efficient strategies for mining informative negatives x_k are a key factor [205] for successful training of ranking-based DML models. Since NCE essentially translates to a ranking of image triplets $(x_i, \tilde{x}_i, x_k)$, its learning signal is also impaired if the negatives x_k are uninformative, i.e. $d_\phi(x_i, x_k)$ being large. To this end, we control the contribution of negatives $x_k \in \mathcal{X}$ to $\mathcal{L}_{\text{NCE}}$ by a weight factor $w(d) = \min(\lambda, q^{-1}(d))$. Here, $q(d) = d^{n-2}\left[1 - \frac{1}{4}d^2\right]^{\frac{n-3}{2}}$ is the distribution of pairwise distances on the D-dimensional unit hypersphere and λ a cut-off parameter (For more details, see Sec. 5.1). Similar to [256], $w(d)$ helps to equally weigh negatives from the whole range of possible distances in Φ and, in particular, increases the impact of harder negatives. Thus, our distance-adapted NCE loss becomes

$$\mathcal{L}_{\text{DaNCE}} = \frac{1}{Z} \sum_{x_i \in \mathcal{X}} -\log \frac{\exp(\phi(x_i;\theta)^\top \phi(\tilde{x}_i;\theta)/\tau)}{\sum_{x_k \in \mathcal{N}} \exp(w(d_\phi(x_i, x_k)) \cdot \phi(x_k;\theta)^\top \phi(x_i;\theta)/\tau)} \,. \tag{4.8}$$

NCE-based objectives learn best using large sets of negatives [90]. However, due to the stochastic, batch-wise optimization procedure of deep learning, $\mathcal{N}$ in (4.8) is typically restricted to samples from the current training batch, whose size is resticted by the available GPU memory. To alleviate this limitation, we follow [90] and realize $\mathcal{N}$ as a large memory queue $\mathcal{Q}$ storing embedding representations for all training samples. Since updating the embedding representations $\phi(x_i;\theta)$ for all samples $x_i \in \mathcal{X}$ after each training iteration s is computationally prohibitive, the representations $\phi^{*,s}(x_i) \in \mathcal{Q}$ are gradually updated

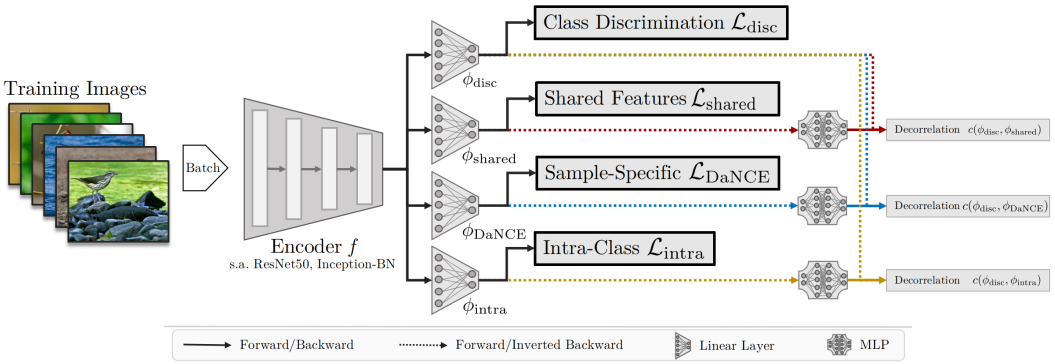


Figure 4.3: *Architecture of our proposed model.* Each task $\mathcal{L}_\bullet$ optimizes an individual embedding $\phi_\bullet$ implemented as a linear layer with a shared underlying feature encoder f. Pairwise decorrelation $c(\cdot,\cdot)$ of the embeddings utilizing the mapping ψ based on a two-layer MLP encourages each task to further emphasize on its targeted data characteristics. Gradient inversion R is applied during the backward pass to each embedding head.

during training by running-averages using the computed representations from a training mini-batch at s. The update follows

$$\phi^{*,s+1}(x_i) = \mu\phi^{*,s}(x_i) + (1-\mu)\phi(x_i;\theta) \tag{4.9}$$

with μ being a fixed, scalar factor weighing the former and newly inferred representation of x_i based on the current model state at iteration s with parameters weights θ. Other embeddings are still represented by older network states. Substituting the actual embedding representation $\phi(x_k;\theta)$ of negative instances x_k in (4.8) with representations sampled from the memory queue $\mathcal{Q}$ transforms (4.8) to

$$\mathcal{L}_{\text{DaNCE}} = \frac{1}{Z}\sum_{x_i\in\mathcal{X}} -\log \frac{\exp(\phi(x_i;\theta)^\top \phi(\tilde{x}_i;\theta)/\tau)}{\sum_{x_k\in\mathcal{Q}} \exp(w(d_\phi(x_i,x_k))\cdot \phi^{*,s}(x_k)^\top \phi(x_i;\theta)/\tau)}\,. \tag{4.10}$$

Consequently, using and maintaining the memory queue constitutes a trade-off between leveraging larger sets of negatives for computing the loss (4.10) and evaluating it using negatives based on the most current inferred representations.

Joint optimization for maximal feature diversity. The tasks presented in Sec. 4.1 are formulated to extract mutually complementary information from our training data. In order to capture their learned features in a single model to obtain a rich image representation, we now discuss how to jointly optimize these tasks by extending the architecture we introduced in Sec. 3.1.3. For completeness and readability, we re-iterate both notation and reasoning behind the (adjusted) joint optimization framework already provided in this section.
While each task targets a semantically different concept of features, their driving learning signals are based on potentially contradicting ranking constraints on the learned embed-

ding. Thus, aggregating these signals to optimizing a joint, single embedding function ϕ may entail detrimental interference between them. In order to circumvent this issue, we learn a dedicated embedding for each task, as often conducted in multi-task optimization [189, 194]. Thus, we introduce ϕ^{disc}, ϕ^{shared}, ϕ^{intra} and ϕ^{DaNCE} with $\phi^{\bullet} : \mathbb{R}^{D_f} \mapsto \Phi^{\bullet} \subset \mathbb{R}^{D}$ with $\phi^{\bullet}_i = \phi^{\bullet}(f_i; \theta_{\bullet}) = \phi^{\bullet}(f(x_i; \omega); \theta_{\bullet})$. Here, $\bullet$ indicates a particular task (i,e. disc, shared, intra, DaNCE) and consequently $\theta_{\bullet}$ the corresponding trainable parameters of the embedding representations.
As all embeddings share the same feature extractor f, each task still benefits from the aggregated learning signals. Additionally, as there may still be redundant overlap in the information captured by each task, we mutually decorrelate these representations, thus maximizing the diversity of the overall training signal. Again, we minimize the mutual information of two embedding functions ϕ^a, ϕ^b (a, b represent two of our introduced tasks) by maximizing their correlation c, followed by a gradient reversal. For that, we learn a mapping $\psi : \mathbb{R}^{D} \mapsto \mathbb{R}^{D}$ from ϕ^a_i to ϕ^b_i given an image x_i and compute the correlation $c(\phi^a_i, \phi^b_i) = \|(R(\phi^a_i) \odot \psi(R(\phi^b_i)))\|^2_2$ with $\odot$ being the point-wise product. Note, that $\phi^{\bullet}$ is regularized to the unit-hypersphere and thus trivial solutions are prevented. R denotes a gradient reversal operation, which inverts the resulting gradients during back-propagation. Maximizing c results in ψ aiming to make ϕ^a_i and ϕ^b_i comparable. However, through the subsequent gradients reversal, we actually decorrelate the embedding functions. Joint training of all tasks is finally performed by minimizing

$$\mathcal{L} = \mathcal{L}_{\text{disc}} + \alpha_1 \mathcal{L}_{\text{shared}} + \alpha_2 \mathcal{L}_{\text{intra}} + \alpha_3 \mathcal{L}_{\text{DaNCE}} - \sum_{(\phi_a, \phi_b) \in \mathcal{P}} \rho_{a,b} \cdot c(\phi_a, \phi_b) \tag{4.11}$$

where $\mathcal{P}$ denotes the pairs of embeddings to be decorrelated. We found

$$\mathcal{P} = \{(\phi_{\text{disc}}, \phi_{\text{DaNCE}}), (\phi_{\text{disc}}, \phi_{\text{shared}}), (\phi_{\text{disc}}, \phi_{\text{intra}})\} \,, \tag{4.12}$$

to work best, which decorrelates each of our introduced novel tasks with the standard class-discriminative task. Our experiments showed that further decorrelation among our learning tasks does not result in additional benefits. The weighting parameters $\rho_{a,b}$ adjusting the degree of decorrelation between the embeddings are set to the same, constant value in our implementation. Fig. 4.3 provides an overview of our model architecture. Finally, to also combine our learned embedding representations we concatenate the individual task embeddings, thus forming an ensemble representation.

Computational Costs. We train all tasks using the same mini-batch to avoid computational overhead. While optimizing each learner on an individual batch can further alleviate training signal interference [201, 261], training time increases significantly. Across datasets, we measure an increase in training time by $10 - 15\%$ per epoch compared to class-discriminative training only. This is comparable to or lower than other recently proposed DML methods, which perform a full clustering on the dataset [194, 201] after each epoch, compute extensive embedding statistics [105] or simultaneously train generative models [141].

Backbone →	Inception-V1 + BN			ResNet50		
Parameter ↓	CUB200	CARS196	SOP	CUB200	CARS196	SOP
$\rho_{a,b}, \alpha_i$	300, 0.1	100, 0.1	50, 0.1	1500, 0.3	100, 0.1	300, 0.2
τ	0.01	0.01	0.01	0.01	0.01	0.01
ϵ, ν	70, 0.3	160, 0.3	100, 0.3	60, 0.3	70, 0.3	70, 0.3

Table 4.1: *Hyperparameters*. The parameter ϵ denotes the epoch at which the learning rate is annealed by ν. Decorrelation weights $\rho_{a,b}$ and weightings α_i (eq. 4.11) are the same for all pairs in $\mathcal{P}$ (see eq. 4.12). τ denotes the temperature in $\mathcal{L}_{\text{DaNCE}}$ (eq. 4.10).

4.3 Experiments

Following we first present our implementation details and the benchmark datasets. Next, we evaluate our proposed model and study how our learning tasks complement each other and improve over baseline performances. Finally, we discuss our results in the context of the current state-of-the-art and conduct analysis and ablation experiments.

Implementation details. We follow the common training protocol of [194, 201, 256] for implementations utilizing a ResNet50-backbone. The shorter image axis is resized to 256, followed by a random crop to 224×224 and a random horizontal flip with $p = 0.5$. During evaluation, only a center crop is taken after resizing. The embedding dimension is set to $D = 128$ for each task embedding. For model variants using the Inception-V1 with Batch-Normalization[103], we follow [105, 251] and use $D = 512$. Resizing, cropping and flipping is done in the same way as for ResNet50 versions. For training, we use Adam[122] with learning rate 10^{-5} and a weight decay of $5 \cdot 10^{-4}$. For ablations, we use no learning rate scheduling, while our final model is trained using scheduling values determined by cross-validation. The implementation is done using the PyTorch framework[180], and experiments are performed on compute clusters containing NVIDIA Titan X, Tesla V4, P100 and V100, always limited to 12GB VRAM following the standard training protocol [256]. For DiVA, we utilise the triplet-based margin loss [256] with fixed margin $\gamma = 0.2$ and $\beta = 1.2$. The utilized hyperparameters for DiVA are listed in Tab. 4.1. Training is run for 200 epochs on CUB200/CARS196 and 150 epochs on SOP.

Datasets. We evaluate the performance on three common benchmark datasets with standard training/test splits (see e.g. [194, 201, 251, 256]): *CARS196*[130], which contains 16,185 images from 196 car classes. The first 98 classes containing 8054 images are used for training, while the remaining 98 classes with 8131 images are used for testing. *CUB200-2011*[239] with 11,788 bird images from 200 classes. Training/test sets contain the first/last 100 classes with 5864/5924 images respectively. *Stanford Online Products (SOP)*[171] provides 120,053 images divided in 22,634 product classes. 11318 classes with 59551 images

Dataset →		CUB200-2011[239]			CARS196[130]			SOP[171]		
Approach ↓	Dim	R@1	R@2	NMI	R@1	R@2	NMI	R@1	R@10	NMI
Margin[256] (orig, R50)	128	63.6	74.4	69.0	79.6	86.5	69.1	72.7	86.2	90.7
Margin[256] (ours, IBN)	512	63.6	74.7	68.3	79.4	86.6	66.2	76.6	89.2	89.8
DiVA (IBN, D & Da)	512	64.5	76.0	68.8	80.4	87.7	67.2	77.0	89.4	**90.1**
DiVA (IBN, D & S)	512	65.1	76.4	69.0	81.5	88.3	66.8	77.2	89.6	90.0
DiVA (IBN, D & I)	512	64.9	75.8	68.4	80.6	87.9	67.4	76.9	89.4	89.9
DiVA (IBN, D & Da & I)	510	65.3	76.5	68.3	82.2	89.1	67.8	75.8	89.0	89.8
DiVA (IBN, D & S & I)	510	65.5	76.4	68.4	82.1	89.4	67.2	77.0	89.3	89.7
DiVA (IBN, D & Da & S)	510	65.9	76.7	68.9	82.6	89.6	68.0	77.4	89.6	90.1
DiVA (IBN, D & Da & S & I)	512	**66.4**	**77.2**	**69.6**	**83.1**	**90.0**	**68.1**	**77.5**	**90.3**	**90.1**

Table 4.2: *Comparison of our proposed method using different combinations of learning tasks.* IBN (Inception-V1 with Batch-Normalization), and R50(ResNet50) denote the backbone architecture. No learning rate scheduling is used. Our tasks are denoted by D(*iscriminative*), S(*hared*), I(*ntra-Class*) & and Da(*NCE*). For fair comparison, the dimensionality per task embedding depends on the number of tasks incorporated to ensure a total of $D = 512$. Two tasks therefore each use $D = 256$, three use $D = 170$ and when four tasks are combined, each use $D = 128$.

are used for training, while the remaining 11316 classes with 60502 images are used for testing.

4.3.1 Performance study of multi-feature DML

We now compare our model and the complementary benefit of our proposed feature learning tasks for supervised DML. Tab. 5.1 evaluates the performance of our model based on margin loss [256], a triplet based objective with an additionally learnable margin, and distance-weighted triplet sampling [256]. We use Inception-V1 with Batchnorm and a maximal aggregated embedding dimensionality of 512. Thus, if two tasks are utilized, each embedding has $D = 256$, in case of three tasks 170 and four tasks result in $D = 128$. No learning rate scheduling is used. Evaluation is conducted on CUB200-2011 [239], CARS196 [130] and SOP [171]. Retrieval performance is measured through Recall@k[107] and clustering quality via Normalized Mutual Information (NMI) [151]. While our results vary between possible task combinations, we observe that the generalization of our model consistently increases with each task added to the joint optimization. Our strongest model including all proposed tasks improves the generalization performance by 2.8% on CUB200-2011, 3.7% on CARS196 and 0.9% on SOP. This highlights that *(i)* purely discriminative supervised learning disregards valuable training information and *(ii)* carefully designed learning tasks are able to capture this information for improved generalization to unknown test classes. We further analyze our observations in the ablation experiments.

Dataset →		CUB200-2011[239]			CARS196[130]			SOP[171]		
Approach ↓	Dim	R@1	R@2	NMI	R@1	R@2	NMI	R@1	R@2	NMI
HTG[275]	512	59.5	71.8	-	76.5	84.7	-	-	-	-
HDML[277]	512	53.7	65.7	62.6	79.1	87.1	69.7	68.7	83.2	89.3
Margin[256]	128	63.6	74.4	69.0	79.6	86.5	69.1	72.7	86.2	90.8
HTL[71]	512	57.1	68.8	-	81.4	88.0	-	74.8	88.3	-
DVML[141]	512	52.7	65.1	61.4	82.0	88.4	67.6	70.2	85.2	90.8
MultiSim[251]	512	65.7	77.0	-	84.1	90.4	-	78.2	90.5	-
D&C[201]	128	65.9	76.6	69.6	84.6	90.7	70.3	75.9	88.4	90.2
MIC[194]	128	66.1	76.8	69.7	82.6	89.1	68.4	77.2	89.4	90.0
Significant increase in network parameter:										
HORDE[105]+Contr.[86]	512	66.3	76.7	-	83.9	90.3	-	-	-	-
Softtriple[183]	512	65.4	76.4	-	84.5	90.7	70.1	78.3	90.3	**92.0**
Ensemble Methods:										
A-BIER[175]	512	57.5	68.7	-	82.0	89.0	-	74.2	86.9	-
Rank[250]	1536	61.3	72.7	66.1	82.1	89.3	71.8	**79.8**	**91.3**	90.4
DREML[261]	9216	63.9	75.0	67.8	86.0	91.7	**76.4**	-	-	-
ABE[120]	512	60.6	71.5	-	85.2	90.5	-	76.3	88.4	-
Inception-BN										
Ours (DiVA-IBN-512)	512	66.8	77.7	70.0	84.1	90.7	68.7	78.1	90.6	90.4
ResNet50										
Ours (Margin[256]-R50-512)	512	64.4	75.4	68.4	82.2	89.0	68.1	78.3	90.0	90.1
Ours (DiVA-R50-512)	512	**69.2**	**79.3**	**71.4**	**87.6**	**92.9**	72.2	**79.6**	**91.2**	90.6

Table 4.3: *Comparison to the state-of-the-art methods on CUB200-2011*[239], *CARS196*[130] *and SOP*[171]. DiVA-Arch-Dim describes the backbone used with DiVA (IBN: Inception-V1 with Batchnorm, R50: ResNet50) and the total training and testing embedding dimensionality. For fair comparison, we also ran a standard ResNet50 with embedding dimensionality of 512. As can be seen, DiVA significantly outperforms other methods on CUB200 and CARS196 while achieving competitive performance on SOP. Even with the weaker IBN-backbone we reach state-of-the-art on CUB200 and comparable results on CARS196 and SOP.

4.3.2 Comparison to state-of-the-art approaches

Next, we compare our strongest model trained with the same hyperparameters and a fixed learning rate scheduling per benchmark (Tab. 4.1) to the current state-of-the-art approaches in DML. For fair comparison to the different methods, we report result both using Inception-BN (IBN) and ResNet50 (R50) as backbone architecture. As Inception-BN is typically trained with embedding dimensionality of 512, we restrict each embedding to $D = 128$ for direct comparison with non-ensemble methods. Thus we deliberately impair the potential of our model due to a significantly lower capacity per task, compared to the standard $D = 512$. For comparison with ensemble approaches and maximal performance, we use a ResNet50 [194, 201, 256] architecture and the corresponding standard

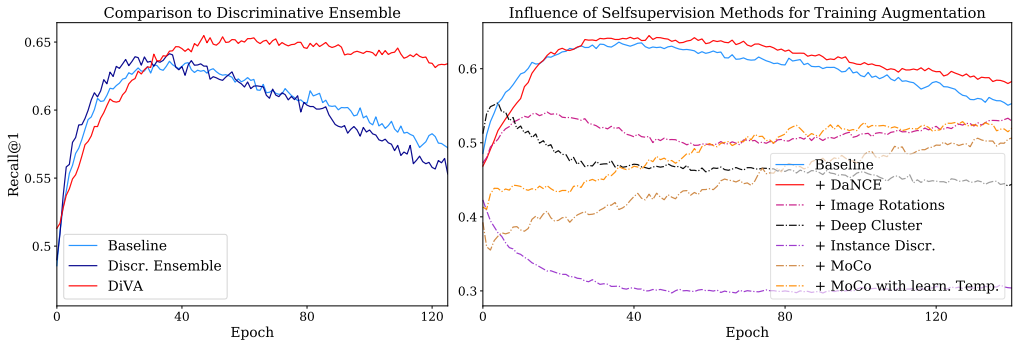


Figure 4.4: *Analysis of complementary tasks for supervised learning.* (left): Performance comparison between class-dicsriminative training only (Baseline), ensemble of class-discriminative learners (Discr. Ensemble) and our proposed DiVA, which exhibits a large boost in performance. (right): Evaluation of self-supervised learning approaches combined with standard discriminative DML.

dimensionality $D = 128$ per task. Fig. 5.2 summarizes our results. We significantly improve over methods with comparable backbone architectures and achieve new state-of-the-art results with our ResNet50-ensemble. In particular we outperform the strongest ensemble methods, including DREML [261] which utilize a much higher total embedding dimensionality. The large improvement is explained by the diverse and mutually complementary learning signals contributed by each task in our ensemble. In contrast, previous ensemble methods rely on the same, purely class-discriminative training signal for each learner. Note that some approaches strongly differ from the standard training protocols and architectures, resulting in more parameters and much higher GPU memory consumption, such as Rank [250] (32GB), ABE [120] (24GB), Softtriple [183] and HORDE [105]. Additionally, Rank [250] employs much larger batch-sizes to increase the number of classes per batch. This is especially crucial on the SOP dataset, which greatly benefits from higher class coverage due to its vast amount of classes, as shown by [196]. Nevertheless, our model outperforms these methods - in some cases even in its constrained version (IBN-512).

4.3.3 Ablation studies

In this section we conduct ablation experiments for various parts of our model. For every ablation we again use the Inception-BN network. The dimensionality setting follows the performance study in sec. 4.3.1. Again, we train each model with a fixed learning rate for fair comparison among ablations.

Influence of distance-adaption in DaNCE. To evaluate the benefit of our extension from $\mathcal{L}_{\text{nce}}$ [84, 90] to $\mathcal{L}_{\text{DaNCE}}$, we compare both versions in combination with standard supervised DML (i.e. class-discriminative features) in Fig. 4.4 (right). Our experiment indicates two positive effects: *(i)* The training convergence with our extended objective is much faster and *(ii)* the performance differs greatly between employing $\mathcal{L}_{\text{nce}}$ and

Methods →	Baseline	DiVA	No De-correlation	Separated models
Recall@1 →	63.6	66.4	65.6	48.7

Table 4.4: *Ablation studies*. We compare standard margin loss as baseline and DiVA performance against ablations of our model: no decorrelation between embeddings (No-Decorrelation.) and training an independent model for each task (Separated models). Total embedding dimensionality is 512.

$\mathcal{L}_{\text{DaNCE}}$. In fact, using the standard NCE objective is even detrimental to learning, while our extended version improves over the only discriminatively trained baseline. We attribute this to both the slow convergence of $\mathcal{L}_{\text{nce}}$ which is not able to support the faster discriminative learning and to emphasizing harder negatives in $\mathcal{L}_{\text{DaNCE}}$. In particular the latter is an important factor in ranking based DML [205], as during training more and more negatives become uninformative. To tackle this issue, we also experimented with learning the temperature parameter τ. While convergence speed increases slightly, we find no significant benefit in final generalization performance.

Evaluation of self-supervision methods. Fig. 4.4 compares DaNCE to other methods from self-supervised representation learning. For that purpose we train the discriminative task with either DeepCluster [30], RotNet [72] or Instance Discrimination [258]. We observe that neither of these tasks is able to provide complementary information to improve generalization. DeepCluster, trained with 300 pseudo classes for classification, actually aims at approximating the class-discriminative learning signal while RotNet is strongly dependent on the variance of the training classes and converges very slowly. Instance discrimination seems to provide a contradictory training signal to the supervised task. These results are in line with previous works [93] which report difficulties to directly combine both supervised and self-supervised learning for improved test performance. In contrast, we explicitly adapt NCE to DML in our proposed objective DaNCE.

Comparison to purely class-discriminative ensemble. We now compare DiVA to an ensemble of class-discriminative learner (Discr. Ensemble) based on the same model architecture using embedding decorrelation in Fig. 4.4. While the discriminative ensemble improves over the baseline, the amount of captured data information eventually saturates and, thus, performs significantly worse compared to our multi-feature DiVA ensemble. Further, our ablation reveals that joint optimization of diverse learning tasks regularizes training and reduces overfitting effects which eventually occur during later stages of DML training.

Benefit of task decorrelation. The role of decorrelating the embedding representations of each task during learning is analyzed by comparison to a model trained without this constraint. Firstly, Tab. 4.4 demonstrates that omitting the decorrelation still outperforms the standard margin loss ('Baseline') by 2.1% while operating on the same total embedding dimensionality. This proves that learning diverse features significantly

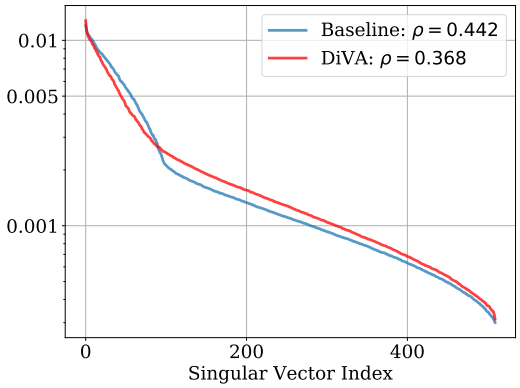


Figure 4.5: *Singular Value Spectrum.* We analyze the singular value spectrum of DiVA embeddings and that of a network trained with the standard discriminative task. Consistent with [196] we find that our improvements in generalization performance (as shown in Tab. 5.2 and Tab. 5.1) are reflected by a reduced spectral decay.

improves generalization. Adding the de-corralation constraint then additionally boosts performance by 1.2%, as now each task is further encouraged to capture distinct data characteristics.

Learning without feature sharing. To highlight the importance of feature sharing among our learning tasks, we train an individual, independent model for the class-discriminative, class-shared, sample-specific and intra-class task. At testing time, we combine their embeddings similar to our proposed model. Tab. 4.4 shows a dramatic drop in performance to 48% for the disconnected ensemble ('Separately Trained'), proving that sharing the complementary information captured from different data characteristics is a crucial element of our model and mutually benefits learning. Without the class-discriminative signal, the other tasks lack the concept of an object class, which hurts the aggregation of embeddings. In addition, the supervised task again suffers from strong overfitting to the training data. However, sharing the learned information aggregates their training signal and results in an overall improvement in generalization.

Generalization and representation compression. In chapter 2 we link improved DML generalization to a decreased compression [228] of the representation. Our findings suggests that the number of directions with significant variance [196, 233] of a representation correlates with the generalization ability in DML. To this end, we analyze our model using the spectral decay ρ (lower is better) which is computed as the KL-divergence between the normalized singular value spectrum and a uniform distribution. Fig. 4.5 compares the spectral decays of our model and a standard supervised baseline model. As expected, due to the diverse information captured, our model learns a more complex representation which results in a significantly lower value of ρ and better generalization.

4.4 Discussion

In this chapter we proposed several learning tasks which complement the class-discriminative training signal of standard, supervised Deep Metric Learning (DML) for improved generalization to unknown test distributions. Each of our tasks is designed to capture different characteristics of the training data: class-discriminative, class-shared, intra-class and sample-specific features. For the latter, we adapted contrastive self-supervised learning to the needs of supervised DML. Jointly optimizing all tasks results in a diverse overall training signal which is further amplified by mutual decorrelation between the individual tasks. Unifying these distinct representations greatly boosts generalization over purely discriminatively trained models. Our experiments and ablation studies show significantly boosted generalization performance, improving over existing state-of-the-art DML approaches.

5 Adaptive Triplet Sampling using Reinforcement Learning

Similarity learning typically learns a representation which maps similar images close together and dissimilar images far apart. This task is naturally formulated by ranking tasks, in which individual pairs of images are compared[86, 171, 217] or contrasted against a third image[205, 245, 256]. These triplet-based objective functions constitutes the basis of most Deep Metric Learning (DML) algorithms[156, 175, 194, 201, 261]. However, with growing training set size, leveraging every single triplet for learning becomes computationally infeasible, limiting training to only a subset of all possible triplets. Thus, a careful selection of those triplets which drive learning best is crucial. This raises the question: How to determine *which* triplets to present *when* to our model during training?

As training progresses, more and more triplet relations will be correctly represented by the model. Thus, ever fewer triplets will still provide novel, valuable information. Conversely, leveraging only triplets which are hard to learn[58, 205, 275] but therefore informative, impairs optimization due to high gradient variance[256]. Consequently, a reasonable mixture of triplets with varying difficulty would provide an informative and stable training signal. Now, the question remains, *when* to present which triplet? Sampling from a fixed distribution over difficulties may serve as a simple proxy[256] and is a typical remedy in representation learning in general[21, 123]. However, *(i)* choosing a proper distribution is difficult; *(ii)* the abilities and state of our model evolves as training progresses and, thus, a fixed distribution cannot optimally support every stage of training; and *(iii)* triplet sampling should actively contribute to the learning objective rather than being chosen independently. Since a manually predefined sampling distribution does not fulfill these requirements, we need to *learn* and adapt it while training a representation.

Such online adaptation of the learning algorithm and the parameters that control it during training is typically framed as a teacher-student setup and optimized using Reinforcement Learning (RL). When modeling a flexible sampling process (the student), a controller network (the teacher) learns to adjusts the sampling such that the DML model is steadily provided with an optimal training signal.

Subsequently, we present how to learn a novel triplet sampling strategy which is able to effectively support the learning process of a DML model at every stage of training. To this end, we model a sampling distribution so it is easily adjustable to yield triplets of arbitrary mixtures of difficulty. To adapt to the training state of the DML model, we employ Reinforcement Learning to update the adjustment policy. Directly optimizing the policy so it improves performance on a held-back validation set, adjusts the sampling process to optimally support DML training. Experiments show that our adaptive sampling strategy

significantly improves over fixed, manually designed triplet sampling strategies on multiple datasets. Moreover, we perform diverse analyses and ablations to provide additional insights into our method.
This chapter is based on our publication *'PADS: Policy-Adapted Sampling for Visual Similarity Learning'* [195].

5.1 Preliminaries: Static Triplet Sampling Strategies in DML

While triplet-based ranking losses have proven to be powerful, the number of possible triplets $(x_i, x_j, x_k) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X}$ grows dramatically with the size of the training set. Thus, training quickly becomes infeasible, turning efficient triplet sampling strategies into a key component for successful learning as discussed here. The notation in this chapter follows the notation introduced in chapter 2.
When performing DML using triplet-based ranking losses, distances $d_{ik} \triangleq d_\phi(x_i, x_k) = \|\phi(x_i) - \phi(x_k)\|_2$. between an anchor and negative samples, respectively d_{ij} between the anchor and positive samples decreasingly violate a given, fixed triplet margin $\beta \in \mathbb{R}_{>0}$ (i.e. $d_{ik} - d_{ij} < \beta$) as training progresses. Naively employing random triplet sampling entails many of the selected triplets being uninformative, as distances in Φ are strongly biased towards larger distances d due to its regularization to the unit-hypersphere $\mathbb{S}^D$. Consequently, recent sampling strategies explicitly leverage triplets which violate the triplet margin and, thus, are difficult and informative.

(Semi-)Hard negative sampling. Hard negative sampling methods focus on triplets violating the margin β the most, i.e. by sampling negatives

$$x_k^* = \underset{x_k \in \mathcal{X}: d_{ik} < d_{ij}}{\arg\min} \; d_{ik} \; . \tag{5.1}$$

While it speeds up convergence, it may result in collapsed models[205] due to a strong focus on few data outliers and very hard negatives. Facenet[205] proposes a relaxed, semi-hard negative sampling strategy restricting the sampling set to a single mini-batch $\mathcal{B}$ and to those which are closest to the margin β without violating it. Hence, they employ considerably less hard negatives samples

$$x_k^* = \underset{x_k \in \mathcal{B}: d_{ik} > d_{ij}}{\arg\min} \; d_{ik} \tag{5.2}$$

for learning. Based on this idea, different online[179, 217] and offline[88] strategies emerged.

(Static) Distance-based sampling. By considering the hardness of a negative, one can successfully discard easy and uninformative triplets. However, triplets that are too hard lead to noisy learning signals due to overall high gradient variance [256]. As a remedy, to control the variance while maintaining sufficient triplet utility, sampling can be extended to also consider easier negatives, i.e. introducing a sampling distribution $x_k \sim p(x_k|x_i)$

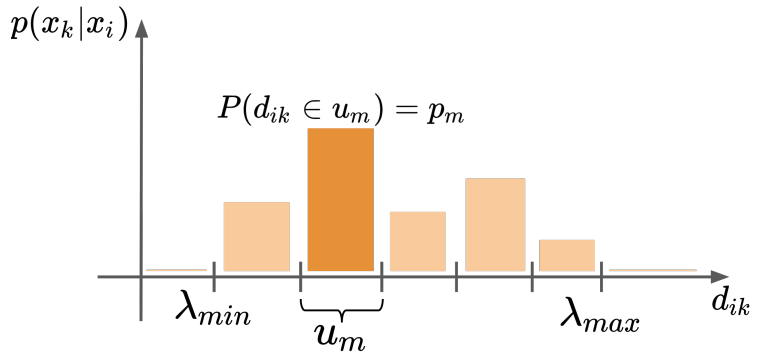


Figure 5.1: *Sampling distribution $p(x_k|x_i)$ for selecting negatives x_k given an anchor x_i.* We discretize the distance interval $U = [\lambda_{\text{min}}, \lambda_{\text{max}}]$ into M equi-sized bins u_m with individual sampling probabilities p_m.

over the range of distances d_{ik} between anchor and negatives. Wu et al. [256] propose to sample from a static uniform prior on the range of d_{ik}, thus equally considering negatives from the whole spectrum of difficulties. As pairwise distances in Φ are strongly biased towards larger d_{ik}, their sampling distribution requires to weigh $p(x_k|x_i)$ inversely to the analytical distance distribution $q(d)$ on Φ, i.e.

$$q(d) \propto d^{D-2}\left[1 - \frac{1}{4}d^2\right]^{\frac{D-3}{2}}, \tag{5.3}$$

for large $D \geq 128$ [225]. Distance-based sampling from the static, uniform prior is then performed by

$$x_k \sim p(x_k|x_i) \propto \min\left(\lambda, q^{-1}(d_{ik})\right) \tag{5.4}$$

with λ being a clipping hyperparameter for regularization.

5.2 PADS: Learning an Adaptive Sampling Strategy

Distance-based sampling of negatives x_k has proven to offer a good trade-off between fast convergence and a stable, informative training signal. However, a static sampling distribution $p(x_k|x_i)$ provides a stream of training data independent of the the changing needs of a DML model during learning. While samples of mixed difficulty may be useful at the beginning, later training stages are calling for samples of increased difficulty, as e.g. analyzed by curriculum learning[15]. Unfortunately, as different models and even different model intializations[76] exhibit distinct learning dynamics, finding a generally applicable learning schedule is challenging. Thus, again, heuristics[85] are typically employed, inferring changes after a fixed number of training epochs or iterations. To provide an optimal training signal, however, we rather want $p(x_k|x_i)$ to adapt to the training state of the DML model than merely the training iteration. Such an adaptive negative sampling allows for

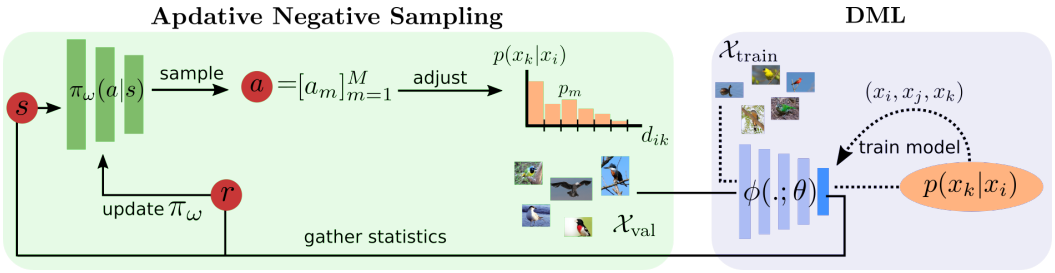


Figure 5.2: *Overview of approach.* Blue denotes the standard Deep Metric Learning (DML) setup using triplets (x_i, x_j, x_k). Our proposed adaptive negative sampling is shown in green: (1) We compute the current training state s using $\mathcal{X}_{val}$. (2) Conditioned on s, our policy $\pi_\omega(a|s)$ predicts adjustments to p_m. (3) We perform bin-wise adjustments of $p(x_k|x_i)$. (4) Using the adjusted $p(x_k|x_i)$ we train the DML model. (5) Finally, π_ω is updated based on the reward r.

adjustments which directly facilitate maximal DML performance. Since manually designing such a strategy is difficult, *learning* it is the most viable option.
Subsequently, we first present how to find a parametrization of $p(x_k|x_i)$ that is able to represent arbitrary, potentially multi-modal distributions, thus being able to sample negatives x_k of any mixture of difficulty needed. Using this, we can learn a policy which effectively alters $p(x_k|x_i)$ to optimally support learning of the DML model. We refer to this strategy as *PADS*: Policy-adapted Sampling.

5.2.1 Modelling a flexible sampling distribution

Since learning benefits from a diverse distribution $p(x_k|x_i)$ of negatives, uni-modal distributions (e.g. Gaussians, Binomials, χ^2) are insufficient. Thus, we utilize a discrete probability mass function

$$p(x_k|x_i) \triangleq P(d_{ik} \in u_m) = p_m \ , \tag{5.5}$$

where the bounded interval $U = [\lambda_{\text{min}}, \lambda_{\text{max}}]$ of possible distances d_{ik} is discretized into M disjoint equidistant bins $u_1, \ldots, u_M$,

$$u_m = [\frac{m-1}{M}, \frac{m}{M}] \ . \tag{5.6}$$

The probability of drawing x_k from bin u_m is p_m with $p_m \geq 0$ and $\sum_m p_m = 1$. Fig. 5.1 illustrates this discretized sampling distribution.
This representation of the negative sampling distribution effectively controls *which* samples are used to learn ϕ. As ϕ changes during learning, $p(x_k|x_i)$ should also adapt to always provide the most useful training samples, i.e. to control *when* to use *which* sample. Hence the probabilities p_m need to be updated while learning ϕ. We subsequently solve this task

by learning a *stochastic adjustment policy* π_ω for the p_m, implemented as a neural network parametrized by ω.

5.2.2 Learning an adjustment policy for $p(x_k|x_i)$

Our sampling process based on $p(x_k|x_i)$ should provide optimal training signals for learning ϕ at every stage of training. Thus, we adjust the p_m by a multiplicative update $a \in \mathcal{A}$ conditioned on the current representation (or state) $s \in \mathcal{S}$ of ϕ during learning. We introduce a conditional distribution $\pi_\omega(a|s)$ to control which adjustment to apply at which state s of training ϕ. To learn π_ω, we measure the utility of these adjustments for learning ϕ using a reward signal $r = r(s,a)$. We now first describe how to model each of these components, before presenting how to efficiently optimize the adjustment policy π_ω alongside ϕ.

Adjustments a. To adjust $p(x_k|x_i)$, $\pi_\omega(a|s)$ proposes adjustments a to the p_m. To lower the complexity of the action space, we use a limited set of actions $\mathcal{A} = \{\alpha, 1, \beta\}$ to individually decrease, maintain, or increase the probabilities p_m for each bin u_m, i.e.

$$a \triangleq [a_m \in \{\alpha, 1, \beta\}]_{m=1}^{M} \,. \tag{5.7}$$

Further, α, β are fixed constants $0 < \alpha < 1, \beta > 1$ and $\frac{\alpha+\beta}{2} = 1$. Updating $p(x_k|x_i)$ is then performed by bin-wise updates $p_m \leftarrow p_m \cdot a_m$ followed by re-normalization. Using a multiplicative adjustment accounts for the exponential distribution of distances on Φ (cf. Sec. 5.1).

Training states s. Adjustments a depend on the present state $s \in \mathcal{S}$ of the representation ϕ. Unfortunately, we cannot use the current model weights θ of the embedding network, as the dimensionality of s would be to high, thus making optimization of π_ω infeasible. Instead, we represent the current training state using representative statistics describing the learning progress: running averages over Recall@1[107], NMI[151] and average distances between and within classes on a fixed held-back validation set $\mathcal{X}_{\text{val}}$. Additionally we use past parametrizations of $p(x_k|x_i)$ and the relative training iteration (cf. Implementation details, Sec. 5.4).

Rewards r. An optimal sampling distribution $p(x_k|x_i)$ yields triplets whose training signal consistently improves the evaluation performance of ϕ while learning. Thus, we compute the reward r for for adjustments $a \sim \pi_\omega(a|s)$ by directly measuring the relative improvement of $\phi(\cdot;\theta)$ over $\phi(\cdot;\theta')$ from the previous training state. This improvement is quantified by DML evaluation performance $e(\phi(.;\theta), \mathcal{X}_{\text{val}})$, i.e. evaluation the embedding ϕ with parameters θ on the evaluation set $\mathcal{X}_{\text{val}}$ using the evaluation metric e. More precisely, we define r as

$$r = \text{sign}\big(e(\phi(.;\theta), \mathcal{X}_{\text{val}}) - e(\phi(.;\theta'), \mathcal{X}_{\text{val}}))\big) \tag{5.8}$$

where θ was reached from θ' after H DML training iterations using $p(x_k|x_i)$. We choose e to be the sum of Recall@1[107] and NMI[151]. Both metrics are in the range $[0, 1]$ and target slightly different performance aspects. Further, similar to [102], we utilize the sign function for consistent learning signals even during saturated training stages.

Learning of π_ω. Adjusting $p(x_k|x_i)$ is a stochastic process controlled by actions a sampled from $\pi_\omega(a|s)$ based on a current state s. This defines a Markov Decision Process (MDP) [221] which is naturally optimized by Reinforcement Learning. For a detailed introduction to Reinforcement Learning, we refer the reader to Sutton et al. [221]. The policy objective $J(\omega)$ is formulated to maximize the total expected reward

$$R(\tau) = \sum_l r_l(a_l, s_l) \tag{5.9}$$

over training episodes of tuples $\tau = \{(a_l, s_l, r_l)|l = 0, \ldots, L]\}$ collected from sequences of L time-steps, i.e.

$$J(\omega) = \mathbb{E}_{\tau\sim\pi_\omega(\tau)}[R(\tau)] \tag{5.10}$$

Hence, π_ω is optimized to predict adjustments a for $p(x_k|x_i)$ which yield high rewards and thereby improving the performance of ϕ. Common approaches use episodes τ comprising long state trajectories which potentially cover multiple training epochs[62]. As a result, there is a large temporal discrepancy between model and policy updates. However, in order to closely adapt $p(x_k|x_i)$ to the learning of ϕ, this discrepancy needs to be minimized. In fact, our experiments show that single-step episodes, i.e. $L = 1$, are sufficient for optimizing π_ω to infer meaningful adjustments a for $p(x_k|x_i)$. Such a setup is also successfully adopted by contextual bandits [138] [1]. In summary, our training episodes τ consists of updating $p(x_k|x_i)$ using a sampled adjustment a, performing H DML training iterations based on the adjusted $p(x_k|x_i)$ and updating π_ω using the resulting reward r. Optimizing Eq. 5.10 is then performed by standard RL algorithms which approximate different variations of the policy gradient [221] based on the gain $G(s, a)$, i.e.

$$\nabla_\omega J(\omega) = \mathbb{E}_{\tau\sim\pi_\omega(\tau)}[\nabla_\omega \log \pi_\omega(a|s)G(s, a)] \; . \tag{5.11}$$

The choice of the exact form of $G = G(s, a)$ gives rise to different optimization methods, e.g REINFORCE [255] ($G = R(\tau)$), Advantage Actor Critic (A2C) [221] ($G = A(s, a)$), etc. Other RL algorithms, such as TRPO [206] or PPO [207] replace Eq. (5.10) by surrogate objective functions. Fig. 5.2 provides an overview over the learning procedure. Moreover, in the supplementary material we compare different RL algorithms and summarizes the learning procedure in Alg. 1 using PPO [207] for policy optimization.

[1] Opposed to bandits, in our RL setup, actions which are sampled from π_ω influence *future* training states of the learner. Thus, the policy implicitly learns state-transition dynamics.

Initialization of $p(x_k|x_i)$. We find that an initialization with a slight emphasis towards smaller distances d_{ik} works best. However, as shown in Tab. 5.7, also other initializations work well. In addition, the limits of the distance interval $U = [\lambda_{\text{min}}, \lambda_{\text{max}}]$ can be controlled for additional regularization as done in [256]. This means ignoring values above λ_{max} and clipping values below λ_{min}, which is analyzed in Tab. 5.5.
Self-Regularization: As noted in [194], the utilization of intra-class features can be beneficial to generalization. Our approach easily allows for a learnable inclusion of such features. As positive samples are generally closest to anchors, we can merge positive samples into the set of negative samples and have the policy learn to place higher sampling probability on such low-distance cases. We find that this additionally improves generalization performance.

Computational costs. Computational overhead over fixed sampling strategies[205, 256] comes from the estimation of r requiring a forward pass over $\mathcal{X}_{\text{val}}$ and the computation of the evaluation metrics. For example, setting $M = 30$ increases the computation time per epoch by less than 20%.

5.3 Related Works

Metric learning has become the leading paradigm for learning distances between images with a broad range of applications, including image retrieval[141, 161, 256], image classification [65, 276], face verification[99, 143, 205] or human pose analysis[46, 154]. Ranking losses formulated on pairs[86, 217], triplets[71, 205, 245, 256] or even higher order tuples of images[37, 171, 250] emerged as the most widely used basis for DML [196]. As with the advent of CNNs datasets are growing larger, different strategies are developed to cope with the increasing complexity of the learning problem.

Complexity management in DML. The main line of research are negative sampling strategies [88, 205, 256] based on distances between an anchor and a negative image. FaceNet [205] leverages only the hard negatives in a mini-batch. Wu et al. [256] sample negatives uniformly over the whole range of distances to avoid large variances in the gradients while optimization. Harwood et al. [88] restrict and control the search space for triplets using pre-computed sets of nearest neighbors by linearly regressing the training loss. Each of them successfully enable effective DML training. However, these works are based on fixed and manually predefined sampling strategies. In contrast, we learn an adaptive sampling strategy to provide an optimal input stream of triplets conditioned on the training state of our model.
Orthogonal to sampling negatives from the training set is the generation of hard negatives in form of images [58] or feature vectors [275, 277]. Thus, these approaches also resort to hard negatives, while our sampling process yields negatives of any mixture of difficulty depending on the model state.
Finally, proxy based techniques reduce the complexity of the learning problem by learning one [161] or more [183] virtual representatives for each class, which are used as negatives.

Dataset		CUB200-2011[239]			CARS196[130]			SOP[171]			
Approach	Dim	R@1	R@4	NMI	R@1	R@4	NMI	R@1	R@10	R@100	NMI
Margin[256] + $\mathcal{U}$-dist (orig)	128	63.6	83.1	69.0	79.6	90.1	**69.1**	72.7	86.2	93.8	**90.7**
Margin[256] + $\mathcal{U}$-dist ($\beta = 1.2$)	128	63.5	84.4	68.1	80.1	91.9	67.6	74.6	87.5	94.2	90.7
Margin[256] + $\mathcal{U}$-dist ($\beta = 0.6$)	128	63.0	83.0	66.9	79.7	91.8	67.1	73.5	87.2	93.9	89.3
Margin[256] + *PADS* (Ours)	128	**67.3**	**85.9**	**69.9**	**83.5**	**93.8**	68.8	**76.5**	**89.0**	**95.4**	89.9
Triplet[205] + semihard (orig)	64	42.6	66.4	55.4	51.5	73.5	53.4	66.7	82.4	91.9	**89.5**
Triplet[205] + semihard (ReImp)	128	60.6	82.1	65.5	71.9	88.5	64.1	73.5	87.5	94.9	89.2
Triplet[205] + $\mathcal{U}$-dist (ReImp)	128	62.2	82.8	66.3	78.0	91.4	65.7	73.9	87.7	94.5	89.3
Triplet[205] + *PADS* (Ours)	128	**64.0**	**84.3**	**67.8**	**79.9**	**92.3**	**67.1**	**74.8**	**88.2**	**95.0**	**89.5**

Table 5.1: Comparison of our proposed adaptive negative sampling (*PADS*) against common static negative sampling strategies: semihard negative mining[171] (*semihard*) and static distance-based sampling ($\mathcal{U}$*-dist*)[256] using triplet[205] and margin loss[256]. *Dim* the dimensionality of ϕ.

Thus, these approaches approximate the negative distributions, while our sampling adaptively yields individual negative samples.

Adaptive learning. Curriculum Learning [15] gradually increases the difficulty of the the samples presented to the model. Hacohen et al. [85] employ a batch-based learnable scoring function to provide a batch-curriculum for training, while we learn how to adapt a sampling process to the training state. Graves et al. [**cl_tasks**] divide the training data into fixed subsets before learning in which order to use them from training. Further, Gopal et al. [81] employs an empirical online importance sampling distribution over inputs based on their gradient magnitudes during training. Similarly, Shreyas et al. [204] learn an importance sampling over instances. In contrast, we learn an online policy for selecting triplet negatives, thus instance relations. Meta Learning aims at learning how to learn. It has been successfully applied for various components of a learning process, such as activation functions [184], input masking [62], self-supervision [27], finetuning [212], loss functions [102], optimizer parameters [4] and model architectures [181, 259]. In this work, we learn a sampling distribution to improve triplet-based learning.

5.4 Experiments

In this section we provide implementation details, evaluations on standard metric learning datasets, ablations studies and analysis experiments.

Implementation details. We follow the training protocol of [256] with ResNet50. During training, images are resized to 256×256 with random crop to 224×224 and random horizontal flipping. For completeness, we also evaluate on Inception-BN [103] following standard practice in the supplementary. The initial learning rates are set to 10^{-5}. We choose triplet parameters according to [256], with $\gamma = 0.2$. For margin loss, we evaluate margins $\beta = 0.6$ and $\beta = 1.2$. Our policy π is implemented as a two-layer

Dataset		CUB200-2011[239]				CARS196[130]				SOP[171]			
Approach	Dim	R@1	R@2	R@4	NMI	R@1	R@2	R@4	NMI	R@1	R@2	R@4	NMI
HTG[275]	512	59.5	71.8	81.3	-	76.5	84.7	90.4	-	-	-	-	-
HDML[277]	512	53.7	65.7	76.7	62.6	79.1	87.1	92.1	69.7	68.7	83.2	92.4	89.3
HTL[71]	512	57.1	68.8	78.7	-	81.4	88.0	92.7	-	74.8	88.3	94.8	-
DVML[141]	512	52.7	65.1	75.5	61.4	82.0	88.4	93.3	67.6	70.2	85.2	93.8	90.8
A-BIER[175]	512	57.5	68.7	78.3	-	82.0	89.0	93.2	-	74.2	86.9	94.0	-
MIC[194]	128	66.1	76.8	85.6	69.7	82.6	89.1	93.2	68.4	77.2	89.4	95.6	90.0
D&C[201]	128	65.9	76.6	84.4	69.6	84.6	90.7	94.1	70.3	75.9	88.4	94.9	90.2
Margin[256]	128	63.6	74.4	83.1	69.0	79.6	86.5	90.1	69.1	72.7	86.2	93.8	90.8
Margin[256] + *PADS*	128	**67.3**	**78.0**	**85.9**	**69.9**	83.5	89.7	93.8	68.8	76.5	89.0	95.4	89.9
Significant increase in network parameter:													
HORDE[105]+contr. [86]	512	66.3	76.7	84.7	-	83.9	90.3	94.1	-	-	-	-	-
SOFT-TRIPLE[183]	512	65.4	76.4	84.5	-	84.5	90.7	94.5	70.1	78.3	90.3	95.9	**92.0**
Ensemble Methods:													
Rank[250]	1536	61.3	72.7	82.7	66.1	82.1	89.3	93.7	71.8	**79.8**	**91.3**	**96.3**	90.4
DREML[261]	9216	63.9	75.0	83.1	67.8	**86.0**	**91.7**	**95.0**	**76.4**	-	-	-	-
ABE[120]	512	60.6	71.5	79.8	-	85.2	90.5	94.0	-	76.3	88.4	94.8	-

Table 5.2: Comparison to the state-of-the-art DML methods on CUB200-2011[239], CARS196[130] and SOP[171]. *Dim* denotes the dimensionality of ϕ.

fully-connected network with ReLU-nonlinearity inbetween and 128 neurons per layer. Action values are set to $\alpha = 0.8, \beta = 1.25$. Episode iterations H are determined via cross-validation within [30,150]. The sampling range $[\lambda_{\text{min}}, \lambda_{\text{min}}]$ of $p(x_k|x_i)$ is set to [0.1, 1.4], with $M = 30$. The sampling probability of negatives corresponding to distances outside this interval is set to 0. For the input state we use running averages of validation recall, NMI and average intra- and interclass distance based on running average lengths of 2, 8, 16 and 32 to account for short- and longterm changes. We also incorporate the metrics of the previous 20 iterations. Finally, we include the sampling distributions of the previous iteration and the training progress normalized over the total training length. For optimization, we utilize an A2C + PPO setup with ratio limit $\epsilon = 0.2$. The history policy is updated every 5 policy iterations. For implementation we use the PyTorch framework[180] on a single NVIDIA Titan X.

	Init.	Reference	fix π_ω	fix last $p(x_n|x_a)$
R@1	$\neq$	**65.4**	64.3	59.0
R@1	=	65.4	**65.8**	57.6

Table 5.3: Transferring a fixed trained policy π_ω and fixed final distribution $p(x_n|x_a)$ to training runs with different ($\neq$) and the same network initialization (=). *Reference* denotes the training run from which π_ω and $p(x_n|x_a)$ is obtained.

Benchmark datasets. We evaluate the performance on three common benchmark datasets. For each dataset the first half of classes is used for training and the other half

Dataset	CUB200-2011[239]		CARS196[130]	
Metrics	R@1	NMI	R@1	NMI
Ours	**67.3**	**69.9**	**83.5**	**68.8**
linear CL	59.1	63.1	72.2	64.0
non-linear CL	63.6	68.4	78.1	66.8

Table 5.4: Comparison to curriculum learning strategies with predefined linear and non-linear progression of $p(x_k|x_i)$.

is used for testing. Further, we use a random subset of 15% of the training images for our validation set $\mathcal{X}_{\text{val}}$. We use:
CARS196[130], with 16,185 images from 196 car classes.
CUB200-2011[239], 11,788 bird images from 200 classes.
Stanford Online Products (SOP)[171], containing 120,053 images divided in 22,634 classes.

5.4.1 Results

In Tab. 5.1 we apply our adaptive sampling strategy on two widely adopted basic ranking losses: triplet[205] and margin loss[256]. For each loss, we compare against the most commonly used static sampling strategies, semi-hard[205] (semihard) and distance-based sampling[256] ($\mathcal{U}$-dist) on the CUB200-2011, CARS196 and SOP dataset. We measure image retrieval performance using recall accuracy R@k[107] following [175]. For completeness we additonally show the normalized mutual information score (NMI)[151], despite not fully correlating with retrieval performance. For both losses and each dataset, our learned negative sampling significantly improves the performance over the non-adaptive sampling strategies. Especially the strong margin loss greatly benefits from the adaptive sampling, resulting in boosts up to 3.8% on CUB200-2011, 3.4% on CARS196 and 1.9% on SOP. This clearly demonstrates the importance of adjusting triplet sampling to the learning process a DML model, especially for smaller datasets.
Next, we compare these results with the current state-of-the-art in DML which extend these basic losses using diverse additional training signals (MIC[194], DVML[141], HORDE[105], A-BIER[175]), ensembles of embeddings (DREML[261], D&C[201], Rank[250]) and/or significantly more network parameters (HORDE[105], SOFT-TRIPLE[183]). Tab. 5.2 shows that our results, despite not using such additional extensions, compete and partly even surpass these strong methods. On CUB200-2011 we outperform all methods, including the powerful ensembles, by at least 1.2% in Recall accuracy. On CARS196[130] we rank second behind the top performing non-ensemble method D&C[201]. On SOP[171] we lose 0.7% to MIC[194] which, in turn, we surpass on both CUB200-2011 and CARS196. This highlights the strong benefit of our adaptive sampling.

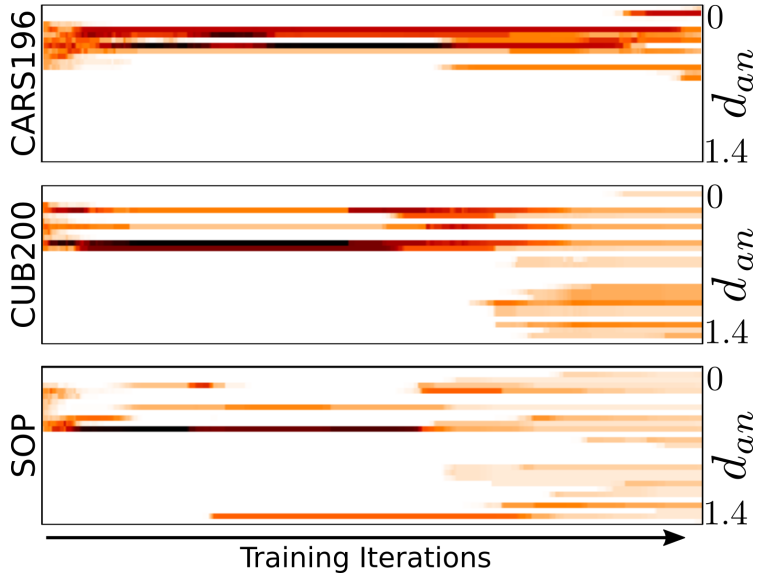


Figure 5.3: *Averaged progression of $px_k|x_i)$ over multiple training runs on CUB200-2011, CARS196 and SOP.*

5.4.2 Analysis

We now present various analysis experiments providing detailed insights into our learned adaptive sampling strategy.

Training progression of $p(x_k|x_i)$. We now analyze in Fig. 5.3 how our adaptive sampling distribution progresses during training by averaging the results of multiple training runs with different network initializations. While on CARS196 the distribution $p(x_k|x_i)$ strongly emphasizes smaller distances d_{ik}, we observe on CUB200-2011 and SOP generally a larger variance of $p(x_k|x_i)$. Further, on each dataset, during the first half of training $p(x_k|x_i)$ quickly peaks on a sparse set of bins u_m, as intuitively expected, since most triplets are still informative. As training continues, $p(x_k|x_i)$ begins to yield both harder and easier negatives, thus effectively sampling from a wider distribution. This observation confirms the result of Wu et al. [256] which proposes to ease the large gradient variance introduced by hard negatives with also adding easier negatives. Moreover, for each dataset we observe a different progression of $p(x_k|x_i)$ which indicates that manually designing similar sampling strategies is difficult, as also confirmed by our results in Tab. 5.1 and 5.4.

Transfer of π_θ and $p(x_k|x_i)$. Tab. 5.3 investigates how well a trained policy π_ω or final sampling distribution $p(x_k|x_i)$ from a reference run transfer to differently ($\neq$) or equally ($=$) initialized training runs. We find that applying a fixed trained policy (*fix* π_ω) to a new training run with the same network initialization ($=$) improves performance by 0.4% due to the immediate utility of π_ω for learning ϕ as π_ω is already fully adapted to the

$[\lambda_{\text{min}}, \lambda_{\text{max}}]$	$[0, 2]$	$[0.1, 1.4]$	$[0.25, 1.0]$	$[0.5, 1.4]$
Recall@1	64.7	**65.7**	64.8	63.7
NMI	67.5	**69.2**	68.2	67.5

Table 5.5: Varying the interval $U = [\lambda_{\text{min}}, \lambda_{\text{max}}]$ of distances d_{ik} used for learning $p(x_k|x_i)$. The number of bins u_m is kept fixed to $M = 30$.

Num. bins M	10	30	50	100
Recall@1	63.8	**65.7**	65.3	64.9
NMI	67.8	**69.2**	68.7	68.6

Table 5.6: Varying the number of bins u_m used to discretize the range of distances $U = [0.1, 1.4]$ used for learning $p(x_k|x_i)$.

reference learning process. In contrast, applying the trained policy to a differently initialized training run ($\neq$) drops performance by 1.5%. Since the fixed π_ω cannot adapt to the learning states of the new model, its support for optimizing ϕ is diminished. Note that the policy has only been trained on a single training run, thus it cannot fully generalize to different training dynamics. This shows the importance of an adaptive sampling.
Next, we investigate if the distribution $p(x_k|x_i)$ obtained at the end of training can be regarded as an optimal sampling distribution over d_{ik}, as π_ω is fully trained. To this end we fix and apply the distribution $p(x_k|x_i)$ after its last adjustment by π_ω (*fix last* $p(x_k|x_i)$) in training the reference run. As intuitively expected, in both cases performance drops strongly as *(i)* we now have a static sampling process and *(ii)* the sampling distribution is optimized to a specific training state. Given our strong results, this proves that our sampling process indeed adapts to the learning of ϕ.

Curriculum learning. To compare our adaptive sampling with basic curriculum learning strategies, we pre-define two sampling schedules: *(1)* A *linear* increase of negative hardness, starting from a semi-hard distance intervall[205] and *(2)* a *non-linear* schedule using distance-based sampling[256], where the distribution is gradually shifted towards harder negatives. We visualize the corresponding progression of the sampling distribution in the supplementary material. Tab. 5.4 illustrates that both fixed, pre-defined curriculum schedules perform worse than our learned, adaptive sampling distribution by at least 3.6% on CUB200-2011. On CARS196 the performance gap is even larger. The strong difference in datasets further demonstrates the difficulty of finding broadly applicable, effective fixed sampling strategies.

5.4.3 Ablation studies

Subsequently we ablate different parameters for learning our sampling distribution $p(x_k|x_i)$ on the CUB200-2011 dataset. More ablations are shown in the appendix. To make the fol-

Init. Distr.	$\mathcal{U}_{[0.1,1.4]}$	$\mathcal{N}(0.5, 0.05)$	$\mathcal{U}_{[0.3,0.7]}$
Recall@1	63.9	65.0	**65.7**
NMI	67.0	68.6	**69.2**

Table 5.7: Comparison of $p(x_k|x_i)$-initializations on distance interval $U = [0.1, 1.4]$. $\mathcal{U}_{[a,b]}$ denotes uniform emphasis in $[a, b]$ with low probabilities outside the interval. $\mathcal{N}(\mu, \sigma)$ denotes a normal distribution.

lowing experiments comparable, no learning rate scheduling was applied, as convergence may significantly change with different parameter settings. In contrast, the results in Tab 5.1-5.2 are obtained with our best parameter settings *and* a fixed learning rate scheduling. Without scheduling, our best parameter setting achieves a recall value of 65.7 and NMI of 69.2 on CUB200-2011.

Distance interval U. As presented in Sec. 5.2.1, $p(x_k|x_i)$ is defined on a fixed interval $U = [\lambda_{\text{min}}, \lambda_{\text{max}}]$ of distances. Similar to other works[88, 256], this allows us to additionally regularize the sampling process by clipping the tails of the true range of distances $[0, 2]$ on Φ. Tab. 5.5 compares different combinations of $\lambda_{\text{min}}, \lambda_{\text{max}}$. We observe that, while each option leads to significant performance boost compared to the static sampling strategies, an interval $U = [0.1, 1.4]$ results in the most effective sampling process.

Number of bins M. Next, we analyze the impact of the resolution of the interval U in Tab. 5.6, i.e. the number of bins M. This affects the flexibility of $p(x_k|x_i)$, but also the complexity of the actions a to be predicted. As intuitively expected, increasing M allows for better adaption and performance until the complexity grows too large.

Initialization of $p(x_k|x_i)$. Finally, we analyze how the initialization of $p(x_k|x_i)$ impacts learning. Tab. 5.7 compares the performance using different initial distributions, such as a neutral uniform initialization (i.e. random sampling) ($\mathcal{U}_{[0.1,1.4]}$), emphasizing semi-hard negatives x_k early on ($\mathcal{U}_{[0.3,0.7]}$) or a proxy to [256] ($\mathcal{N}(0.5, 0.05)$). We observe that our learned sampling process benefits from a meaningful, but generic initial configuration of $p(x_k|x_i)$, $\mathcal{U}_{[0.3,0.7]}$, to effectively adapt the learning process of ϕ.

Performance with Inception-BN. For fair comparison, we also evaluate using Inception-V1 with Batch-Normalization [103]. We follow the standard pipeline (see e.g. [161, 183]), utilizing Adam [122] with images resized and random cropped to 224x224. The learning rate is set to 10^{-5}. We retain the size of the policy network and other hyperparameters. The results on CUB200-2011[239] and CARS196[130] are listed in Table 5.8. On CUB200, we achieve results competitive to previous state-of-the-art methods. On CARS196, we achieve a significant boost over baseline values and competitive performance to the state-of-the-art.

Dataset		CUB200-2011[239]				CARS196[130]			
Approach	Dim	R@1	R@2	R@4	NMI	R@1	R@2	R@4	NMI
HTG[275]	512	59.5	71.8	81.3	-	76.5	84.7	90.4	-
HDML[277]	512	53.7	65.7	76.7	62.6	79.1	87.1	92.1	69.7
HTL[71]	512	57.1	68.8	78.7	-	81.4	88.0	92.7	-
DVML[141]	512	52.7	65.1	75.5	61.4	82.0	88.4	93.3	67.6
A-BIER[175]	512	57.5	68.7	78.3	-	82.0	89.0	93.2	-
MIC[194]	128	66.1	76.8	85.6	69.7	82.6	89.1	93.2	68.4
D&C[201]	128	65.9	76.6	84.4	69.6	84.6	90.7	94.1	70.3
Margin[256]	128	63.6	74.4	83.1	69.0	79.6	86.5	90.1	69.1
Reimpl. Margin[256], IBN	512	63.8	75.3	84.7	67.9	79.7	86.9	91.4	67.2
Ours(Margin[256] + *PADS*, IBN)	512	66.6	77.2	85.6	68.5	81.7	88.3	93.0	68.2
Significant increase in network parameter:									
HORDE[105]+Contr.[86]	512	66.3	76.7	84.7	-	83.9	90.3	94.1	-
SOFT-TRIPLE[183]	512	65.4	76.4	84.5	-	84.5	90.7	94.5	70.1
Ensemble Methods:									
Rank[250]	1536	61.3	72.7	82.7	66.1	82.1	89.3	93.7	71.8
DREML[261]	9216	63.9	75.0	83.1	67.8	**86.0**	**91.7**	**95.0**	**76.4**
ABE[120]	512	60.6	71.5	79.8	-	85.2	90.5	94.0	-

Table 5.8: Comparison to the state-of-the-art DML methods on CUB200-2011[239] and CARS196[130] using the Inception-BN Backbone (see e.g. [161, 183]) and embedding dimension of 512.

Validation set $\mathcal{X}_{\text{val}}$: The validation set $\mathcal{X}_{val}$ is sampled from the training set $\mathcal{X}_{train}$, composed as either a fixed disjoint, held-back subset or repetitively re-sampled from $\mathcal{X}_{train}$ during training. Further, we can sample $\mathcal{X}_{val}$ across all classes or include entire classes. We found (Tab. 5.9 (d)) that sampling $\mathcal{X}_{val}$ from each class works much better than doing it per class. Further, resampling $\mathcal{X}_{val}$ provides no significant benefit at the cost of an additional hyperparameter to tune.

Composition of states s and target metric e. Choosing meaningful target metrics $e(\phi(\cdot;\theta), \mathcal{X}_{\text{val}})$ for computing rewards r and a representative composition of the training state s increases the utility of our learned policy π_ω. To this end, Tab. 5.10 compares different combinations of state compositions and employed target metrics e. We observe that incorporating information about the current structure of the embedding ϕ into s, such as intra- and inter-class distances, is most crucial for effective learning and adaptation. Moreover, also incorporating performance metrics into s which directly represent the current performance of the model ϕ, e.g. Recall@1 or NMI, additional adds some useful information.

Validation Set:	$\mathcal{X}_{\text{val}}^{\text{By}}$	$\mathcal{X}_{\text{val}}^{\text{Per}}$	$\mathcal{X}_{\text{val}}^{\text{By, R}}$	$\mathcal{X}_{\text{val}}^{\text{Per, R}}$
Recall@1	62.6	65.7	63.0	**65.8**
NMI	67.7	69.2	67.8	**69.6**

Table 5.9: Composition of $\mathcal{X}_{\text{val}}$. Superscript By/Per denotes usage of entire classes/sampling across classes. R denotes re-sampling during training with best found frequency of $\frac{1}{50 \text{ epochs}}$.

Frequency of updating π_ω. We compute the reward r for an adjustment a to $p(x_k|x_i)$ every H DML training iterations. High values of H reduce the variance of the rewards r, however, at the cost of slow policy updates which result in potentially large discrepancies to updating ϕ. Tab. 5.11 shows that choosing H from the range $[30, 70]$ results in a good trade-off between the stability of r and the adaptation of $p(x_k|x_i)$ to ϕ. Moreover, we also show the result for setting $M = \infty$, i.e. using the initial distribution throughout training without adaptation. Fixing this distribution performs worse than the reference method Margin loss with static distance-based sampling[256]. Nevertheless, frequently adjusting $p(x_k|x_i)$ leads to significant superior performance, which indicates that our policy π_ω effectively adapts $p(x_k|x_i)$ to the training state of ϕ.

Importance of long-term information for states s. For optimal learning, s should not only contain information about the current training state of ϕ, but also about some history of the learning process. Therefore, we compose s of a set of running averages over different lengths $\mathcal{R}$ for various training state components, as discussed in the implementation details of the main paper. Tab. 5.12 confirms the importance of long-term information for stable adaptation and learning. Moreover, we see that the set of moving averages $\mathcal{R} = \{2, 8, 16, 32\}$ works best.

Visual curriculum evaluations. In Fig. 5.4 we visually illustrate the fixed curriculum schedules which we applied for the comparison experiment in Sec. 5.3 of our main paper. We evaluated various schedules - Linear progression of sampling intervals starting at semi-hard negatives going to hard negatives, and progressively moving $\mathcal{U}$-dist[256] towards harder negatives. The schedules visualized were among the best performing ones to work for both CUB200 and CARS196 dataset.

Reward metrics e / Composition of state s	NMI	R@1	R@1 + NMI
Recall, Dist., NMI	63.9	65.5	**65.6**
	68.5	68.9	69.2
Recall, Dist.	65.0	**65.7**	64.4
	68.5	**69.2**	**69.4**
Recall, NMI	63.7	63.9	64.2
	68.4	68.2	68.5
Dist., NMI	65.3	65.3	65.1
	68.8	68.7	68.5
Dist.	65.3	65.5	64.3
	68.8	69.1	68.6
Recall	64.2	65.1	64.9
	67.8	69.0	68.4
NMI	64.3	64.8	63.9
	68.7	**69.2**	68.4

Table 5.10: Comparison of different compositions of the training state s and reward metric e. *Dist.* denotes average intra- and inter-class distances. Recall in state composition denotes all Recall@k-values, whereas for the target metric only Recall@1 was utilized.

H	10	30	50	70	100	∞	[256]
R@1	64.4	**65.7**	65.4	65.2	65.1	61.9	63.5
NMI	68.3	**69.2**	69.2	68.9	69.0	67.0	68.1

Table 5.11: Evaluation of the policy update frequency H, i.e. the number of DML training iterations performed before updating the policy π_θ using a reward r.

$\mathcal{R}$	2	2, 32	2, 8, 16, 32	2, 8, 16, 32, 64
R@1	64.5	65.4	**65.7**	65.6
NMI	68.6	69.1	69.2	**69.3**

Table 5.12: Evaluation of various sets $\mathcal{R}$ of moving average lengths to analyze the benefit of long-term learning progress information added to training states s.

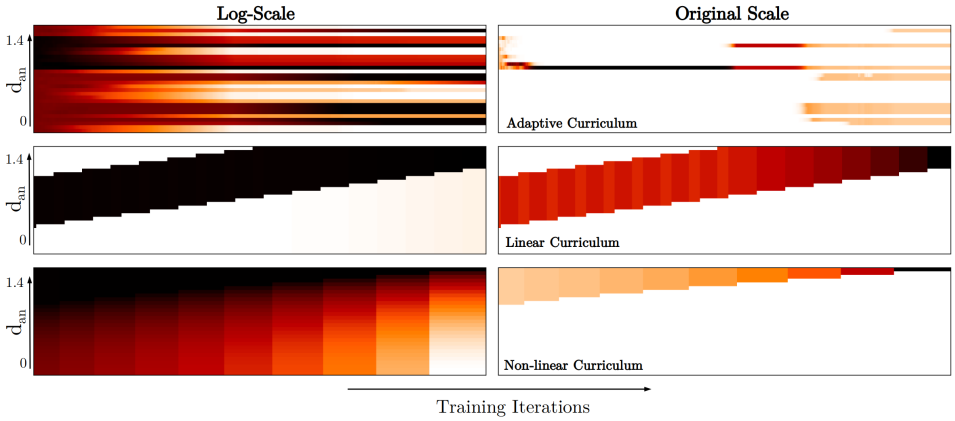


Figure 5.4: *Visual comparison between fixed sampling curriculums and a learned progression of* $p(x_k|x_i)$ *by PADS.* Left: log-scale over $p(x_k|x_i)$, right: original scale. Top row: learned sampling schedule (PADS); middle row: linear shift of a sampling interval from semi-hard[205] negatives to hard negatives; bottom row: shifting a static distance-based sampling[256] to gradually sample harder negatives.

Comparison of RL algorithms. We evaluate the applicability of the following RL algorithms for optimizing our policy π_ω (Eq. 4 in the main paper):

- REINFORCE algorithm[255] with and without Exponential Moving Average (EMA)
- Advantage Actor Critic (A2C)[221]
- Rainbow Q-Learning[95] without extensions (vanilla) and using Priority Replay and 2-Step updates
- Proximal Policy Optimization (PPO)[207] applied to REINFORCE with EMA and to A2C.

For a comparable evaluation setting we use the CUB200-2011[239] dataset without learning rate scheduling and fixed 150 epochs of training. Within this setup, the hyperparameters related to each method are optimized via cross-validation. Tab. 5.13 shows that all methods, except for vanilla Q-Learning, result in an adjustment policy π_ω for $p(x_k|x_i)$ which outperforms static sampling strategies. Moreover, policy-based methods in general perform better than Q-Learning based methods with PPO being the best performing algorithm. We attribute this to the reduced search space (Q-Learning methods need to evaluate in state-actions space, unlike policy-methods, which work directly over the action space), as well as not employing replay buffers, i.e. not acting off-policy, since state-action pairs of previous training iterations may no longer be representative for current training stages.

Qualitative UMAP visualization. Figure 5.5 shows a UMAP[153] embedding of test image features for CUB200-2011[239] learned by our model using PADS. We can see

Approach	R@1	NMI
Margin[256]	63.5	68.1
REINFORCE	64.2	68.5
REINFORCE, EMA	64.8	68.9
REINFORCE, A2C	65.0	69.0
PPO, EMA	65.4	69.0
PPO, A2C	**65.7**	**69.2**
Q-Learn	63.2	67.9
Q-Learn, PR/2-Step	64.9	68.5

Table 5.13: Comparison of different RL algorithms. For policy-based algorithms (REINFORCE, PPO) we either use Exponential Moving Average (EMA) as a variance-reducing baseline or employ Advantage Actor Critic (A2C). In addition, we also evaluate Q-Learning methods (vanilla and Rainbow Q-Learning). For the Rainbow setup we use Priority Replay and 2-Step value approximation. Margin loss[256] is used as a representative reference for static sampling strategies.

clear groupings for birds of the same and similar classes. Clusterings based on similar background is primarily due to dataset bias, e.g. certain types of birds occur only in conjunction with specific backgrounds.

5.4.4 Typical image retrieval failure cases

Fig. 5.6 shows nearest neighbours for good/bad test set retrievals. Even though the nearest neighbors do not always share the same class label as the anchor, all neighbors are very similar to the bird species depicted in the anchor images. Failures are due to very subtle differences.

5.5 Discussion

This chapter presented a learned adaptive triplet sampling strategy using Reinforcement Learning. We optimized a teacher network to adjust the negative sampling distribution to the ongoing training state of a DML model. By training the teacher to directly improve the evaluation metric on a held-back validation set, the resulting training signal optimally facilitates DML learning. Our experiments show that our adaptive sampling strategy improves significantly over static sampling distributions. Thus, even though only built on top of basic triplet losses, we achieve competitive or even superior performance compared to the state-of-the-art of DML on multiple standard benchmarks sets.

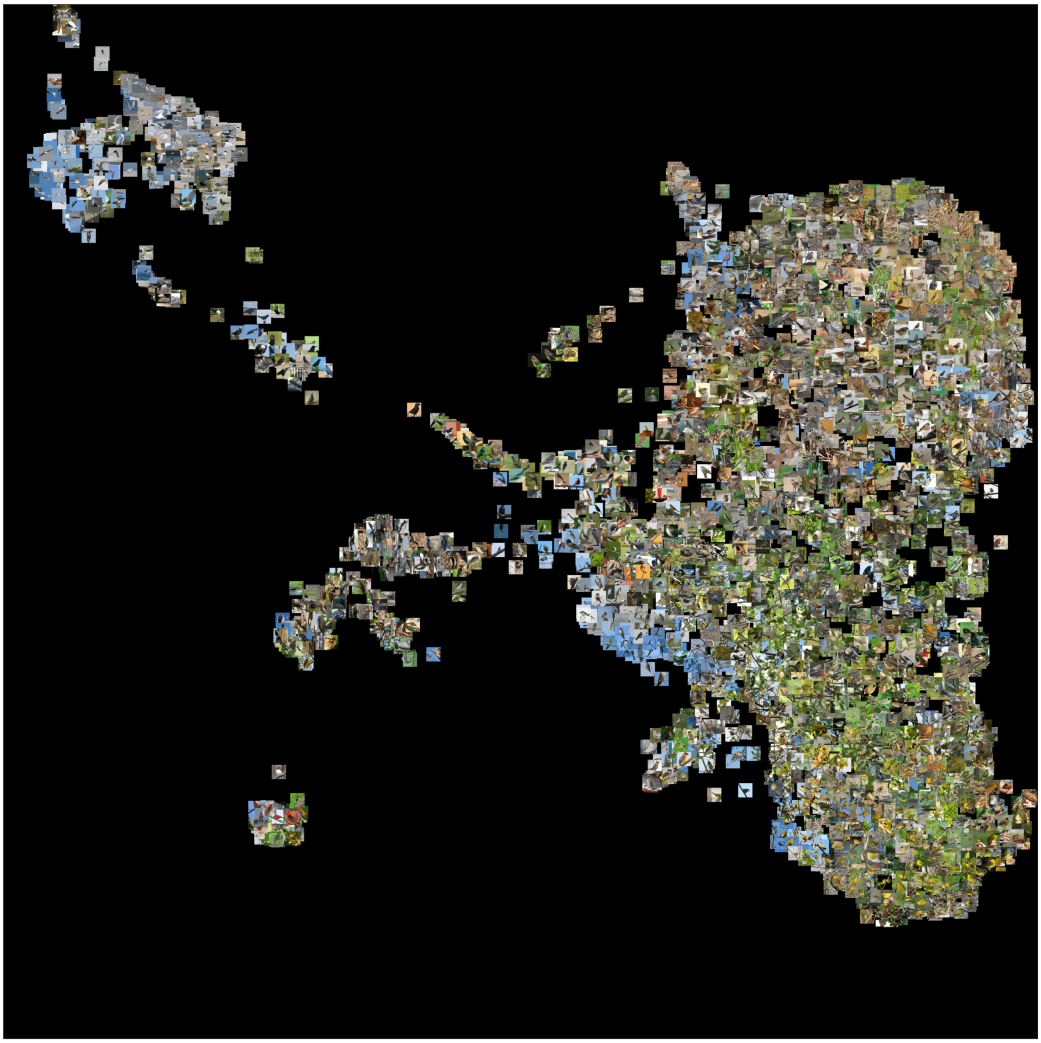


Figure 5.5: *UMAP embedding* based on the image embeddings $\phi(\cdot;\theta)$ obtained from our proposed approach on CUB200-2011[239] (Test Set).

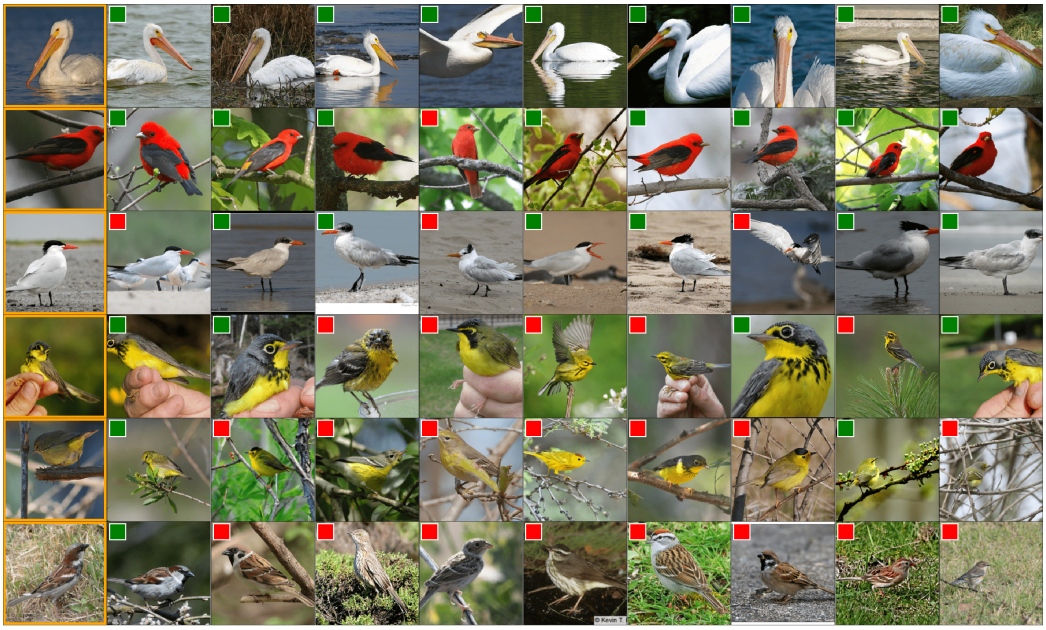


Figure 5.6: *Selection of good and bad nearest neighbour retrieval cases on CUB200-2011 (Test).* Orange bounding box marks query images, green/red boxes denote correct/incorrect retrievals.

6 Unsupervised Representation Learning from Reliable Image Similarities

The driving force of deep learning is supervised training using vast amounts of tediously labeled training samples, such as object bounding boxes for visual recognition. Since easily accessible visual data is growing exponentially, manual labeling of training samples constitutes a bottleneck to utilizing all this valuable data. Consequently, there has recently been great interest in weakly supervised [274], self-supervised [166], and unsupervised [154, 260] approaches to representation learning. Fundamental computer vision problems like classification [197, 269], object detection [270] and image segmentation [185] all directly depend on such learned representations to find similar objects or group related image areas.
To learn a characteristic representation of images and the relations between them, different degrees of supervision can be considered: *(i)* supervised learning using samples with class labels [149, 205, 256] as, for instance, covered in the previous chapters; *(ii)* problem specific surrogate tasks such as colorization [140], permutations [166], or transitivity [249], and *(iii)* unsupervised feature learning [21]. Regardless of the training signal, be it unaries such as class labels [12], binary similarity constraints between samples [217] or sample ordering constraints [188], a dataset of N training samples gives rise to N^2 pairwise relations, exploitable for learning our representation. In the absence of supervisory information, these relations need to be automatically inferred during training. However, the vast majority of these inferred pairwise relations turn out to be unreliable as discussed in Sect. 6.1, Fig. 6.2, and Fig. 6.3. Despite the danger of diminished performance due to learning from spurious relations, recent approaches on unsupervised representation learning [21, 30], nevertheless, do not question the reliability of these relations. Now, assuming that only a small fraction of correct relations per sample can be identified reliably (i.e. we are left with at most $O(N)$ class labels or pairwise link constraints), how can we discover those few reliable relations, when no label or guiding side information is available?
Subsequently, we propose a novel approach to visual representation learning that explicitly identifies and leverages reliable image relations without the need for annotations, supervision, problem-specific surrogate tasks for self-supervision, or pre-training. By extracting compact groups of images we are able to harness reliable similarities. Subsequently, we divide these compact groups constituting the overall learning problem into smaller, (potentially overlapping) subproblems, such that each contains only reliable dissimilarities between their groups. Thus, whereas the complicated global problem suffers from many of the N^2 relations not being reliable, we ensure that the samples in each

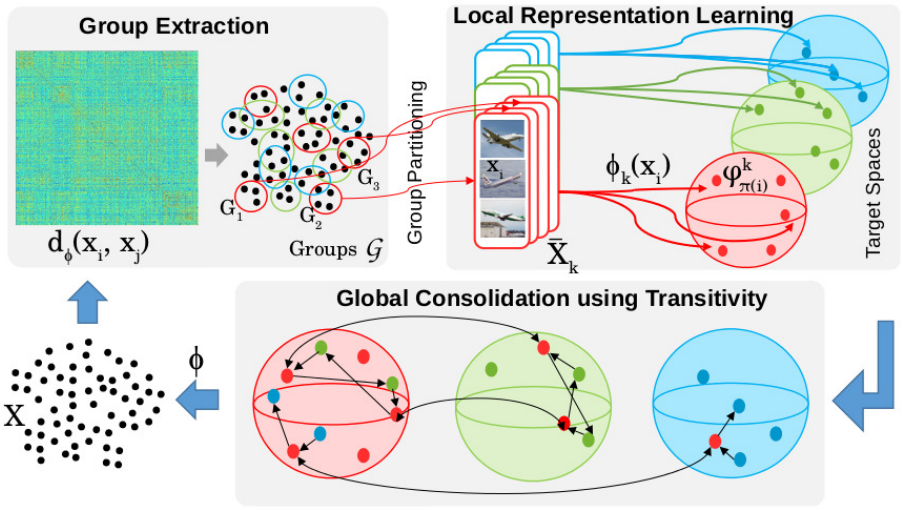


Figure 6.1: *Overview of our iterative learning procedure.* We first find reliable similarity constraints by forming compact groups. To avoid unreliable dissimilarities, we partition the data into sets of mutually dissimilar groups, $\bar{\mathcal{X}}_k$. Based on these (dis-)similarity constraints between/within groups, we learn a local representation $\phi_k(x_i)$ for subset $\bar{\mathcal{X}}_k$. Finally, we exploit sparse couplings between the local representations to arrive at a consolidated global representation. This iterative procedure improves the overall representation by successively adding reliable constraints into the learning process.

subproblem are either reliably similar or dissimilar. Optimization is then performed by learning a mapping from the images onto dedicated target embedding locations, sampled to reflect the structure and distribution estimated from the reliable relations for each subset. Next, coupling the local subproblems by utilizing transitivity between their samples allows us to consolidate the learned individual representations into a concerted global representation using well-known techniques from Deep Metric Learning. Finally, by alternating between extracting reliable relations and learning, we successively incorporate more reliable relations and in turn more data which ultimately improves our image representation (cf. Fig. 6.1).

We evaluate our model on challenging benchmarks and achieve state-of-the-art performance on the ImageNet [50] dataset, thus proving the scalability of our approach. Further, our approach performs comparably to the state-of-the-art in transfer learning on PASCAL VOC [61] indicating its general applicability. By performing ablation and analysis studies, we finally provide insights into our learning procedure.

This chapter is based on our publication *'Unsupervised Representation Learning by Discovering Reliable Image Relations'* [155].

6.1 Discovering and Learning from Reliable Similarities

Let us now learn a representation $\phi : \mathbb{R}^{D'} \to \Phi \subset \mathbb{R}^{D}$ that allows to relate D' dimensional image samples $x_i, x_j \in \mathcal{X}$ to another. As discussed in the previous chapters, this problem

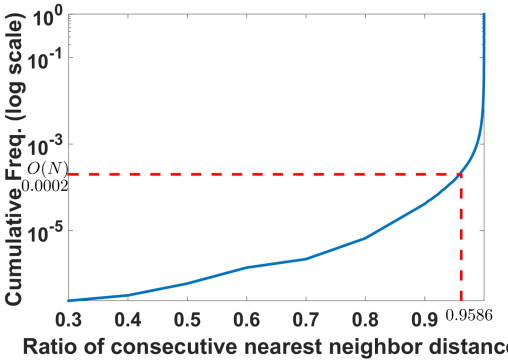


Figure 6.2: *Nearest neighbour distance ratios.* For most of the N^2 pairwise relations, the ratio $d(x_i,x_j)/d(x_i,x_{j+1})$ (for sorted neighbors x_j) is close to 1. Only $O(N)$ have a robust ordering. This analysis is based on $N = 5000$ samples from the STL-10[43] dataset using the Euclidean distances based on an unsupervised, learned representation.

is typically addressed by similarity learning and formulated as learning image distances $d_\phi(x_i, x_j) = \|\phi(x_i;\theta) - \phi(x_j;\theta)\|_2$. Following earlier notation, θ denotes the learnable parameters of the representation. Hence, this naturally yields a representation ϕ such that given image relations $d(x_i, x_j)$ are reflected and preserved under the embedding ϕ. Thus, learning ϕ is propelled by pairwise relationships between images indicated by $d(x_i, x_j)$: In supervised training $d(x_i, x_j)$ is typically defined on the basis of manually provided class labels y_i (thus, $d(x_i, x_j) \in \{0, 1\}$), weak user feedback, problem specific surrogate tasks, dense triplet ranking constraints such as $d(x_i, x_j) < d(x_i, x_k)$, or other sparse partial ordering constraints.
Without being externally provided with information driving the training process, we have to infer learning constraints ourselves during training. Subsequently, we now show how to identify learning constraints which are likely to agree with ground-truth, thus constitute a reliable substitute for manual supervision signals.

6.1.1 Identifying reliable relations

Regardless of the origin of $d(x_i, x_j)$, only a small number of all possible N^2 pairwise relations (for N training samples) may be feasibly provided by manual annotation. For the particular case of unsupervised learning only a small number of the pairwise relations can be inferred correctly for training with high confidence. Let's consider a triplet of images (x_i, x_j, x_k) with the ground-truth distance between x_i and x_j being small and between x_i and x_k being large, i.e. $d(x_i, x_j) < d(x_i, x_k)$. A learned distance function d_ϕ is correct, if it obeys these ground-truth constraints. Furthermore to ensure robustness to noise these constraints should be obeyed reliably by a clear margin $d(x_i, x_j)/d(x_i, x_k) \overset{!}{\ll} 1$. In Fig. 6.2 we plot this ratio for consecutive nearest neighbors. This is the same ratio used in [147] to measure the reliability of a matching similar image. We observe that for only $O(N)$ pairwise distances the ratio is significantly smaller than 0.95. All other relations would not even have an opportunity to exhibit above correctness constraints reliably and,

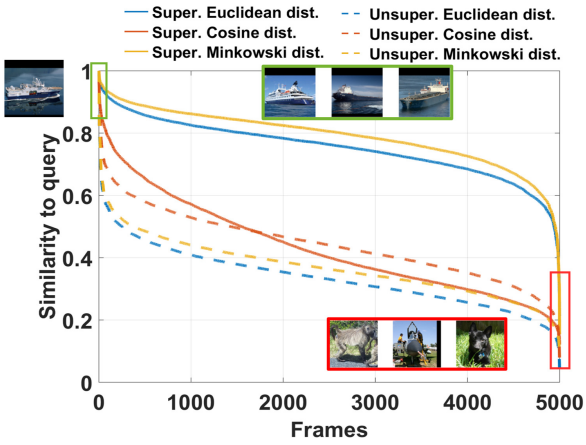


Figure 6.3: *Sorted pairwise similarities for a query image based on different representations and distance metrics resulting from supervised and unsupervised training on STL-10[43].* Only the strong (dis-)similarities at both ends are reliable to provide a robust ordering.

thus, would be falsely identified to be correct rather than spurious. Further, in Fig. 6.3 we plot the sorted similarities for a query image. Observe from Fig. 6.2 that above triplet constraints can only be fulfilled where there is significant slope in Fig. 6.3. Thus, only relations from both ends, where we have strong (dis-)similarities, can be considered to be reliable. These relations are significantly less susceptible to change under noise than the vast majority, as analyzed in Fig. 6.4. However, recent work on unsupervised learning has nevertheless simply relied on all pairwise relations [21] inferred during training at the cost of incorporating corrupted relations. In contrast, we now present an approach for unsupervised representation learning which explicitly aims at extracting and leveraging these reliable relations.

6.1.2 Outline of our iterative representation learning:

We first decompose the training set into K subsets $\bar{\mathcal{X}}_k \subset \mathcal{X}, k = 1, \ldots, K$ of images by extracting (Sec. 6.1.3) and distributing (Sec. 6.1.4) potentially overlapping compact groups of images to the subproblems $\bar{\mathcal{X}}$ which exhibit reliable mutual similarity, based on the representation from the previous training iteration. Within each $\bar{\mathcal{X}}_k$ all mutual image relations are reliable. In each iteration, learning a representation ϕ (Sec. 6.1.5) then proceeds as follows: For each subset $\bar{\mathcal{X}}_k$ we seek an embedding $\phi_k(x_i) \triangleq \phi_k(x_i; \theta_k)$[1] with $\phi_k : \mathbb{R}^{D'} \to \Phi_k \subset \mathbb{R}^D$. To learn ϕ_k, we randomly sample target points $\varphi_t^k \in \Phi_k$ such that the distribution of their pairwise distances matches those of the constituents x_i in the groups comprised by the subproblem $\bar{\mathcal{X}}_k$. Learning the mapping from images x_i to targets φ_t^k then yields the local representations ϕ_k (Sec. 6.1.5). In a final step, all these local representations are merged into a single overall representation ϕ by formulating triplet learning problems exploiting transitivity relations between those samples x_i which are shared among subsets $\bar{\mathcal{X}}_k$ (Sec. 6.1.5). Based on representation ϕ, reliable relations are

[1] Subsequently, we omit the explicit dependence on the model parameters θ_k for readibility.

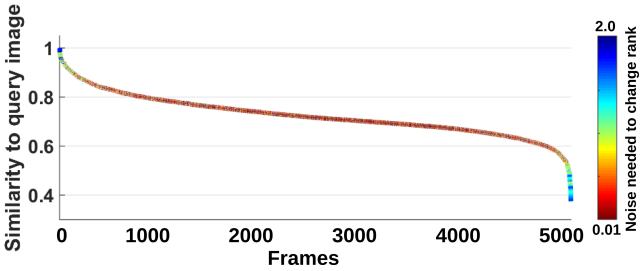


Figure 6.4: *Sorted pairwise image similarities for an STL-10 query image.* Color indicates amount of noise (gaussian, $\mu = 0$, $\sigma^2 = 0.01, \ldots, 2$) needed for each image to change its rank w.r.t. query.

then again extracted to serve as input for the next iteration. Since no annotations are provided, the first iteration of our training starts *from scratch* with a randomly initialized CNN and random assignments.

6.1.3 Compact groups for finding reliable similarities

Every training iteration builds upon the distances learned in the previous round. Since the majority of inferred relations between our training samples is not reliable, how can we find the ones we can rely upon without annotations? In Fig. 6.2 to Fig. 6.4 we empirically demonstrate that only a few nearest neighbours (NN) can robustly be identified. Unfortunately, even these do not always give rise to correct relations. For instance, two samples may spuriously have a small distance or a sample may be an outlier, thus corrupting the nearest neighbors for a given sample x_i. This issue can however be alleviated by forming compact groups of images. Fig. 6.5 (a) shows that considering dense groups of h samples increases the chance of inferring correct similarity relations, following the intuition that in dense areas of our feature space pairwise relations do not arise accidentally but due to actual commonalities. For different group sizes h on ImageNet, the plot illustrates the average percentage of correct image relations in a group based on their ground-truth labels (blue). As we observe, the chance that h samples appear erroneously close to another (forming a compact group) becomes increasingly unlikely as h increases. On the other hand, large compact groups are scarce and therefore only a small number of samples can be covered for large h (red).

Let the largest pairwise distance, $d^G_{\max} = \max_{x_i, x_j \in G} d_\phi(x_i, x_j)$, between members of a group G represent its compactness. Further, when building groups of random samples, it is highly unlikely to form a group with low compactness, i.e. correctly mutually close samples. Thus, we demand $d^G_{\max}$ for each G to be smaller than the p-th percentile of the compactness of a set of randomly built groups of equal size. Consequently, we extract reliable groups G by starting with a seed x_i and add its nearest neighbors as long as the resulting compactness $d^G_{\max}$ is below the p-th percentile compactness of the randomly sampled groups of equal size. Thus, we grow G to be as large as possible.

Repeating this process with all samples $x_i \in \mathcal{X}$, we denote $\mathcal{G}$ as the resulting set of all groups G with reliable pairwise similarity relationships. Note, since all groups are build

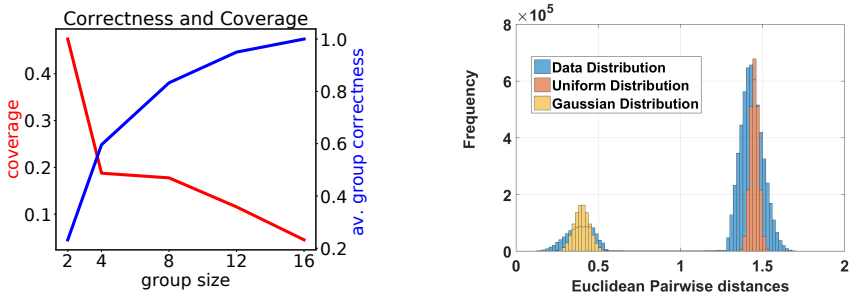


Figure 6.5: a) *Group correctness vs. data coverage w.r.t group size.* blue: average correctness of relations in a group. red: data coverage for groups G of different size extracted on ImageNet using our representation. We call a relation correct if it links images with the same label. Note that we only use labels for evaluation in this plot, but not for extracting our groups. Coverage: fraction of all samples that can be covered by compact groups of a given size relative to the overall number of extracted groups G. b) *Data- vs. Target-Distribution.* Distribution of all pairwise intra-group and inter-group distances (blue) based on a fully supervised trained representation, of points uniformly sampled from an ℓ_2 unit sphere (orange), and of a Gaussian distribution (yellow). Evidently, most data points are far apart, approximated by the orange mode. But there are also characteristic compact hubs approximated by the yellow mode (which has been magnified for the purpose of this illustration). Note that the sampled distributions can approximate the data distribution.

independently, typically there is overlap between groups G, i.e. G not being mutually disjoint.

6.1.4 Reliable dissimilarity by partitioning groups

The compact groups in $\mathcal{G}$ provide small distances that we can reliably use for learning a representation. However, the relationships *between* the groups can be arbitrary and many are unlikely to be reliable as discussed above and in Fig. 6.2. Therefore, to further increase reliability, instead of using all relations in $\mathcal{G} \subset \mathcal{X}$ for learning our representation, we partition them into K subsets of L groups each, i.e. $\bar{\mathcal{X}}_k = \{G_1^k, \ldots, G_L^k\}$. By distributing overlapping groups across different $\bar{\mathcal{X}}_k$ while maximizing the distance between groups within a subset, we gathers the reliable dissimilarity relationships from the tail of the distance distributions shown in Fig. 6.3. Thus, all relations in each subset are as reliable as possible.

Following the work of CliqueCNN [12], we formally partition $\mathcal{G}$ using the following criteria: *(i)* all groups $G \in \bar{\mathcal{X}}_k$ should be mutually distant, *(ii)* partially overlapping groups should be distributed across different $\bar{\mathcal{X}}_k$ to establish couplings between the subproblems exploitable for transitivity relations, and *(iii)* the union of all subsets $\bigcup_{k=1}^{K} \bar{\mathcal{X}}_k$ should cover $\mathcal{X}$ as much as possible to maximize the usage of training samples. Using these constraints, we formulate the partitioning process as the following optimization problem [12].

Let $\boldsymbol{C} \in \{0,1\}^{|\mathcal{G}|\times|\mathcal{X}|}$ be the assignment matrix of samples x_i to groups G. Furthermore,

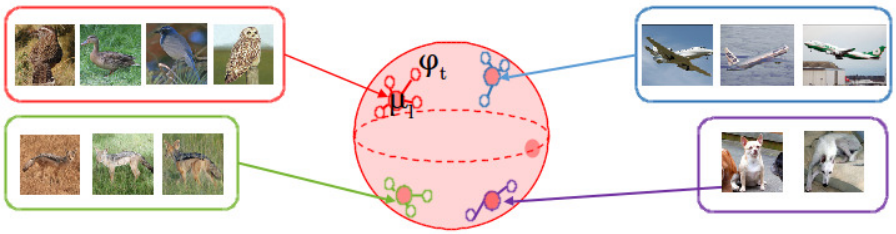


Figure 6.6: *Sampling target points in* Φ_k. Large distances between groups G_l are captured by sampling centroids μ_l uniformly from the surface of a hypersphere. Then target points φ_t for the $x_i \in G_l$ are sampled from a Gaussian around μ_l to represent the compact groups' constituents.

the column vector $a_k \in \{0,1\}^{|\mathcal{G}|\times 1}$ indicates groups assigned to subset $\bar{\mathcal{X}}_k$. Then $\boldsymbol{A} = (a_1, \ldots, a_K) \in \{0,1\}^{|\mathcal{G}|\times K}$ are the assignments of groups to all $\bar{\mathcal{X}}_k$. Moreover, $S \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$ contains the pairwise mean similarities between any two groups. With $\mathbf{1}$ denoting the all-ones vector, the partitioning objective is the formulated as

$$\begin{aligned} \min_{\mathbf{A}} \quad & \operatorname{tr}(\mathbf{A}^\top \mathbf{S} \boldsymbol{A}) - \operatorname{tr}(\boldsymbol{A}^\top \operatorname{diag}(\boldsymbol{S}) \boldsymbol{A}) \\ & - \lambda_1 \sum_k^K ||a_k^\top \boldsymbol{C}||_p^p - \lambda_2 ||\mathbf{1}\boldsymbol{A}^\top \boldsymbol{C}||_p^p \\ \text{s.t.} \quad & \boldsymbol{A}^\top \mathbf{1} = L\mathbf{1} \end{aligned} \tag{6.1}$$

where $\operatorname{tr}(\boldsymbol{A}^\top \boldsymbol{S} \boldsymbol{A})$ is regularized for the diagonal elements of S and enforces constraint *(i)*, minimal similarity, i.e. maximal distance between groups in $\bar{\mathcal{X}}_k$, $||a_k^\top \boldsymbol{C}||_p^p$ maximizes *(ii)*, the distribution of groups across the subsets $\bar{\mathcal{X}}_k$, and $||\mathbf{1}\mathbf{A}^\top \boldsymbol{C}||_p^p$ enforces *(iii)*, maximal coverage of training samples. As discussed in detail in [12], this optimization problem can be efficiently solved with $p = \frac{1}{16}$ for the penalty terms to approximate the non-linear step-function. The hyperparameters λ_1, λ_2 are weighting terms for adjusting relative impact of individual constraints. CliqueCNN [12] has addressed a similar problem to find a discrete partitioning of images into surrogate classes for optimizing a standard classification task similar to DeepCluster [30]. However, both [12] and [30] are using all resulting surrogate classes without considering their reliability. Thus, they are inevitably prone to introducing noise into the learning process. In contrast, we seek a partitioning of the set of already extracted groups $\mathcal{G}$ into subsets to further increase the reliability of our relations which we use to formulate the following learning problem.

6.1.5 Reliable unsupervised similarity learning

Now we learn a representation $\phi_k(x_i)$ that preserves all the reliable relations in $\bar{\mathcal{X}}_k$. In the previous chapters we showed that DML is a powerful tool to learn similarities between datapoints. However, in Sec. 5.1 we discussed the crucial need for hard-negative sampling strategies in order to fully unfold the potential of ranking-based methods. Moreover, the

large-scale datasets, which are commonly employed in unsupervised representation learning settings due to the cheaply available data pose a computational challenge to most DML approaches. Instead we follow the idea of [21] and randomly sample prototype locations $\varphi_t^k \in \Phi_k$, such that the distribution of their pairwise distances in the embedding spaces Φ_k matches the distribution of distances between the samples in $\bar{\mathcal{X}}_k$. Given an inferred assignment $t = \pi(i)$ between the datapoints x_i of groups $G \in \bar{\mathcal{X}}_k$ and the targets φ_t^k, optimizing the embedding functions ϕ_k can then be formulated as an efficient regression problem.

Sampling the targets φ_t^k. Following the practice of similarity learning, we restrict Φ_k to the surface of a D dimensional unit hypersphere. Then, we sample the targets φ_t^k from Φ_k to consider small Gaussian distributed hubs accounting for the compact groups G. More precisely, we first sample centroids $\mu_l, l = 1, \ldots, L$ uniformly from Φ_k. Next, we sample φ_t^k from the Gaussians $\mathcal{N}(\mu_l, \Sigma)$ with fixed covariance matrix Σ as illustrated in Fig. 6.6. Note that this sampling process is designed to approximate the empirical data distribution as illustrated in Fig. 6.5 (b). Using the representation of a fully supervised model as a proxy for ground-truth image relations (for this experiment only), we observe that the distribution of pairwise intra-group and inter-group distances (blue distribution) exhibits two distinctive modes as intuitively expected: A large mode representing mediocre to large inter-group distances and a minor second mode of small intra-group distances reflecting dense neighborhoods of mutually similar samples. Sampling φ_t^k based on Gaussians $\mathcal{N}(\mu_l, \Sigma)$ uniformly distributed on a hypersphere explicitly approximates this distribution (orange and yellow distributions). In contrast, other approaches such as [21] rely on a data independent prior which does not sufficiently account for dense neighbourhoods of highly similar datapoints and consequently diminishes the expressiveness of the representation to be learned.

Learning the representations ϕ_k. Learning ϕ_k is now formulated as establishing correspondences $t = \pi(i)$ between the x_i from groups $G \in \bar{\mathcal{X}}_k$ and the targets φ_t^k. This requires to minimize the distances between $\phi_k(x_i)$ and $\varphi_{\pi(i)}^k$. We obtain ϕ_k and π by minimizing the objective

$$\mathcal{L}_{\text{local}}^k := \sum_{x_i \in G:\, G \in \bar{\mathcal{X}}_k} \|\phi_k(x_i) - \varphi_{\pi(i)}^k\|_2 \,. \tag{6.2}$$

We solve this problem by alternating between two steps: *(i)* Find optimal assignments π based on the current representation ϕ_k. Since global solvers for such an assignment problem typically exhibit the prohibitive cost of $\mathcal{O}(N^3)$ in the number of input samples, we adopt the efficient algorithm of [21] which uses stochastic local updates to approximate the standard Hungarian method [133] which is often employed for solving assignment problems. *(ii)* Given an assignment, we optimize the representation ϕ_k by minimizing the distances $\|\phi_k(x_i) - \varphi_{\pi(i)}^k\|_2$, thus updating the weights θ of our embedding network. By alternating between these two steps (we re-assign between x_i and φ_t^k every 3 epochs),

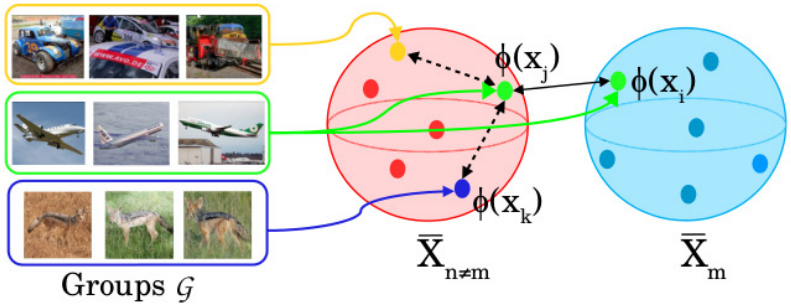


Figure 6.7: *Triplet constraints for consolidating subproblems.* A reliable relations between two subsets $\bar{\mathcal{X}}_m$ and $\bar{\mathcal{X}}_{n\neq m}$ couples them locally: solid black arrow between samples x_i and x_j which appear in the same group G (green group). Further, we find $x_k \in \bar{\mathcal{X}}_{n\neq m}$ which are reliably dissimilar to x_j and have not been used for optimizing ϕ_m (dashed black lines). We now have a triplet (x_i, x_j, x_k) relating the previously unknown x_k to the subset $\bar{\mathcal{X}}_m$.

the model needs to reason about which targets apply for which images and, thus, which images should be placed next to each other. Thus it needs to find an optimal assignment of groups G to the target points preserving their relations and further needs to infer meaningful relations between groups G. At initialization, we start with randomly initialized parameters θ and random assignments $\pi(i)$.
Using this learning process, we obtain a representation ϕ_k for each of our subsets $\bar{\mathcal{X}}_k$, each having learned its own image distances $d_{\phi_k}(x_i, x_j)$.

Coupling subproblems to consolidate their representations. We now have an ensemble of K representations ϕ_k, each representing a different subset of the data. The goal is now to consolidate their learned relationships $d_{\phi_k}(\cdot,\cdot)$ into a global representation $d_\phi(\cdot,\cdot)$ reflecting all of the data. Therefore, we look for reliable relations between different subsets to establish links that allows to locally transfer relations from one subset into the other, respectively from one representation to other. Groups G, which are (partially) shared among subsets due to their overlap and distribution to different subsets by the partitioning, act as anchors for such links as illustrated in Fig. 6.7. Using transitivity we thus find triplets (x_i, x_j, x_k) across subsets which allows to transfer information about previously unknown datapoints and the corresponding relations from one local representation to another.
Let x_i be a member of a subset $\bar{\mathcal{X}}_m$, but not of $\bar{\mathcal{X}}_n$, i.e. $x_i \in \bar{\mathcal{X}}_m \wedge x_i \notin \bar{\mathcal{X}}_n$. Similarly let $x_j \in \bar{\mathcal{X}}_n \wedge x_j \notin \bar{\mathcal{X}}_m$ and $x_k \in \bar{\mathcal{X}}_n \wedge x_k \notin \bar{\mathcal{X}}_m$. Assume $\exists G \in \mathcal{G} : x_i, x_j \in G$, thus providing a reliable similarity between x_i, x_j and consequently between both subsets. Similarly, we assume x_j, x_k to have a reliable dissimilarity. Using transitivity the triplet (x_i, x_j, x_k) thus implies an ordering constraint under the representation ϕ_m, i.e., we want $d_{\phi_m}(x_i, x_j) \overset{!}{<} d_{\phi_m}(x_i, x_k)$. Note that these are additional relations imputed from $\bar{\mathcal{X}}_{n\neq m}$, which were previously not present in $\bar{\mathcal{X}}_m$. Let $\mathcal{T}_m$ be the set of all such triplets deduced by transitivity between $\bar{\mathcal{X}}_m$ and $\bar{\mathcal{X}}_{n\neq m}$. Incorporating this additional information to refine

ϕ_m is now naturally formulated as a standard triplet ranking problem, which we already discussed e.g. by (2.8) in chapter 2. Based on this we get the transfer objective

$$\mathcal{L}^m_{\text{transfer}} = \sum_{(x_i,x_j,x_k)\in\mathcal{T}_m} \Big[d_{\phi_m}(x_i,x_j) - d_{\phi_m}(x_i,x_k) + \beta \Big]_+ \tag{6.3}$$

for a given representation ϕ_m. The parameter β controls the margin between x_j and x_k with respect to x_i. Note that the relations between $\bar{\mathcal{X}}_m$ and the other subsets are only *sparse*. Thus the potentially large computational complexity of the triplet ranking loss does not dominate the complexity of the overall approach.
However, optimizing this objective alone would ignore and potentially forget about the dense relationships in $\bar{\mathcal{X}}_m$ exploited by $\mathcal{L}^m_{\text{local}}$. Thus we combine both objectives,

$$\mathcal{L}^m_{\text{refine}} = \mathcal{L}^m_{\text{local}} + \mathcal{L}^m_{\text{transfer}}. \tag{6.4}$$

to avoid forgetting while still incorporating the new information. Optimizing (6.4) retains the constraints from $\bar{\mathcal{X}}_m$ while incorporating couplings to other subsets $\bar{\mathcal{X}}_{n\neq m}$ to improve ϕ_m. Due to the additional inter-set relations, ϕ_m effectively now covers more of $\mathcal{X}$ than before the refinement.
In the last training iteration, one representation $\phi \in \{\phi_k\}_k$ becomes the final global representation. We conducted experiments using different aggregation strategies (averaging, random selection, etc.) and observed that after the final iteration all ϕ_k capture the dataset $\mathcal{X}$ nearly equally well, allowing to randomly select one.

Initialization. As our overall iterative approach starts from random network initialization, initially we have no representation ϕ provided to extract reliable (dis)similarities for the grouping process in Sect. 6.1.3. Therefore, we train the first iteration by optimizing only the problem $\mathcal{L}_{\text{local}}$ (6.2) based on the whole training set and randomly sample individual target points for each image, yielding our first representation ϕ_{init}. The iterative approach then gradually learns stronger representations ϕ from iteration to iteration by capturing more and more reliable relationships in our dataset.

Pseudo-Code for summary. For additional clarity we now present a pseudo-code overview for our iterative approach (cf. Algorithm 3).

6.2 Related Works

Due to the difficulty of extracting reliable relations from data, many recent label-free approaches resort to generic prior assumptions on the data distribution or single image- and class-based tasks. (Deep) clustering methods [12, 30] rely on a predefined number of pseudo-classes which typically is estimated by heuristics and further focus on similarity constraints only. Our model explicitly models both similarity and dissimilarity constraints estimated from data itself. Bojanowski et al.[21] find a mapping between images

Algorithm 2: *Unsupervised Representation Learning by Extracting Reliable Image Relations.*

Input : $\mathcal{X}$: Set of training images
Parameter : T: Number of training iterations; K: Number of subsets $\bar{\mathcal{X}}_k$
Output : ϕ: global representation

Train initial representation ϕ_{init}:
$\varphi_i^{\text{init}} \sim$ uniform distribution on unit hypersphere; $\forall x_i \in \mathcal{X}$
$\phi_{\text{init}} \leftarrow \min \mathcal{L}_{\text{local}}$ `// Eq.2`
$\phi \leftarrow \phi_{\text{init}}$ `// initialize` ϕ

Iterative learning:
for $iter \leftarrow 1$ **to** T **do**
  Extract reliable image relations
  $G^h_{\text{rand}} \leftarrow$ `SampleRandomGroups(`$\mathcal{X}, \phi$`)` `// different sizes` h
  $\mathcal{G} = \{\}$
  for $i \leftarrow 1$ **to** $|\mathcal{X}|$ **do**
    $G \leftarrow$ `BuildMaximalReliableGroup(`x_i, G^h_{rand}`)`
    $\mathcal{G} \leftarrow \mathcal{G} \cup G$
  $\{\bar{\mathcal{X}}_k\}_{k=1}^K \leftarrow$ `PartitionGroups(`$\mathcal{G}, K$`)` `// Sec.3.4`

  Train local representations ϕ_k `// Sec. 3.5.2`
  for $k \leftarrow 1$ **to** K **do**
    $\mu_l^k \sim$ uniform distribution on unit hypersphere; $\forall G \in \bar{\mathcal{X}}_k$
    $\varphi_t^k \sim \mathcal{N}(\mu_l^k, \Sigma)$; $\forall x_i \in G$ `// gaussian around` μ_l^k
    while *not converged* **do**
      $\pi \leftarrow$ `LocalReassignment(`$\phi_k, \bar{\mathcal{X}}_k, \varphi_t^k$`)`
      $\theta_k \leftarrow$ `RegressTargets(`$\bar{\mathcal{X}}_k, \varphi^k_{\pi(i)}$`)`

  Refine local representations ϕ_k `// Sec. 3.5.3`
  for $k \leftarrow 1$ **to** K **do**
    $\mathcal{T}_k \leftarrow$ `SampleTriplets(`$\{\bar{\mathcal{X}}_s\}_{s=1}^K$`)`
    $\mu_l^k \sim$ uniform distribution on unit hypersphere; $\forall G \in \bar{\mathcal{X}}_k$
    $\varphi_t^k \sim \mathcal{N}(\mu_l^k, \Sigma)$; $\forall x_i \in G$ `// gaussian around` μ_l^k
    while *not converged* **do**
      $\pi \leftarrow$ `LocalReassignment(`$\phi_k, \bar{\mathcal{X}}_k, \varphi_t^k$`)`
      $\theta_k \leftarrow$ `RegressTargetsAndRefine(`$\bar{\mathcal{X}}_k, \varphi^k_{\pi(i)}, \mathcal{T}_k$`)`

  Update global ϕ by randomly choosing $k^* \in \{1, \ldots, K\}$
  $\phi \leftarrow \phi_{k^*}$

and a uniformly discretized target space, thus enforcing their representation to resemble a distribution of pairwise relationships independent of the actual data structure. Sanakoyeu et al. [12] cluster data into small surrogate classes to perform a global classification task, however, not considering that similar images may end up in competing, different classes. Thus, inferred relationships during training suffer from contradicting training signals.

	Method	Acc@1
RGB input	Supervised [21]	59.7
	Random [166]	12.0
	Colorization [272]	35.2
	Jigsaw Puzzles [166]	38.1
	BiGAN [55]	32.2
	RotNet [72]	43.8
gradient (sobel) input	Supervised [21]	57.4
	NAT [21]	36.0
	Deep Cluster [30]	44.0
	Ours (Initialization)	30.0
	Ours (Round 1)	39.1
	Ours (Round 2)	44.6
	Ours (Round 3)	45.8
	Ours (Round 4)	**46.0**
	Ours (mean±std)	45.8±0.3

Table 6.1: *Comparison of our method to other state of the art unsupervised learning approaches on the ImageNet dataset.* We report classification accuracy (*Acc*@1).

Also DeepCluster [30] follows this strategy based on disjoint k-means clustering, thus enforcing clear distinct boundaries which potentially disagree with the real data distribution. In contrast, our grouping process does not enforce hard class boundaries and is able to adapt to the data structure. Moreover by splitting groups into reliable subproblems and constructing a learning problem following their distance distribution, our groups corroborate their training signal. Dosovitski et al. [57] cast distance learning as an exemplar classification task utilizing heavy data augmentations at the cost of poor scalability to large data collections.
Self–supervised learning approaches aim to leverage data itself by typically solving surrogate tasks based on temporal [248] and spatial [166] coherence. These approaches are either domain specific or operate on images independently thus missing out on their relationships. Our work, in contrast, explicitly models relationships between images. Gidaris et al. [72] exploit image geometry and classify rotations applied to input images. Even though they report good results on image classification tasks on large datasets, this task is conceptually dependent on large variations in the underlying data distribution to avoid trivial image representation, potentially missing out fine-grained relationships between images.

6.3 Experiments

Following we evaluate the performance of our model on large scale datasets and the usability of our learned representation for the tasks of classification, detection and segmen-

Method	Class (%mAP)	Det (%mAP)	Seg (%mIoU)
Supervised	79.9	56.8	48.0
Random	57.0	44.5	30.1
Colorization [272]	65.6	46.9	35.6
BIGAN [55]	60.1	46.9	34.9
Jigsaw Puzzle [166]	67.6	53.2	37.6
NAT [21]	65.3	49.4	-
Split-Brain [271]	67.1	46.7	36.0
Counting [167]	67.7	51.4	36.6
RotationNet [72]	73.0	54.4	39.1
DeepCluster [30]	73.7	55.4	**45.1**
Ours	**74.2**	**55.6**	44.6
Ours (mean±std)	74.1±0.3	55.5±0.2	44.5±0.3

Table 6.2: *Comparing our model to state-of-the-art unsupervised approaches on PASCAL VOC 2007 classification, detection, and segmentation (measured in mean average precision and mean intersection over union.*

tation. Further, we present ablation and analysis experiments providing insights into our iterative learning process.

6.3.1 Implementation details and benchmarking

We now evaluate our learned image representation on the ImageNet and PASCAL VOC dataset. We test on different vision tasks, such as image classification, objection detection, and semantic segmentation and compare our approach against state-of-the-art unsupervised methods. As preprocessing we convert our images to gradient images (obtained using a sobel filter) to avoid trivial solutions based on color, thus following the protocol of recent approaches [21, 30]. These also conducted experiments on supervised ImageNet classification and concluded that gradient images yield similar performance in comparison to RGB inputs, thus indicating a fair comparison. If not stated otherwise, for all experiments the number of subsets $\bar{\mathcal{X}}_k$ used for training is fixed to $K = 5$. In the grouping stage we set p to be the 3% percentile. We set the dimensionality of ϕ_k to $D = 2048$ and choose the parameters $\lambda_1, \lambda_2, \Sigma$ using cross-validation on the training set. The margin parameter γ is set to 0.2 as suggested in various works [256]. We train our model using stochastic gradient descent using an initial learning rate of 0.01 and momentum of 0.9. Building groups can be efficiently performed using the FAISS [109] library for fast nearest-neighbor retrieval with GPU support, thus leading to no significant computational overhead.

ImageNet. We evaluate the ability of our model to capture differences between objects and its ability to scale to large image collections on the ImageNet dataset [50]. This dataset is composed of 1.2M images distributed over 1,000 categories including subtle category

Method	Acc@1
Target coding [263] (supervised)	73.2
Wang et al. [242]	68.2
CliqueCNN [12]	69.3
Exemplar CNN [57]	75.4
Discr. Attr. [101]	**76.8**
Chang et al. [31]	74.6
Ours	75.3

Table 6.3: *Comparison to other approaches based on average classification accuracy on STL-10, using the unlabeled split for training.*

boundaries (such as dog breeds). For fair comparison with other methods we use the AlexNet [**alexnet**] architecture with batch normalization. Our evaluation follows the standard protocol for unsupervised ImageNet pretraining: First, performing our unsupervised training on a randomly initialized network without using any labels. Afterwards, the convolutional layers are fixed while the last layers are randomly reinitialized and trained using ImageNet labels. Table 6.1 compares our results with other state-of-the-art unsupervised approaches. Our method converges to its final performance of 46.0% after 4 training iterations (excluding initialization). Hence, we are significantly improving upon all other unsupervised approaches including DeepCluster [30] by 2%, which is trained on all training data at the cost of also incorporating unreliable noisy information. In contrast, our model successfully leverages more and more reliable image relations over the iterations, thus alleviating the issue of noise. Additionally we report mean and standard deviation of our approach (Ours (mean$\pm$std)) over 5 runs. Note that all of the other methods are only reporting their best run.

PASCAL VOC. We now illustrate the generalization capability of our learned representation on different transfer learning tasks. We utilize our representation trained on ImageNet without labels (using the same architecture as above) and fine tune it on the PASCAL VOC 2007 [61] classification, detection, and segmentation tasks (VOC 2012). For transfer learning we use the framework of Krähenbühl et al. [129] for classification experiments, the Fast R–CNN [74] framework for object detection and the method of Long et al. [210] for semantic segmentation. Our results are summarized in Table 6.2. On all transfer learning tasks, i.e. classification, detection and segmentation, our approach achieves comparable results to the unsupervised state-of-the-art which further demonstrates the expressive power of our representation. Additionally we report mean and standard deviation of our approach (Ours (mean$\pm$std)) over 5 runs. Note that all of the other methods are only reporting their best run.

Method	Acc@1
Ours (Initialization)	67.5
Ours (Round 1)	71.2
Ours (Round 2)	72.8
Ours (Round 3)	75.2
Ours (Round 4)	**75.3**
No decomposition into $\bar{\mathcal{X}}_k$ (Round 1)	69.1
Triplets Only (Round 1)	62.6

Table 6.4: *Average classification accuracy on STL-10 for ablations of our approach.*

6.3.2 Ablation studies

We now show ablation experiments on the STL-10 dataset to evaluate the individual parts of our iterative approach, summarized in Tab. 6.3 and 6.4.

STL-10 Performance. We contrast our approach to state-of-the-art methods on STL. For a fair comparison, we use the same network architecture as [57] and train our model on the unlabeled split of the dataset. For this experiment we set $D = 128$. Tab. 6.3 shows that our proposed approach achieves competitive performance to unsupervised state-of-the-art approaches, ExemplarCNN [57], Discriminative Attributes [101] and Chang et al. [31]. However, in contrast to ExemplarCNN [57] and Discriminative Attributes [101] whose learning procedures rely on dense, costly (instance-level-)exemplar classification tasks, our approach leverages only a sparse set of reliable constraints and thus scales to large datasets as shown on ImageNet. Chang et al. [31] leverages individual images in combination with strong augmentations, thus neglecting valuable information from direct relations between training samples. Further, they are operating on a more powerful network architecture leading to unfair comparison in their favor. Note that our approach outperforms CliqueCNN [12] by 6% which also learns from relations inferred by a grouping of images, however, without considering their reliability. This strongly indicates that the concept of reliability actually helps to reduce the amount of noise introduced into the learning process.

Divide-and-Conquer. The performance of our representation increases with each iteration and improves over our initial representation by 7.8%, cf. Tab. 6.4. Now, to evaluate the effect of incorporating reliable dissimilarities, we train our model using only a single, global learning problem, $K = 1$, (No decomposition) for all groups for one iteration after initialization. As a result there are only reliable relations within the $G \in \mathcal{G}$, but lots of unreliable relationships between them. Due to the former, performance slightly increases over the initialization by 1.6%. The latter explains the 2.1% lower performance compared to our divide-and-conquer strategy. This highlights the importance of modelling reliable dissimilarities when learning visual representations.

Number of Subsets $\bar{\mathcal{X}}_k$	Acc@1
no partitioning	69.1
2 subsets	69.5
5 subsets	70.8
10 subsets	71.2
20 subsets	71.0

Table 6.5: *Average classification accuracy on STL-10 dataset as number of subsets K is varied.* Results are for one iteration of training after initialization.

Full model vs. triplet learning. In this ablation experiment (Tab.6.4 Triplets Only) we use our extracted reliable (dis-)similarity relationships to mine triplet-constraints as input into a standard triplet-loss framework [256]. For a sample that serves as triplet anchor, reliable similarity relations act as positive constraints and reliable dissimilarity relations act as negative constraints. The aim of this experiment is to contrast our learning objective, Eq. (4), against popular ranking loss approaches. The massive drop of 8.6% in performance can be explained by the dependence of such frameworks on hard-negative mining strategies. Since reliable constraints are only based on high similarities and dissimilarities, the ranking framework obviously has no access to hard constraints, which are very difficult to find reliably *without* supervision [256] or strong pretraining [188]. Note that we are using triplet constraints only for transferring already learned information to refine our representations ϕ_k (Sec. 6.1.5) rather than using them as the driving overall learning signal.

Number of subsets. To evaluate the sensitivity of our approach with respect to the number of subsets $\bar{\mathcal{X}}_k$ used for training, we train multiple models using different values of K. Tab. 6.5 illustrates, that training performance saturates for $K > 10$ subsets. This indicates that further partitioning of the data and thus further maximizing the dissimilarity between groups in $\bar{\mathcal{X}}_k$ has no effect and has been achieved to a sufficient degree.

6.3.3 Analysis of the model:

We now further analyze the effect of iterative training based on the model used for the ImageNet benchmark. The following experiments highlight the ability of our model to turn reliable (dis-)similarities into increasingly correct sample relations while simultaneously harnessing more and more data.

Correctness of image relations in groups G. In Sec. 6.1.3 and Sec. 6.1.4 we gather reliable relations driving the learning of our representation. Fig. 6.5 (a) illustrates that our procedure of extracting groups G for reliable relations increases the probability of finding correct relationships. Moreover, successive iterations of training improve the learned image relations. Fig. 6.8 confirms this by measuring how many relations within a group disagree. One can see that in consecutive iterations, our groups G exhibit more and more

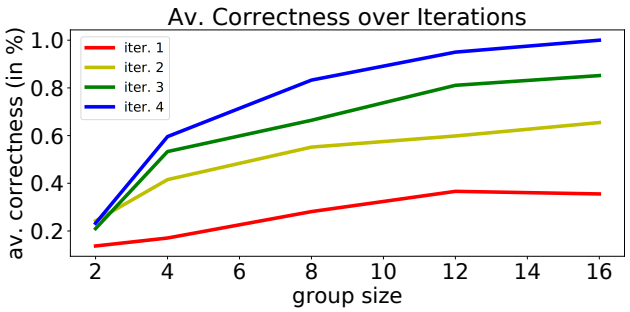


Figure 6.8: *Average correctness for groups G of different size and in different training iterations.* Average correctness is the fraction of members with the same ground-truth ImageNet class label (not used for training).

correct relationships. This proves that our model is able to extend the reliable relations learned from a group G to relations which so far could not be identified as reliable. Consequently the performance of our model increases.

Data coverage. In our proposed method we deliberately explore an orthogonal approach to simply using all training samples by reducing the noise introduced into the training process using only reliable relations for learning. Consequently there is a trade-off between exploring more data and introducing more noise due to unreliable relations. Fig. 6.9 shows the amount of training data covered in each training iteration. Overall and per subset, our grouping process covers more and more data samples, since representations improve and more relations become reliable. Between iteration 1 and iteration 4 the amount of data available for training ϕ is almost doubled to 60%. Moreover, Fig. 6.8 proves that the additional data is meaningful and not just noise, since overall data quality is improving simultaneously. This demonstrates that our model not only reinforces already available reliable relations but is able to generalize its representation to previously unused data, i.e., discovering new reliable relationships. Further, the result that we achieve state-of-the-art performance using 60% of the available data proves *(i)* there exists an alternative way to using all but therefore noisy relations between training samples, *(ii)* we are able to successfully find and exploit reliable samples and *(iii)* the idea of considering data reliability for unsupervised representation learning is not fully solved, thus opening new promising directions for future research.

Fixing the seed of groups G. In Fig. 6.10 and Fig. 6.11 we present examples for groups of size 4 and 8 while training on the ImageNet dataset. To allow for a detailed comparison, we fix the seed elements x_i and show how the constituents of a group change over the iterations. We observe that over the iterations the constituents of each group share more and more meaningful visual features. At the beginning the representation focuses on coarse visual commonalities like rough shape and scene. At the end relationships between constituents are dominated by true (intra-)class-specific features and similar pose.

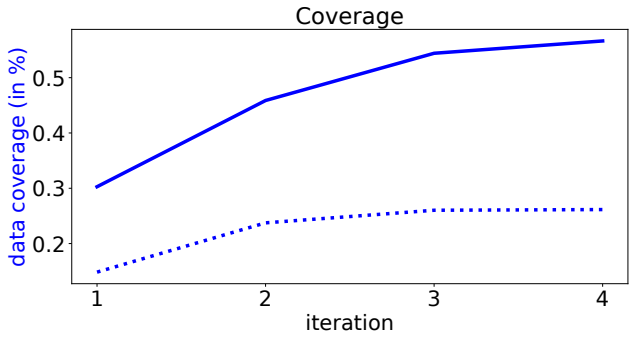


Figure 6.9: *Fraction of training data covered by subsets $\bar{\mathcal{X}}_k$ in successive training iterations.* Solid: overall data coverage. Dashed: mean data coverage per subset.

Method	Training time	Acc@1
Supervised	3 days (Titan X)	59.7
NAT [21]	1 days (Titan X)	36.0
RotNet [72]	2 days (Titan X)	43.8
Jigsaw Puzzle [166]	3 days (Titan X)	38.1
BiGAN [55]	3 days (Titan X)	32.2
DeepCluster [30]	12 days (Titan X)	44.0
Ours	14 days (Titan X)	46.0

Table 6.6: *Training time vs. performance on the ImageNet dataset.* Comparison of training times is based on the reported timings in the manuscripts of each method using a single NVIDIA Titan X (Pascal). Additionally, their performance for Acc@1 on ImageNet is reported.

Typical sources of incorrectness. To explain common sources of incorrect relationships within groups, i.e., group constituents having different labels, we show typical examples in Fig. 6.12. As one can see, disagreements often arise due to subtle differences between the constituents' classes (such as different dog breeds, hedgehogs vs. sea urchins, etc.) and misleading scene settings (such as a buildup of flags imitating a ship's shape, a dog's head above the surface while swimming looking like a duck, etc.).

Analysis of computational cost. As deriving exact computational complexities is often difficult, it is common practice for deep learning based methods to compare their computational complexities based on their training times. For our model learning one representation ϕ_k (including the refinement step) for a subset $\bar{\mathcal{X}}_k$ takes approx. 16h on a Titan X (Pascal), resulting in 14 days of training in total for the Imagenet dataset. Thus, our overall training time is comparable to the current state-of-the-art approach DeepCluster [30] (12 days). Further, Tab. 6.6 compares the computational cost of our method with recent unsupervised representation learning approaches in relation to their performance. As we observe, peak performances on Imagenet are computationally costly. Further, we

observe that RotNet [72] offers the best trade-off between training efficiency and performance.

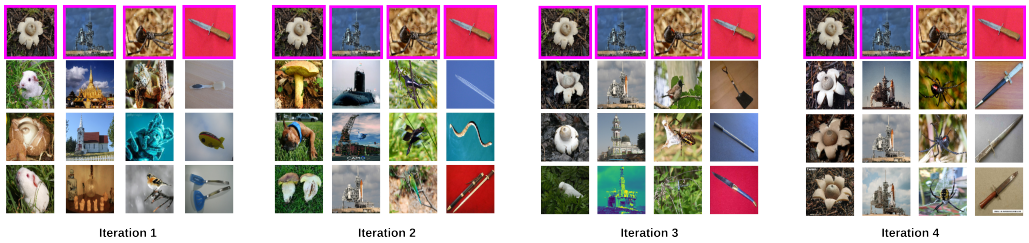


Figure 6.10: *Groups of size* 4 *for fixed seed image (pink) over the iterations while training the ImageNet model.* Each column represents a group G.

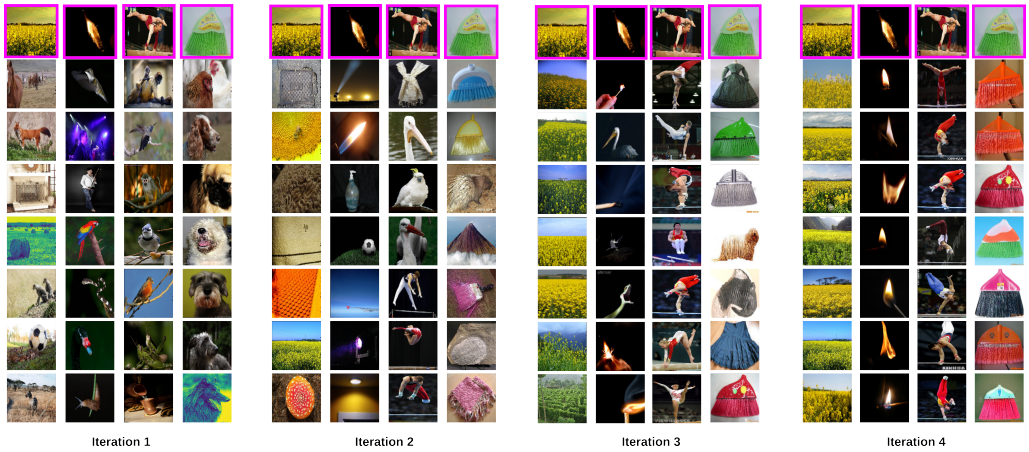


Figure 6.11: *Groups of size* 8 *for fixed seed images (pink) over the iterations while training the ImageNet model.* Each column represents a group G.

Figure 6.12: *Examples of groups with incorrect constituent relationships (i.e. different labels) taken from the last iteration while training on ImageNet.* Orange highlights the source of disagreement. Each column represents a group G.

6.4 Discussion:

In this chapter we presented a novel iterative approach for unsupervised learning of visual feature representations. Experimental evaluation shows that our method yields a representation of competitive or even superior performance on the tasks of unsupervised classification and transfer learning compared to the state-of-the-art. We propose a novel technique for finding reliable relations (i.e. dis-/similarities) between training images which are likely to agree with the ground-truth. This reduces the amount of noise due to erroneous relations introduced into training, which is typically an issue when all possible training data relations are directly used. As using all relations is typically common practice in the vast majority of unsupervised learning literature, our work offers a new direction of future research directions: Instead of solely looking for more powerful surrogate tasks to compensate for missing supervision, we also address the question which training samples and relations can be trusted and which are likely to obstruct learning. This question naturally leads to a trade-off between covering all available training data and exploiting only reliable relations for learning. Hence, future work should focus on further improving the estimation of reliability to increase data coverage while maintaining a high reliability for efficient learning.
We presented a technique for identifying reliable relations resulting in a set of local subproblems. For each subproblem the data samples are either reliably similar or reliably dissimilar. Further, the learning process for each subproblem is formulated to preserve their reliable relations and approximate the actual distribution of distances of the training data. This stands in contrast to previous approaches whose feature space relies on data-independent prior assumptions which potentially disagree with the training data at hand. We then optimize each problem individually before using transitivity relations between them to efficiently merge their learned local representations into a single concerted representation.

7 Learning Continuous Posture Similarities for Unsupervised Video Understanding

The ability to understand human actions is of cardinal importance for our interaction with another. Explaining the activity of another person by observing only individual postures and their temporal transitions in a sequence of video frames has been a long-standing challenge in computer vision. There are numerous applications in problems like activity indexing and search [139, 200, 216], action prediction [115, 236], behavior understanding and transfer [173, 178], abnormality detection [8], and action synthesis and video generation [7, 60, 197, 237].
In this chapter, our goal is to apply similarity learning [205, 256] to learn a representation for human activity based on the finest accessible level, i.e. individual human poses. Hence, our model needs to both capture the characteristic postures and the distinctive transitions between them. Moreover, since providing manual supervision information indicating the similarity between given postures is not feasible, we need to infer such supervision automatically during training. In short,our approach to activity representation and understanding, summarized in Fig. 7.1 (also see video material[1]), entails the following characteristics:

- *Unsupervised*: The most prominent paradigm to video understanding has been supervised action classification [115], since labeling finer entities such as individual poses [3] is tedious. Action classification typically utilizes holistic models and a discriminative approach is trained to classify actions into discrete classes. As a result, such approaches model actions globally in terms of their overall most salient differences, rather than capturing the subtle changes of human posture over time (the clothing of a person may suffice to discriminate running from diving).

- *Model-free*: Multiple works have modeled activity and posture using a predefined model for joint locations (i.e. MoCap [114, 213, 278], depth [191], etc.). However, obtaining this meta information is costly and prevents scaling these approaches to use large unlabeled collections of video data.

- *Continuous in time*: Several approaches have tackled the problem of understanding activity by decomposing it into discrete sub-actions [137, 240, 273] or into a hierarchy

[1]Video material demonstrating the different applications can be found under https://hciweb.iwr.uni-heidelberg.de/compvis/research/tmilbich_iccv17

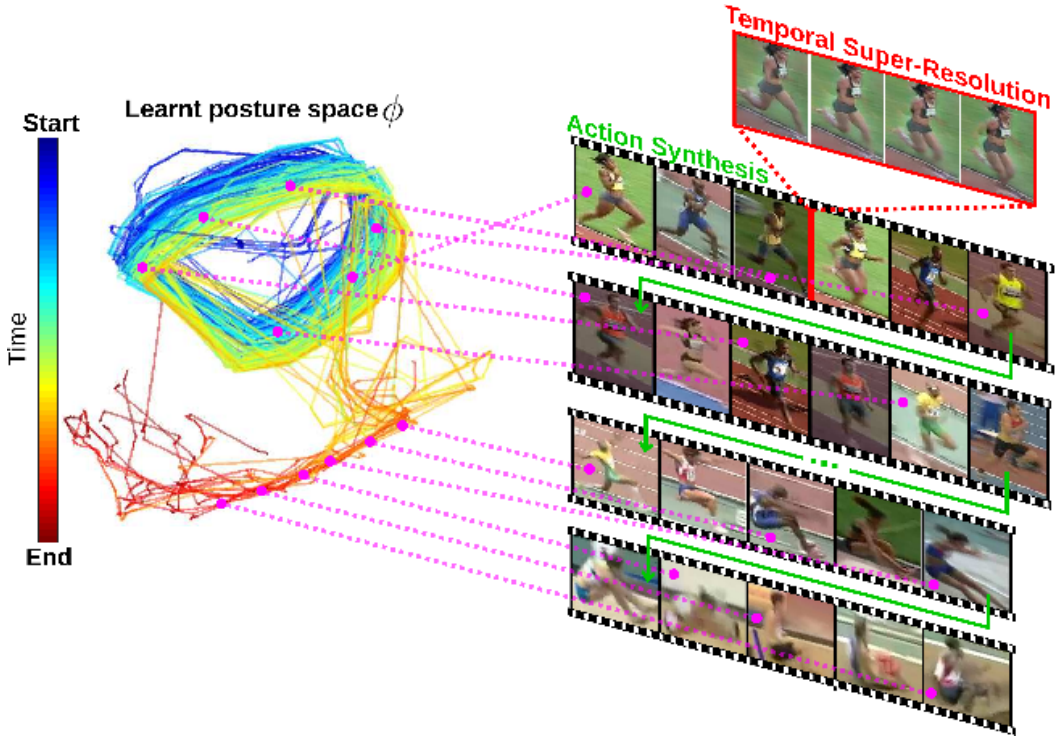


Figure 7.1: *Visualizing all frames of all long-jump sequences using the learnt posture representation ϕ.* Similar frames across different sequences and repetitions across time are mapped nearby, yielding a concise rendering of the overall activity with its characteristic gait cycles. Moreover, intermediate frames can be synthesized (top right) as well as future frames of a sequence (visualized by nearest neighbors from training set (right)).

[219, 278]. Consequently, the detailed, continuous evolution between consecutive postures is neglected.

- *Multi-granular*: A lot of work has focussed separately either on pose matching [12, 148, 229], action classification [5, 63, 115, 244], or mid-level entities (e.g. clusters of postures) [137, 219, 273]. Explaining activities, however, demands to describe overall activity based on fine-granular postures and the transitions in between, thus linking coarse with fine granularities.

- *Fine-grained activity parsing*: Explaining activity on the temporal scale of single postures with all their diverse changes is far more detailed and complex than mere action classification [63, 244]. Previous efforts [83] have approximated posture and its transitions using discrete states in an AND-OR graph and relied on tedious supervision information.

With no supervision information, no predefined model for human posture, and training from scratch, we need to compensate for there being no labels for individual postures. Therefore, we utilize a large number of training video frames and have our learning

framework alternate between proposing pairwise similarities/dissimilarities among postures and resolving transitivity conflicts to bring the relationships into mutual agreement. In contrast to the presented unsupervised framework in chapter 6, we can now exploit the temporal information inherent in the video domain as an inductive bias for inferring our similarity constraints. To propose similarities, a combinatorial sequence matching algorithm is used which can find optimal solutions, but only for small sets of frames. We then resolve the transitivity conflicts between these different subsets of the training data by learning a posture embedding that reconciles the similarity constraints from the different subsets. While the sequence matching is already incorporating information about the natural order of postures in video, a Recurrent Neural Network (RNN) is trained to capture the overall activity and to predict future frames of an activity sequence by synthesizing transitions.
Our experimental results show that our approach is able to successfully explain an activity by understanding how posture continuously changes over time and to model the temporal relationships between postures. Furthermore, our posture representation obtains state-of-the-art results on the problem of zero-shot human pose estimation. Also for the classical application of transfer learning, our model proves worth as a powerful initialization for other supervised human pose estimation methods. In addition, our approach captures the temporal progression of an activity, it can predict future frames, and it also enables a Generative Adversarial Network (GAN) [80] to provide temporal super-resolution.
This chapter is based on our publication *'Unsupervised Video Understanding by Reconciliation of Posture Similarities'* [154].

7.1 Representation Learning for Parsing Activities

We are interested in detailed understanding of human activity without requiring manual interaction or predefined models. Therefore, we can explain overall activity in a video only using the most basic entity that we can directly access: the human posture observed in individual frames $x \in \mathbb{R}^{D'}$. Activity, which emerges at the temporal scale of an entire video sequence, is then represented by individual poses and their characteristic transitions and repetitions on a fine temporal scale. Hence, to model an activity, we learn a posture representation $\phi : \mathbb{R}^{D'} \rightarrow \mathbb{R}^{D}$ which: *(i)* is invariant to changes in environmental conditions such as lighting and background. *(ii)* is invariant to the appearance of persons (clothing and skin color). *(iii)* is continuous in time (consecutive frames are near in feature space). Only then we can understand the essential characteristics of an activity and spot all repetitions of the same pose over time and in different sequences, despite changes in person appearance or environment.
As seen in the previous chapters, similarity, respectively metric learning constitutes an effective way to learn an embedding $\phi(x;\theta)$ with trainable parameters θ. Such approaches exhibit great expressive power to learn highly non-linear representations, however, typically at the cost of requiring lots of manually labeled samples for training. A popular alternative is then to only fine-tune a representation that has been pre-trained for discrete classification on large datasets, such as ImageNet [50]. Unfortunately, the performance

of pre-trained models for transfer learning is heavily task dependent, i.e. how closely the pre-train tasks resembles the target task (cf. Sec. 1.3). In our scenario, the most common pre-train task, discrete classification objective, contradicts our requirements for ϕ, i.e., a discrete classification loss neglects smoothness within the representation space ϕ. In addition, since datasets like ImageNet [50] are composed of single images, pre-trained models fail to encode temporal relationships between frames. This explains the inferior performance of these pre-trained models in Sec. 7.2.
Instead of using a pre-trained representation we seek to learn a representation $\phi(x;\theta)$ that maps similar postures close in feature space while retaining temporal structure of an activity. Ideally, manual supervision of human postures, such as joint annotations, and/or positive links of similar postures within and across sequences together with negative links of dissimilar postures could be used to learn $\phi(x;\theta)$ employing, for example, triplets of similar and dissimilar poses. However, we are lacking these labels, altogether. To overcome this lack of labels we exploit the relationships inherent in large collections of video sequences. We infer the supervision information, which is required to learn $\phi(x;\theta)$, by solving a combinatorial sequence matching problem. The solution of this matching problem then provides us with correspondences of similar and dissimilar postures, which we then impose onto the representation $\phi(x;\theta)$ to learn it.

7.1.1 Sequence matching for self-supervision

We aim to learn an embedding representation $\phi(x;\theta)$ which encodes posture similarity, without being provided with any labels. In order to learn such a representation, we employ a self-supervision strategy, leveraging the temporal information in videos to solve a sequence matching problem and find pair-wise correspondences between frames on a sequence level. Let $\mathcal{S} = (x_j)_{j=1}^{n}$ and $\mathcal{S}' = (x'_{j'})_{j'=1}^{n'}$ denote two sequences of n and n' frames respectively. We want to find a correspondence $\pi : \{1, ..., n\} \mapsto \{0, 1, ..., n'\}$ that matches frames of $\mathcal{S}$ to frames of $\mathcal{S}'$, where the index 0 is used to match outliers. Furthermore, in order to successfully learn $\phi(x;\theta)$ using self-supervision, we want to enforce the following constraints on π: *(i)* Corresponding frames should be similar in appearance. *(ii)* To avoid temporal cross-over and to reduce false positives matches, consecutive correspondences must be chronologically ordered. *(iii)* To avoid only part of a sequence being used and to explore the full span of possible postures therein, one-to-many correspondences should be penalized. *(iv)* Correspondences should be invariant to the sequence frame rate. As we see, most matching constraints are based on temporal information, whose need is demonstrated in Fig. 7.2 and 7.3. These constraints prevent, by definition, to utilize classical sequence matching approaches like the computational costly String Matching

[124] or Dynamic Time Warping [17, 116]. Thus, we follow Buechler et al. [6] and define the following optimization problem which combines all these constraints,

$$\begin{aligned}\min_{\pi:\{1,\dots,n\}\mapsto\{0,1,\dots,n'\}} & \sum_{j=1}^{n}\left\|\phi(x_j;\theta)-\phi(x'_{\pi(j)};\theta)\right\|_2^2 \\ & + \lambda_1\sum_{j=1}^{n-1}\mathbf{1}_{\pi(j)>\pi(j+1)} \\ & + \lambda_2\sum_{j=1}^{n-1}\mathbf{1}_{\pi(j)=\pi(j+1)} \\ & + \lambda_3\sum_{j=1}^{n-1}\mathbf{1}_{\pi(j)+1<\pi(j+1)}[\pi(j+1)-\pi(j)] \end{aligned} \tag{7.1}$$

where $\mathbf{1}_\bullet$ denotes the indicator function and $\lambda_1, \lambda_2, \lambda_3$ penalize the violations of the different temporal constraints. The use of inequalities ensures constraint *(iv)*. Similar to [6], we can convert the optimization problem in Eq. (7.1) into an Integer Linear Program (ILP). In order to do so, we define a matrix $\boldsymbol{Z} \in \{0,1\}^{n\times n'\times n'}$, where $z_{j,j'_1,j'_2} \triangleq \mathbf{1}_{\pi(j)=j'_1\wedge\pi(j+1)=j'_2}$. A non-zero z indicates matches for two consecutive frames starting at position j. The ILP is then

$$\begin{aligned}\max_{\boldsymbol{Z}\in\{0,1\}^{n\times n'\times n'}} & \sum_{j=1}^{n-1}\sum_{j'_1,j'_2=0}^{n'} z_{j,j'_1,j'_2}p_{j,j'_1,j'_2} \\ \text{subject to } & \sum_{j'_1,j'_2=0}^{n'} z_{j,j'_1,j'_2} = 1 \\ & \sum_{j'_1=0}^{n'} z_{j,j'_1,j'_2} = \sum_{j'_3=0}^{n'} z_{j+1,j'_2,j'_3} \end{aligned} \tag{7.2}$$

where p_{j,j'_1,j'_2} is the sum of all terms in Eq. (7.1) with $z_{j,j'_1,j'_2} = 1$. To obtain reliable self-supervision information, we obtain an exact solution of this ILP using a branch-and-cut algorithm [176] following [6]. However, exactly solving this problem for pairs of long sequences (e.g. $n > 500$) is a costly operation with exponential worst case complexity in n, making it computationally infeasible. To circumvent this high cost when matching $\mathcal{S}$ onto $\mathcal{S}'$, we break the target sequence $\mathcal{S}'$ into k equal length sub-sequences of length $n' \approx 40$ and find an exact solution for Eq. (7.2) on each local sub-sequence in parallel. Thus, the overall computational cost is reduced by a factor of k, making it a feasible to tackle long sequences. However we obtain only local correspondences at sub-sequence level and thus, discarding important relationships between different sub-sequences that compose the overall activity. To compensate for this shortcoming, we optimize $\phi(x;\theta)$ with the different, local sub-sequence solutions in subsequent mini-batches as discussed in

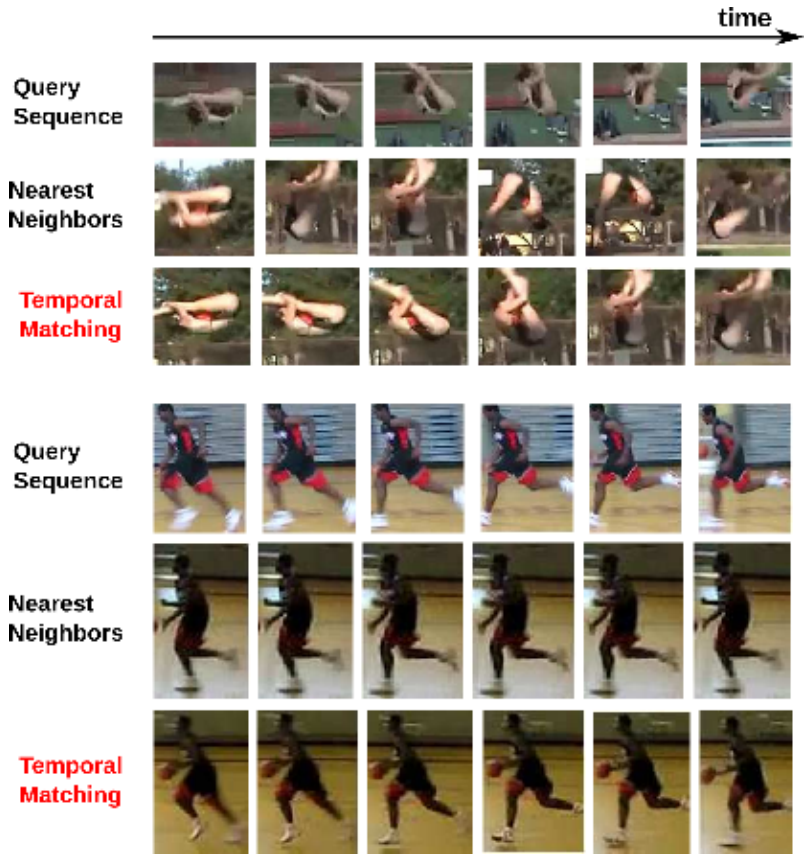


Figure 7.2: *Correspondences for two query sequences obtained using standard nearest neighbors (first row) and our sequence matching with temporal constraints (second row).* Note how the temporal constraints provide much more accurate correspondences.

Sec. 7.1.2. Our model then reconciles the local sub-sequence correspondences. Thus, we benefit from combining the computational feasibility of exact local sub-sequence matching with the power of stochastic gradient decent optimization, which aggregates lots of local observations in one concerted representation ϕ.

7.1.2 From local correspondences to a globally consistent posture representation

We now learn a joint representation $\phi(x;\theta)$ that reconciles all the sub-problems. Given proper learning constraints, we can utilize powerful similarity learning frameworks, such as ranking-based losses (cf. Sec. 2.2.1).
To provide the required supervision information to learn a representation for encoding human activity in a fully unsupervised manner, the mini-batches for training are composed just by pairs of sequences $\{\mathcal{S}, \mathcal{S}'\}$. We find these pairs by first randomly choosing $\mathcal{S}$ and sampling $\mathcal{S}'$ from a set of nearest neighbour sequences $\mathcal{S}_{\text{nn}}$ to $\mathcal{S}$, thus sorting out totally unrelated sequences. $\mathcal{S}_{\text{nn}}$ is constructed using simple sequence descriptors by tem-

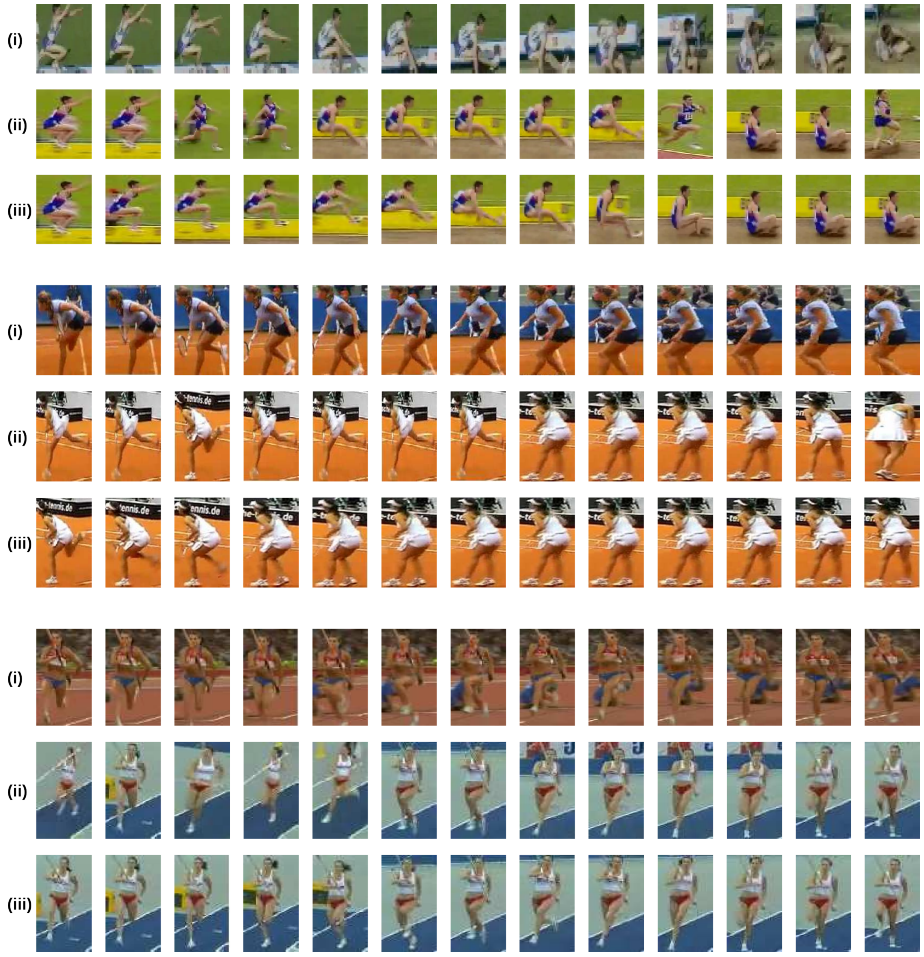


Figure 7.3: *Correspondences for query sequences of three different categories (rows (i)).* Assignments are obtained using standard nearest neighbors (rows (ii)) and our sequence matching with temporal constraints (rows (iii)). Note how the temporal constraints provide much more accurate correspondences of posture.

porally pooling similarities over all frames of a video. After breaking $\mathcal{S}'$ into equal-length sub-sequences, the ILP in Eq. (7.2) yields a solution $\boldsymbol{Z}^*$. These are exact pair-wise correspondences between $\mathcal{S}$ and a particular sub-sequence of $\mathcal{S}'$. We then use these correspondences $\boldsymbol{Z}^*$ to generate triplets $t = (x_i, x_j, x_k) \in \mathcal{T}$. Here, (x_i, x_j, x_k) consists of a randomly sampled anchor image $x_i \in \mathcal{S}$ and its positive correspondence $x_j = \pi(x_i) \in \mathcal{S}'$, together with a randomly chosen negative $x_k \in \mathcal{S}'$. We randomly sample negatives based on the q-th percentile of the similarity distribution of sequence $\mathcal{S}'$ to x_j. That is, we compute the similarity of x_k to each frame of $\mathcal{S}'$, and sample negatives from frames with a lower similarity than the q-th percentile. We thus include increasingly hard negatives by decreasing q over epochs. Note that by sampling positives and negatives from the same sequence and by comparing the same sequence with different other sequences, relationships within sequences are also implicitly established. Using this triplet self-supervision

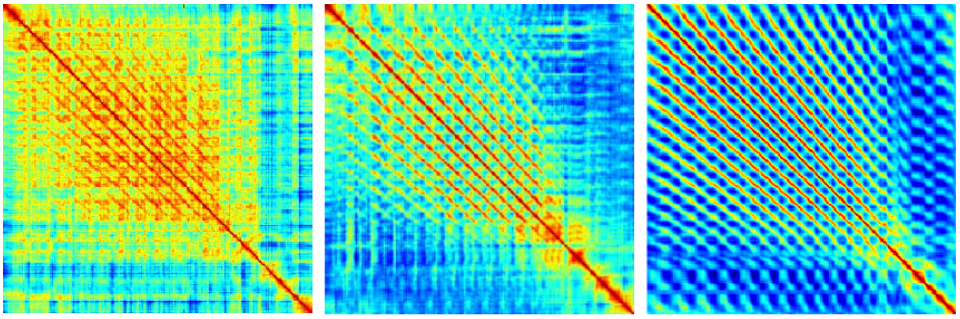


Figure 7.4: *Similarity matrices for a long jump sequence of Olympic Sports dataset:* computed using (a) VGG-S pre-trained on Imagenet, (b) CliqueCNN [12], (c) Ours. The diagonal structures correspond to repetitions of gait cycles during running.

we update the model parameter θ of $\phi(x;\theta)$ via back-propagation and the triplet ranking loss [205, 247],

$$\mathcal{L}(\mathcal{T};\theta)\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\mathcal{L}'(t;\theta) \tag{7.3}$$

$$\mathcal{L}'(t;\theta) = \left[\|\phi(x_i;\theta)-\phi(x_j;\theta)\|_2 - \|\phi(x_i;\theta)-\phi(x_k;\theta)\|_2 + \beta\right]_+ , \tag{7.4}$$

with β controlling the margin between x_j and x_k with respect to x_i. The matching algorithm from Eq. (7.2) provides the self-supervision information needed to train the representation ϕ, whereas the training using Eq. (7.3) yields the posture embedding required to compute the similarities in Eq. (7.2). Alg. 3 outlines this iterative procedure. We found that using HOG-LDA [87] to initialize ϕ for the first epoch provides a decent initialization leading to a speed-up compared to a random ϕ. Learning a single posture representation that captures characteristic similarities across and within sequences is thus decomposed into a series of mini-batch optimizations. For each, Eq. (7.2) provides a matching that is locally, within the respective sub-sequences, optimal. The stochastic optimization of the CNN then consolidates, over a number of mini-batches, the transitivity conflicts between the local solutions to arrive at a single posture representation. That way, both approaches combine their strengths and weaknesses in an ideal manner. Fig. 7.4(a-c) show an excerpt of similarity matrices from different models. Note the significantly improved signal-to-noise ratio in (c).

7.1.3 Recurrent neural networks for learning temporal transitions

We now have an algorithm which effectively learns an overall posture representation $\phi(x;\theta)$ for an activity based on relationships within and across sequences. This detailed posture encoding is an ideal basis to also deal with coarser temporal scales, enabling to learn the complete temporal structure of an activity without requiring any prior model and to synthesize future frames of a sequence. As ϕ maps into the continuous representa-

tion space Φ, also the transitions between the embeddings of consecutive sequence frames will be continuous. To explain or synthesize overall activity on basis of the progression of a video sequence in Φ, Recurrent Neural Network (RNN) are an ideal model class. RNNs are recurrent functions which naturally lend themselves to process and encode inherently temporally ordered data points. Most RNNs use an internal, hidden state vector $h \in \mathbb{R}^{D''}$ with dimensionality $D'' \gg D$ to represent the current state of the already processed temporal sequence and to effectively learn the variability of the individual temporal transitions (cf. Sect. 7.2). By recursively applying the RNN to a given video sequence $(x_1, \ldots, x_s, \ldots, x_S)$, the hidden h_s state at time step s typically gets updated by the general form

$$h_s = f_a(f_b(x_s) + f_c(h_{s-1})) \ , \tag{7.5}$$

with f_a, f_b, f_c being some functions to incorporate the information in x_s to the state h_s. Due to significant computational costs, using RNNs on video data requires a meaningful image representation. Here, we can use the structured representation that we learn in Sect. 7.1.2. In this work, we incarnate our RNN as a widely used LSTM (Long-Short-Term-Memory) [97]. In order for the LSTM to learn an activity with all its temporal transition between individual postures, we formulate a regression task for future embedding prediction based on our representation $\phi(x; \theta)$. To successfully predict the embeddings of future frames based on the observation of a small preceding sub-sequence, the LSTM needs to learns typical progressions of the complete activity.
Let $\mathcal{C}^{s,L} = (x_{s-L}, \ldots, x_s) \subset \mathcal{S}$ be an L frame sub-sequence of a video sequence $\mathcal{S}$ up to the s-th frame. Further, let h_s be the hidden state of the LSTM after consecutively processing the individual embeddings of the frames $\mathcal{C}^{s,L}$. To estimate a potential future embedding $\hat{\phi}(x_{s+1}) \in \Phi$, we learn a predictor function $g : \mathbb{R}^{D''} \rightarrow \mathbb{R}^{D}$ on the hidden states h^s, thus reverting to the embedding dimensionality D with

$$\hat{\phi}(x_{s+1}) = g(h_s; \theta') \tag{7.6}$$

and θ' denoting the corresponding trainable parameters of g. Applying the predictor recurrenlty will then allow us to continue and generate activity represented as a sequence

Algorithm 3: Unsupervised learning of a consistent posture representation using local correspondences.

Data: $\{\mathcal{S}_i\}_{i=1}^{S}, \theta$
`// Unlabeled video sequences and randomly initialized` θ
Result: $\{\theta\}$
while *not converged* **do**
  $(\mathcal{S}, \mathcal{S}') \leftarrow \{\mathcal{S}_i\}_{i=1}^{S}$ `// Training batch`
  $\boldsymbol{Z}^* \leftarrow \arg\min$ Eq. (7.2) `// Find assignment`
  $\mathcal{T} \leftarrow (\mathcal{S}, \mathcal{S}', \boldsymbol{Z}^*)$ `// Sample triplets`
  $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}(\mathcal{T}; \theta)$ `// Update` θ

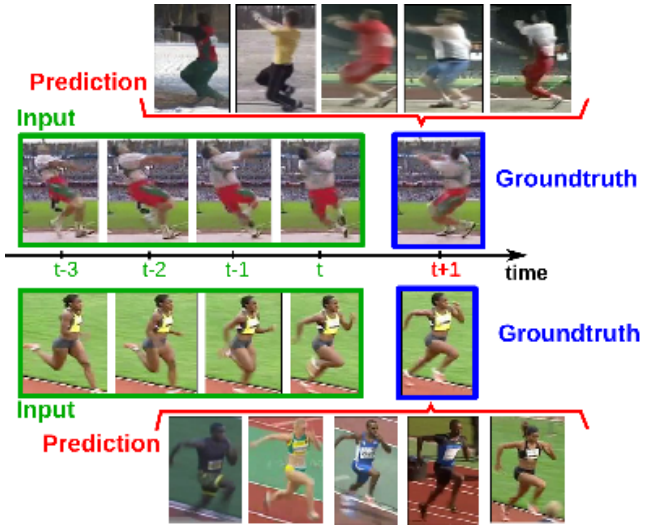


Figure 7.5: *Visualizing the Top* 5 *predictions (red) for the next frame* $s+1$ *given four previous frames on basis of our representation* ϕ *(green).* Actual successor in blue.

of embeddings in Φ. For learning, we formulate our future embedding prediction as an regression problem to estimate the embedding of the next ground-truth frame x_{s+1} to $C^{s,L}$,

$$\mathcal{L}_{\text{predict}}(h_s, x_{s+1}; \theta') = \|g(h_s; \theta') - \phi(x_{s+1}; \theta)\|_2^2 \,. \tag{7.7}$$

The trained LSTM is then able to hypothesize transitions to future frames based on a small subsequence of an activity as demonstrated in Fig. 7.5.

7.2 Experimental Evaluation

To evaluate our approach at all granularity levels of an activity, we report quantitative results for posture retrieval in Sect. 7.2.1 and human pose estimation (HPE) in Sec. 7.2.2-7.2.3. In addition, we provide qualitative results for activity understanding in Sect. 7.2.4, and for temporal super-resolution, and action synthesis in Sec. 7.2.5 to 7.2.7.

7.2.1 Posture retrieval: Olympic sports Dataset

The Olympic Sports (OS) dataset [163] is a compilation of video sequences of 16 different sports, containing more than 170000 frames overall in 300 video sequences. During training, each training mini-batch is generated by sampling two random sequences $\mathcal{S}, \mathcal{S}'$ and solving the ILP in Eq. (7.2) to obtain correspondences $\boldsymbol{Z}^*$. Solving the ILP takes $\sim$ 0.1sec (IBM CPLEX Optimization framework) compared to $\sim$ 0.5sec to process a minibatch on a NVIDIA Titan X (Pascal). We then use these correspondences to sample 300 triplets, where the initial percentile $q = 100$ is decreased by 10 every epoch. In each frame the approach of [64] yields person bounding boxes. During training we utilize VGG-S up to the fc6 layer and stack a 128-dimensional fc7 layer on top together with an L_2-normalization

HOG-LDA [87]	Ex-CNN [57]	VGG-S Imagenet [214]	Doersch et. al [54]
0.62	0.56	0.64	0.58
Shuffle&Learn [160]	CliqueCNN [12]	Ours (Scratch)	Ours (Imagenet)
0.63	0.79	**0.83**	**0.83**

Table 7.1: *Avg. AUC for each method on Olympic Sports dataset.*

layer following standard setups of deep metric learning as discussed in chapter 2, which is our representation $\phi(x;\theta)$. We use *Caffe* [108] for our implementation. To evaluate our representation on fine-grained posture retrieval we utilize the annotations provided by [12] and follow their evaluation protocol, using their annotations only for testing. We compare our method with CliqueCNN [12], the triplet formulation of Shuffle&Learn [160], the tuple approach of Doersch et. al [54], VGG-S [214], and HOG-LDA [87]. For completeness we also include a version of our model that was initialized with Imagenet pre-trained weights [214]. (*i*) For CliqueCNN, Shuffle& Learn, and Doersch et. al methods we use the models downloaded from their respective project websites. *(ii)* Exemplar-CNN is trained using the best performing parameters reported in [57] and the 64c5-128c5-256c5-512f architecture. Then we use the output of fc4 and compute 4-quadrant max pooling. During training of our approach, each image in the training set is augmented by performing random translation, scaling and rotation to improve invariance.

In Tab. 7.1 we show the average Area Under the Curve(AuC) measure over all categories for the different methods. When compared with the best method so far [12], the proposed approach improves the performance by 4%, although the method in [12] was even pre-trained on Imagenet. This improvement is due to the cross sequence relationships enforced by the sequence matching, which enforce a representation which is invariant to background and environmental factors, encoding only posture. In addition, when compared to the state-of-the-art methods that leverage tuples [54] or triplets [160] for training a CNN from scratch, our approach shows 20% higher performance. This is explained by the more detailed similarity relationships encoded in the cross-sequence correspondences obtained by the sequence matching approach, which uses temporal constraints to obtain high quality relationships of similarity and dissimilarity. It is noteworthy, that our randomly initialized VGG-S trained with our self-supervision strategy yields equivalent performance to a version with pre-trained Imagenet weights as initialization. Thus, the proposed self-supervision circumvents the use of the 1.2M labelled Imagenet samples. To the best of our knowledge, this is the first time that a self-supervised method performs equivalently without the widely adopted Imagenet pre-training strategy.

7.2.2 Zero-Shot human pose estimation

After evaluating the proposed method for fine-grained posture retrieval, we tackle the problem of zero-shot pose estimation on the LSP dataset. That is, we transfer the pose representation learnt on Olympic Sports to the LSP dataset without any further training and retrieve similar poses based on their similarity. The LSP [110] dataset is one of the

Method	T	UL	LL	UA	LA	H	Total
Ground Truth	93.7	78.8	74.9	58.7	36.4	72.4	69.2
Chu et al. [42]	98.4	95.0	92.8	88.5	81.2	95.7	90.9
CliqueCNN [12]	80.1	50.1	45.7	27.2	12.6	45.5	43.5
VGG-S [214]	**82.0**	48.2	41.8	32.4	15.8	53.6	47.0
Ours Scratch	73.0	45.1	41.6	26.2	12.2	44.4	43.0
Shuffle&Learn [160]	60.4	33.2	28.9	16.8	7.1	33.8	30.0
Ours (Imagenet)	81.3	**54.6**	**48.8**	**36.1**	**19.1**	**56.9**	**50.0**

Table 7.2: *PCP measure for each method on Leeds Sports dataset for zero-shot pose estimation.*

most widely used benchmarks for pose estimation. For evaluation we use the representation to compute visual similarities and find nearest neighbours to a query frame. Since the evaluation is zero-shot, joint labels are not available. At test time we therefore estimate the joint coordinates of a query person by finding the most similar frame from the training set and taking its joint coordinates. We then compare our method with VGG-S [214] pre-trained on Imagenet, the triplet approach of Misra et. al (Shuffle&Learn) [160] and CliqueCNN [12]. In addition, we also report an upper bound on the performance that can be achieved by zero-shot evaluation using ground-truth similarities. Here the most similar pose for a query is given by the frame which is closest in average distance of ground-truth pose annotations. This is the best one can achieve without a parametric model for pose (the performance gap to 100% shows the discrepancy between poses in test and train set). For completeness, we compare with a fully supervised state-of-the-art approach for pose estimation [42]. For computing similarities we now use the the intermediate *pool5* layer of VGG-S as our representation $\phi(x;\theta)$, provided that our model is transferred from another dataset [264].
In Tab. 7.2 we show the PCP@0.5 obtained by the different methods. For a fair comparison with CliqueCNN [12] (which was pre-trained on Imagenet), we include a version of our method trained using Imagenet initialization. Since in this experiment we are transferring our model from another dataset, we expect that Imagenet pre-training increases performance. Our approach significantly improves the visual similarities learned using both Imagenet pre-trained VGG-S and CliqueCNN [12], obtaining a performance boost of at least 3% in PCP score. In addition, when trained from scratch without any pre-training on Imagenet our model outperforms the model of [160] by 13%, due to the fact that the cross-sequence correspondences obtained by our approach encode finer relationships between samples. Finally, it is notable that even though our pose representation is *transferred from a different dataset* without fine-tuning on LSP, it obtains state-of-the-art performance in the realm of unsupervised methods.

7.2.3 Self-supervision as pre-training for human pose estimation

In addition to the zero-shot learning experiment we also evaluate our approach on the challenging MPII Pose dataset [3] which is a state of the art benchmark for evaluation

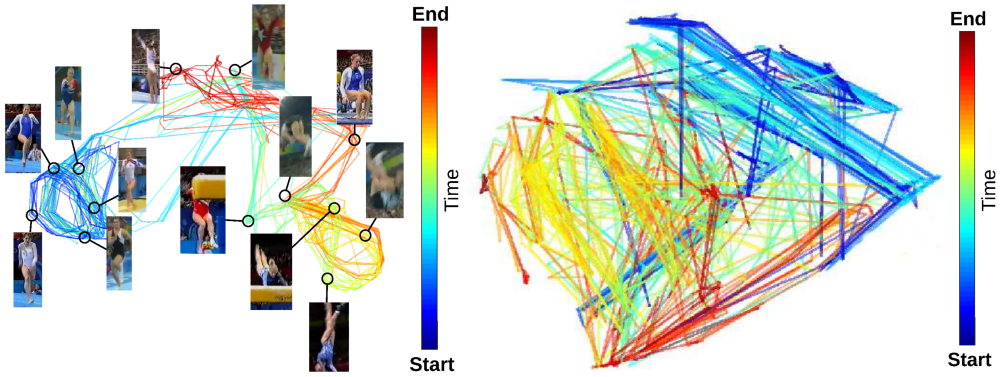
Approach	Head	Neck	Shoulders	Elbows	Wrists	Hip	Knees	Ankles	Thorax	Pelvis	Total
Rand. Init.	79.5	87.1	71.6	52.1	34.6	64.1	58.3	51.2	85.5	70.1	65.4
Imagenet Init.	87.2	93.2	85.2	69.6	52.0	81.3	69.7	62.0	93.4	86.6	78.0
S&L [160]	75.8	86.3	75.0	59.2	42.2	73.3	63.1	51.7	87.1	79.5	69.3
Ours (scratch)	80.6	88.4	74.8	56.9	41.6	73.3	63.6	56.9	88.6	79.9	70.5
Ours (Imagenet)	**90.2**	**93.8**	**86.3**	**70.4**	**58.6**	**82.4**	**73.2**	**67.4**	**93.7**	**88.4**	**80.4**

Table 7.3: *PCKh@0.5 measure on MPII Pose benchmark dataset using different initializations for the DeepPose approach [229].*

of articulated human pose estimation. The dataset includes around 25K images containing over 40K people with annotated body joints. MPII Pose is a particularly challenging dataset because of the clutter, occlusion and number of persons appearing in images. We are interested in evaluating how far our self-supervised approach can still boost a parametric approach that is trained with extensive supervision. Thus, we report the performance obtained by DeepPose [229], when trained using as initialization each of the following models: Random initialization, Shuffle&Learn [160], Imagenet and our approach trained on OS (scratch and Imagenet pretraining). For this experiment the Alexnet [131] architecture is used like in [229]. Following the standard evaluation metric on MPII dataset, Tab. 7.3 shows the PCKh@0.5 obtained by training DeepPose (stg-1) using their best reported parameters with the different initializations.
The performance obtained on MPII Pose benchmark shows that our unsupervised feature representation successfully scales to challenging datasets, successfully dealing with clutter, occlusions and multiple persons. In particular, when comparing our unsupervised initialization with a random initialization we obtain a 5.1% performance boost, which indicates that our features encode a robust notion of pose that is robust to the clutter present in MPII dataset. Furthermore, we obtain a 1.2% improvement over the Shuffle&Learn [160] approach.

7.2.4 Visualizing the activity representation

Whereas previous experiments have evaluated our representation on the level of individual poses, we now analyze the ability of our representation $\phi(x;\theta)$ to also capture transitions between successive postures. Our model is trained using temporally aligned correspondences across sequences, thus representing not only relationships between different sequences but also encoding temporal transitions of pose within a sequence. Therefore, ϕ captures all the regularity of posture. It maps similar postures of different persons and repetitions of the same posture in a sequence, e.g., repeated gait cycles, to the same spot in feature space. Moreover, successive poses are also mapped to similar representations, so an activity has a smooth trajectory over ϕ. To visually demonstrate the ability of $\phi(x;\theta)$ to capture the fine-grained pose interactions over time and between sequences we project the high dimensional representation ϕ to a 2D plot using the t-SNE procedure [150]. Fig. 7.6(a) shows a mapping of all instances of vault activity. Successive postures within each video are connected by straight lines and color encodes the time within the sequence. The



(a) (b)

Figure 7.6: *Visualizing the learned posture representation ϕ and the progression of an activity (indicated by color). All* frames of *all* vault sequences are shown (a) Our representation successfully learns the inherent structure of an action, e.g. repetitive gait and spinning cycles (blue, orange loops). Also, on a coarser temporal scale, repeated postures are brought near (outstretched arms before/after jump; cyan and red). Vector quantization of ϕ yields mutually dissimilar, characteristic poses shown in frames. (b) Representation obtained using [160]. As this model discards cross-sequence interactions, it misses the regularity of related postures across time and sequences.

learned representation captures the repetitive structure of running and spinning (blue and orange loops) and the characteristic transitions between. Additionally, the regularity of ϕ allows to group repetitive postures and to provide a condensed overview of an activity. Therefore, we employ standard agglomerative clustering [68] to extract prominent mutually dissimilar posture that span an activity. We show the representative of each cluster on its corresponding location in Fig. 7.6(a). Moreover, we compare our representation with the state-of-the-art approach of Misra et al. [160] which introduces a temporal verification problem and learns to find the correct temporal order of triplets of postures within a sequence, Fig. 7.6(b). This shows that modeling only posture interactions within a sequence [160] and excluding cross-sequences correspondences as proposed in Sect. 7.1 fails to capture the temporal evolution of an activity and degrades temporal structure.

7.2.5 Inferring temporal super-resolution

The previous section has demonstrated that our representation $\phi(x;\theta)$ successfully encodes posture and provides a basis for modeling the characteristic progression of an activity on the finest accessible level—individual frames. To further demonstrate the fine granularity at which activity is captured, we now unfold transitions between postures in consecutive frames, that is, we obtain super-resolution between two consecutive frames. Whereas [208] aggregates frames from within the same sequence, we bring all the different sequences with all their variability together. Since our representation maps related poses close to another, we can employ a local linear interpolation between successive postures to infer intermediate frames. Then the Generative Adversarial Network (GAN) of [56] is

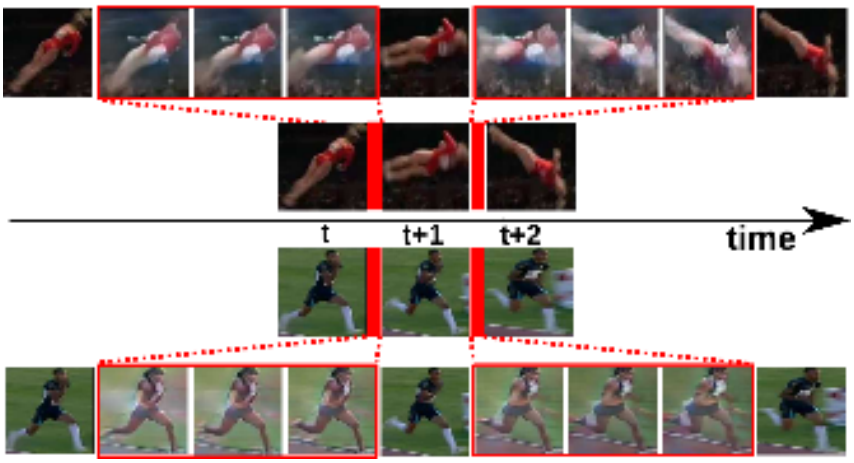
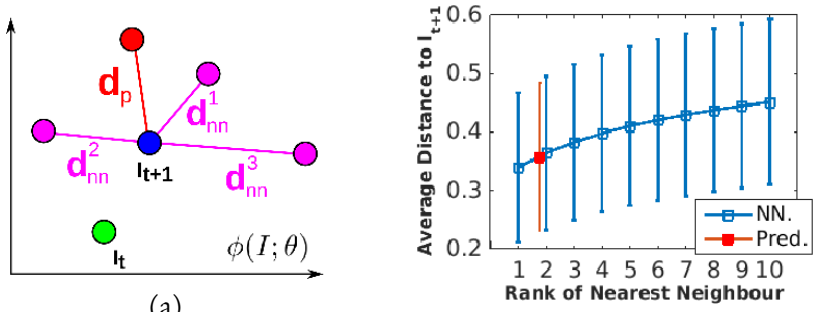


Figure 7.7: *Inferring intermediate frames between consecutive postures.* For each transition we interpolate representations ϕ at regular intervals in between and invert interpolated features with [56].

used to invert the feature back to an image. We use the implementation of [56] for both the generator and discriminator networks, where our learned activity representation $\phi(x;\theta)$ acts as the encoder network. To jointly train the three networks we use the DeePSiM-loss [56] considering adversarial and euclidean terms on both the image and feature domain. This inversion of our representation $\phi(x;\theta)$ creates images for the synthesized intermediate frames, allowing us to go past the limited temporal scale of given video sequences. Fig. 7.7 shows temporal super-resolution results for two different activities. The continous progression of activity is preserved due to the continuity of our pose representation $\phi(x;\theta)$. It has finer temporal granularity than an individual video, since it interleaves a large number of related sequences, providing a truly continuous activity representation.

7.2.6 Activity understanding using LSTMs

So far we have provided a comprehensive analysis of our activity representation, demonstrating its ability to understand actions on the fine-grained scale of single postures. Now, we evaluate the capability of our method to understand activity. To this end, we train an RNN on top of the posture representation ϕ to yield a sequence-level encoding h_s, as discussed in Sect. 7.1.3. We employ an LSTM, trained on sub-sequences $\mathcal{C}^{s,L}$ of length $L = 4$, densely sampled from all video sequences to predict the next succeeding frame embedding $\hat{\phi}(x_{s+1})$. During training, we sample mini-batches to cover the overall diversity of activity, so that all constituent postures are equally represented for learning the LSTM. Fig. 7.5 shows exemplary predictions from different activities. Let us quantify the quality of predicted next frames. Given the true successor embedding $\phi(x_{s+1};\theta)$, we identify its nearest neighbor in all the videos. We then compute the distance between these two frames and average it over all videos. The same is then done for the second, third, etc. nearest neighbor. Similarly, we compute the distance of our prediction and



(a) (b)

Figure 7.8: (a) Setup of quantitative evaluation. Blue is the actual next frame, red the prediction, purple the nearest neighbours. (b) LSTM evaluation: Comparing our average prediction error (red) against the average distance of $\phi(x_{t+1};\theta)$ to each of its 10 nearest neighbours embeddings (blue). The error bars indicate the standard deviation of the measurements.

$\phi(x_{s+1};\theta)$ and also average it. Fig. 7.8 (b) compares the resulting error of our prediction against that of the k nearearest neighbor from the dataset. Our prediction is better than actually observing the next frame and picking its second nearest neighbor. Despite the large variability of an activity this shows that the temporal progression of an activity has been well captured to yield favorable predictions of a successive frame.

7.2.7 Video understanding by action synthesis

We have just seen predictions of the next frame x_{s+1} of a sequence. By recursively adding this predicted frame and then predicting a next successive frame we can iteratively synthesize an overall activity frame by frame. For visualization of the predicted next posture, we choose the nearest neighbor from the training set. Fig. 7.9 summarizes the synthesis of a snatch activity initialized at the green posture. One can see that our model successfully infers the temporal ordering of the activity from its beginning until the end.

7.3 Discussion

In this chapter we presented an unsupervised approach for understanding activity by means of its most fine-grained temporal constituents, individual human postures. A combinatorial sequence matching algorithm proposes relations between frames from subsets of the training set, which are then used to learn a single concerted pose embedding that reconciles transitivity conflicts using a similarity learning approach. Without any manual annotation, the model learns a structured representation of postures and their temporal development. The model not only enables retrieval of similar postures but also temporal super-resolution. Additionally, based on a recurrent formulation, future frames and activities can be synthesized.

Figure 7.9: *Synthesizing the 'clean and jerk' activity from the Olypmic Sports dataset [163] by recursively predicting the next posture.* Green indicates initial image and every 5th frame is shown.

8 Conclusion

In this thesis, we addressed the problem of learning abstract representations that effectively describe and allow to compare between objects, one of the cornerstones of many Computer Vision applications. The quality of object representations is typically measured by their generalization performance, thus how accurately unseen object instances are captured. Particularly challenging is the grand goal of effective out-of-distribution generalization, i.e. the transfer of a representation to objects which are far from the training data distribution – to date poorly understood and an open research problem. The first part of this work was devoted to analyzing and advancing out-of-distribution generalization of object representations learned by Deep Metric Learning approaches. To this end, we challenged the prevailing learning paradigm of class-discriminative training, which typically results in representations that are highly specialized only on the discriminative features of the training objects. Consequently, they are unlikely to also describe novel unseen objects which naturally hinders out-of-distribution generalization.

To identify driving factors of out-of-distribution generalization, we conducted a large-scale study of the current field of Deep Metric Learning (DML), arguably the main research direction of similarity learning. In particular, by focusing on the structure of representations, we uncovered a strong relationship between generalization and the (information) compression of a representation. Representations which are less compressed, i.e. representations which describe data more comprehensively (and less specialized), exhibit stronger generalization performance. Naturally, such representations are more expressive and capture a more diverse set of object features which increases the chance to also capture out-of-distribution objects. This finding is particularly interesting as it is diametrically opposed to the class-discriminative learning paradigm, which actually aims at learning strongly compressed object representations. Following this property, we combined classic class-discriminative learning with learning tasks focusing on complementary object features, thus directly increasing the overall expressiveness of object representations. In particular, we introduced the concept of class-shared features, i.e. characteristics which are explicitly common among training classes – and neglected by discriminative learning. Our evaluations clearly show that they have a significant impact on generalization performance, outperforming the pure class-discriminative state-of-the-art. Motivated by this result, we extended our DML training framework to include even more additional learning tasks, eventually merging various, diverse learning signals into a single model, which set a new state-of-the-art in DML. Besides training objectives, also the data sampling process is known to affect learning of deep models. Hence, to provide a comprehensive work on representation generalization, we also addressed data sampling strategies for the widely used triplet-based metric learning. Similar to the discriminative paradigm discussed above,

predefined and fixed heuristics have been used so far to complement learning without explicitly taking generalization into account. Although such heuristics are efficient and can provide informative data samples at least at some stages of the learning process, a model's training state continually changes. Hence, this actually asks also for a change in the learning signal provided by the input triplets. As a solution, we proposed the first learnable selection policy for triplets during training, while explicitly targeting generalization to unknown samples. Using Reinforcement Learning, we adapt the sampling policy to the current training state of a DML model. While Reinforcement Learning often introduces substantial computational overhead, our sampling policy is based on the efficient formulation of heuristic-based strategies. Therefore, we are able to maintain efficient learning, while simultaneously explicitly optimizing generalization performance. In conclusion, we showed that to improve out-of-distribution generalization, it has to be considered more explicitly in the learning formulations. Although effective, our proposed frameworks are only specific solutions to the broader insights on out-of-generalization generalization provided in this work. Hence, we hope to stimulate future lines of research on this topic.
The learning tasks that have been introduced to advance representation generalization are formulated to only reuse already provided class annotations or to not require any supervision at all. The reason for this is partly that we simply cannot tell what kind of shared features can be learned from data, thus what kind of supervision would be required. On the other hand, providing sufficient supervision information is often tedious and costly. Both of these reasons are actually common issues for machine learning in general. This supervision dependency issue is particularly unfortunate, since Deep Learning has proven to scale extremely well with the amount of training data, which, for example in the case of images or videos, seems to be available in unlimited quantities. As a consequence, bottlenecks in providing the required supervision information often directly represent bottlenecks in model performance. To overcome these problems, the research area of unsupervised learning is devoted to learning from pure data without the need for supervisory information – often using highly specific surrogate tasks as a substitute for annotations. In the second part of this thesis, we investigated the application of well-established similarity learning, respectively Deep Metric Learning techniques, to unsupervised representation learning, both for the domain of static images and complex video sequences. To provide such models with the necessary similarity constraints for learning, we explored how to extract reliable sample relations from the data itself, i.e. relations that are likely to be correct. This is orthogonal to related work in unsupervised learning, which typically does not consider the reliability of the inferred constraints, despite the risk of degraded performance due to learning from erroneous self-supervision. Indeed, our experimental results show that reliability benefits learning and the eventually learned representations. By alternating between refining our representations and inferring more and more similarity relations, our models show significant performance improvements. In particular in the domain of video sequences depicting human activity, our proposed model enables applications like temporal super-resolution, zero-shot human pose estimation, and transfer learning.

Bibliography

1. P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. "Geometric approximation via coresets". *Combinatorial and Computational Geometry* 52, 2005, pp. 1–30.
2. T. Ahonen, A. Hadid, and M. Pietikainen. "Face Description with Local Binary Patterns: Application to Face Recognition". *IEEE Transactions Pattern Analysis and Machine Intelligence (TPAMI)* 28, 2006, pp. 2037–2041.
3. M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. "2D Human Pose Estimation: New Benchmark and State of the Art Analysis". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.
4. M. Andrychowicz, M. Denil, S. Gómez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. de Freitas. "Learning to learn by gradient descent by gradient descent". In: *Advances in Neural Information Processing Systems*. 2016.
5. B. Antic and B. Ommer. "Learning Latent Constituents for Recognition of Group Activities in Video". In: *European Conference on Computer Vision (ECCV)*. 2014.
6. B. Antic, U. Büchler, A.-S. Wahl, M. E. Schwab, and B. Ommer. "Spatiotemporal Parsing of Motor Kinematics for Assessing Stroke Recovery". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2015, pp. 467–475.
7. B. Antic and B. Ommer. "Per-Sample Kernel Adaptation for Visual Recognition and Grouping". In: *International Conference on Computer Vision (ICCV)*. 2015.
8. B. Antic and B. Ommer. "Video parsing for abnormality detection". In: *International Conference on Computer Vision (ICCV)*. 2011.
9. P. Bachman, R. D. Hjelm, and W. Buchwalter. "Learning Representations by Maximizing Mutual Information Across Views". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche-Buc, E. B. Fox, and R. Garnett. 2019.
10. S. Baker and I. Matthews. "Lucas-Kanade 20 Years On: A Unifying Framework". 56, 2004, pp. 221–255.
11. M. Bar. "The proactive brain: memory for predictions". *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 2009.
12. M. A. Bautista, A. Sanakoyeu, E. Tikhoncheva, and B. Ommer. "Cliquecnn: Deep unsupervised exemplar learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2016.
13. H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. "Speeded-Up Robust Features (SURF)". *Comput. Vis. Image Underst.* 110, 2008, pp. 346–359.

14. A. Bellet and A. Habrard. “Robustness and generalization for metric learning”. *Neurocomputing* 151, 2015, pp. 259–267.
15. Y. Bengio, J. Louradour, R. Collobert, and J. Weston. “Curriculum Learning”. In: *International Conference on Machine Learning (ICML)*. 2009.
16. A. Berg and J. Malik. *Geometric Blur for Template Matching*. 2001.
17. D. J. Berndt and J. Clifford. “Using dynamic time warping to find patterns in time series.” In: *KDD workshop*. Vol. 10. Seattle, WA. 1994.
18. D. Berthelot, C. Raffel, A. Roy, and I. J. Goodfellow. “Understanding and Improving Interpolation in Autoencoders via an Adversarial Regularizer”. In: *7th International Conference on Learning Representations (ICLR)*. 2019.
19. B. Bhattarai, G. Sharma, and F. Jurie. “CP-mtML: Coupled Projection Multi-Task Metric Learning for Large Scale Face Retrieval”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
20. I. Biedermann. “Recognition-by-components: A theory of human image understanding”. *Psychological Review* 94:2, 1987, pp. 115–147.
21. P. Bojanowski and A. Joulin. “Unsupervised Learning by Predicting Noise”. In: *International Conference on Machine Learning (ICML)*. 2017.
22. D. Bouchacourt, R. Tomioka, and S. Nowozin. “Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations”. In: *AAAI 2018*. 2018.
23. X. Bouthillier, C. Laurent, and P. Vincent. “Unreproducible Research is Reproducible”. In: *International Conference on Machine Learning (ICML)*. 2019.
24. B. Brattoli, U. Buchler, A.-S. Wahl, M. E. Schwab, and B. Ommer. “Lstm self-supervision for detailed behavior analysis”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.
25. A. Brock, J. Donahue, and K. Simonyan. “Large Scale GAN Training for High Fidelity Natural Image Synthesis”, 2018.
26. A. Brutzkus and A. Globerson. “Why do Larger Models Generalize Better? A Theoretical Perspective via the XOR Problem”. In: *International Conference on Machine Learning (ICML)*. 2019.
27. U. Büchler, B. Brattoli, and B. Ommer. “Improving Spatiotemporal Self-Supervision by Deep Reinforcement Learning”. In: *European Conference on Computer Vision (ECCV)*. 2018.
28. F. Cakir, K. He, X. Xia, B. Kulis, and S. Sclaroff. “Deep Metric Learning to Rank”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.
29. J. Canny. “A Computational Approach to Edge Detection”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 8, 1986, pp. 679–698.

30. M. Caron, P. Bojanowski, A. Joulin, and M. Douze. "Deep Clustering for Unsupervised Learning of Visual Features". *European Conference on Computer Vision*, 2018.

31. J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan. "Deep unsupervised learning with consistent inference of latent representations". *Pattern Recognition (PR)* 77, 2018, pp. 438–453.

32. O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. 2010.

33. N. V. Chawla, N. Japkowicz, and A. Kotcz. "Editorial: Special Issue on Learning from Imbalanced Data Sets". *SIGKDD Explor. Newsl.* 2004.

34. G. Chechik, V. Sharma, U. Shalit, and S. Bengio. "Large Scale Online Learning of Image Similarity Through Ranking". *Journal on Machine Learning Research* 11, 2010, pp. 1109–1135.

35. S. Chen, Y. Li, and N. M. Kwok. "Active vision in robotic systems: A survey of recent developments". *The International Journal of Robotics Research* 30:11, 2011, pp. 1343–1377.

36. T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations". In: *International Conference on Machine Learning (ICML)*. 2020.

37. W. Chen, X. Chen, J. Zhang, and K. Huang. "Beyond triplet loss: a deep quadruplet network for person re-identification". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

38. X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets". In: *International Conference on Neural Information Processing Systems*. 2016.

39. T. Ching, D. Himmelstein, B. Beaulieu-Jones, A. Kalinin, B. Do, G. Way, E. Ferrero, P. Agapow, M. Zietz, M. Hoffman, W. Xie, G. Rosen, B. Lengerich, J. Israeli, J. Lanchantin, S. Woloszynek, A. Carpenter, A. Shrikumar, J. Xu, and C. Greene. "Opportunities and obstacles for deep learning in biology and medicine". *Journal of The Royal Society Interface* 15, 2018, p. 20170387.

40. K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches". In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. 2014.

41. S. Chopra, R. Hadsell, and Y. LeCun. "Learning a similarity metric discriminatively, with application to face verification". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2005.

42. X. Chu, W. Yang, W. Ouyang, C. Ma, A. L. Yuille, and X. Wang. "Multi-Context Attention for Human Pose Estimation". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

43. A. Coates, A. Ng, and H. Lee. "An analysis of single-layer networks in unsupervised feature learning". In: *International Conference on Artificial Intelligence and Statistics*. 2011.
44. M. Cohen, G. Alvarez, K. Nakayama, and T. Konkle. "Visual search for object categories is predicted by the representational architecture of high-level visual cortex". *Journal Neurophysiology*, 2017, pp. 388–402.
45. C. Cortes and V. Vapnik. "Support-Vector Networks". *Machine Learning* 20, 1995, pp. 273–297.
46. H. Coskun, D. J. Tan, S. Conjeti, N. Navab, and F. Tombari. "Human Motion Analysis with Deep Metric Learning". In: *European Conference on Computer Vision (ECCV)*. 2018.
47. G. Csurka. *Domain Adaptation in Computer Vision Applications*. 1st. Springer Publishing Company, Incorporated, 2017.
48. Y. L. Cun. *A Theoretical Framework for Back-Propagation*. 1988.
49. N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2005.
50. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009.
51. J. Deng, J. Guo, N. Xue, and S. Zafeiriou. *ArcFace: Additive Angular Margin Loss for Deep Face Recognition*. 2019.
52. G. N. Desouza and A. C. Kak. "Vision for mobile robot navigation: a survey". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 24:2, 2002, pp. 237–267.
53. J. Dicarlo, D. Zoccolan, and N. Rust. "How Does the Brain Solve Visual Object Recognition?" *Neuron* 73, 2012, pp. 415–34.
54. C. Doersch, A. Gupta, and A. A. Efros. "Unsupervised visual representation learning by context prediction". In: *International Conference on Computer Vision (ICCV)*. 2015.
55. J. Donahue, P. Krähenbühl, and T. Darrell. "Adversarial Feature Learning". In: *International Conference on Learning Representations (ICLR)*. 2017.
56. A. Dosovitskiy and T. Brox. "Generating Images with Perceptual Similarity Metrics based on Deep Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2016.
57. A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox. "Discriminative unsupervised feature learning with convolutional neural networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2014.
58. Y. Duan, W. Zheng, X. Lin, J. Lu, and J. Zhou. "Deep Adversarial Metric Learning". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

59. V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. C. Courville. "Adversarially Learned Inference". In: *International Conference on Learning Representations (ICLR)*. 2017.

60. A. A. Efros, A. C. Berg, G. Mori, J. Malik, et al. "Recognizing Action at a Distance." In: *International Conference on Computer Vision (ICCV)*. Vol. 3. 2003.

61. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The PASCAL Visual Object Classes Challenge 2007 (VOC2007)*. 2007.

62. Y. Fan, F. Tian, T. Qin, J. Bian, and T.-Y. Liu. "Neural Data Filter for Bootstrapping Stochastic Gradient Descent". In: *International Conference on Learning Representations (ICLR)*. 2017.

63. C. Feichtenhofer, A. Pinz, and A. Zisserman. "Convolutional Two-Stream Network Fusion for Video Action Recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

64. P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. "Object Detection with Discriminatively Trained Part-Based Models". *IEEE Transactions on Pattern Analysis Machine Intelligence (TPAMI)* 32, 2010, pp. 1627–1645.

65. Z. Feng, C. Xu, and D. Tao. "Self-Supervised Representation Learning by Rotation Feature Decoupling". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

66. C. Finn, P. Abbeel, and S. Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *International Conference on Machine Learning (ICML)*. 2017.

67. P. Fletcher, C. Frith, S. Baker, T. Shallice, R. Frackowiak, and R. Dolan. "The mind's eye - precuneus activation in memory-related imagery". *Neuroimage*, 1995, pp. 195–200.

68. H. Frigui and R. Krishnapuram. "Clustering by competitive agglomeration". *Pattern recognition* 30:7, 1997.

69. Y. Ganin and V. Lempitsky. "Unsupervised Domain Adaptation by Backpropagation". In: *International Conference on Machine Learning (ICML)*. 2015.

70. L. Gautheron, A. Habrard, E. Morvan, and M. Sebban. "Metric Learning from Imbalanced Data". In: *International Conference on Tools with Artificial Intelligence (ICTAI)*. 2019.

71. W. Ge. "Deep metric learning with hierarchical triplet loss". In: *European Conference on Computer Vision (ECCV)*. 2018.

72. S. Gidaris, P. Singh, and N. Komodakis. "Unsupervised Representation Learning by Predicting Image Rotations". In: *International Conference on Learning Representations (ICLR)*. 2018.

73. A. Gionis, P. Indyk, and R. Motwani. "Similarity Search in High Dimensions via Hashing". In: *International Conference on Very Large Data Bases*. 1999.

74. R. Girshick. “Fast R-CNN”. In: *International Conference on Computer Vision (ICCV)*. 2015.

75. R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

76. X. Glorot and Y. Bengio. “Understanding the difficulty of training deep feedforward neural networks.” In: JMLR Proceedings. 2010.

77. J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. “Neighbourhood Components Analysis”. In: *International Conference on Neural Information Processing Systems (NeurIPS)*. 2004.

78. M. A. Goodale and A. D. Milner. “Separate visual pathways for perception and action”. *Trends in Neurosciences* 15, 1992.

79. I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. The MIT Press, 2016.

80. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2014.

81. S. Gopal. “Adaptive Sampling for SGD by Exploiting Side Information”. In: *International Conference on Machine Learning (ICML)*. 2016.

82. H. Greenspan, B. van Ginneken, and R. M. Summers. “Guest Editorial Deep Learning in Medical Imaging: Overview and Future Promise of an Exciting New Technique”. *IEEE Transactions on Medical Imaging* 35:5, 2016, pp. 1153–1159.

83. A. Gupta, P. Srinivasan, J. Shi, and L. S. Davis. “Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos”. In: *Conference on Computer Vision and Pattern Recognition*. 2009.

84. M. Gutmann and A. Hyvärinen. “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models”. In: *International Conference on Artificial Intelligence and Statistics*. 2010.

85. G. Hacohen and D. Weinshall. “On The Power of Curriculum Learning in Training Deep Networks”. In: 2019.

86. R. Hadsell, S. Chopra, and Y. LeCun. “Dimensionality Reduction by Learning an Invariant Mapping”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2006.

87. B. Hariharan, J. Malik, and D. Ramanan. “Discriminative decorrelation for clustering and classification”. In: *European Conference on Computer Vision (ECCV)*. 2012.

88. B. Harwood, B. Kumar, G. Carneiro, I. Reid, T. Drummond, et al. “Smart mining for deep metric learning”. In: *International Conference on Computer Vision (ICCV)*. 2017.

89. K. He, G. Gkioxari, P. Dollár, and R. Girshick. “Mask R-CNN”. In: *International Conference on Computer Vision (ICCV)*. 2017.

90. K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. "Momentum Contrast for Unsupervised Visual Representation Learning". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

91. K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

92. M. N. Hebart, C. Y. Zheng, F. Pereira, and C. I. Baker. "Revealing the multidimensional mental representations of natural objects underlying human similarity judgements". *Nature Human Behaviour*, 2020.

93. D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song. "Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty". In: *Advances in Neural Information Processing Systems 32*. 2019.

94. A. Hermans, L. Beyer, and B. Leibe. *In Defense of the Triplet Loss for Person Re-Identification*. 2017. arXiv: 1703.07737 [cs.CV].

95. M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. G. Azar, and D. Silver. "Rainbow: Combining Improvements in Deep Reinforcement Learning". In: *AAAI Conference on Artificial Intelligence, (AAAI)*. 2018.

96. M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *International Conference on Neural Information Processing Systems*. 2017.

97. S. Hochreiter and J. Schmidhuber. "Long Short-Term Memory". *Neural Computing* 9, 1997, pp. 1735–1780.

98. A. E. Hoerl and R. W. Kennard. "Ridge Regression: Biased Estimation for Nonorthogonal Problems". *Technometrics* 42, 2000, pp. 80–86.

99. J. Hu, J. Lu, and Y. Tan. "Discriminative Deep Metric Learning for Face Verification in the Wild". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

100. M. Huai, H. Xue, C. Miao, L. Yao, L. Su, C. Chen, and A. Zhang. "Deep Metric Learning: The Generalization Analysis and an Adaptive Algorithm". In: *International Joint Conference on Artificial Intelligence, IJCAI-19*. 2019.

101. C. Huang, C. C. Loy, and X. Tang. "Unsupervised Learning of Discriminative Attributes and Visual Representations". *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

102. C. Huang, S. Zhai, W. Talbott, M. Á. Bautista, S.-Y. Sun, C. Guestrin, and J. Susskind. "Addressing the Loss-Metric Mismatch with Adaptive Loss Alignment". In: *International Conference on Machine Learning (ICML)*. 2019.

103. S. Ioffe and C. Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". *International Conference on Machine Learning (ICML)*, 2015.

104. D. R. J., G. R. Fink, M. E Rolls Booth, A. Holmes, R. S. Frackowiak, and K. J. Friston. "How the brain learns to see objects and faces in an impoverished context". *Nature* 389, 1997, pp. 596–599.

105. P. Jacob, D. Picard, A. Histace, and E. Klein. "Metric Learning With HORDE: High-Order Regularizer for Deep Embeddings". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

106. J. Janai, F. Güney, A. Behl, and A. Geiger. *Computer Vision for Autonomous Vehicles: Problems, Datasets and State-of-the-Art*. Foundations, Trends in Computer Graphics, and Vision, 2020.

107. H. Jegou, M. Douze, and C. Schmid. "Product quantization for nearest neighbor search". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 33, 2011, pp. 117–128.

108. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. "Caffe: Convolutional Architecture for Fast Feature Embedding". *arXiv preprint arXiv:1408.5093*, 2014.

109. J. Johnson, M. Douze, and H. Jégou. "Billion-scale similarity search with GPUs". *ArXiv*, 2017.

110. S. Johnson and M. Everingham. "Learning Effective Human Pose Estimation from Inaccurate Annotation". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

111. T. B. Johnson and C. Guestrin. "Training Deep Models Faster with Robust, Approximate Importance Sampling". In: *Advances in Neural Information Processing Systems 31*. 2018.

112. I. T. Jolliffe. *Principal Component Analysis and Factor Analysis*. Springer New York, 1986. ISBN: 978-1-4757-1904-8.

113. D. Kahneman. *Thinking, fast and slow*. Farrar, Straus and Giroux, 2011.

114. A. Kapur, A. Kapur, N. Virji-Babul, G. Tzanetakis, and P. F. Driessen. "Gesture-based affective computing on motion capture data". In: *International Conference on Affective Computing and Intelligent Interaction*. 2005.

115. A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. "Large-scale video classification with convolutional neural networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

116. E. Keogh. "Exact indexing of dynamic time warping". In: *International Conference on Very Large Data Bases*. 2002.

117. N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. "On large-batch training for deep learning: Generalization gap and sharp minima". In: *International Conference on Learning Representations (ICLR)*. 2017.

118. S. Kim, D. Kim, M. Cho, and S. Kwak. "Proxy Anchor Loss for Deep Metric Learning". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

119. S. Kim, M. Seo, I. Laptev, M. Cho, and S. Kwak. "Deep Metric Learning Beyond Binary Supervision". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

120. W. Kim, B. Goyal, K. Chawla, J. Lee, and K. Kwon. "Attention-Based Ensemble for Deep Metric Learning". In: *European Conference on Computer Vision (ECCV)*. 2018.

121. D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. "Semi-Supervised Learning with Deep Generative Models". In: *International Conference on Neural Information Processing Systems*. 2014.

122. D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: 2015.

123. D. P. Kingma and M. Welling. "Auto-encoding variational bayes". In: *International Conference on Learning Representations (ICLR)*. 2013.

124. D. E. Knuth, J. H. Morris Jr, and V. R. Pratt. "Fast pattern matching in strings". *SIAM journal on computing* 6:2, 1977, pp. 323–350.

125. G. Koch, R. Zemel, and R. Salakhutdinov. "Siamese Neural Networks for One-shot Image Recognition". In: *International Conference on Machine Learning (ICML) - Deep Learning Workshop*. 2015.

126. T. Konno and M. Iwazume. "Cavity Filling: Pseudo-Feature Generation for Multi-Class Imbalanced Data Problems in Deep Learning". *ArXiv*, 2019.

127. S. Kornblith, J. Shlens, and Q. V. Le. "Do better ImageNet models transfer better?" In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

128. D. Kotovenko, A. Sanakoyeu, S. Lang, and B. Ommer. "Content and Style Disentanglement for Artistic Style Transfer". In: *International Conference on Computer Vision (ICCV)*. 2019.

129. P. Krähenbühl, C. Doersch, J. Donahue, and T. Darrell. "Data-dependent initializations of convolutional neural networks". *International Conference on Learning Representations (ICLR)*, 2016.

130. J. Krause, M. Stark, J. Deng, and L. Fei-Fei. "3d object representations for fine-grained categorization". In: *International Conference on Computer Vision (ICCV) Workshops*. 2013.

131. A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *International Conference on Neural Information Processing Systems*. 2012.

132. A. Krogh and J. A. Hertz. "A Simple Weight Decay Can Improve Generalization". In: *Advances in Neural Information Processing Systems*. 1992.

133. H. Kuhn. "The Hungarian method for the assignment problem". *Naval research logistics quarterly* 2, 1955.

134. B. Kulis. "Metric Learning: A Survey". *Foundations and Trends® in Machine Learning* 5, 2013, pp. 287–364.

135. U. L.G. and M. Mishkin. "Two cortical visual systems". *Analysis of Visual Behavior*, 1982, pp. 549–578.

136. S. Laine and T. Aila. "Temporal Ensembling for Semi-Supervised Learning." In: *International Conference on Learning Representations (ICLR)*. 2017.

137. T. Lan, Y. Zhu, A. Roshan Zamir, and S. Savarese. "Action recognition by hierarchical mid-level action elements". In: *International Conference on Computer Vision (ICCV)*. 2015.

138. J. Langford and T. Zhang. "The Epoch-Greedy Algorithm for Multi-armed Bandits with Side Information". In: *Advances in Neural Information Processing Systems 20*. 2008.

139. I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. "Learning realistic human actions from movies". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2008.

140. G. Larsson, M. Maire, and G. Shakhnarovich. "Colorization as a Proxy Task for Visual Understanding". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

141. X. Lin, Y. Duan, Q. Dong, J. Lu, and J. Zhou. "Deep Variational Metric Learning". In: *The European Conference on Computer Vision (ECCV)*. 2018.

142. G. Linden, B. Smith, and J. York. "Amazon.Com Recommendations: Item-to-Item Collaborative Filtering". *IEEE Internet Computing* 7, 2003, pp. 76–80.

143. W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. "SphereFace: Deep Hypersphere Embedding for Face Recognition". *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

144. W. Liu, Y. Wen, Z. Yu, and M. Yang. "Large-Margin Softmax Loss for Convolutional Neural Networks". In: *International Conference on International Conference on Machine Learning*. 2016.

145. Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma. "A Survey of Content-Based Image Retrieval with High-Level Semantics". *Pattern Recognition* 40, 2007, pp. 262–282.

146. S. P. Lloyd. "Least squares quantization in PCM". *IEEE Transactions on Information Theory* 28, 1982, pp. 129–136.

147. D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision* 60, 2004, pp. 91–110.

148. F. Lv and R. Nevatia. "Single View Human Action Recognition using Key Pose Matching and Viterbi Path Searching". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2007.

149. Q. Ma, C. Bai, J. Zhang, Z. Liu, and S. Chen. "Supervised learning based discrete hashing for image retrieval". *Pattern Recognition (PR)* 92, 2019, pp. 156–164.

150. L. v. d. Maaten and G. Hinton. "Visualizing data using t-SNE". *Journal of Machine Learning Research* 9:Nov, 2008, pp. 2579–2605.

151. C. Manning, P. Raghavan, and H. Schütze. "Introduction to information retrieval". *Natural Language Engineering* 16, 2010, pp. 100–103.

152. A. Martin. "The representation of object concepts in the brain". *Annual review of psychology* 58, 2007, pp. 25–45.

153. L. McInnes, J. Healy, N. Saul, and L. Grossberger. "UMAP: Uniform Manifold Approximation and Projection". *The Journal of Open Source Software* 3:29, 2018, p. 861.

154. T. Milbich, M. Bautista, E. Sutter, and B. Ommer. "Unsupervised video understanding by reconciliation of posture similarities". In: *International Conference on Computer Vision (ICCV)*. 2017.

155. T. Milbich, O. Ghori, F. Diego, and B. Ommer. "Unsupervised Representation Learning by Discovering Reliable Image Relations". *Pattern Recognition (PR)* 102, 2020.

156. T. Milbich, K. Roth, H. Bharadhwaj, S. Sinha, Y. Bengio, B. Ommer, and J. P. Cohen. "DiVA: Diverse Visual Feature Aggregation for Deep Metric Learning". In: *European Conference on Computer Vision (ECCV)*. 2020.

157. T. Milbich, K. Roth, B. Brattoli, and B. Ommer. "Sharing Matters for Generalization in Deep Metric Learning". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

158. B. Mirzasoleiman, J. Bilmes, and J. Leskovec. *Coresets for Accelerating Incremental Gradient Methods*. 2020. URL: https://openreview.net/forum?id=SygRikHtvS.

159. I. Misra and L. van der Maaten. *Self-Supervised Learning of Pretext-Invariant Representations*. 2020.

160. I. Misra, C. L. Zitnick, and M. Hebert. "Shuffle and learn: unsupervised learning using temporal order verification". In: *European Conference on Computer Vision*. 2016.

161. Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh. "No fuss distance metric learning using proxies". In: *International Conference on Computer Vision (ICCV)*. 2017.

162. K. Musgrave, S. Belongie, and S.-N. Lim. *A Metric Learning Reality Check*. 2020. arXiv: 2003.08505 [cs.CV].

163. J. C. Niebles, C.-W. Chen, and L. Fei-Fei. "Modeling temporal structure of decomposable motion segments for activity classification". In: *European Conference on Computer Vision (ECCV)*. 2010.

164. H. Noh, T. You, J. Mun, and B. Han. "Regularizing Deep Neural Networks by Noise: Its Interpretation and Optimization". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. 2017.

165. M. Noroozi and P. Favaro. "Unsupervised learning of visual representations by solving jigsaw puzzles". In: *European Conference on Computer Vision (ECCV)*. 2016.

166. M. Noroozi and P. Favaro. "Unsupervised Learning of Visual Representions by solving Jigsaw Puzzles". In: *European Conference on Computer Vision (ECCV)*. 2016.

167. M. Noroozi, H. Pirsiavash, and P. Favaro. "Representation Learning by Learning to Count". In: *International Conference on Computer Vision (ICCV)*. 2017.

168. M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash. "Boosting self-supervised learning via knowledge transfer". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

169. M. Norouzi, D. J. Fleet, and R. Salakhutdinov. "Hamming Distance Metric Learning". In: *International Conference on Neural Information Processing Systems*. 2012.

170. H. Oh Song, S. Jegelka, V. Rathod, and K. Murphy. "Deep metric learning via facility location". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

171. H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. "Deep metric learning via lifted structured feature embedding". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

172. A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I. J. Goodfellow. "Realistic Evaluation of Deep Semi-Supervised Learning Algorithms". In: *International Conference on Neural Information Processing Systems*. 2018.

173. N. M. Oliver, B. Rosario, and A. P. Pentland. "A Bayesian computer vision system for modeling human interactions". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 22:8, 2000, pp. 831–843.

174. A. Oord, Y. Li, and O. Vinyals. "Representation Learning with Contrastive Predictive Coding". In: 2018. URL: arXiv%20preprint%20arXiv:1807.03748.

175. M. Opitz, G. Waltner, H. Possegger, and H. Bischof. "Deep metric learning with BIER: Boosting independent embeddings robustly". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.

176. M. Padberg and G. Rinaldi. "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems". *SIAM review* 33:1, 1991.

177. S. J. Pan and Q. Yang. "A Survey on Transfer Learning". *IEEE Transactions on Knowledge and Data Engineering* 22, 2010, pp. 1345–1359.

178. M. Pantic, A. Pentland, A. Nijholt, and T. S. Huang. "Human computing and machine understanding of human behavior: A survey". In: *Artifical Intelligence for Human Computing*. Springer, 2007, pp. 47–71.

179. O. M. Parkhi, A. Vedaldi, and A. Zisserman. "Deep Face Recognition". In: *British Machine Vision Conference*. 2015.

180. A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. "Automatic differentiation in PyTorch". In: *Advances in Neural Information Processing Systems (NeurIPS) Workshops*. 2017.

181. H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean. "Efficient Neural Architecture Search via Parameter Sharing". *International Conference on Machine Learning (ICML)*, 2018.

182. J. Pu, Y.-G. Jiang, J. Wang, and X. Xue. "Which Looks Like Which: Exploring Inter-class Relationships in Fine-Grained Visual Categorization". In: *European Conference on Computer Vision (ECCV)*. 2014.

183. Q. Qian, L. Shang, B. Sun, J. Hu, H. Li, and R. Jin. "SoftTriple Loss: Deep Metric Learning Without Triplet Sampling". In: *International Conference on Computer Vision (ICCV)*. 2019.

184. P. Ramachandran, B. Zoph, and Q. V. Le. "Searching for Activation Functions". *CoRR* abs/1710.05941, 2017.

185. J. F. Randrianasoa, C. Kurtz, É. Desjardin, and N. Passat. "Binary Partition Tree construction from multiple features for image segmentation". *Pattern Recognition (PR)* 84, 2018, pp. 237–250.

186. S. Ravi and H. Larochelle. "Optimization as a Model for Few-Shot Learning". In: *International Conference on Learning Representations (ICLR)*. 2017.

187. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

188. C.-X. Ren, J.-Z. Li, P. Ge, and X.-L. Xu. "Deep metric learning via subtype fuzzy clustering". *Pattern Recognition (PR)* 90, 2019, pp. 210–219.

189. S. Ren, K. He, R. B. Girshick, and J. Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015.

190. S. Ren, K. He, R. Girshick, and J. Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett. 2015.

191. M. Reyes, G. Dominguez, and S. Escalera. "Featureweighting in dynamic time-warping for gesture recognition in depth data". In: *International Conference on Computer Vision Workshops (ICCV Workshops)*. 2011.

192. L. G. Roberts. *Machine Perception of Three-Dimensional Solids*. Outstanding Dissertations in the Computer Sciences. Garland Publishing, New York, 1963.

193. O. Ronneberger, P.Fischer, and T. Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. 2015.

194. K. Roth, B. Brattoli, and B. Ommer. "MIC: Mining Interclass Characteristics for Improved Metric Learning". In: *International Conference on Computer Vision (ICCV)*. 2019.

195. K. Roth, T. Milbich, and B. Ommer. "PADS: Policy-Adapted Sampling for Visual Similarity Learning". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

196. K. Roth, T. Milbich, S. Sinha, P. Gupta, B. Ommer, and J. P. Cohen. "Revisiting Training Strategies and Generalization Performance in Deep Metric Learning". In: *International Conference on Machine Learning (ICML)*. 2020.

197. J. C. Rubio, A. Eigenstetter, and B. Ommer. "Generative Regularization with Latent Topics for Discriminative Object Recognition". *Pattern Recognition* 48:12, 2015.

198. Y. Rubner, C. Tomasi, and L. J. Guibas. "The Earth Mover's Distance as a Metric for Image Retrieval". *International Journal of Computer Vision* 40, 2000, pp. 99–121.

199. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. "Learning Representations by Back-Propagating Errors". In: *Neurocomputing: Foundations of Research*. MIT Press, 1988, pp. 696–699.

200. S. Sadanand and J. J. Corso. "Action bank: A high-level representation of activity in video". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

201. A. Sanakoyeu, V. Tschernezki, U. Buchler, and B. Ommer. "Divide and Conquer the Embedding Space for Metric Learning". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

202. K. van de Sande, T. Gevers, and C. Snoek. "Evaluating Color Descriptors for Object and Scene Recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 32, 2010, pp. 1582–1596.

203. B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. "Application of Dimensionality Reduction in Recommender Systems: A case study". In: *WebKDD Workshop at the ACM SIGKKD*. 2000.

204. S. Saxena, O. Tuzel, and D. DeCoste. "Data Parameters: A New Family of Parameters for Learning a Differentiable Curriculum". In: *Advances in Neural Information Processing Systems*. 2019.

205. F. Schroff, D. Kalenichenko, and J. Philbin. "Facenet: A unified embedding for face recognition and clustering". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

206. J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. "Trust Region Policy Optimization". In: *International Conference on Machine Learning (ICML)*. 2015.

207. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. "Proximal Policy Optimization Algorithms". *CoRR*, 2017.

208. O. Shahar, A. Faktor, and M. Irani. "Super-Resolution from a Single Video". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

209. J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004. ISBN: 0521813972.

210. E. Shelhamer, J. Long, and T. Darrell. “Fully Convolutional Networks for Semantic Segmentation”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

211. H. Shi, Y. Yang, X. Zhu, S. Liao, Z. Lei, W. Zheng, and S. Li. “Embedding Deep Metric for Person Re-identification: A Study Against Large Variations”. In: *European Conference on Computer Vision (ECCV)*. 2016.

212. T. Shi, J. Steinhardt, and P. Liang. “Learning where to sample in structured prediction”. In: *Artificial Intelligence and Statistics*. 2015.

213. L. Sigal, A. O. Balan, and M. J. Black. “Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion”. *International journal of computer vision* 87:1-2, 2010.

214. K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *International Conference on Learning Representations (ICLR)*. 2015.

215. S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le. “Don’t decay the learning rate, increase the batch size”. In: *International Conference on Learning Representations (ICLR)*. 2018.

216. S. W. Smoliar and H. Zhang. “Content based video indexing and retrieval”. *IEEE Multimedia* 1:2, 1994, pp. 62–72.

217. K. Sohn. “Improved deep metric learning with multi-class n-pair loss objective”. In: *Advances in Neural Information Processing Systems*. 2016.

218. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. *Journal of Machine Learning Research (JMLR)*, 2014.

219. B. Su, J. Zhou, X. Ding, H. Wang, and Y. Wu. “Hierarchical Dynamic Parsing and Encoding for Action Recognition”. In: *European Conference on Computer Vision (ECCV)*. 2016.

220. Ö. Sümer, T. Dencker, and B. Ommer. “Self-supervised Learning of Pose Embeddings from Spatiotemporal Relations in Videos”. In: *International Conference on Computer Vision (ICCV)*. 2017.

221. R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018.

222. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. “Going Deeper with Convolutions”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

223. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. “Going deeper with convolutions”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

224. Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

225. *The sphere game in n dimensions*. http://faculty. madisoncollege.edu/alehnen/sphere/hypers.htm., 2017.

226. B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. "YFCC100M: The New Data in Multimedia Research". *Commun. ACM* 59, 2016, pp. 64–73.

227. Y. Tian, D. Krishnan, and P. Isola. "Contrastive Multiview Coding". In: *European Conference on Computer Vision (ECCV)*. 2020.

228. N. Tishby and N. Zaslavsky. *Deep Learning and the Information Bottleneck Principle*. 2015.

229. A. Toshev and C. Szegedy. "Deeppose: Human pose estimation via deep neural networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

230. M. Turk and A. Pentland. "Eigenfaces for Recognition". *J. Cognitive Neuroscience* 3, 1991, pp. 71–86.

231. S. Ullman and S. Soloviev. "Computation of Pattern Invariance in Brain-like Structures". *Neural Netw.* 12, 1999, pp. 1021–1036.

232. E. Ustinova and V. Lempitsky. "Learning deep embeddings with histogram loss". In: *Advances in Neural Information Processing Systems*. 2016.

233. V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, D. Lopez-Paz, and Y. Bengio. "Manifold Mixup: Better Representations by Interpolating Hidden States". In: ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 6438–6447.

234. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. "Extracting and Composing Robust Features with Denoising Autoencoders". In: *International Conference on Machine Learning 2008 (ICML)*. 2008.

235. O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. "Matching Networks for One Shot Learning". In: *International Conference on Neural Information Processing Systems*. 2016.

236. C. Vondrick, H. Pirsiavash, and A. Torralba. "Anticipating visual representations from unlabeled video". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

237. C. Vondrick, H. Pirsiavash, and A. Torralba. "Generating videos with scene dynamics". In: *Advances In Neural Information Processing Systems*. 2016.

238. J. Wagemans, J. H. Elder, M. Kubovy, S. E. Palmer, M. A. Peterson, M. Singh, and R. von der Heydt. "A century of Gestalt psychology in visual perception: I. Perceptual grouping and figure ground organization." *Psychological bulletin* 138, 2012, p. 1172.

239. C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. "The caltech-ucsd birds-200-2011 dataset", 2011.

240. J. Walker, A. Gupta, and M. Hebert. "Patch to the future: Unsupervised visual prediction". In: *Conference on Computer Vision and Pattern Recognition (CVPR*. 2014.

241. J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li. "Deep Learning for Content-Based Image Retrieval: A Comprehensive Study". In: *ACM International Conference on Multimedia*. 2014.

242. D. Wang and X. Tan. "Unsupervised feature learning with C-SVDDNet". *Pattern Recognition (PR)* 60, 2016, pp. 473–485.

243. H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. "CosFace: Large Margin Cosine Loss for Deep Face Recognition". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

244. H. Wang, A. Klaser, C. Schmid, and C.-L. Liu. "Action Recognition by Dense Trajectories". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

245. J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin. "Deep metric learning with angular loss". In: *International Conference on Computer Vision (ICCV)*. 2017.

246. M. Wang and W. Deng. "Deep Visual Domain Adaptation: A Survey". *Neurocomputing* 312, 2018, pp. 135–153.

247. X. Wang and A. Gupta. "Unsupervised learning of visual representations using videos". In: *International Conference on Computer Vision (ICCV)*. 2015.

248. X. Wang and A. Gupta. "Unsupervised Learning of Visual Representations Using Videos". In: *International Conference on Computer Vision (ICCV)*. 2015.

249. X. Wang, K. He, and A. Gupta. "Transitive Invariance for Self-Supervised Visual Representation Learning". In: *International Conference on Computer Vision (ICCV)*. 2017.

250. X. Wang, Y. Hua, E. Kodirov, G. Hu, R. Garnier, and N. M. Robertson. "Ranked List Loss for Deep Metric Learning". *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

251. X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott. "Multi-Similarity Loss with General Pair Weighting for Deep Metric Learning". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

252. K. Q. Weinberger and L. K. Saul. "Distance Metric Learning for Large Margin Nearest Neighbor Classification". *Journal of Machine Learning Research* 10, 2009, pp. 207–244.

253. K. R. Weiss, T. Khoshgoftaar, and D. Wang. "A survey of transfer learning". *Journal of Big Data* 3, 2016, pp. 1–40.

254. Y. Weiss, A. Torralba, and R. Fergus. "Spectral Hashing". In: *International Conference on Neural Information Processing Systems*. 2008.

255. R. J. Williams. "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning". *Machine Learning*, 1992.

256. C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl. "Sampling matters in deep embedding learning". In: *International Conference on Computer Vision (ICCV)*. 2017.

257. Z. Wu, Y. Xiong, X. Y. Stella, and D. Lin. "Unsupervised Feature Learning via Non-Parametric Instance Discrimination". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

258. Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. "Unsupervised Feature Learning via Non-Parametric Instance Discrimination". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

259. S. Xie, H. Zheng, C. Liu, and L. Lin. "SNAS: Stochastic Neural Architecture Search". In: 2019.

260. W. Xiong, L. Zhang, B. Du, and D. Tao. "Combining local and global: Rich and robust feature pooling for visual recognition". *Pattern Recognition (PR)* 62, 2017, pp. 225–235.

261. H. Xuan, R. Souvenir, and R. Pless. "Deep Randomized Ensembles for Metric Learning". In: *European Conference on Computer Vision (ECCV)*. 2018.

262. H. Xuan, A. Stylianou, and R. Pless. "Improved Embeddings with Easy Positive Triplet Mining". In: *Winter Conference on Applications of Computer Vision (WACV)*. 2020.

263. S. Yang, P. Luo, C. Change Loy, K. W. Shum, and X. Tang. "Deep Representation Learning with Target Coding". In: *AAAI Conference on Artificial Intelligence (AAAI)*. 2015.

264. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. "How transferable are features in deep neural networks?" In: *Advances in Neural Information Processing Systems*. 2014.

265. B. Yu, T. Liu, M. Gong, C. Ding, and D. Tao. "Correcting the Triplet Selection Bias for Triplet Loss". In: *European Conference on Computer Vision (ECCV)*. 2018.

266. T. Yuan, W. Deng, J. Tang, Y. Tang, and B. Chen. "Signal-to-Noise Ratio: A Robust Distance Metric for Deep Metric Learning". *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

267. Y. Yuan, K. Yang, and C. Zhang. "Hard-aware deeply cascaded embedding". In: *International Conference on Computer Vision (ICCV)*. 2017.

268. A. Zhai and H.-Y. Wu. "Classification is a Strong Baseline for Deep Metric Learning". In: *British Machine Learning Conference (BMVC)*. 2019.

269. J. Zhang, K. Mei, Y. Zheng, and J. Fan. "Learning multi-layer coarse-to-fine representations for large-scale image classification". *Pattern Recognition (PR)* 91, 2019, pp. 175–189.

270. P. Zhang, W. Liu, Y. Lei, and H. Lua. "Hyperfusion-Net: Hyper-densely reflective feature fusion for salient object detection". *Pattern Recognition (PR)* 93, 2019, pp. 521–533.

271. R. Zhang, P. Isola, and A. A. Efros. "Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

272. R. Zhang, P. Isola, and A. A. Efros. "Colorful Image Colorization". In: *European Conference on Computer Vision (ECCV)*. 2016.

273. W. Zhang, M. Zhu, and K. G. Derpanis. "From actemes to action: A strongly-supervised representation for detailed action understanding". In: *International Conference on Computer Vision (ICCV)*. 2013.

274. Y. Zhang, Y. Bai, M. Ding, Y. Li, and B. Ghanem. "Weakly-supervised object detection via mining pseudo ground truth bounding-boxes". *Pattern Recognition (PR)* 84, 2018, pp. 68–81.

275. Y. Zhao, Z. Jin, G.-j. Qi, H. Lu, and X.-s. Hua. "An Adversarial Approach to Hard Triplet Generation". In: *European Conference on Computer Vision (ECCV)*. 2018.

276. X. Zhe, S. Chen, and H. Yan. "Directional statistics-based deep metric learning for image classification and retrieval". *Pattern Recognition* 93, 2018.

277. W. Zheng, Z. Chen, J. Lu, and J. Zhou. "Hardness-Aware Deep Metric Learning". *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

278. F. Zhou, F. De la Torre, and J. K. Hodgins. "Hierarchical aligned cluster analysis for temporal clustering of human motion". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 35:3, 2013.

279. D. Zoccolan, N. Oertelt, J. Dicarlo, and D. Cox. "A rodent model for the study of invariant visual object recognition". *National Academy of Sciences of the United States of America* 106, 2009, pp. 8748–8753.