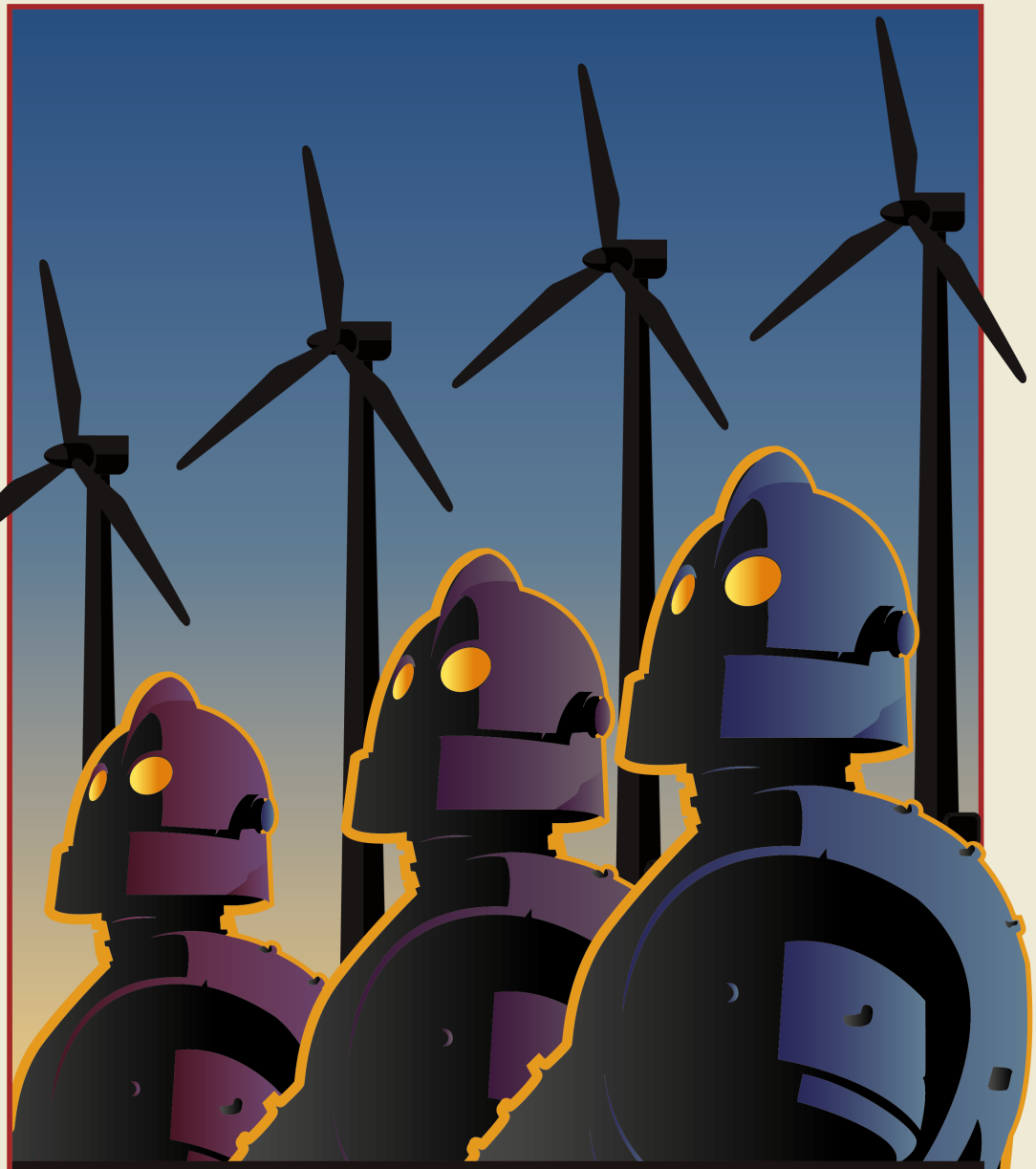
Symmetry-based learning from limited data

Sosnovik, I.

Publication date
2023
Document Version
Final published version

IVAN SOSNOVIK



SYMMETRY-BASED LEARNING FROM LIMITED DATA

Symmetry-Based Learning From Limited Data

Ivan Sosnovik

This book was typeset by the author using LaTeX 2_ε.

Symmetry-Based Learning From Limited Data

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Aula der Universiteit
op donderdag 15 juni 2023, te 14.00 uur

door Ivan Sosnovik
geboren te Moskou

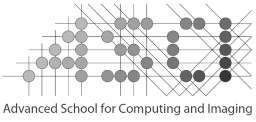
Promotiecommissie

Promotor:	Prof. dr. ir. A. W. M. Smeulders	Universiteit van Amsterdam
Copromotor:	Prof. dr. C. G. M. Snoek	Universiteit van Amsterdam
Overige leden:	Prof. dr. M. Worring	Universiteit van Amsterdam
	Prof. dr. M. Welling	Universiteit van Amsterdam
	dr. J.H. Jacobsen	Apple Inc.
	dr. E. Gavves	Universiteit van Amsterdam
	Prof. dr. T. Lindeberg	KTH Stockholm

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

UNIVERSITEIT VAN AMSTERDAM

The work described in this thesis has been carried out within the graduate school ASCI, at the Video and Image Sense lab, and the Delta lab of the University of Amsterdam.



CONTENTS

1

INTRODUCTION

1.1 LEARNING TO SOLVE

While mathematics is beautiful by itself, it allows us to come up with elegant solutions to the most sophisticated problems. What once started as a dream of flying later combined with mathematics, advances in engineering, and electronics to give us planes and rockets. We developed control theory to use planes safely, launch rockets to the most distant places, and protect ourselves from various threats. The derivative disciplines of control theory such as pattern recognition, machine learning and reinforcement learning, allow us to diagnose diseases, entertain us, meet new people, save money, draw, and invent. With the rise of computers, what was previously seen as an unsolvable problem, nowadays is just a matter of compute power. What previously required a concrete and specific algorithm, can now be derived by a computer from data, derived in a similar way as we have learned the concepts of the world ourselves. And this is the focus of our research.

In the end, the Artificial Neural Network is a mathematical model, inspired by the way human brains work. It takes input in the form of a table of numbers and applies simple linear and non-linear functions many times one after another until an answer is calculated. Then the answer is compared to the label data indicating what is to be the correct outcome. The parameters of neural networks are adjusted by the discrepancy to make it come closer to the ground truth. After repeating such an operation a sufficient number of times for all given examples, the neural network will hopefully produce the correct output. Neural networks are among the most powerful algorithms of the day. With them, we can build an image classifier, a music synthesizer, a text translator and even an autopilot. It can be hard to reach a formal solution for a given task. However, software libraries are now so efficient that a program that will train a neural network for this task can sometimes be written even in a day.

Training of a model depends on many factors and may take days or even months. One of the most important factors which affect the final result is the training data. Groundbreaking research which resulted in the Turing Award of 2018 [119] required Dr. Yann LeCun to collect a dataset of handwritten digits [97]. Nowadays, this dataset can be downloaded within a fraction of a second. Modern devices acquire tons of information from various modalities every minute. Such an enormous amount of data allows us to train models which are woven into our daily routine to make it more comfortable.

We can go even further, we can build models which will analyze information that was either limited or even absent in the past. Living in society, raised by generations of educated people, we are able to predict facts even without experiencing them again and again. A basic concern for everyday life is a drive to safety. We learn how to live

safely, and how to interact with the world without too much risk. In fact, in our human life, we learn a lot about security and safety not from our own experience, but from education. Therefore, we aim to bring into neural networks experience which captures the knowledge handed down over generations. Here we do not consider rules of knowledge as documented since ancient times in law or in encyclopedias since Comenius or Diderot and d'Alembert [40], but experience captured in ways to handle the generalities in the data.

A significant part of our education takes the form of rules. Rules help us to choose the best option fast, they serve as guidelines. A significant part of our education is physical laws. Such laws act as constraints. The constraints tell us what outcomes are possible and what are impossible at all. With just a few physical laws in mind, we can readily predict where the cars around us will be next, and how they slow down. For example, while driving, we are able to turn to both sides, but some actions are highly not recommended for the sake of safety. We do not have to try all possible options and check their outcomes. Instead, we can follow the rules, guidelines and best practices to make driving as effective as possible without spending too much time. Guidelines and constraints are a part of every education, after which, with some practice, you can learn new things in various domains: be it skydiving, photography, snowboarding, cooking, Uruguayan tango or Rubik's cube.

While neural networks are ultimate learners when it comes to large amounts of data, they lack the knowledge of physical laws. Current neural networks make no distinction between instructional rules and physical laws. They treat them the same: all rules are learned implicitly as much as they appear from the data. When physical impossibilities would be available to the network, such knowledge would allow neural networks to learn from fewer examples. It will make trained models more reliable and safer, as the constraints of our world are part of their constraints as well.

1.2 SYMMETRIES

Rather than focusing on physical laws themselves we focus on their key ingredient — symmetries. Symmetries of the observed world are the foundations of Special Relativity Theory and Field Theory [12].

Symmetries are transformations of the space that map it to itself unchanged [13]. For example, a ball can be rotated by an arbitrary angle and it will still be a ball that looks exactly the same. A square can be rotated by 90°. A palindrome can be read left-to-right and right-to-left. They are the symmetries of the objects. If we know all admissible symmetries, we do not need to learn all variations in appearance but simply specify one example and say "et cetera" by specifying all other admissible appearances from there. Ideally, in the absence of variations in the recording, to cover all variation of one object it suffices to specify one example if we were certain we have covered all the symmetries. Firstly, the appearance effect of all symmetries is not fully predictable. Hence, one cannot rely on one observation. When the light comes from the right, rotating a square object will cause unknown changes in appearance also depending on the paint of the object. Secondly, symmetries may be too wide a class and hence reduce discriminability. Pictures of humans are almost exclusively standing up vertically or

lying down horizontally. When we assume the full rotational symmetry, we effectively insert pictures which are physically possible but rare in practice as the intermediate rotation poses are not stable. We will study the effectiveness of symmetry classes in classification distinction and effectiveness throughout this research

Symmetries set important constraints on how the information about the object can be retrieved and interpreted. In this research, we focus on two constraints: invariance and equivariance. An invariant measure or a function is one that does not change when the input is changed [13]. For example, the mass of an object is invariant under its orientation or illuminations, so is its class. A function is equivariant if it changes accordingly with the input [13]. For instance, the mass of a balloon increases proportional to the additional air we pump in it. Combinations of invariant and equivariant functions are of high interest to us as they allow for strong and flexible reasoning about the observed world.

1.3 RESEARCH QUESTIONS

1.3.1 *Scale-Equivariant Networks*

There are very frequent transformations, which occur in real life and we are very well aware of them. However, neural networks do not consider them as simple as we do. Objects may rotate and move in all possible directions which results in a significant variation in the way they are represented in a video or in an image. For an object classifier, the correct answer should not depend on the position of the target object, on its orientation, and ideally also not on the illumination nor on minor occlusions. For an object-segmentation network or an object detector, the prediction should change with the position of the object changes. Such behaviors can be learned from large training datasets. This is not an option for extreme yet very important problems: uncommon car accidents, personalized treatment of diseases or analysis of traumas. A reliable network should tackle the principle transformations in a predictable manner regardless of the amount of training data.

Can we build neural networks which are aware of the principle geometric symmetries without learning them?

There are just three symmetry transformations of the plane that keep shape unchanged: translation, rotation and scaling [13]. Equipping neural networks with translation equivariance was a groundbreaking result [97]. After years of subsequent improvements and modifications of the original idea, what once was available to big companies and only for document classification. Nowadays, it allows billions of users to search for any visual information they require. In 2016 Cohen and Welling [25] extended the symmetry property of neural networks to rotation. This idea allowed for more accurate image classifiers [25, 156] and object trackers [63], it also demonstrated that it is possible to learn effectively from less when it comes to medical data [152]. Last but not least transformation, scaling, is one of the key focuses of this research.

Scale transformations occur in various image and video analysis tasks. They are frequent as a result of the varying distances between the camera and the object. It s an example of transformations that allow us to better understand the geometry of the world:

smaller otherwise similarly looking objects tell us that they are farther from us than their larger counterparts. However, for Convolutional Neural Networks such differences due to transformations over scale cause confusion and lead to incorrect predictions. In image classification it is usually sufficient to make the prediction independent, in other words invariant, to scale transformations. Be it a scan of a document, a group photo or a satellite image, the target object may vary in size, but its class should stay the same. If we consider videos from a dashcam, recordings of sports events, and streams of visual data from drones' sensors, the objects of interests may move very fast with respect to the camera. Thus, they rapidly change their retinal size, sometimes by a factor of 10 or even more [138]. In all of these cases, the visual analysis system should be equivariant to such transformations. Both the predicted information about the size and the distance to the object should be updated accordingly when the input changes.

To answer the research question we develop a theory of scale-equivariant convolutional neural networks. We demonstrate the idea of filter reparametrization and an algorithm for fast change of filter scales without back-and-forth resizing of 2-dimensional tensors. We demonstrate how the most common building blocks of neural networks can be extended to scale equivariance: convolution, batch normalization, nonlinearities, spatial pooling. We also demonstrate how one can transform a scale-equivariant neural network into a scale-invariant function. We present a recipe for how to make an arbitrary CNN-based model scale equivariant, by simply drop-in replacing its building blocks with the proposed ones. In our experiments, we demonstrate, that by using our recipe, one can train more accurate image classifiers which set new state-of-the-art results on several benchmarks. We also demonstrate, that in visual object tracking, one can achieve much more accurate results when it comes to significant size variations and keep it real-time. We demonstrate that scale-equivariant neural networks consider scale variations as a valuable source of information and not as a disturbing factor. As a result, they learn from less data and learn to be more robust.

1.3.2 *Accurate Discrete Symmetry-Based Neural Networks*

While transformation equivariance and invariance serve as very essential inductive biases, help to learn from less data, and be better interpretable and reliable, these properties are mathematical abstractions. The core of scale-equivariant convolutional neural networks is rooted in symmetries of continuous algebraic functions. Although the observed world can be described with continuous functions, they are not mathematically identical to the latter [88, 157], which already demonstrates a gap between the theory and the observed world. In addition, we do not analyze the analog world directly. Instead, we digitize the world first either with a digital camera or by performing discretization later in the computer. Thus, the signals from the world come in the form of arrays, and discrete tables with discrete numbers. It results in a discrepancy between the symmetries of the real world and the symmetries we assume while building the models.

How to match the scale symmetries of the observed world with the scale symmetries of mathematical structures?

Starting from the first principles, we derive general constraints, under which scale-equivariant neural networks are indeed equivariant under scale transformations of input,

parametrized as images or frames of a video. We find that there are two distinct cases. For integer scale factors, which transform the original image into another one that lies on exactly the same grid, the exact solution exists. It causes no extra error, and it matches with what is also called dilated convolution. For all other cases, there is no exact solution. However, we demonstrate that it is possible to learn an approximate solution by minimizing the discrepancy between the theoretical and real output of the layer when it's calculated on real data. After analyzing the solutions we conclude that the optimal convolutional filter is very sparse and thus allows for faster operations.

In our experiments, we compare various scale-equivariant and scale-invariant models in terms of their accuracy and performance in image classification and video object tracking. We demonstrate that by simply replacing one set of filters with another set we improve the model. After comparing several models side-by-side we conclude that the lower the discrepancy between the real symmetries and the mathematical symmetries are, the better the model performs. And the best-performing model is the model which uses the filter of two folds: learned and integer-scaled. Thus this model is a solution to the research question.

1.3.3 *Functional Variations in Neural Networks*

Although scale-equivariant neural networks do not cover the whole set of possible transformations, they have a very important property to the end-user of the technology—robustness. Scale-equivariant networks process scale transformations of the input in a predictable, pre-defined, theoretically justified way. Thus, they allow for building a robust CNN-based model. If we develop an autopilot for an autonomous vehicle, ideally we would like to know what the scenarios to be processed are. This is impossible mostly because of the complexity of the world. However, we know, that if our scale-equivariant model is effective at classifying traffic signs and it is accurate at tracking pedestrians and other vehicles, it will also be accurate on these tasks when the objects change in size because of the increased or decreased distance to them. Thus, we can rely on this technology, because this predictable behavior serves as a constraint built into the models and it minimizes the risk of incorrect outcomes due to distance.

In the same way, we can robustify our models with respect to other well-grounded symmetry transformations: translation, rotation, horizontal and vertical flips. But is our model robust to more often, yet subtle variations such as snow, frosted glass, rain of various severity, a combination of fog and insufficient illumination or partial occlusion? We can train the autopilot by providing more and more data from many extreme scenarios, but it increases the cost of the technology significantly and making it less accessible to a wider audience.

Scale, rotation or translation symmetries can be baked into neural networks for more robust behavior mostly because these transformations are well-defined mathematical structures, a part of the group of affine transformations. These transformations are the most studied and the most frequent. However, even transformations which slightly differ from affine transformations are hard to put into neural networks from the start. Thus, the models have to learn them from the data, which raises the question, whether they are reliable enough to be deployed.

How to make models robust to subtle variations that go beyond affine transformations?

To answer this question, we start with the simple idea that the usual way of making neural networks robust to unseen variations, is to train them using sophisticated data augmentation. The output of a convolutional layer will, however, stay the same if we transform the filter instead of the input image. A rotation in the input image can be compensated by a reverse rotation of the filter. For a flipped input, there is a filter with a corresponding flip. If we simulate a snow effect on the input, there is a special augmentation of the filter which generates the same output. The plethora of possible transformations of the filter lies in the vicinity of the original filter in the functional space. Indeed, we can represent the filter as a function of some hidden parameters, perform a Taylor expansion of the filter in functional space with respect to these parameters, slightly vary them, and take only the linear part of this series. Thus, for every set of subtle input transformations, there is a set of linear transformations of the filter.

We introduce a new class of convolutional neural networks, which consist not only of trainable convolutional filters but also involve a wide set of filter transformations, which are all utilized during training. We also propose an algorithm for fast implementation of these new models, which allows for a reasonable trade-off between the training speed and the resulting robustness. We demonstrate that by using our algorithm, one can take any convolutional model, replace the standard convolutional layers with the proposed ones, retrain the model, and gain a significant boost in accuracy on unseen transformations. Our models demonstrate increased robustness to a wide set of real-life transformations: occlusion, scale, blur, noise, viewpoint variations and some natural phenomena.

1.3.4 *Unsupervised Learning With Symmetries*

So far, we considered the cases when the model processes something after deployment which was unseen at the training stage. We demonstrated that if the unseen image can be transformed back to a sample from the training subset with some symmetry, our models can still process it correctly. It allows us to build complex yet reliable models. This approach resembles how students solve problems, having learned how to solve similar problems during the course. When we, students of the past, start working with the world of problems, we no longer see equations, we no longer solve the tasks we were shown in the classroom. Instead, we have a feeling, an intuition, some common sense, that if there is a problem, the solution will have some structure. It will have some symmetry. Sometimes, this symmetry is not absolute. Sometimes, the problem can be solved even sooner than we can understand that there is some underlying beautiful law. However, after observing a lot of data, solving many problems, finding thousands of previously unseen solutions, we distill this knowledge to a new law, to a new method, to a new best practice. And finally, the next generations of students will learn about it in their classrooms.

With this process, we see that there are at least two distinct cases of learning. In the supervised learning task [97], we compare new data to what was already learned prior to the task. In tasks such as unsupervised [58], self-supervised [83] or zero short learning [167], the direct comparison to previously seen data is not feasible. We saw that

for supervised learning a very effective tool for performing accurate and robust learning is to assume that the unseen data will be related to the previously seen data via some symmetry transformation. We can also assume that once we are given some data for the second case of learning, its data also has internal symmetries with respect to itself.

How to make models learn from unlabeled data by assuming symmetries?

As this research question is general, we consider a special case. We focus on video summarization. The task of video summarization is to select a subset of the video which serves as a summary of the source. How to summarize? What to highlight? What information seems more important than the other? These questions cannot be answered by comparing the data to the previously seen dataset. Instead, we consider every video as a set of short clips. We project each clip to some hidden space where we assume that they form symmetric clusters, such as circles, and then select a subset that shares the same structure. The model learns how to project clips and how to select from symmetric structures, by comparing clips one to one another, from one video to clips from another video. We suggest an effective implementation of the proposed approach. In our experiments, we demonstrate that our approach allows for improving a wide range of video sumamrizers just by changing the way they learn.

1.4 CO-AUTHORSHIP AND ROLES

The thesis is composed of the following original contributions.

CHAPTER 2 This chapter is based on "Scale-Equivariant Steerable Networks", published in *International Conference on Learning Representations (ICLR)*, 2020 [144], by Ivan Sosnovik, Michał Szmaja and Arnold W. M. Smeulders.
Contribution of authors

- Ivan Sosnovik: all aspects
- Michał Szmaja: technical implementation
- Arnold W. M. Smeulders: supervision and insights

CHAPTER 3 This chapter is based on "Scale Equivariance Improves Siamese Tracking ", published in *Winter Conference on Applications of Computer Vision (WACV)*, 2021 [142], by Ivan Sosnovik, Artem Moskalev and Arnold W. M. Smeulders.
Contribution of authors

- Ivan Sosnovik: all aspects
- Artem Moskalev: theoretical and technical implementation
- Arnold W. M. Smeulders: supervision and insights

CHAPTER 4 This chapter is based on "DISCO: Accurate Discrete Scale Convolutions", published in *The British Machine Vision Conference (BMVC)*, 2021 [140], by Ivan Sosnovik, Artem Moskalev and Arnold W. M. Smeulders. (**Best Paper Award**) and "How To Transform Kernels for Scale-Convolutions", published in *International Conference on Computer Vision (ICCV), 2nd Visual Inductive Priors for Data-Efficient Deep Learning Workshop*, 2021 [141], by Ivan Sosnovik, Artem Moskalev and Arnold W. M. Smeulders.
Contribution of authors

- Ivan Sosnovik: theoretical implementation
- Artem Moskalev: technical implementation
- Arnold W. M. Smeulders: supervision and insights

CHAPTER 5 This chapter is based on "Wiggling Weights to Improve the Robustness of Classifiers", by Sadaf Gulshad, Ivan Sosnovik and Arnold W. M. Smeulders.

Contribution of authors

- Sadaf Gulshad: all aspects
- Ivan Sosnovik: theoretical and technical implementation
- Arnold W. M. Smeulders: supervision and insights

CHAPTER 6 This chapter is based on "Learning to Summarize Videos by Contrasting Clips", by Ivan Sosnovik, Artem Moskalev, Cees Kaandorp and Arnold W. M. Smeulders.
Contribution of authors

- Ivan Sosnovik: all aspects
- Artem Moskalev: theoretical implementation
- Cees Kaandorp: technical implementation
- Arnold W. M. Smeulders: supervision and insights

During our research we published other papers which are not a part of this thesis but influenced our broader understanding of the problems of Artificial Intelligence.

- "Semi-conditional normalizing flows for semi-supervised learning", published in *First workshop on Invertible Neural Networks and Normalizing Flows at International Conference on Machine Learning (ICML)*, 2019 [2], by Andrei Atanov, Alexandra Volokhova, Arsenii Ashukha, Ivan Sosnovik, Dmitry Vetrov

- "Relational Prior for Multi-Object Tracking", published in *International Conference on Computer Vision (ICCV), 2nd Visual Inductive Priors for Data-Efficient Deep Learning Workshop*, 2021 [112], by Artem Moskalev, Ivan Sosnovik and Arnold W. M. Smeulders.
- "Built-in Elastic Transformations for Improved Robustness", preprint 2021 [61], by Sadaf Gulshad, Ivan Sosnovik, Arnold W. M. Smeulders.
- "Two is a crowd: tracking relations in videos", preprint 2021 [113], by Artem Moskalev, Ivan Sosnovik and Arnold W. M. Smeulders.
- "PIE: Pseudo-Invertible Encoder", preprint 2021 [7], by Jan Jetze Beitler, Ivan Sosnovik and Arnold W. M. Smeulders.
- "Contrasting quadratic assignments for set-based representation learning", published in *European Conference on Computer Vision (ECCV)*, 2022 [114], by Artem Moskalev, Ivan Sosnovik, Volker Fischer and Arnold W. M. Smeulders.
- "LieGG: Studying Learned Lie Group Generators", published in *Conference on Neural Information Processing Systems (NeurIPS)*, 2022, by Artem Moskalev, Ivan Sosnovik, Volker Fischer, Arnold W. M. Smeulders

2

SCALE-EQUIVARIANT STEERABLE NETWORKS

2.1 INTRODUCTION

Scale transformations occur in many image and video analysis tasks. They are a natural consequence of the variable distances among objects, or between objects and the camera. Such transformations result in significant changes in the input space which are often difficult for models to handle appropriately without careful consideration. At a high level, there are two modeling paradigms which allow a model to deal with scale changes: models can be endowed with an internal notion of scale and transform their predictions accordingly, or instead, models can be designed to be specifically invariant to scale changes. In image classification, when scale changes are commonly a factor of 2, it is often sufficient to make class prediction independent of scale. However, in tasks such as image segmentation, visual tracking, or object detection, scale changes can reach factors of 10 or more. In these cases, it is intuitive that the ideal prediction should scale proportionally to the input. For example, the segmentation map of a nearby pedestrian should be easily converted to that of a distant person simply by downscaling.

Convolutional Neural Networks (CNNs) demonstrate state-of-the-art performance in a wide range of tasks. Yet, despite their built-in translation equivariance, they do not have a particular mechanism for dealing with scale changes. One way to make CNNs account for scale is to train them with data augmentation [6]. This is, however, suitable only for global transformations. As an alternative, [72] and [146] use the canonical coordinates of scale transformations to reduce scaling to well-studied translations. While these approaches do allow for scale equivariance, they consequently break translation equivariance.

Several attempts have thus been made to extend CNNs to both scale and translation symmetry simultaneously. Some works use input or filter resizing to account for scaling in deep layers [84, 171]. Such methods are suboptimal due to the time complexity of tensor resizing and the need for interpolation. In [54] the authors pre-calculate filters defined on several scales to build scale-invariant networks, while ignoring the important case of scale equivariance. In contrast, [163] employ the theory of semigroup equivariant networks with scale-space as an example; however, this method is only suitable for integer downscale factors and therefore limited.

In this paper we develop a theory of scale-equivariant networks. We demonstrate the concept of steerable filter parametrization which allows for scaling without the need for tensor resizing. Then we derive scale-equivariant convolution and demonstrate a fast algorithm for its implementation. Furthermore, we experiment to determine to what degree the mathematical properties actually hold true. Finally, we conduct a set of

experiments comparing our model with other methods for scale equivariance and local scale invariance.

The proposed model has the following advantages compared to other scale-equivariant models:

1. It is equivariant to scale transformations with arbitrary discrete scale factors and is not limited to either integer scales or scales tailored by the image pixel grid.
2. It does not rely on any image resampling techniques during training, and therefore, produces deep scale-equivariant representations free of any interpolation artifacts.
3. The algorithm is based on the combination of tensor expansion and 2-dimensional convolution, and demonstrates the same computation time as the general CNN with a comparable filter bank.

2.2 PRELIMINARIES

Before we move into scale-equivariant mappings, we discuss some aspects of equivariance, scaling transformations, symmetry groups, and the functions defined on them. For simplicity, in this section, we consider only 1-dimensional functions. The generalization to higher-dimensional cases is straightforward.

EQUIVARIANCE Let us consider some mapping g. It is equivariant under L_θ if and only if there exists L'_θ such that $g \circ L_\theta = L'_\theta \circ g$. In case L'_θ is the identity mapping, the function g is invariant.

In this paper we consider scaling transformations. In order to guarantee the equivariance of the predictions to such transformations, and to improve the performance of the model, we seek to incorporate this property directly inside CNNs.

SCALING Given a function $f : R \to R$, a scale transformation is defined as follows:

$$L_s[f](x) = f(s^{-1}x), \quad \forall s > 0 \tag{2.1}$$

We refer to cases with $s > 1$ as *upscale* and to cases with $s < 1$ as *downscale*. If we convolve the downscaled function with an arbitrary filter ψ and perform a simple change of variables inside the integral, we get the following property:

$$\begin{aligned}
[L_s[f] \star \psi](x) &= \int_R L_s[f](x')\psi(x' - x)dx' \\
&= \int_R f(s^{-1}x')\psi(x' - x)dx' \\
&= s \int_R f(s^{-1}x')\psi(s(s^{-1}x' - s^{-1}x))d(s^{-1}x') \\
&= sL_s[f \star L_{s^{-1}}[\psi]](x)
\end{aligned} \tag{2.2}$$

In other words, *convolution of the downscaled function with a filter can be expressed through a convolution of the function with the correspondingly upscaled filter where*

downscaling is performed afterwards. Equation 2.2 shows us that the standard convolution is not scale-equivariant.

STEERABLE FILTERS In order to make computations simpler, we reparametrize $\psi_\sigma(x) = \sigma^{-1}\psi(\sigma^{-1}x)$, which has the following property:

$$L_{s^{-1}}[\psi_\sigma](x) = \psi_\sigma(sx) = s^{-1}\psi_{s^{-1}\sigma}(x) \tag{2.3}$$

It gives a shorter version of Equation 2.2:

$$L_s[f] \star \psi_\sigma = L_s[f \star \psi_{s^{-1}\sigma}] \tag{2.4}$$

We will refer to such a parameterization of filters as *Steerable Filters* because the scaling of these filters is the transformation of its parameters. Note that we may construct steerable filters from any function. This has the important consequence that it does not restrict our approach. Rather it will make the analysis easier for discrete data. Moreover, note that any linear combination of steerable filters is still steerable.

SCALE-TRANSLATION GROUP All possible scales form the scaling group S. Here we consider the discrete scale group, i.e. scales of the form $\dots a^{-2}, a^{-1}, 1, a, a^2, \dots$ with base a as a parameter of our method. Analysis of this group by itself breaks the translation equivariance of CNNs. Thus we seek to incorporate scale and translation symmetries into CNNs, and, therefore consider the Scale-Translation Group H. It is a semidirect product of the scaling group S and the group of translations $T \cong R$. In other words: $H = \{(s,t)|s \in S, t \in T\}$. For multiplication of group elements, we have $(s_2, t_2) \cdot (s_1, t_1) = (s_2 s_1, s_2 t_1 + t_2)$ and for the inverse $(s_2, t_2)^{-1} \cdot (s_1, t_1) = (s_2^{-1} s_1, s_2^{-1}(t_1 - t_2))$. Additionally, for the corresponding scaling and translation transformations, we have $L_{st} = L_s L_t \neq L_t L_s$, which means that the order of the operations matters.

From now on, we will work with functions defined on groups, i.e. mappings $H \to R$. Note, that simple function $f : R \to R$ may be considered as a function on H with constant value along the S axis. Therefore, Equation 2.4 holds true for functions on H as well. One thing we should keep in mind is that when we apply L_s to functions on H and R we use different notations. For example $L_s[f](x') = f(s^{-1}x')$ and $L_s[f](s', t') = f((s,0)^{-1}(s', t')) = f(s^{-1}s', s^{-1}t')$

GROUP-EQUIVARIANT CONVOLUTION Given group G and two functions f and ψ defined on it, G-equivariant convolution is given by

$$[f \star_G \psi](g) = \int_G f(g') L_g[\psi](g') d\mu(g') = \int_G f(g')\psi(g^{-1}g') d\mu(g') \tag{2.5}$$

Here $\mu(g')$ is the Haar measure also known as invariant measure [49]. For $T \cong R$ we have $d\mu(g') = dg'$. For discrete groups, the Haar measure is the counting measure, and integration becomes a discrete sum. This formula tells us that the output of the convolution evaluated at point g is the inner product between the function f and the transformed filter $L_g[\psi]$.

2.3 SCALE-EQUIVARIANT MAPPINGS

Now we define the main building blocks of scale-equivariant models.

SCALE CONVOLUTION In order to derive scale convolution, we start from group equivariant convolution with $G = H$. We first use the property of semidirect product of groups which splits the integral, then choose the appropriate Haar measures, and finally use the properties of steerable filters. Given the function $f(s,t)$ and a steerable filter $\psi_\sigma(s,t)$ defined on H, a scale convolution is given by:

$$\begin{aligned}[f \star_H \psi_\sigma](s,t) &= \int_S \int_T f(s',t') L_{st}[\psi_\sigma](s',t') d\mu(s') d\mu(t') \\ &= \sum_{s'} \int_T f(s',t') \psi_{s\sigma}(s^{-1}s', t'-t) dt' \\ &= \sum_{s'} [f(s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t)\end{aligned} \tag{2.6}$$

And for the case of C_{in} input and C_{out} output channels we have:

$$[f \star_H \psi_\sigma]_m(s,t) = \sum_{n=1}^{C_{\text{in}}} \sum_{s'} [f_n(s',\cdot) \star \psi_{n,m,s\sigma}(s^{-1}s',\cdot)](t), \quad m = 1 \dots C_{\text{out}} \tag{2.7}$$

Let us proof its equivariance.

Proof. Let us first show that scale-convolution is equivariant to translations.

$$\begin{aligned}[L_{\hat{t}}[f] \star_H \psi_\sigma](s,t) &= \sum_{s'} [L_{\hat{t}}[f](s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t) \\ &= \sum_{s'} L_{\hat{t}}[f(s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t) \\ &= L_{\hat{t}}\Big\{ \sum_{s'} [f(s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)] \Big\}(t) \\ &= L_{\hat{t}}[f \star_H \psi_\sigma](s,t)\end{aligned} \tag{2.8}$$

Now we show that scale convolution is equivariant to scale transformations:

$$\begin{aligned}[L_{\hat{s}}[f] \star_H \psi_\sigma](s,t) &= \sum_{s'} [L_{\hat{s}}[f](s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t) \\ &= \sum_{s'} L_{\hat{s}}[f(\hat{s}^{-1}s',\cdot) \star \psi_{\hat{s}^{-1}s\sigma}(s^{-1}s',\cdot)](t) \\ &= \sum_{s''} [f(s'',\cdot) \star \psi_{\hat{s}^{-1}s\sigma}(\hat{s}s^{-1}s'',\cdot)](\hat{s}^{-1}t) \\ &= [f \star_H \psi_\sigma](\hat{s}^{-1}s, \hat{s}^{-1}t) \\ &= L_{\hat{s}}[f \star_H \psi_\sigma](s,t)\end{aligned} \tag{2.9}$$

Finally, we can use the property of semidirect product of groups

$$\begin{aligned} L_{\hat{s}\hat{t}}[f] \star_H \psi_\sigma &= L_{\hat{s}} L_{\hat{t}}[f] \star_H \psi_\sigma = L_{\hat{s}}[L_{\hat{t}}[f] \star_H \psi_\sigma] \\ &= L_{\hat{s}} L_{\hat{t}}[f \star_H \psi_\sigma] = L_{\hat{s}\hat{t}}[f \star_H \psi_\sigma] \end{aligned} \tag{2.10}$$

□

In [90] the authors prove that a feed-forward neural network is equivariant to transformations from G if and only if it is constructed from G-equivariant convolutional layers. Thus Equation 2.7 shows the most general form of scale-equivariant layers which allows for building scale-equivariant convolutional networks with such choice of S. We will refer to models using scale-equivariant layers with steerable filters as Scale-Equivariant Steerable Networks, or shortly *SESN*[1]

NONLINEARITIES In order to guarantee the equivariance of the network to scale transformations, we use scale equivariant nonlinearities. We are free to use simple point-wise nonlinearities. Indeed, point-wise nonlinearities ν, like ReLU, commute with scaling transformations:

$$\begin{aligned} [\nu \circ L_s[f]](s', x') &= \nu(L_s[f](s', x')) = \nu(f(s^{-1}s', s^{-1}x')) \\ &= \nu[f](s^{-1}s', s^{-1}x') = [L_s \circ \nu[f]](s', x') \end{aligned} \tag{2.11}$$

POOLING Until now we did not discuss how to convert an equivariant mapping to an invariant one. One way to do this is to calculate the invariant measure of the signal. In case of translation, such a measure could be the maximum value for example.

First, we propose the maximum scale projection defined as $f(s, x) \to \max_s f(s, x)$. This transformation projects the function f from H to T. Therefore, the representation stays equivariant to scaling, but loses all information about the scale itself.

Second, we are free to use spatial max-pooling with a moving window or global max pooling. Transformation $f(s, x) \to \max_x f(s, x)$ projects the function f from H to S. The obtained representation is invariant to scaling in the spatial domain, however, it stores the information about scale.

Finally, we can combine both of these pooling mechanisms in any order. The obtained transformation produces a scale invariant function. It is useful to utilize this transformation closer to the end of the network, when the deep representation must be invariant to nuisance input variations, but already has very rich semantic meaning.

2.4 IMPLEMENTATION

In this paragraph we discuss an efficient implementation of Scale-Equivariant Steerable Networks. We illustrate all algorithms in Figure 1. For simplicity we assume that zero padding is applied when it is needed for both the spatial axes and the scale axis.

FILTER BASIS A direct implementation of Equation 2.7 is impossible due to several limitations. First, the infinite number of scales in S calls for a discrete approximation.

1 pronounced 'season'

We truncate the scale group and limit ourselves to N_S scales and use discrete translations instead of continuous ones. Training of SESN involves searching for the optimal filter in functional space which is a problem by itself. Rather than solving it directly, we choose a complete basis of N_b steerable functions $\Psi = \{\psi_{s^{-1}\sigma,i}\}_{i=1}^{N_b}$ and represent the convolutional filter as a linear combination of basis functions with trainable parameters $w = \{w_i\}_{i=1}^{N_b}$. In other words, we do the following substitution in Equation 2.7: $\psi_\sigma \to \kappa = \sum_i w_i \Psi_i$

In our experiments we use a basis of 2D Hermite polynomials with 2D Gaussian envelope, as it demonstrates good results. Assuming that the center of the filter is the point $(0, 0)$ in coordinates (x, y), we use the filters of the following form:

$$\psi_\sigma(x, y) = A\frac{1}{\sigma^2}H_n\Big(\frac{x}{\sigma}\Big)H_m\Big(\frac{y}{\sigma}\Big)\exp\Big[-\frac{x^2+y^2}{2\sigma^2}\Big] \tag{2.12}$$

Here A is a constant independent on σ, H_n — Hermite polynomial of the n-th order. We iterate over increasing pairs of n, m to generate the required number of functions. The basis is pre-calculated for all scales and fixed. For filters of size $V \times V$, the basis is stored as an array of shape $[N_b, S, V, V]$.

CONV T→H If the input signal is just a function on T with spatial size $U \times U$, stored as an array of shape $[C_{\text{in}}, U, U]$, then Equation 2.7 can be simplified. The summation over S degenerates, and the final result can be written in the following form:

$$\texttt{convTH}(f, w, \Psi) = \texttt{squeeze}(\texttt{conv2d}(f,\ \texttt{expand}(w \times \Psi))) \tag{2.13}$$

Here w is an array of shape $[C_{\text{out}}, C_{\text{in}}, N_b]$. We compute filter $w \times \Psi$ of shape $[C_{\text{out}}, C_{\text{in}}, S, V, V]$ and expand it to shape $[C_{\text{out}}, C_{\text{in}}S, V, V]$. Then we use standard 2D convolution to produce the output with $C_{\text{out}}S$ channels and squeeze it to shape $[C_{\text{out}}, S, U, U]$. Note that the output can be viewed as a stack of feature maps, where all the features in each spatial position are vectors of S components instead of being scalars as in standard CNNs.

CONV H→H The function on H has a scale axis and therefore there are two options for choosing weights of the convolutional filter. The filter may have just one scale and, therefore, does not capture the correlations between different scales of the input function; or, it may have a non-unitary extent K_S in the scale axis and capture the correlation between K_S neighboring scales. We refer to the second case as *interscale interaction*.

It the first case w has shape $[C_{\text{out}}, C_{\text{in}}, N_b]$ and Equation 2.7 degenerates in the same way as before

$$\texttt{convHH}(f, w, \Psi) = \texttt{squeeze}(\texttt{conv2d}(\texttt{expand}(f),\ \texttt{expand}(w \times \Psi))) \tag{2.14}$$

We expand f to an array of shape $[C_{\text{in}}S, U, U]$ and expand $w \times \Psi$ to have shape $[C_{\text{out}}S, C_{\text{in}}S, V, V]$. The result of the convolution is then squeezed in the same way as before.

In the case of interscale interaction, w has shape $[C_{\text{out}}, C_{\text{in}}, K_S, N_b]$. We iterate over all scales in interaction, shift f for each scale, choose a corresponding part of w, and apply `convHH` to them. We sum the obtained K_S results afterwards.

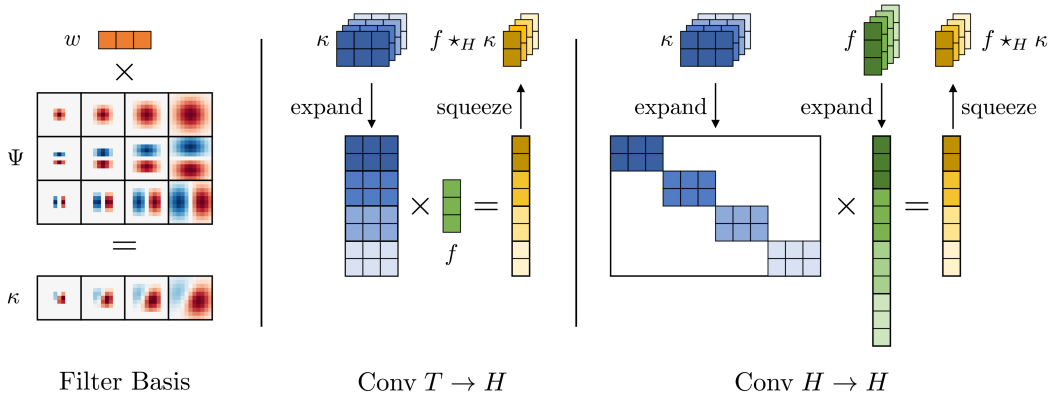


Figure 1: Left: the way steerable filters are computed using a steerable filter basis. Middle and right: a representation of scale-convolution using Equation 2.13 and Equation 2.14. As an example we use input signal f with 3 channels. It has 1 scale on T and 4 scales on H. It is convolved with filter $\kappa = w \times \Psi$ without scale interaction, which produces the output with 2 channels and 4 scales as well. Here we represent only channels of the signals and the filter. Spatial components are hidden for simplicity.

2.5 RELATED WORK

Various works on group-equivariant convolutional networks have been published recently. These works have considered roto-translation groups in 2D [25, 76, 154, 161] and 3D [89, 149, 160] and rotation equivariant networks in 3D [23, 26, 44]. In [50] authors describe the algorithm for designing steerable filters for rotations. Rotation steerable filters are used in [27, 155, 156] for building equivariant networks. In [79] the authors build convolutional blocks locally equivariant to arbitrary k-parameter Lie group transformation by using a steerable basis. And in [116] the authors discuss the approach for learning steerable filters from data. To date, the majority of papers on group equivariant networks have considered rotations in 2D and 3D, but have not payed attention to scale symmetry. As we have argued above, it is a fundamentally different case.

Many papers and even conferences have been dedicated to image scale-space — a concept where the image is analyzed together with all its downscaled versions. Initially introduced in [78] and later developed by [105, 121, 158] scale space relies on the scale symmetry of images. The differential structure of the image [88] allows one to make a connection between image formation mechanisms and the space of solutions of the 2-dimensional heat equation, which significantly improved the image analysis models in the pre-deep learning era.

One of the first works on scale equivariance and local scale invariance in the framework of CNNs was proposed by [171] named SiCNN. The authors describe the model with siamese CNNs, where the filters of each instance are rescaled using interpolation techniques. This is the simplest case of equivariance where no interaction between different scales is done in intermediate layers. In SI-ConvNet by [84] the original network is modified such that, in each layer, the input is first rescaled, then convolved and

Method	Equivariance	Admissible Scales	Approach	Interscale
SiCNN	✓	Grid	Filter Rescaling	✗
SI-ConvNet	✗	Grid	Input Rescaling	✗
SEVF	✓	Grid	Input Rescaling	✓
DSS	✓	Integer	Filter Dilation	✓
SS-CNN	✗	Any	Steerable Filters	✗
SESN, Ours	✓	Any	Steerable Filters	✓

Table 1: Comparing SESN to SiCNN [171], SI-ConvNet [84], SEVF [110], DSS [163] and SS-CNN [54]. "Interscale" refers to the ability of capturing interscale interactions with kernels of non-unitary scale extent. "Grid" stands for the scales which generate images which lie exactly on the initial pixel grid.

rescaled back to the original size. Finally, the response with the maximum values is chosen between the scales. Thus, the model is locally scale-invariant. In [110], in the SEVF model, the input of the layers is rescaled and convolved multiple times to form vector features instead of scalar ones. The length of the vector in each position is the maximum magnitude of the convolution, while the direction of the angle encodes the scale of the image which gave this response. These scale-equivariant networks rely on image rescaling which is quite slow. [163] (DSS) generalize the concept of scale-space to deep networks. They use filter dilation to analyze the images on different scales. While this approach is as fast as the standard CNN, it is restricted only to integer downscale factors $2, 4, 8 \dots$. In [54], while discussing SS-CNN the authors use scale-steerable filters to deal with scale changes. The paper does not discuss equivariance, which is an important aspect for scale.

We summarize the information about these models in Table 1. In contrast to other scale-equivariant models, SESN uses steerable filters which allows for fast scale-convolution with no limitation of flexibility. With the framework of Scale-Equivariant Convolutional Networks we are free to build both equivariant and invariant models of different kinds.

2.6 EXPERIMENTS

In this section we conduct the experiments and compare various methods for working with scale variations in input data. Alongside SESN, we test local scale invariant SI-ConvNet and SS-CNN, scale equivariant SiCNN, SEVF and DSS. For SEVF, DSS and SS-CNN we use the code provided by authors, while for others we reimplement the main buildings blocks.

SESN allows for training several times faster than other methods which rely on image rescaling.

2.6.1 *Equivariance error*

We have presented scale-convolution which is equivariant to scale transformation and translation for continuous signals. While translation equivariance holds true even for dis-

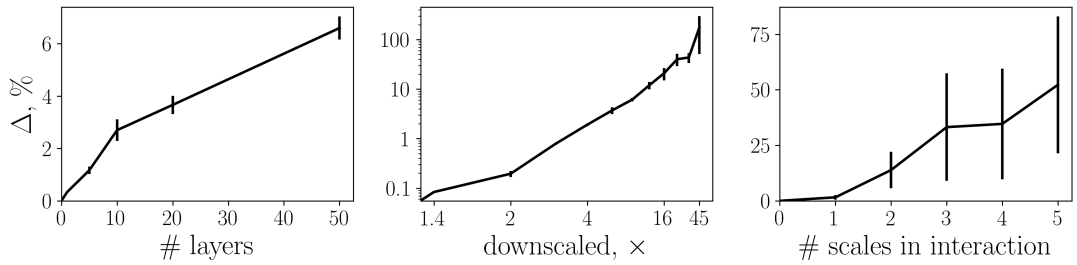


Figure 2: Equivariance error Δ as a function of the number of layers (left), downscaling applied to the input image (middle), and as a function of number of scales in interscale interactions (right). The bars indicate standard deviation.

cretized signals and filters, scale equivariance may not be exact. Therefore, before starting any experiments, we check to which degree the predicted properties of scale-convolution hold true. We do so by measuring the difference $\Delta = \|[L_s\Phi(f) - \Phi L_s(f)\|_2^2 / \|L_s\Phi(f)\|_2^2$, where Φ is scale-convolution with randomly initialized weights.

In case of perfect equivariance the difference is equal to zero. We calculate the error on randomly sampled images from the STL-10 dataset [22]. The results are represented in Figure 2. The networks on the left and on the middle plots do not have interscale interactions. The networks on the middle and on the right plots consist of just one layer. We use $N_S = 5, 13, 5$ scales for the networks on the left, the middle, and the right plots respectively. While discretization introduces some error, it stays very low, and is not much higher than 6% for the networks with 50 layers. The difference, however, increases if the input image is downscaled more than 16 times. Therefore, we are free to use deep networks. However, we should pay extra attention to extreme cases where scale changes are of very big magnitude. These are quite rare but still appear in practice. Finally, we see that using SESN with interscale interaction introduces extra equivariance error due to the truncation of S. We will build the networks with either no scale interaction or interaction of 2 scales.

2.6.2 *MNIST-scale*

Following [54, 84, 110] we conduct experiments on the MNIST-scale dataset. We rescale the images of the MNIST dataset [97] to $0.3 - 1.0$ of the original size and pad them with zeros to retain the initial resolution. The scaling factors are sampled uniformly and independently for each image. The obtained dataset is then split into 10,000 for training, 2,000 for evaluation and 50,000 for testing. We generate 6 different realizations and fix them for all experiments.

As a baseline model we use the model described in [54], which currently holds the state-of-the-art result on this dataset. It consists of 3 convolutional and 2 fully-connected layers. Each layer has filters of size 7×7. We keep the number of trainable parameters almost the same for all tested methods. This is achieved by varying the number of channels. For scale equivariant models we add scale projection at the end of the convolutional block.

Method	(28×28)	(28×28) +	(56×56)	(56×56) +	# Params
CNN	2.56 ± 0.04	1.96 ± 0.07	2.02 ± 0.07	1.60 ± 0.09	495 K
SiCNN	2.40 ± 0.03	1.86 ± 0.10	2.02 ± 0.14	1.59 ± 0.03	497 K
SI-ConvNet	2.40 ± 0.12	1.94 ± 0.07	1.82 ± 0.11	1.59 ± 0.10	495 K
SEVF Scalar	2.30 ± 0.06	1.96 ± 0.07	1.87 ± 0.09	1.62 ± 0.07	494 K
SEVF Vector	2.63 ± 0.09	2.23 ± 0.09	2.12 ± 0.13	1.81 ± 0.09	475 K
DSS Scalar	2.53 ± 0.10	2.04 ± 0.08	1.92 ± 0.08	1.57 ± 0.08	494 K
DSS Vector	2.58 ± 0.11	1.95 ± 0.07	1.97 ± 0.08	1.57 ± 0.09	494 K
SS-CNN	2.32 ± 0.15	2.10 ± 0.15	1.84 ± 0.10	1.76 ± 0.07	494 K
SESN Scalar	2.10 ± 0.10	1.79 ± 0.09	1.74 ± 0.09	1.50 ± 0.07	495 K
SESN Vector	$\mathbf{2.08 \pm 0.09}$	$\mathbf{1.76 \pm 0.08}$	$\mathbf{1.68 \pm 0.06}$	$\mathbf{1.42 \pm 0.07}$	495 K

Table 2: Classification error of different methods on MNIST-scale dataset, lower is better. In experiment we use image resolution of 28×28 *and* 56×56*. We test both the regime without data augmentation, and the regime with scaling data augmentation, denoted with "+". All results are reported as mean ± std over 6 different fixed realizations of the dataset. The best results are* ***bold****.*

For SiCNN, DSS, SEVF and our model, we additionally train counterparts where after each convolution, an extra projection layer is inserted. Projection layers transform vector features in each spatial position of each channel into scalar ones. All of the layers have now scalar inputs instead of vector inputs. Therefore, we denote these models with "Scalar". The original models are denoted as "Vector". The exact type of projection depends on the way the vector features are constructed. For SiCNN, DSS, and SESN, we use maximum pooling along the scale dimension, while for SEVF, it is a calculation of the L_2-norm of the vector. We report hyperparameters of the used architectures in Table 3.

All models are trained with the Adam optimizer [87] for 60 epochs with a batch size of 128. Initial learning rate is set to 0.01 and divided by 10 after 20 and 40 epochs. We conduct the experiments with 4 different settings. Following the idea discussed in [54], in addition to the standard setting we train the networks with input images upscaled to 56×56 using bilinear interpolation. This results in all image transformations performed by the network becoming more stable, which produces less interpolation artifacts. For both input sizes we conduct the experiments without data augmentation and with scaling augmentation, which results in 4 setups in total. We run the experiments on 6 different realizations of MNIST-scale and report mean ± std calculated over these runs.

The obtained results are summarized in Table 2. The reported errors may differ a bit from the ones in the original paper because of the variations in generated datasets and slightly different training procedure. Nevertheless, we try to keep our configuration as close as possible to [54] which currently demonstrated the best classification accuracy on MNIST-scale. For example, SS-CNN reports error of 1.91 ± 0.04 in [54] while it has 1.84 ± 0.10 in our experiments.

SESN significantly outperforms other methods in all 4 regimes. "Scalar" versions of it already outperform all previous methods, and "Vector" versions make the gain even

Method	Conv 1	Conv 2	Conv 3	FC 1	# Scales
CNN	32	63	95	256	1
SiCNN	32	63	95		7
SI-ConvNet	32	63	95		7
SEVF Scalar	32	63	95		8
SEVF Vector	23	45	68		8
DSS	32	63	95		4
SS-CNN	30	60	90		6
SESN	32	63	95		4

Table 3: Number of channels in convolutional layers, number of units in fully-connected layers and number of scales used by different models.

Method	Error, %	# Params
WRN	11.48	11.0 M
SiCNN	11.62	11.0 M
SI-ConvNet	12.48	11.0 M
DSS	11.28	11.0 M
SS-CNN	25.47	10.8 M
SESN-A	10.83	11.0 M
SESN-B	**8.51**	11.0 M
SESN-C	14.08	11.0 M
Harm WRN	9.55	11.0 M

*Table 4: Classification error on STL-10. The best results are **bold**. We additionally report the current best result achieved by Harm WRN from [151].*

more significant. The global architectures of all models are the same for all rows, which indicates that the way scale convolution is done plays an important role.

2.6.3 *STL-10*

In order to evaluate the role of scale equivariance in natural image classification, we conduct the experiments on STL-10 dataset [22]. This dataset consists of 8,000 training and 5,000 testing labeled images. Additionally, it includes 100,000 unlabeled images. The images have a resolution of 96×96 pixels and RGB channels. Labeled images belong to 10 classes such as bird, horse or car. We use only the labeled subset to demonstrate the performance of the models in the low data regime.

The dataset is normalized by subtracting the per-channel mean and dividing by the per-channel standard deviation. During training, we augment the dataset by applying 12 pixel zero padding and randomly cropping the images to size 96×96. Additionally, random horizontal flips with probability 50% and Cutout [37] with 1 hole of 32 pixels are used.

Method	Block 1	Block 2	Block 3	# Scales
CNN	16	32	64	1
SiCNN	16	32	64	3
SI-ConvNet	16	32	64	3
SEVF	11	23	45	3
DSS	16	32	64	4
SS-CNN	11	22	44	3
SESN	16	32	64	3

Table 5: Number of channels in convolutional blocks and number of scales used by different models in Section 2.6.2. We report the number of channels up to the widening factor.

As a baseline we choose WideResNet [173] with 16 layers and a widening factor of 8. We set dropout probability to 0.3 in all blocks. We train SESN-A with just vector features. For SESN-B we use maximum scalar projection several times in the intermediate layers, and for SESN-C we use interscale interaction.

All models are trained for 1000 epochs with a batch size of 128. We use SGD optimizer with Nesterov momentum of 0.9 and weight decay of $5 \cdot 10^{-4}$. The initial learning rate is set to 0.1 and divided by 5 after 300, 400, 600 and 800 epochs. We report configurations of the models in Table 5.

The results are summarized in Table 4. We found SEVF training unstable and therefore do not include it in the table. Pure scale-invariant SI-ConvNet and SS-CNN demonstrate significantly worse results than the baseline. We note the importance of equivariance for deep networks. We also find that SESN-C performs significantly worse than SESN-A and SESN-B due to high equivariance error caused by interscale interaction. SESN-B significantly improves the results of both WRN and DSS due to the projection between scales. The maximum scale projection makes the weights of the next layer to have a maximum receptive field in the space of scales. This is an easy yet effective method for capturing the correlations between different scales. This experiment shows that scale-equivariance is a very useful inductive bias for natural image classification with deep neural networks.

To the best of our knowledge, the proposed method achieves a new state-of-the-art result on the STL-10 dataset in the supervised learning setting. The previous lowest error is demonstrated in [151]. The authors propose Harm WRN — a network where the convolutional kernels are represented as a linear combination of Discrete Cosine Transform filters.

2.7 TIME PERFORMANCE

We report the average time per epoch of different methods for scale equivariance and local scale invariance in Table 6. Experimental setups from Section 2.6.2 are used. We used 1 Nvidia GeForce GTX 1080Ti GPU for training the models.

The methods relying on image rescaling techniques during training (SiCNN, SI-ConvNet, SEVF) demonstrate significantly worse time performance that the ones, using either steerable filters or filter dilation. Additionally, we see that our method outperforms SS-CNN by a wide margin. Despite the similar filter sizes and comparable number of parameters between SS-CNN and SESN Scalar, the second one demonstrates significantly better results due to the algorithm proposed in Section 2.4. Finally, DSS performs slightly faster in some cases than our method as each convolution involves less FLOPs. Dilated filters are sparse, while steerable filters are dense.

Method	28×28, s	56×56, s
CNN	3.8	3.8
SiCNN Scalar	13.5	18.9
SiCNN Vector	15.3	22.8
SI-ConvNet	18.4	33.1
SEVF Scalar	21.0	38.4
SEVF Vector	25.4	46.0
DSS Scalar	3.9	5.0
DSS Vector	3.9	4.8
SS-CNN	14.8	16.6
SESN Scalar	3.8	5.1
SESN Vector	3.8	6.8

Table 6: Average time per epoch during training on input data with resolution 28×28 *and* 56×56*.*

2.8 DISCUSSION

In this paper, we have presented the theory of Scale-Equivariant Steerable Networks. We started from the scaling transformation and its application to continuous functions. We have obtained the exact formula for scale-equivariant mappings and demonstrated how it can be implemented for discretized signals. We have demonstrated that this approach outperforms other methods for scale-equivariant and local scale-invariant CNNs. It demonstrated new state-of-the-art results on MNIST-scale and on the STL-10 dataset in the supervised learning setting.

We suppose that the most exciting possible application of SESN is in computer vision for autonomous vehicles. Rapidly changing distances between the objects cause significant scale variations which makes this well suited for our work. We especially highlight the direction of siamese visual tracking where the equivariance to principle transformations plays an important role.

3

SCALE EQUIVARIANCE IMPROVES SIAMESE TRACKING

3.1 INTRODUCTION

Siamese trackers turn tracking into similarity estimation between a template and the candidate regions in the frame. The Siamese networks are successful because the similarity function is powerful: it can learn the variances of appearance very effectively, to the degree that even the association of the frontside of an unknown object to its backside is usually successful. And, once the similarity is effective, the location of the candidate region is reduced to simply selecting the most similar candidate.

Mathematically, one of the key ingredients of success of the similarity function is translation equivariance, i.e. a translation in the input image is to result in the proportional translation in feature space. Non-translation-equivariant architectures will induce a positional bias during training, so the location of the target will be hard to recover from the feature space [98, 177]. In real life scenarios, the target will undergo more transformations than just translation, and unless the network has an internal mechanism to handle them, the similarity may degrade. We start from the position that equivariance to common transformations should be the guiding principle in designing conceptually simple yet robust trackers. To that end, we focus on scale equivariance for trackers in this paper.

Measuring scale precisely is crucial when the camera zooms its lens or when the target moves into depth. However, scale is also important in distinguishing among objects in general. In following a marching band or in analysing a soccer game, or, in general, when many objects in the video have a similar appearance (a crowd, a heard, team sports), the similarity power of Siamese trackers has a hard time locating the right target. In such circumstances spatial-scale equivariance will provide a richer and hence more discriminative descriptor, which is essential to differentiate among several but similar candidates in an image. And, even, as we will show, when the sequence does not show variation over scale, proper scale measurement is important to keep the target bounding box stable in size.

The common way to implement scale into a tracker is to train the network on a large dataset where scale variations occur naturally. However, as was noted in [96], such training procedures may lead to learning groups of re-scaled duplicates of almost the same filters. And, as a consequence, inter-scale similarity estimation becomes unreliable, see Figure 3 top. Scale-equivariant models have an internal notion of scale and a built-in weight sharing among different filter scales. Thus, scale equivariance aims to produce the same distinction for all sizes, see Figure 3 bottom.

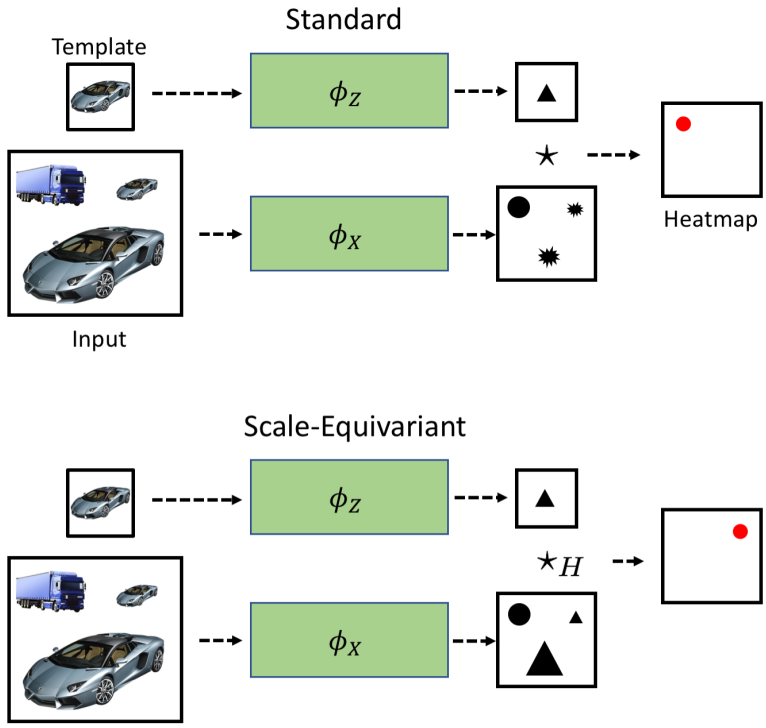


Figure 3: The standard version (top) and the scale-equivariant version (bottom) of a basic tracker. The scale-equivariant tracker has an internal notion of scale which allows for the distinction between similar objects which only differ in scale.

In this paper, we aim to equip the Siamese network with spatial and scale equivariance built-in from the start to capture the natural variations of the target *a priori*. We aim to improve a broad class of tracking algorithms by enhancing their capacity of candidate distinction. We adopt recent advances [144] in convolutional neural networks (CNNs) which handle scale variations explicitly and efficiently.

While scale-equivariant convolutional models have led to success in image classification [144, 161], we focus on their usefulness in *object localization*. Where scale estimation has been used in the localization for tracking, it typically relies on brute-force multi-scale detection with an obvious computational burden [10, 33], or on a separate network to estimate the scale [31, 100]. Both approaches, will require attention to avoid bias and the propagation thereof through the network. Our new method treats scale and scale equivariance as a desirable fundamental property, which makes the algorithm conceptually easier. Hence, scale equivariance should be easy to merge into an existing network for tracking. Then, scale equivariance will enhance the performance of the tracker without further modification of the network or extensive data augmentation during the learning phase.

We make the following contributions:

- We propose the theory for scale-equivariant Siamese trackers, and we provide a simple recipe of how to make a wide range of existing trackers scale-equivariant.
- We implement a scale-equivariant variant of a popular Siamese tracker — extending SiamFC [10] to scale-equivariant behaviour.

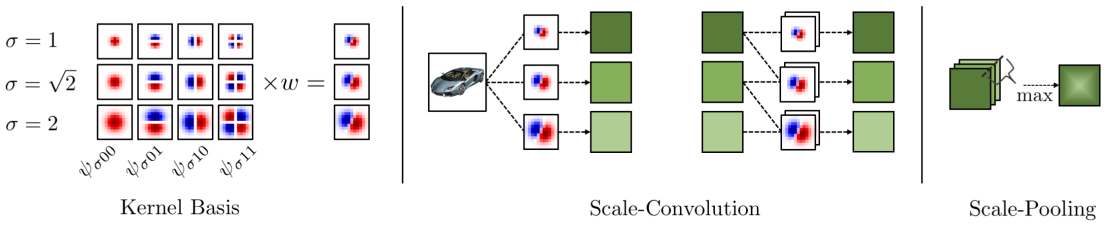


Figure 4: Left: convolutional kernels use a fixed kernel basis on multiple scales, each with a set of trainable weights. Middle: a representation of scale-convolution using Equation 3.6 for the first and all subsequent layers. Right: a scheme of scale max-pooling, which transforms a 3D-signal into a 2D one without loosing scale equivariance. As an example we use a basis of 4 functions and 3 scales with a step of $\sqrt{2}$. Only one channel of each convolutional layer is demonstrated for simplicity.

- We demonstrate the advantage of scale-equivariant Siamese trackers over their conventional counterparts on popular benchmarks.

3.2 RELATED WORK

SIAMESE TRACKING The challenge of learning to track arbitrary objects can be addressed by deep similarity learning [10]. The common approach is to employ Siamese networks to compute the embeddings of the original patches. The embeddings are then fused to obtain a location estimate. Such a formulation is general, allowing for a favourable flexibility in the design of the tracker. The pioneering works of Tao *et al.* [148] and of Bertinetto *et al.* [10] employ off-line trained CNNs as feature extractors. They compare dot-product similarities between the feature map of the template with the maps coming from the current frame. Bertinetto *et al.* [10] additionally measures similarities on multiple scales. Li *et al.* [100] considers tracking as a one-shot detection problem to design Siamese region-proposal-networks [125] by fusing the features from a fully-convolutional backbone. The recent ATOM [31] and DIMP [11] trackers employ a multi-stage tracking framework, where an object is coarsely localized by the online *classification* branch, and subsequently refined in its position by the *estimation* branch. From a Siamese perspective, in both [11, 31] the object embeddings are first fused to produce an initial location and subsequently processed by the IoU-Net [82] to enhance the precision of the bounding box.

The aforementioned references have laid the foundation for most of the state-of-the-art trackers. These methods share an implicit or explicit attention to translation equivariance for feature extraction. The decisive role of translation equivariance is noted in [10, 98, 177]. Bertinetto *et al.* [10] utilize fully-convolutional networks where the output directly commutes with a shift in the input image as a function of the total stride. Li *et al.* [98] suggest a training strategy to eliminate the spatial bias introduced in non-fully-convolutional backbones. Along the same line, Zhang and Peng [177] demonstrated that deep state-of-the-art models developed for classification are not directly applicable for localization. And hence these models are not directly applicable to tracking as they induce positional bias, which breaks strict translation equivariance. We argue that

transformations, other then translation, such as rotation may be equally important for certain classes of videos like sports and following objects in the sea or in the sky. And we argue that scale transformation is common in the majority of sequences due to the changing distances between objects and the camera. In this paper, we take on the latter class of transformations for tracking.

EQUIVARIANT CNNS Various works on transformation-equivariant convolutional networks have been published recently. They extend the built-in property of translation-equivariance of conventional CNNs to a broader set of transformations. Mostly considered was roto-translation, as demonstrated on image classification [25, 27, 76, 129, 131, 154], image segmentation [156] and edge detection [161].

One of the first works on scale-translation-equivariant convolutional networks was by Marcos *et al.* [110]. In order to process images on multiple scales, the authors resize and convolve the input of each layer multiple times, forming a stack of features which corresponds to variety of scales. The output of such a convolutional layer is a vector whose length encodes the maximum response in each position among different scales. The direction of the vector is derived from the scale, which gave the maximum. The method has almost no restrictions in the choice of admissible scales. As this approach relies on rescaling the image, the obtained models are significantly slower compared to conventional CNNs. Thus, this approach is not suitable for being applied effectively in visual object tracking.

Worrall & Welling [163] propose Deep Scale-Spaces, an equivariant model which generalizes the concept of scale-space to deep networks. The approach uses filter dilation to analyze the images on different scales. It is almost as fast as a conventional CNN with the same width and depth. As the method is restricted to integer scale factors it is unsuited to applications in tracking where the scene dictates arbitrary scale factors.

Almost simultaneously, three papers [8, 144, 181] were proposed to implement scale-translation-equivariant networks with arbitrary scales. What they have in common is that they use a pre-calculated and fixed basis defined on multiple scales. All filters are then calculated as a linear combination of the basis and trainable weights. As a result, no rescaling is used. We prefer to use [144], as Sosnovik *et al.* propose an approach for building general scale-translation-equivariant networks with an algorithm for the fast implementation of the scale-convolution.

To date, the application of scale-equivariant networks was mostly demonstrated in image classification. Almost no attention was paid to tasks that involve object localization, such as visual object tracking. As we have noted above, it is a fundamentally different case. To the best of our knowledge, we demonstrate the first application of transformation-equivariant CNNs to visual object tracking.

3.3 SCALE-EQUIVARIANT TRACKING

In this work, we consider a wide range of modern trackers which can be described by the following formula:

$$h(z, x) = \phi_X(x) \star \phi_Z(z) \tag{3.1}$$

where z, x are the template and the input frame, and ϕ_X, ϕ_Z are the functions which process them, and $\star$ is the convolution operator. The resulting value $h(z, x)$ is a heatmap that can be converted into a prediction by relatively simple calculations. Functions ϕ_X, ϕ_Z here can be parametrized as feed-forward neural networks. For our analysis, it is both suitable if the weights of these networks are fixed or updated during training or inference. This pipeline describes the majority of siamese trackers such as [10, 98, 100, 148] or the trackers based on correlation filters [32, 33].

3.3.1 *Convolution is all you need*

Let us consider some mapping g. It is equivariant under a transformation L if and only if there exists L' such that $g \circ L = L' \circ g$. If L' is the identity mapping, then the function g is invariant under this transformation. A function of multiple variables is equivariant when it is equivariant with respect to each of the variables. In our analysis, we consider only transformations that form a transformation group, in other words, $L \in G$.

Theorem 1. *A function given by Equation 3.1 is equivariant under a transformation L from group G if and only if ϕ_X and ϕ_Z are constructed from G-equivariant convolutional layers and $\star$ is the G-convolution.*

Proof. Let us fix $z = z_0$ and introduce a function $h_X(x) = h(x, z_0) = \phi_X(x) \star \phi_Z(z_0)$. This function is a feed-forward neural network. All its layers but the last one are contained in ϕ_X and the last layer is a convolution with $\phi_Z(z_0)$. According to [90] a feed-forward neural network is equivariant under transformations from G if and only if it is constructed from G-equivariant convolutional layers. Thus, the function h_X is equivariant under transformations from G if and only if

- The function ϕ_X is constructed from G-equivariant convolutional layers
- The convolution $\star$ is the G-convolution

If we then fix $x = x_0$, we can show that a function $h_Z(z) = h(x_0, z) = \phi_X(x_0) \star \phi_Z(z)$ is equivariant under transformations from G if and only if

- The function ϕ_Z is constructed from G-equivariant convolutional layers
- The convolution $\star$ is the G-convolution

The function h is equivariant under G if and only if both the function h_X and the function h_Z are equivariant. □

A simple interpretation of this theorem is that *a tracker is equivariant to transformations from G if and only if it is fully G-convolutional*. The necessity of fully-convolutional trackers is well-known in tracking community and is related to the ability of the tracker to capture the main variations in the video — the translation. In this paper, we seek to extend this ability to scale variations as well. Which, due to Theorem 1 boils down to using scale-convolution and building fully scale-translation convolutional trackers.

3.3.2 *Scale Modules*

Given a function $f : R \to R$, a scale transformation is defined as follows:

$$L_s[f](t) = f(s^{-1}t), \quad \forall s \geq 0 \tag{3.2}$$

where cases with $s > 1$ are refered to as upscale and with $s < 1$ as downscale. Standard convolutional layers and convolutional networks are translation equivariant but not scale-equivariant [144].

PARAMETRIC SCALE-CONVOLUTION In order to build scale-equivariant convolutional networks, we follow the method proposed by Sosnovik *et al.* [144]. We begin by choosing a complete basis of functions defined on multiple scales. Choosing the center of the function to be the point $(0, 0)$ in coordinates (u, v), we use functions of the following form:

$$\psi_{\sigma nm}(u, v) = A\frac{1}{\sigma^2}H_n\Big(\frac{u}{\sigma}\Big)H_m\Big(\frac{v}{\sigma}\Big)e^{-\frac{u^2+v^2}{2\sigma^2}} \tag{3.3}$$

Here H_n — Hermite polynomial of the n-th order. A is a constant used for normalization. In order to build a basis of N functions, we iterate over increasing pairs of n and m. We build such a basis for a chosen set of equidistant scales σ and fix it:

$$\Psi_\sigma = \Big\{\psi_{\sigma 00},\ \psi_{\sigma 01},\ \psi_{\sigma 10},\ \psi_{\sigma 11} \dots\Big\} \tag{3.4}$$

Kernels of convolutional layers are parametrized by trainable weights w in the following way:

$$\kappa_\sigma = \sum_i \Psi_{\sigma i} w_i \tag{3.5}$$

As a result, each kernel is defined on multiple scales and no image interpolation is used. Given a function of scale and translation $f(s, t)$ and a kernel $\kappa_\sigma(s, t)$, a scale convolution is defined as:

$$[f \star_H \kappa_\sigma](s, t) = \sum_{s'} [f(s', \cdot) \star \kappa_{s\cdot\sigma}(s^{-1}s', \cdot)](t) \tag{3.6}$$

The result of this operation is a stack of features each of which corresponds to a different scale. We end up with a 3-dimensional representation of the signal — 2-dimensional translation + scale. We follow [144] and denote scale-convolution as $\star_H$ in order to distinguish it from the standard one. Figure 4 demonstrates how a kernel basis is formed and how scale-convolutional layers work.

FAST 1×1 SCALE-CONVOLUTION An essential building block of many backbone deep networks such as ResNets [67] and Wide ResNets [173] is a 1×1 convolutional layer. We follow the interpretation of these layers proposed in [102] — it is a linear combination of channels. Thus, it has no spatial resolution. In order to build a scale-equivariant counterpart of 1×1 convolution, we do not utilize a kernel basis. As we pointed out before, the signal is stored as a 3 dimensional tensor for each channel. Therefore, for a kernel defined on N_S scales, the convolution of the signal with this kernel is just a 3-dimensional convolution with a kernel of size 1×1 in spatial dimension, and

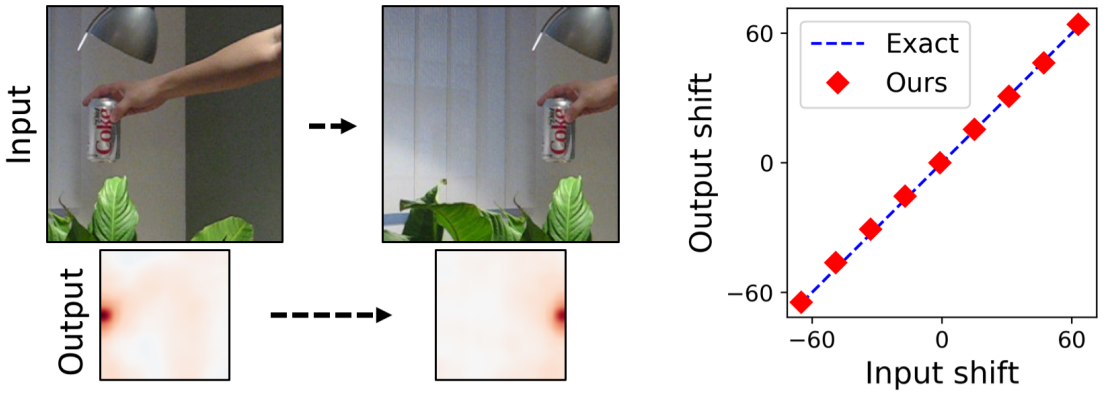


Figure 5: Left: two samples from the simulated sequence. The input image is a translated and cropped version of the source image. The output is the heatmap produced by the proposed model. The red color represents the place where the object is detected. Right: correspondence between the input and the output shifts.

with N_S values in depth. This approach for 1×1 scale-convolution is faster than the special case of the algorithm proposed in [144].

PADDING Although zero padding is a standard approach in image classification for saving the spatial resolution of the image, it worsens the localization properties of convolutional trackers [98, 177]. Nevertheless, a simple replacement of standard convolutional layers with scale-equivariant ones in very deep models is not possible without padding. Scale-equivariant convolutional layers have kernels of a bigger spatial extent because they are defined on multiple scales. We use circular padding during training and zero padding during testing in our models.

The introduced padding does not affect the feature maps which are obtained with kernels defined on small scales. In Figure 5 we demonstrate that it does not violate the translation equivariance of a network. We choose an image and select a sequence of translated and cropped windows inside of it. We process this sequence with a deep model that consists of the proposed convolutional layers and follows the inference procedure described in [177]. We derive the predicted location of the object and compare its value to the input shift. The input and the output translations have almost identical values.

SCALE-POOLING In order to capture correlations between different scales and to transform a 3-dimensional signal into a 2-dimensional one, we utilize global max pooling along the scale axis. This operation does not eliminate the scale-equivariant properties of the network. We found that it is useful to additionally incorporate this module in the places where conventional CNNs have spatial max pooling or strides. The mechanism of scale-pooling is illustrated in Figure 4.

NON-PARAMETRIC SCALE-CONVOLUTION The convolutional operation which results in the heatmap of a tracker is non-parametric. Both the input and the kernel come from neural networks. Thus, the approach described in Equation 3.6 is not suitable for

this case. Given two functions f_1, f_2 of scale and translation the non-paramteric scale convolution is defined as follows:

$$[f_1 \star_H f_2](s,t) = L_{s^{-1}}[L_s[f_1] \star f_2](t) \tag{3.7}$$

Here L_s is rescaling implemented as bicubic interpolation. Although it is a relatively slow operation, it is used only once in the tracker and does not heavily affect the inference time. Let us prove that non-parametric scale convolution is equivariant under scale-translation.

Proof. A function given by Equation 3.7 is equivariant under scale transformations of f_1, indeed

$$\begin{aligned}[L_{\hat{s}}[f_1] \star_H f_2](s,t) &= L_{s^{-1}}[L_{s\hat{s}}[f_1] \star f_2](t) \\ &= L_{\hat{s}} L_{(s\hat{s})^{-1}}[L_{s\hat{s}}[f_1] \star f_2](t) \\ &= L_{\hat{s}}[f_1 \star_H f_2](s\hat{s}, t)\end{aligned} \tag{3.8}$$

For a pair of scale and translation $s, \hat{t}$ we have the following property of the joint transformation $L_s T_{\hat{t}} = T_{\hat{t}s} L_s$ from [144], where $T_{\hat{t}}$ is the translation operator defined as $T_{\hat{t}}[f](t) = f(t - \hat{t})$. Now we can show the following:

$$\begin{aligned}[T_{\hat{t}}[f_1] \star_H f_2](s,t) &= L_{s^{-1}}[L_s[T_{\hat{t}}[f_1]] \star f_2](t) \\ &= L_{s^{-1}}[T_{\hat{t}s} L_s[f_1] \star f_2](t) \\ &= L_{s^{-1}} T_{\hat{t}s}[L_s[f_1] \star f_2](t) \\ &= T_{\hat{t}} L_{s^{-1}}[L_s[f_1] \star f_2](t) \\ &= T_{\hat{t}}[f_1 \star_H f_2](t)\end{aligned} \tag{3.9}$$

Therefore, a function given by Equation 3.7 is also equivariant under translations of f_1. The equivariance of the function with respect to a joint transformation follows from the equivariance to each of the transformations separately [144].

We proved the equivariance with respect to f_1. The proof with respect to f_2 is analogous. □

3.3.3 *Extending a Tracker to Scale Equivariance*

We present a recipe to extend a tracker into scale equivariance.

1. The first step is to estimate to what degree objects change in size in this domain, and then to select a set of scales $\sigma_1, \sigma_2, \ldots \sigma_N$. This is a domain-specific hyperparameter. For example, a domain with significant scale variations requires a broader span of scales, while for more smooth sequences the set may consist of just 3 scales around 1.

2. For a tracker which can be described by Equation 3.1, derive ϕ_X and ϕ_Z.

3. For the networks represented by ϕ_X and ϕ_Z all convolutional layers need to be replaced with scale-convolutional ones. The basis for these layers is based on the chosen scales $\sigma_1, \sigma_2, \ldots \sigma_N$.

4. (Optional) Scale-pooling can be included to additionally capture inter-scale correlations between all scales.

5. The connection operation $\star$ needs to be replaced with a non-parametric scale-convolution.

6. (Optional) If the tracker only searches over spatial locations, scale-pooling needs to be included at the very end.

The obtained tracker produces a heatmap $h(z, x)$ defined on scale and translation. Therefore, each position is assigned with a vector of features that has both the measure of similarity and the scale relation between the candidate and the template. If additional scale-pooling is included, then all scale information is just aggregated in the similarity score

Note that the overall structure of the tracker, as well as the training and inference procedures are not changed. Thus, the recipe allows for a simple extension of a tracker with little cost of modification.

3.4 SCALE-EQUIVARIANT SIAMFC

While the proposed algorithm is applicable to a wide range of trackers, in this work, we focus on Siamese trackers. As a baseline we choose SiamFC [10]. This model serves as a starting point for modifications for the many of the modern high-performance Siamese trackers.

3.4.1 *Architecture*

Given the recipe, here we discuss the actual implementation of scale-equivariant SiamFC tracker (SE-SiamFC). In the first step of the recipe we assess the range of scales in the domain (dataset). In sequences presented in most of the tracking benchmarks, like OTB or VOT, objects change their size relatively slowly from one frame to the other. The maximum scale change usually does not exceed a factor of $1.5 - 2$. Therefore, we use 3 scales with a step of $\sqrt{2}$, as the basis for the scale-convolutions. The next step in the recipe is to represent the tracker as it is done in Equation 3.1. SiamFC localizes the object as the coordinate *argmax* of the heatmap $h(z, x) = \phi_Z(z) \star \phi_X(x)$, where $\phi_Z = \phi_X$ are convolutional Siamese backbones. Next, in step number 3, we modify the backbones by replacing standard convolutions by scale-equivariant ones. We follow step 4 and utilize scale-pooling in the backbones in order to capture additional scale correlations between features of various scales. According to step 5 the connecting correlation is replaced with non-parameteric scale-convolution. SiamFC computes its similarity function as a 2-dimensional map, therefore, we follow step 6 and add extra scale-pooling in order to transform a 3-dimensional heatmap into a 2-dimensional one. Now, we can use exactly the same inference algorithm as in the original paper [10]. We use the standard approach of scale estimation, based on greedy selection of the best similarity for 3 different scales.

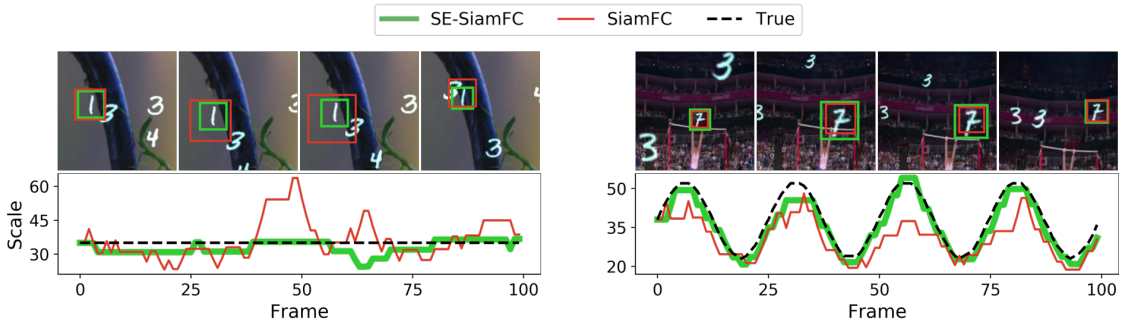


Figure 6: Top: examples of simulated T-MNIST and S-MNIST sequences. Bottom: scale estimation for equivariant and non-equivariant models. In S-MNIST example SE-SiamFC can estimate the scale more accurately. In T-MNIST example, our model better preserves the scale of the target unchanged, while non-scale-equivariant model is prone to oscillations in its scale estimate.

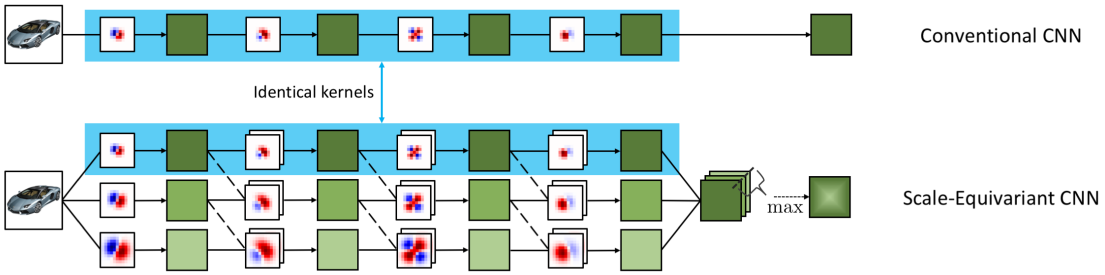


Figure 7: The visualization of the weight initialization scheme from a pretrained model. Dashed connections are initialized with 0*.*

3.4.2 *Weight Initialization*

An important ingredient of a successful model training is the initialization of its weights. A common approach is to use weights from an Imagenet [36] pre-trained model [98, 100, 177]. In our case, however, it requires additional steps, as there are no available scale-equivariant models pre-trained on the Imagenet. We present a method for initializing a scale-equivariant model with weights from a pre-trained conventional CNN. The key idea is that a scale-equivariant network built according to Section 3.3.3 contains a sub-network that is identical to the one of the non-scale-equivariant counterpart. As the kernels of scale-equivariant models are parameterized with a fixed basis and trainable weights, our task is to initialize these weights.

We begin by initializing the inter-scale correlations by setting to 0 all weights responsible for these connections. At this moment, up to scale-pooling, the scale-equivariant model consists of several networks parallel but yet disconnect to one another where the only difference is the size of their filters. For the convolutional layers with a non-unitary spatial extent, we initialize the weights such that the kernels of the smallest scale match

Tracker	T/T	T/S	S/T	S/S	# Params
SiamFC	0.64	0.62	0.64	0.63	999 K
SE-SiamFC	**0.76**	**0.69**	**0.77**	**0.70**	999 K

Table 7: AUC for models trained on T-MNIST and S-MNIST. T/S indicates that the model was trained on T-MNIST and tested on S-MNIST datasets. Bold numbers represent the best result for each of the training/testing scenarios.

those of the source model. Given a source kernel $\kappa'(u, v)$ and a basis $\Psi_{\sigma i}(u, v)$ with $\sigma = 1$ weights w_i are chosen to satisfy the linear system derived from Equation 3.5:

$$\kappa_1(u, v) = \sum_i \Psi_{1i}(u, v) w_i = \kappa'(u, v), \quad \forall u, v \tag{3.10}$$

As the basis is complete by construction, its matrix form is invertible. The system has a unique solution with respect to w_i:

$$w_i = \sum_{u,v} \Psi_{1i}^{-1}(u, v) \kappa'(u, v) \tag{3.11}$$

All 1×1 scale-convolutional layers are identical to standard 1×1 convolutions after zeroing out inter-scale correlations. We copy these weights from the source model. See Figure 7 for more details.

3.5 EXPERIMENTS AND RESULTS

Tracker	Year	OTB-2013		OTB-2015	
		AUC	Prec.	AUC	Prec.
SINT [148]	2016	0.64	0.85	-	-
SiamFC [10]	2016	0.61	0.81	0.58	0.77
DSiam [62]	2017	0.64	0.81	-	-
StructSiam [176]	2018	0.64	0.88	0.62	0.85
TriSiam [42]	2018	0.62	0.82	0.59	0.78
SiamRPN [100]	2018	-	-	0.64	0.85
SiamFC+ [177]	2019	0.67	0.88	0.64	0.85
SE-SiamFC	Ours	**0.68**	0.90	**0.66**	0.88

Table 8: Performance comparisons on OTB-2013, OTB-2015. Bold numbers represent the best result for each of the benchmarks.

Tracker	Year	VOT16			VOT17		
		EAO	A	R	EAO	A	R
SINT [148]	2016	-	-	-	-	-	-
SiamFC [10]	2016	0.24	0.53	0.46	0.19	0.50	0.59
DSiam [62]	2017	-	-	-	-	-	-
StructSiam [176]	2018	0.26	-	-	-	-	-
TriSiam [42]	2018	-	-	-	0.20	-	-
SiamRPN [100]	2018	0.34	0.56	0.26	0.24	0.49	0.46
SiamFC+ [177]	2019	0.30	0.54	0.38	0.23	0.50	0.49
SE-SiamFC	Ours	**0.36**	0.59	0.24	**0.27**	0.54	0.38

Table 9: Performance comparisons on VOT16 and VOT17 benchmarks. Bold numbers represent the best result for each of the benchmarks.

3.5.1 *Translation-Scaling MNIST*

To test the ability of a tracker to cope with translation and scaling, we conduct an experiment on a simulated dataset with controlled factors of variation. We construct the datasets of translating (T-MNIST) and translating-scaling (S-MNIST) digits. In particular, to form a sequence we randomly sample up to 8 MNIST digits with backgrounds from the GOT10k dataset [77]. Then, on each of the digits in the sequence independently, a smoothed Brownian motion model induces a random translation. Simultaneously, for S-MNIST, a smooth scale change in the range $[0.67, 1.5]$ is induced by the sine rule:

$$s_i(t) = \frac{(h-l)}{2}\Big[\sin(\frac{t}{4}+\beta_i)+1)\Big]+l \tag{3.12}$$

where $s_i(t)$ is the scale factor of i-th digit in the t-th frame, h, l are upper and lower bounds for scaling, and $\beta_i \in [0, 100]$ is a phase, sampled randomly for each of the digits. In total, we simulate 1000 sequences for training, and 100 for validation. Each sequence has a length of 100 frames.

We compare two configurations of the tracker: (i) SiamFC with a shallow backbone and (ii) its scale-equivariant version SE-SiamFC. We conduct the experiments according to 2×2 scenarios: the models are trained on either S-MNIST or T-MNIST and are tested on either of them. The results are listed in Table 7.

For both T-MNIST and S-MNIST, we use architectures described in Table 10. 2D BatchNorm and ReLU are inserted after each of the convolutional layers except the last one. We do not use max pooling to preserve strict translation-equivariance.

We train both models for 50 epochs using SGD with a mini-batch of 8 images and exponentially decay the learning rate from 10^{-2} to 10^{-5}. We set the momentum to 0.9 and the weight decay to $5 \cdot 10^{-4}$. A binary cross-entropy loss as in [10] is used. The inference algorithm is the same for both SiamFC and SE-SiamFC and follows the original implementation [10].

As can be seen from Table 7, the equivariant version outperforms its non-equivariant counterpart in all four scenarios. The experiment on S-MNIST varying the scale of an

Stage	SiamFC	SE-SiamFC
Conv1	$[3 \times 3, 96, s = 2]$	
Conv2	$[3 \times 3, 128, s = 2]$	
Conv3	$[3 \times 3, 256, s = 2]$	
Conv4	$[3 \times 3, 256, s = 1]$	
Connect	Cross-correlation	Non-parametric scale-convolution
# Params	999 K	999 K

Table 10: Architectures used in T/S-MNIST experiment. All convolutions in SE-SiamFC are scale-convolutions.

artificial object, shows that the scale-equivariant model has superior ability to precisely follow the change in scale compared to the conventional one. The experiment on T-MNIST shows that (proper) measurement of scale is important even in the case when the sequence does not show change in scale, where the observed scale in SE-SiamFC fluctuates much less than it does in the baseline (see Figure 6).

3.5.2 *Benchmarking*

We compare the scale-equivariant tracker against a non-equivariant baseline on popular tracking benchmarks. We test SE-SiamFC with a backbone from [177] against other popular Siamese trackers on OTB-2013, OTB-2015, VOT16 and VOT17. The benchmarks are chosen to allow direct comparison with the baseline [177].

IMPLEMENTATION DETAILS The parameters of our model are initialized with weights pre-trained on Imagenet by a method described in Section 3.4.2. We use the same training procedure as in the baseline.

For OTB and VOT experiments we used architectures described in Table 11. We use the baseline [177] with Cropping Inside Residual (CIR) units. SE-SiamFC is constructed directly from the baseline as described in the paper. In Table 11 the kernel size refers to the smallest scale $\sigma = 1$ in the network. The sizes of the kernels, which correspond to bigger scales are 9×9 for Conv1 and 5×5 for other layers. Figure 9 gives a qualitative comparison of the proposed method and the baseline.

The pairs for training are collected from the GOT10k [77] dataset. We adopt the same prepossessing and augmentation techniques as in [177]. The inference procedure remains unchanged compared to the baseline.

OTB We test on the OTB-2013 [164] and OTB-2015 [165] benchmarks. Each of the sequences in the OTB datasets carries labels from 11 categories of difficulty in tracking

Stage	SiamFC+	SE-SiamFC
Conv1	$\left[7 \times 7, 64, s = 2\right]$	$\left[7 \times 7, 64, s = 2\right]$
	max pool $\left[2 \times 2, s = 2\right]$	
Conv2	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64, i = 2 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
Conv3	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 128, sp \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 3$
Connect	Cross-correlation	Non-parametric scale-convolution
# Params	1.44 M	1.45 M

Table 11: Architectures used in OTB/VOT experiments. All convolutions in SE-SiamFC are scale-convolutions. s refers to stride, sp denotes scale pooling, i — is the size of the kernel in a scale dimension.

the sequence like: occlusion, scale variation, in-pane rotation, *etc.* We employ a standard one-pass evaluation (OPE) protocol to compare our method with other trackers by area under the success curve (AUC) and precision.

The results are reported in Table 8. Our scale-equivariant tracker outperforms its non-equivariant counterpart by more than 3% on OTB-2015 in both AUC and precision, and by 1.4% on OTB-2013. When summarized per different label of difficulty (see Figure 8), the proposed scale-equivariant tracker will improve all sequence types, not just the ones labeled with "scale variation".

We attribute this to the fact that "scale variation" tag in OTB benchmark only indicates the sequences with a relatively big change in scale factors, while up to a certain degree, scaling is present in almost any video sequence. Moreover, scaling may be present implicitly, in the form of the same patterns being observed on multiple scales. An ability of our model to exploit this leads to a better utilization of trainable parameters and a more discriminative Siamese similarity as a result.

VOT We next evaluate our tracker on VOT2016 and VOT2017 datasets [91]. The performance is evaluated in terms of average bounding box overlap ratio (A), and the robustness (R). These two metrics are combined into the Expected Average Overlap (EAO), which is used to rank the overall performance.

The results are reported in Table 9. On VOT2016 our scale-equivariant model shows an improvement from 0.30 to 0.36 in terms of EAO, which is a 20% gain compared to the non-equivariant baseline. On VOT2017, the increase in EAO is 17%.

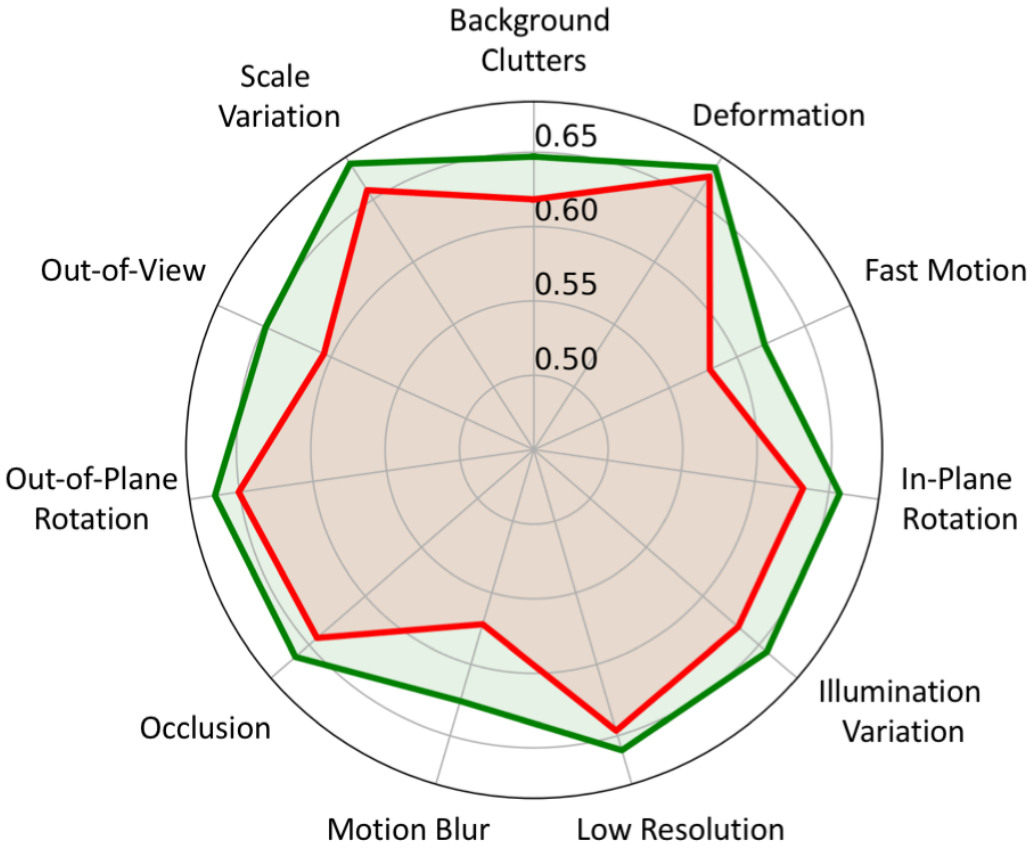


Figure 8: Comparison of AUC on OTB-2013 with different factors of variations. The red polygon corresponds to the baseline SiamFC+ and the green polygon — to SE-SiamFC.

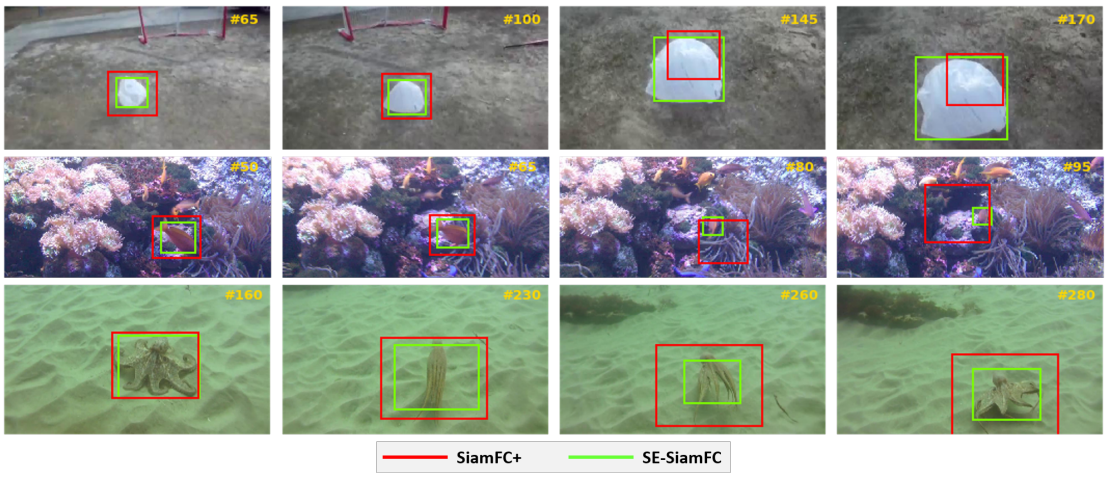


Figure 9: Qualitative comparison of the proposed SE-SiamFC with the baseline SiamFC+ on 3 sequences (bag, fish2, octopus) from VOT2016. Our method handles rapid scale change better and accurately models an object's size.

We qualitatively investigate the sequences with the largest performance gain (see Figure 9). We observed that the most challenging factor for our baseline is the rapid scaling of the object. Even when the target is not completely lost, an imprecise bounding

box heavily influences the overlap with the ground truth and the final EAO. Our scale-equivariant model better adapts to the fast scaling and delivers tighter bounding boxes.

3.6 DISCUSSION

In this work, we argue about the usefulness of additional scale equivariance in visual object tracking for the purpose of enhancing Siamese similarity estimation. We present a general theory that applies to a wide range of modern Siamese trackers, as well as all the components to turn a tracker into a scale-equivariant one. Moreover, we prove that the presented components are both necessary and sufficient to achieve built-in scale-translation equivariance. We sum up the theory by developing a simple recipe for extending existing trackers to scale equivariance. We apply it to develop SE-SiamFC — a scale-equivariant modification of the popular SiamFC tracker.

We experimentally demonstrate that our scale-equivariant tracker outperforms its conventional counterpart on OTB and VOT benchmarks and on the synthetically generated T-MNIST and S-MNIST datasets, where T-MNIST is designed to keep the object at constant scale, and S-MNIST varies the scale in known manner.

The experiments on T-MNIST and S-MNIST show the importance of proper scale measurement for all sequences, regardless whether they have scale change or not. For the standard OTB and VOT benchmarks, our tracker proves the power of scale equivariance. It not only improves the tracking in case of scaling, but also when other factors of variations are present (see Figure 8). It affects the performance in two ways: it prevents erroneous jumps to similar objects at a different size and it provides a better consistent estimate of the scale.

4

DISCO: ACCURATE DISCRETE SCALE CONVOLUTIONS

4.1 INTRODUCTION

Scale is a natural attribute of every object, as basic property as location and appearance. And hence it is a factor in almost every task in computer vision. In image classification, global scale invariance plays an important role in achieving accurate results [84]. In image segmentation, scale equivariance is important as the output map should scale proportionally to the input [1]. And in object detection or object tracking, it is important to be scale-agnostic [126], which implies the availability of both scale invariance as well as scale equivariance as the property of the method. Where scale invariance or equivariance is usually left as a property to learn in the training of these computer vision methods by providing a good variety of examples [103], we aim for accurate scale analysis for the purpose of needing less data to learn from.

Scale of the object can be derived externally from the size of its silhouette, e.g [166], or internally from the scale of its details, e.g [18]. External scale estimation requires the full object to be visible. It will easily fail when the object is occluded and/or when the object is amidst a cluttered background, for example for people in a crowd [138], when proper detection is hard. In contrast, internal scale estimation is build on the scale of common details [135], for example deriving the scale of a person from the scale of a sweater or a face. Where internal scale has better chances of being reliable, it poses heavier demands on the accuracy of assessment than external scale estimation. We focus on improvement of the accuracy of internal scale analysis.

We focus on accurate scale analysis on the generally applicable scale-equivariant convolutional neural networks [8, 144, 163]. A scale-equivariant network extends the equivariant property of conventional convolutions to the scale-translation group. It is achieved by rescaling the kernel basis and sharing weights between scales. While the weight sharing is defined by the structure of the group [25], the proper way to rescale kernels is an open problem. In [8, 144], the authors propose to rescale kernels in the continuous domain to project them later on a pixel grid. This permits the use of arbitrary scales, which is important to many application problems, but the procedure may cause a significant equivariance error [144]. Therefore, Worrall and Welling [163] model rescaling as a dilation, which guarantees a low equivariance error at the expense of permitting only integer scale factors. Due to the continuous nature of observed scale in segmentation, tracking or classification alike, integer scale factors may not cover the range of variations in the best possible way.

In the chapter, we show how the equivariance error affects the performance of SE-CNNs. We make the following contributions:

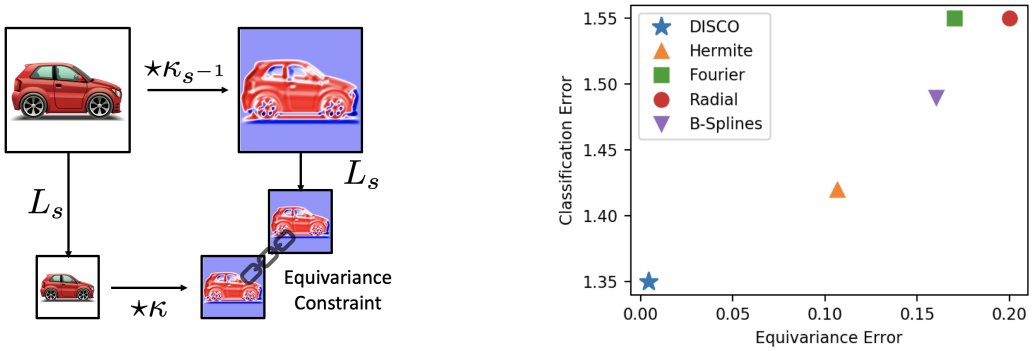


Figure 10: Left: the necessary constraint for scale-equivariance. When it is not satisfied an equivariance error appears. Right: Equivariance error vs. Classification error for scale-equivariant models on MNIST-scale. DISCO achieves the lowest equivariance error and this leads to the best classification accuracy. Alongside DISCO, we test SESN models with Hermite [144], Fourier [181], Radial [54] and B-Spline [8] bases.

- From first principles we derive the best kernels, which minimize the equivariance error.
- We find the conditions when the solution exists and find a good approximation when it does not exist.
- We demonstrate that an SE-CNN with the proposed kernels outperforms recent SE-CNNs in classification and tracking in both accuracy and compute time. We set new state-of-the-art results on MNIST-scale and STL-10.

The proposed approach contains [163] as a special case. Moreover, the proposed kernels can't be derived from [144] and vice versa. The union of our approach and the approach presented in [144] covers the whole set of possible SE-CNNs for a finite set of scales.

4.2 RELATED WORK

GROUP EQUIVARIANT NETWORKS. In recent years, various works on group-equivariant convolution neural networks have appeared. In majority, they consider the roto-translation group in 2D [25, 27, 76, 154, 156, 161], the roto-translation group in 3D [23, 89, 149, 155, 160], the compactified rotation-scaling group in 2D [72] and the rotation group 3D [23, 26, 44]. In [24, 90, 95] the authors demonstrate how to build convolution networks equivariant to arbitrary compact groups. All these papers cover group-equivariant networks for compact groups. In this chapter, we focus the scale-translation group which is an example of a non-compact group.

DISCRETE OPERATORS. Minimization of the discrepancies between the theoretical properties of continuous models and their discrete realizations has been studied for a variety of computer vision tasks. Lindeberg [104, 105] proposed a method for building a scale-space for discrete signals. The approach relied on the connection between the discretized version of the diffusion equation and the structure of images. While this

method considered the scale symmetry of images and significantly improved computer vision models in the pre-deep-learning era, it is not directly applicable to our case of scale-equivariant convolutional networks.

In [39], Diaconu and Worrall demonstrate how to construct rotation-equivariant CNNs on the pixel grid for arbitrary rotations. The authors propose to learn the kernels which minimize the equivariance error of rotation-equivariant convolutional layers. The method relies on the properties of the rotation group and cannot be generalized to the scale-translation group. In this chapter, we show how to minimize the equivariance error for scale-convolution without the use of extensive learning.

SCALE-EQUIVARIANT CNNS. An early work of [84] introduced SI-ConvNet, a model where the input image is rescaled into a multi-scale pyramid. Alternatively, Xu *et al.* [171] proposed SiCNN, where a multi-scale representation is built from rescaling the network filters. While these modified convolutional networks significantly improve image classification, they require run-time interpolation. As a result they are several orders slower than standard CNNs.

In [8, 144, 181] the authors propose to parameterize the filters by a trainable linear combination of a pre-calculated, fixed multi-scale basis. Such a basis is defined in the continuous scale domain and projected on a pixel grid for the set of scale factors. The models do not involve interpolation during training nor inference. As a consequence, they operate within reasonable time. The continuous nature of the bases allows for the use of arbitrary scale factors, but it suffers from a reduced accuracy as the projection on the discrete grid causes an equivariance error.

Worral and Welling [163] propose to model filter rescaling by dilation. This solves the equivariance error of the previous method at the price of permitting only integer scale factors. That makes the method less suited for object tracking, depth analysis and fine-grained image classification, where subtle changes in the image scale are important in the performance. Our approach combines the best of the both worlds as it guarantees a low equivariance error for arbitrary scale factors.

ACCURATE SCALE ANALYSIS. Approaches based on feature pyramids are applied in many tasks [66,103,124,153]. Their implementation require a significant specialisation of the network architecture. Models based on direct scale regression [21, 99, 126] have proved to be accurate in scale analysis, but they rely on a complicated training procedure. Scale-equivariant networks require only a drop-in replacement of the standard convolutions by scale-convolutions, while keeping the training procedure unchanged [8, 142, 144, 163]. We appreciate the universal applicability of scale-equivariant networks. We focus on this particular use in our implementation while the method we set out in this chapter will also apply to other ways of using scale in computer vision.

Existing models for scale-equivariant networks bring computational overhead, which significantly slows down the training and the inference. In this chapter, we present scale-equivariant models which allow for the accurate analysis of scale with a minimum computational overhead while retaining the advantage of being an easy replacement of convolutional layers to improve.

4.3 METHOD

EQUIVARIANCE. A mapping g is equivariant under a transformation L if and only if there exists L' such that $g \circ L = L' \circ g$. If the mapping L' is identity, then g is invariant under transformation L.

SCALE TRANSFORMATIONS. Given a function $f : R \to R$ its scale transformation L_s is defined by

$$L_s[f](t) = f(s^{-1}t), \quad \forall s > 0 \tag{4.1}$$

We refer to cases with $s > 1$ as up-scalings and to cases with $s < 1$ as down-scalings, where $L_{1/2}[f]$ stands for a function down-scaled by a factor of 2.

THE SCALE-TRANSLATION GROUP. We are interested in equivariance under the scale-translation group H and its subgroups. It consists of the translations t and scale transformations s which preserve the position of the center. $H = \{(s, t)\} = S \rtimes T$ is a semi-direct product of a multiplicative group $S = (R^+, +)$ and an additive group $T = (R, +)$. For the multiplication of its elements we have $(s_2, t_2) \cdot (s_1, t_1) = (s_1 s_2, s_2 t_1 + t_2)$. Scale transformation of a function defined on group H consists of a scale transformation of its spatial part as it is defined in the Equation 4.1 and a corresponding multiplicative transformation of its scale part. In other words

$$L_{\hat{s}}[f](s, t) = f(s\hat{s}^{-1}, \hat{s}^{-1}t) \tag{4.2}$$

4.3.1 *Scale-Convolution*

A scale-convolution of f and a kernel κ both defined on scale s and translation t is given by: [144]:

$$[f \star_H \kappa](s, t) = \sum_{s'} [f(s', \cdot) \star \kappa_s(s^{-1}s', \cdot)](\cdot, t) \tag{4.3}$$

where κ_s stands for an s-times up-scaled kernel κ, $\star$ and $\star_H$ are convolution and scale-convolution. The exact way the up-scaling is performed depends on how the down-scaling of the input signal works.

Scale-convolution is equivariant to transformations $L_{\hat{s}}$ from the group H, therefore the following holds true by definition:

$$[L_{\hat{s}}[f] \star_H \kappa] = L_{\hat{s}}[f \star_H \kappa] \tag{4.4}$$

Expanding the left-hand side of this relation by using Equation 4.3, choosing $s = 1$ and replacing $s' \to s'\hat{s}$ we find:

$$[L_{\hat{s}}[f] \star_H \kappa](s, t) = \sum_{s'} [L_{\hat{s}}[f(s', \cdot)] \star \kappa(\hat{s}s', \cdot)](\cdot, t) \tag{4.5}$$

For the right-hand side we have:

$$L_{\hat{s}}[f \star_H \kappa](s, t) = \sum_{s'} L_{\hat{s}}[f(s', \cdot) \star \kappa_{\hat{s}^{-1}}(\hat{s}s', \cdot)](\cdot, t) \tag{4.6}$$

Equating the two sides and choosing f to be zero on all scales but $s = 1$, we obtain the equivariance constraint for the kernels

$$L_s[f] \star \kappa = L_s[f \star \kappa_{s^{-1}}], \quad \forall f, s \tag{4.7}$$

We have found that *the mapping defined by Equation 4.3 is scale-equivariant only if a kernel and its up-scaled versions satisfy Equation 4.7*. Thus, it proves to be the necessary condition for scale-equivariant convolutions. In [8, 144, 181] the opposite, sufficient condition was proved. As a whole it defines the relation between scale convolution and the constraints of its kernels.

4.3.2 *Exact Solution*

In the continuous domain, convolution is defined as an integral over the spatial coordinates. [8, 144, 181] derives a solution for Equation 4.7:

$$\kappa_s(t) = s^{-1}\kappa(s^{-1}t) \tag{4.8}$$

However, when such kernels are calculated and projected on the pixel grid, a discrepancy between the left-hand side and the right-hand side of Equation 4.7 will appear. We refer to such inequality as the *equivariance error*.

We aim at directly solving Equation 4.7 in the discrete domain. In general, for discrete signals down-scaling is a non-invertible operation. Thus L_s is well-defined only for $s < 1$. We start by solving Equation 4.7 for 1-dimensional discrete signals. We then prove its generalization to the 2-dimensional case. Figure 11 illustrates the approach.

Let us consider a discrete signal f represented as a vector $\boldsymbol{f}$ of length N_{in}. It is down-scaled to length $N_{\text{out}} < N_{\text{in}}$ by L_s, which is represented as a rectangular interpolation matrix $\boldsymbol{L}$ of size $N_{\text{out}} \times N_{\text{in}}$. A convolution with a kernel κ is represented as a multiplication with a matrix $\boldsymbol{K}$ of size $N_{\text{out}} \times N_{\text{out}}$, and with a kernel $\kappa_{s^{-1}}$ written as a matrix $\boldsymbol{K}_{s^{-1}}$ of size $N_{\text{in}} \times N_{\text{in}}$. Then Equation 4.7 can be rewritten in matrix form as follows:

$$\boldsymbol{KLf} = \boldsymbol{LK}_{s^{-1}}\boldsymbol{f}, \ \forall \boldsymbol{f} \quad \Longleftrightarrow \quad \boldsymbol{KL} = \boldsymbol{LK}_{s^{-1}} \tag{4.9}$$

Without loss of generality we assume circular boundary conditions. Then the matrix representations $\boldsymbol{K}$ and $\boldsymbol{K}_{s^{-1}}$ are both circulant and their eigenvectors are the column-vectors of the Discrete Fourier Transform $\boldsymbol{F}$ [5, 17, 71]:

$$\boldsymbol{K}_{s^{-1}} = \boldsymbol{F}\text{diag}(\boldsymbol{F}\boldsymbol{\kappa}_{s^{-1}})\boldsymbol{F}^* \tag{4.10}$$

where $\boldsymbol{\kappa}_{s^{-1}}$ is a vector representation of $\kappa_{s^{-1}}$ padded with zeros. After substituting Equation 4.10 into Equation 4.9 and multiplying both sides by $\boldsymbol{F}$ from the right, we get:

$$\boldsymbol{KLF} = \boldsymbol{LF}\text{diag}(\boldsymbol{F}\boldsymbol{\kappa}_{s^{-1}}) \tag{4.11}$$

The left-hand side of the equation is obtained from $\boldsymbol{LF}$ by multiplying it with a diagonal matrix from the right. Thus, each column of the matrix $\boldsymbol{KLF}$ is proportional to the

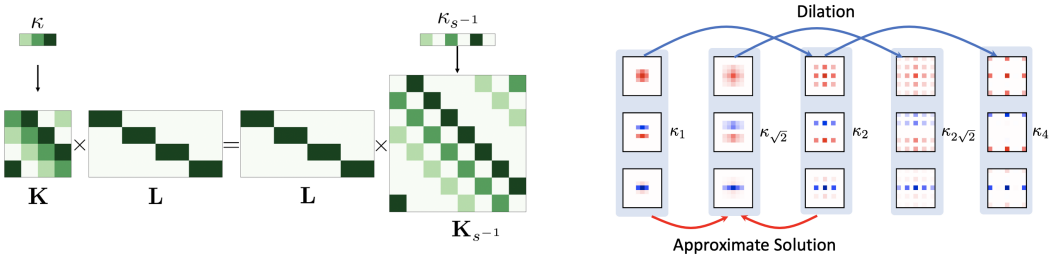


Figure 11: Left: a matrix representation of the 1-dimensional case of the equivariance constraint for $N_{in} = 8$ and $N_{out} = 4$. Right: a multi-scale kernel initialization. $\sqrt{2}$ is the smallest non-integer scale, for which the kernel is approximated by minimizing Equation 4.23, the rest of the kernels can be obtained with dilation.

corresponding column of the matrix $\boldsymbol{LF}$. In Section 4.3.3 we prove that *such a relation is possible if and only if the matrix $\boldsymbol{L}$ performs a down-scaling by an integer scale factor*.

When the requirement is satisfied, the solution with respect to $\boldsymbol{\kappa}_{s^{-1}}$ is the dilation of $\boldsymbol{\kappa}$ by factor s. Such a solution is also known as the *à trous algorithm* [74]:

$$(\boldsymbol{\kappa}_{s^{-1}})_{is} = \sum_i \boldsymbol{F}^*_{ij}(\boldsymbol{KLF})_{1j}/(\boldsymbol{LF})_{1j} = \boldsymbol{\kappa}_i \tag{4.12}$$

4.3.3 *Solutions in 1D*

Lemma 1. *Equation 4.7 has non-trivial solutions with respect to $\boldsymbol{K}_{s^{-1}}$ only if L performs downscaling by an integer factor.*

Proof. Let us consider $\boldsymbol{P}_{\text{in}}$ and $\boldsymbol{P}_{\text{out}}$, matrices of circular shift of rows of sizes $N_{\text{in}} \times N_{\text{in}}$ and $N_{\text{out}} \times N_{\text{out}}$ correspondingly. With no loss of generality we assume circular boundary conditions for convolutions. Thus, matrices $\boldsymbol{K}, \boldsymbol{K}_{s^{-1}}$ are circulant, and therefore $\boldsymbol{K} = \boldsymbol{P}_{\text{out}}\boldsymbol{K}\boldsymbol{P}^T_{\text{out}}$ and $\boldsymbol{K}_{s^{-1}} = \boldsymbol{P}_{\text{in}}\boldsymbol{K}_{s^{-1}}\boldsymbol{P}^T_{\text{in}}$ [107]. If we substitute it into Equation 4.7 we have the following:

$$\boldsymbol{P}^i_{\text{out}}\boldsymbol{K}(\boldsymbol{P}^T_{\text{out}})^i\boldsymbol{L} = \boldsymbol{L}\boldsymbol{P}^j_{\text{in}}\boldsymbol{K}_{s^{-1}}(\boldsymbol{P}^T_{\text{in}})^j, \quad \forall i, j \in Z \tag{4.13}$$

If we multiply it from the left by $(\boldsymbol{P}^T_{\text{out}})^i$ and from the right by $\boldsymbol{P}^j_{\text{in}}$ we get the following equation:

$$\boldsymbol{K}(\boldsymbol{P}^T_{\text{out}})^i\boldsymbol{L}\boldsymbol{P}^j_{\text{in}} = (\boldsymbol{P}^T_{\text{out}})^i\boldsymbol{L}\boldsymbol{P}^j_{\text{in}}\boldsymbol{K}_{s^{-1}} \tag{4.14}$$

We can now multiply Equation 4.14 by a coefficient α_{ij} and then the following holds true:

$$\boldsymbol{K}\sum_{i=1}^{N_{\text{out}}}\sum_{j=1}^{N_{\text{in}}}\alpha_{ij}\boldsymbol{Q}_{ij} = \sum_{i=1}^{N_{\text{out}}}\sum_{j=1}^{N_{\text{in}}}\alpha_{ij}\boldsymbol{Q}_{ij}\boldsymbol{K}_{s^{-1}}, \forall\alpha_{ij} \tag{4.15}$$

where $\boldsymbol{Q}_{ij} = (\boldsymbol{P}_{\text{out}}^T)^i \boldsymbol{L} \boldsymbol{P}_{\text{in}}^j$. Equation 4.15 holds true for all α_{ij}. Which gives us the following system of equations:

$$\begin{cases} \boldsymbol{K}(\boldsymbol{Q}_{00} - \boldsymbol{Q}_{N_{\text{out}}N_{\text{in}}}) = (\boldsymbol{Q}_{00} - \boldsymbol{Q}_{N_{\text{out}}N_{\text{in}}})\boldsymbol{K}_{s^{-1}} \\ \boldsymbol{K}(\boldsymbol{Q}_{01} - \boldsymbol{Q}_{N_{\text{out}},N_{\text{in}}+1}) = (\boldsymbol{Q}_{01} - \boldsymbol{Q}_{N_{\text{out}},N_{\text{in}}+1})\boldsymbol{K}_{s^{-1}} \\ \dots \\ \boldsymbol{K}(\boldsymbol{Q}_{10} - \boldsymbol{Q}_{N_{\text{out}}+1,N_{\text{in}}}) = (\boldsymbol{Q}_{10} - \boldsymbol{Q}_{N_{\text{out}}+1,N_{\text{in}}})\boldsymbol{K}_{s^{-1}} \\ \dots \end{cases} \tag{4.16}$$

Which has non-trivial solutions if the expressions in all brackets are equal to zero. Thus, $\boldsymbol{Q}_{00} = \boldsymbol{Q}_{N_{\text{out}}N_{\text{in}}}$. In other words, $\boldsymbol{L}$ is a row-circulant matrix and N_{in} is divisible by N_{out}. Therefore, the downscaling is performed by an integer factor $N_{\text{in}}/N_{\text{out}}$ □

In order to obtain the solution of Equation 4.7 we represent convolutional matrices by using their eigendecompositions.

$$\begin{aligned} \boldsymbol{K} &= \boldsymbol{F}_{\text{out}} \text{diag}(\boldsymbol{F}_{\text{out}}\boldsymbol{\kappa}) \boldsymbol{F}_{\text{out}}^* \\ \boldsymbol{K}_{s^{-1}} &= \boldsymbol{F}_{\text{in}} \text{diag}(\boldsymbol{F}_{\text{in}}\boldsymbol{\kappa}_{s^{-1}}) \boldsymbol{F}_{\text{in}}^* \end{aligned} \tag{4.17}$$

where $\boldsymbol{F}_{\text{in}}, \boldsymbol{F}_{\text{out}}$ are matrices of the Discrete Fourier Transform of appropriate sizes and $\boldsymbol{\kappa}, \boldsymbol{\kappa}_{s^{-1}}$ are vector representations of convolutional kernels. After substituting the second part of Equation 4.17 into Equation 4.7 we obtain:

$$\boldsymbol{K}\boldsymbol{L} = \boldsymbol{L}\boldsymbol{F}_{\text{in}} \text{diag}(\boldsymbol{F}_{\text{in}}\boldsymbol{\kappa}_{s^{-1}}) \boldsymbol{F}_{\text{in}}^* \tag{4.18}$$

We then multiply both sides of the equation with $\boldsymbol{F}_{\text{in}}$ from the right.

$$(\boldsymbol{K}\boldsymbol{L}\boldsymbol{F}_{\text{in}})_{ij} = \sum_k (\boldsymbol{L}\boldsymbol{F}_{\text{in}})_{ik} \text{diag}(\boldsymbol{F}_{\text{in}}\boldsymbol{\kappa}_{s^{-1}})_{kj} \tag{4.19}$$

As the left hand side is per-column proportional to $\boldsymbol{L}\boldsymbol{F}_{\text{in}}$, we can calculate the solution just by using the first row of each matrix.

$$(\boldsymbol{F}_{\text{in}}\boldsymbol{\kappa}_{s^{-1}})_j = \frac{(\boldsymbol{K}\boldsymbol{L}\boldsymbol{F}_{\text{in}})_{1j}}{(\boldsymbol{L}\boldsymbol{F}_{\text{in}})_{1j}} \tag{4.20}$$

The first row of $\boldsymbol{F}_{\text{in}}$ consists of ones so as the first row of $\boldsymbol{L}\boldsymbol{F}_{\text{in}}$. Additionally, $(\boldsymbol{K}\boldsymbol{L}\boldsymbol{F}_{\text{in}})_{1j} = s^{-1}[\boldsymbol{\kappa}, \boldsymbol{\kappa}]_j$. As the discrete Fourier image of the solution is a scaled concatenated image of the source, the solution is just a dilation of the original kernel [107].

4.3.4 *Solutions in 2D*

We are interested in solving Equation 4.7 with respect to $\kappa_{s^{-1}}$ for any set of κ's which forms a complete basis in the space of square matrices of a certain, fixed size. If the solution exists for any basis, then it exists for a basis of 2-dimensional separable kernels. As the rank of the set of solutions is less or equal to the rank of the initial basis, the

solution is separable as well. Let us consider an image $\boldsymbol{F}$ of size $N_{\text{in}} \times N_{\text{in}}$. Taking into account that its rescaling is a separable operation, the matrix form of Equation 4.7 is:

$$\boldsymbol{K'LFL}^T\boldsymbol{K}^T = \boldsymbol{LK'}_{s^{-1}}\boldsymbol{FK}^T_{s^{-1}}\boldsymbol{L}^T, \quad \forall \boldsymbol{F} \tag{4.21}$$

where $\boldsymbol{K'}$ and $\boldsymbol{K}$ are matrix representations of 1-dimensional components of a separable kernel. As Equation 4.21 holds true for all images, it satisfies $\boldsymbol{F} = \boldsymbol{fc}^T$ and $\boldsymbol{F} = \boldsymbol{cf}^T$ where $\boldsymbol{c}$ is a vector of constants and $\boldsymbol{f}$ is an arbitrary vector. After substituting these functions into Equation 4.21 it degenerates into a system of two independent equations up to a multiplication constant:

$$\begin{cases} \boldsymbol{KL} = \boldsymbol{LK}_{s^{-1}} \\ \boldsymbol{K'L} = \boldsymbol{LK'}_{s^{-1}} \end{cases} \tag{4.22}$$

Thus, if a solution exists for 2-dimensional discrete signals it also exists for the 1-dimensional case.

4.3.5 *Approximate solution*

Let us consider a scale-convolutional layer. One of its hyper-parameters is the set of scales it operates on. For the cases of non-integer scale factors any kernel will introduce an equivariance error into the network. Thus, it is reasonable to use integer scales as reference points and add intermediate scales to cover the required range of scale factors best. Let us choose a set of scales $\{1, \sqrt{2}, 2, 2\sqrt{2}, 4, 4\sqrt{2}, \dots\}$. The set of corresponding kernels is $\{\kappa_1, \kappa_{\sqrt{2}}, \kappa_2, \kappa_{2\sqrt{2}}, \dots\}$. As the smallest kernel is known, all kernels defined on integer scales can be calculated as its dilated versions. And, when kernel $\kappa_{\sqrt{2}}$ is defined, all intermediate kernels $\kappa_{2\sqrt{2}}, \kappa_{4\sqrt{2}}, \dots$ can be calculated by using dilation as well. Thus, the only kernel yet unknown is kernel $\kappa_{\sqrt{2}}$.

The kernel $\kappa_{\sqrt{2}}$ can be calculated as a minimizer of the equivariance error based on the Equation 4.7 as follows:

$$\kappa_{\sqrt{2}} = \arg\min_{E_f} \|L[f] \star \kappa_1 - L[f \star \kappa_{\sqrt{2}}]\|_F^2 + \|L[f] \star \kappa_{\sqrt{2}} - L[f \star \kappa_2]\|_F^2 \tag{4.23}$$

where $L = L_{1/\sqrt{2}}$ is a down-scaling by a factor $\sqrt{2}$.

4.3.6 *Implementation*

To construct a scale-equivariant convolution we parametrize the kernels as a linear combination of a fixed multi-scale basis. The basis is then fixed and only corresponding coefficients are trained. The coefficients are shared for all scales.

We utilize the standard pixel basis on the smallest integer scale. The bases for the rest of the integer scales are computed as a dilation. The basis on the smallest non-integer scale is approximated by applying gradient descent to Equation 4.23. We note that it takes negligible time to compute all of the basis functions before training. See supplementary materials for more details. We refer to scale-convolutions with the proposed bases as

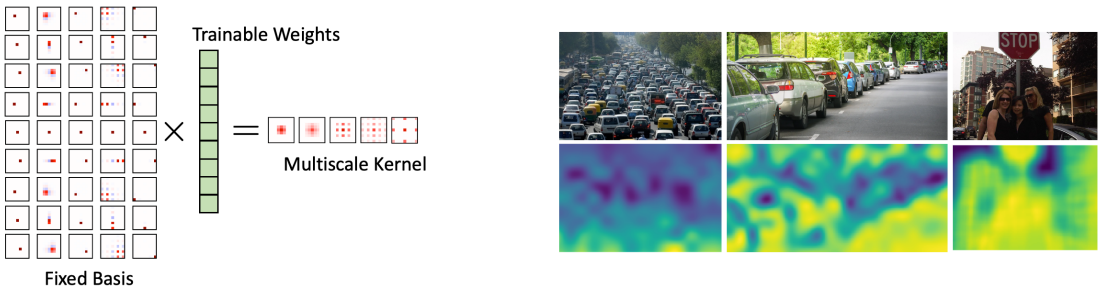


Figure 12: Left: kernels are computed via multiplying a fixed multi-scale basis with trainable weights. Right: images and their scale fields produced by the DISCO model trained to contrast scales.

Discrete Scale Convolutions or shortly DISCO. As DISCO kernels are sparse, they allow for lower computational complexity.

Let us consider a scale-convolutional layer defined on scales $\{1, \sqrt{2}, 2, 2\sqrt{2}, 4, \}$. The kernel on the smallest scale is of size 3×3. As it was noted, as soon as the kernel on the intermediate $\sqrt{2}$ scale is defined, all other kernel can be calculated via dilation.

In scale-convolutional layer the kernels κ are parametrized as follows:

$$\kappa_s = \sum_j \psi_{s,j} w_j \tag{4.24}$$

where $\psi_{s,j}$ is a j-th basis function defined on scale s, and w_j is the corresponding trainable coefficient.

As the basis is fixed during the training, it needs to be defined *a priori*. On the smallest scale all basis functions are just elements of the standard basis, i.e. if $\psi_{1,i}$ is the i-th basis function for the 3×3 filters on the first scale, then $\psi_{1,0}$ is a 3×3 matrix where the only non-zero element is a 1 in the top-left corner, and $\psi_{1,4}$ is a 3×3 matrix with 1 in the center. On the next integer scale 2, the basis is obtained according to Equation 12 of the main paper and computed as a dilation of $\psi_{1,i}$. To obtain non-integer scale bases we start by approximating the first intermediate $\sqrt{2}$ scale basis $\psi_{\sqrt{2},j}$ functions by minimizing the following objective function:

$$\|L[f] \star \psi_{1,j} - L[f \star \psi_{\sqrt{2},j}]\|_F^2 + \|L[f] \star \psi_{\sqrt{2},j} - L[f \star \psi_{2,j}]\|_F^2 \tag{4.25}$$

where f is a random sample from $\mathcal{N}(0, 1)$ and L is an operation of downsampling by a factor of $\sqrt{2}$ by using bicubic interpolation. The basis for the scale $2\sqrt{2}$ is calculated as a dilation of the approximated $\sqrt{2}$ basis. See Figure 12 for more details.

After all basis functions are calculated, the basis is packed into a tensor of size:

$$\texttt{num\_functions} \times \texttt{num\_scales} \times \texttt{height} \times \texttt{width}$$

and used for runtime kernel calculations with the algorithm provided by [144].

4.3.7 *Computational Complexity*

Let us consider a scale-convolutional layer with a set of N_s scales with step $\sigma > 1$. The smallest kernel size is $W \times W$. The computational complexity for calculating the output for one spatial position for the state-of-the-art method from [144] can be estimated as follows:

$$O(SESN) \sim O(W^2(1 + \sigma^2 + \cdots + \sigma^{2N_s-2})) \sim O\left(W^2 \frac{\sigma^{2N_s} - 1}{\sigma^2 - 1}\right) \sim O(W^2 \sigma^{2N_s}) \quad (4.26)$$

In contrast, for DISCO we arrive the following complexity:

$$O(DISCO) \sim O(N_s W^2) \quad (4.27)$$

Thus, where the state of the art SESN convolution grows exponentially in computational complexity with the number of scales, DISCO allows for linear growth.

When using a scale step of $\sqrt{2}$ we achieve a speedup of:

$$\frac{O(SESN)}{O(DISCO)} \sim \frac{2^{N_s}}{N_s}. \quad (4.28)$$

The main reason for the acceleration is that in SESN the filters are dense, as they are rescaled in the continuous domain by using Equation 4 of the main paper, while DISCO filters are sparse as the rescaling is performed by using dilation for the majority of scales. The actual speedup depends on the particular implementation of scale-convolution with such kernels. The current implementation is limited by the functionality of modern deep learning software which is not optimized for sparse filters of a big spatial extent.

4.3.8 *General Solution*

While in many models which consider scale the scale-step is a root of some integer number, it is possible to build a DISCO model with arbitrary scale-steps. Let us consider a scale-convolutional layer defined on scales $\{s_0, as_0, a^2 s_0, \dots a^N s_0\}$ where $a > 1$. In order to construct kernels for such a layer it is first required to calculate a basis $\{\psi_{s_0,j}, \psi_{as_0,j}, \dots \psi_{a^N s_0,j}\}$ for all j. The basis can be calculated as a minimizer of the following objective:

$$\mathcal{L}(\psi_{s_0,j}, \psi_{as_0,j}, \dots \psi_{a^N s_0,j}) = E_f \sum_{\substack{k,l=0 \\ k>l}}^{k,l=N} \| L_{a^{l-k}}[f] \star \psi_{a^l s_0,j} - L_{a^{l-k}}[f \star \psi_{a^k s_0,j}] \|_F^2 \quad (4.29)$$

Model	Basis	MNIST	MNIST+	Equi. error	# Params.
CNN	-	2.02 ± 0.07	1.60 ± 0.09	-	495 K
SiCNN	-	2.02 ± 0.14	1.59 ± 0.03	-	497 K
SI-ConvNet	-	1.82 ± 0.11	1.59 ± 0.10	-	495 K
SEVF	-	2.12 ± 0.13	1.81 ± 0.09	-	475 K
DSS	Dilation	1.97 ± 0.08	1.57 ± 0.09	0.0	494 K
SS-CNN	Radial	1.84 ± 0.10	1.76 ± 0.07	-	494 K
SESN	Hermite	1.68 ± 0.06	1.42 ± 0.07	0.107	495 K
SESN	B-Spline	1.74 ± 0.08	1.49 ± 0.05	0.163	495 K
SESN	Fourier	1.88 ± 0.07	1.55 ± 0.07	0.170	495 K
SESN	Radial	1.74 ± 0.07	1.55 ± 0.10	0.200	495 K
DISCO	Discrete	$\mathbf{1.52 \pm 0.06}$	$\mathbf{1.35 \pm 0.05}$	0.004	495 K

*Table 12: The classification error of various methods on the MNIST-scale dataset, lower is better. We test both the regime with and without data augmentation, where scaling data augmentation is denoted by "+". All results are reported as mean ± std over 6 different, fixed realizations of the dataset. The best results are **bold**.*

4.4 EXPERIMENTS

4.4.1 *Equivariance Error*

To quantitatively evaluate the equivariance error of DISCO versus other methods for scale-convolution [8, 144, 181], we follow the approach proposed in [144]. In particular, we randomly sample images from the MNIST-Scale dataset [144] and pass in through the scale-convolution layer. Then, the equivariance error is calculated as follows:

$$\Delta = \sum_{s} \|L_s \Phi(f) - \Phi(L_s f)\|_2^2 / \|L_s \Phi(f)\|_2^2 \tag{4.30}$$

where Φ is scale-convolution with weights initialized randomly.

The equivariance error for each model is reported in Table 12 and in Figure 10. Note that we can not directly compare against [163] as it only permits integer scale factors. As can be seen, there exists a correlation between an equivariance error and classification accuracy. DISCO model attains the lowest equivariance error.

4.4.2 *Image Classification*

We conduct several experiments to compare various methods for scale analysis in image classification. Alongside DISCO, we test SI-ConvNet [84], SS-CNN [54], SiCNN [171], SEVF [110], DSS [163] and SESN [144]. By relying on the code provided by the authors we additionally reimplement SESN models with other bases such as B-Splines [8], Fourier-Bessel Functions [181] and Log-Radial Harmonics [54, 117].

Interpolation	Nearest	Bilinear	Bicubic
Error	1.36 ± 0.06	1.37 ± 0.05	1.35 ± 0.05

Table 13: Classification accuracy on MNIST-scale for different interpolation methods which are used for approximate basis calculations.

Model	WRN	SiCNN	SI-ConvNet	DSS	SS-CNN	SESN	DISCO
Basis	-	-	-	Dilation	Radial	Hermite	Discrete
Time, s	10	110	55	40	15	165	50
Error	11.48	11.62	12.48	11.28	25.47	8.51	**8.07**

*Table 14: The classification error on STL-10. The best results are in **bold**. The average compute time per epoch is reported in seconds. DISCO sets a new state-of-the-art result in the supervised learning setting.*

MNIST-SCALE. Following [144] we conduct experiments on the MNIST-scale dataset. The dataset consists of 6 splits, each of which contains 10,000 images for training, 2,000 for validation and 50,000 for testing. Each image is a randomly rescaled version of the original from MNIST [97]. The scaling factors are uniformly sampled from the range of $0.3 - 1.0$.

As a baseline model we use the SESN model [144]. It consists of 3 convolutional and 2 fully-connected layers. Each layer has filters of size 7×7. We keep the number of parameters the same for all SESN models and for DISCO. The main difference between the SESN and DISCO models is in the basis for scale-convolutions. We also discovered that average-pooling works slightly better for the DISCO, while for all other methods it either has no effect or worsens the performance. Both SESN and DISCO use the same set of scales in scale convolutions: $\{1, 2^{1/3}, 2^{2/3}, 2\}$

All models are trained with the Adam optimizer [87] for 60 epochs with a batch size of 128. We set the initial learning rate at 0.01 and divide it by 10 after 20 and once more after 40 epochs. We conduct the experiments with 2 different settings: without data augmentation and with scaling augmentation. We run the experiments on 6 different realizations of the MNIST-scale. We report the mean ± standard deviation over these runs.

As can be seen from Table 12, our DISCO model outperforms other scale equivariant networks in accuracy and equivariance error and sets a new state-of-the-art result.

We found in our experiments that the interpolation method which is used to calculate a basis by using equation 4.25 does not affect the final solution. The relative mean squared error between bases is less than one percent. Moreover, DISCO model demonstrates almost the same results on MNIST-scale while various interpolation methods are used. See Table 13 for more results.

STL-10. To demonstrate how accurate scale equivariance helps when the training data is limited, we conduct experiments on the STL-10 [22] dataset. This dataset consists

Equi. Error	STL-10 Error
0.240	8.63
0.082	8.25
0.003	**8.07**

Table 15: Classification accuracy on STL-10 and the equivariance error for the DISCO model with different filters. The first and the second rows correspond to the cases when the basis for the intermediate scale is not optimized.

Model	SiamFC [10]	TriSiam [42]	SiamFC+ [177]	SE-SiamFC+ [142]	DISCO
FPS	-	-	56	14	28
AUC	0.61	0.62	0.67	**0.68**	**0.68**

*Table 16: Performance comparisons on the OTB-13 tracking benchmark. The best results are **bold**. We report the average number of framer per second (FPS) per sequence. Higher FPS and AUC are better.*

of just 8,000 training and 5,000 testing images, divided into 10 classes. Each image has a resolution of 96×96 pixels.

As a baseline we use WideResNet [174] with 16 layers and a widening factor of 8. Scale-equivariant models are constructed according to [144]. All models have the same number of parameters, the same set of scales $\{1, \sqrt{2}, 2\}$ and are trained for the same number of steps. For testing the disco model we use exactly the same setup as described by the authors of [144]. All the models are trained on NVidia GTX 1080 Ti.

The models are trained for 1000 epochs using the SGD optimizer with a Nesterov momentum of 0.9 and a weight decay of $5 \cdot 10^{-4}$. For DISCO, we increase the weight decay to $1 \cdot 10^{-4}$. Tuning weight decay for the other models did not bring any improvement. The learning rate is set to 0.1 at the start and decreased by a factor of 0.2 after the epochs 300, 400, 600 and 800. The batch size is set to 128. During training, we additionally augment the dataset with random crops, horizontal flips and cutout [38].

As can be seen from Table 14, the proposed DISCO model outperforms the other scale-equivariant networks and sets a new state-of-the-art result in the supervised learning setting. Moreover, DISCO is more than 3 times faster than the second-best SESN-model.

We additionally check how accuracy degrades if the basis for the scale of $\sqrt{2}$ is not correctly calculated. While the optimal basis is a minimizer of Equation 4.23, it is possible to stop the stop optimization procedure before convergence and generate then a non-optimal basis. We generated two non-optimal bases which correspond to different moments of the optimization procedure. We report the equivariance error and the classification error on the STL-10 dataset for DISCO with such bases functions in Table 15. It can be seen that lower equivariance errors correspond to lower classification errors.

4.4.3 *Tracking*

To test the ability of DISCO to deliver accurate scale estimation, we choose the task of visual object tracking. We take the recent SE-SiamFC+ [177] tracker and follow the recipe provided in [142] to make it scale-equivariant. We employ the standard one-pass evaluation protocol to compare our method with conventional Siamese trackers and SE-SiamFC+ [142] with a Hermite basis for the scale convolutions. The trackers are evaluated by the usual area-under-the-success-curve (AUC).

The scale-equivariant tracker with DISCO matches the performance of the state-of-the-art SE-SiamFC+, but twice faster as can be seen in Table 16. FPS is measured on Nvidia GTX 1080 Ti for all models.

4.4.4 *Scene Geometry by Contrasting Scales*

We demonstrate the ability of DISCO to propagate scale information through the layers of the network, by presenting a simple approach for geometry estimation of a scene through the use of the intrinsic scale. This is possible because in the DISCO model, we can use high granularity of scale factors and process them more accurately and faster compared to other scale-equivariant models.

We construct a scale-equivariant network with DISCO layers. The weights are initialized from an ImageNet-pretrained network [36] following the approach described in [142]. Next, we strip the classification head of the network and apply global spatial average-pooling. The resulting feature map thus has a dimension $B \times C \times S \times 1 \times 1$, where B, C, S are the batch, channel and scale dimensions respectively. To decode the scale information, we sample the argmax along the scale dimension. Such a tensor has shape $B \times C$ where each element is a scalar that encodes the argmax for each of the objects on each of the channels. Then the tensor is passed to a shallow network, which produces a scale estimate for the input image. The feature extraction network followed by the shallow scale estimator network is denoted as F_θ, where θ is the parameters of the shallow scale estimator, so we do not train the parameters of the feature extractor.

At the core of the method is the scale-contrastive learning algorithm. The model is trained to predict how much one image should be interpolated to match the other. Such an approach does not require any dedicated depth or scale labels. The algorithm is illustrated in Figure 13. First, we sample randomly two scale factors $\gamma_1, \gamma_2 \sim U[0.5, 2.0]$ and apply interpolations $L_{\gamma_1}, L_{\gamma_2}$ to the image $\mathcal{I}$. The transformed images are fed into the network F_θ, which predicts scale estimates $\tilde{\gamma}_1, \tilde{\gamma}_2$ (Figure 13). Then, we minimize the following loss by using the Adam optimizer:

$$\mathcal{L}_{\text{scale}} = E_{\mathcal{I}}\left[\frac{\gamma_2}{\gamma_1} - \frac{\tilde{\gamma}_2}{\tilde{\gamma}_1}\right]^2 = E_{\mathcal{I}}\left[\frac{\gamma_2}{\gamma_1} - \frac{F_\theta(L_{\gamma_2}(\mathcal{I}))}{F_\theta(L_{\gamma_1}(\mathcal{I}))}\right]^2 \longrightarrow \min_\theta \tag{4.31}$$

We train the model on the STL-10 dataset [22] and evaluate it on random images found on the Internet. To infer the scene geometry of the image, we split the image into overlapping patches. For each of them we predict the scale. We provide qualitative results in Figure 13. While the proposed methods was never trained on whole images, it captures the global geometry of the scenes, be it a road or a supermarket.

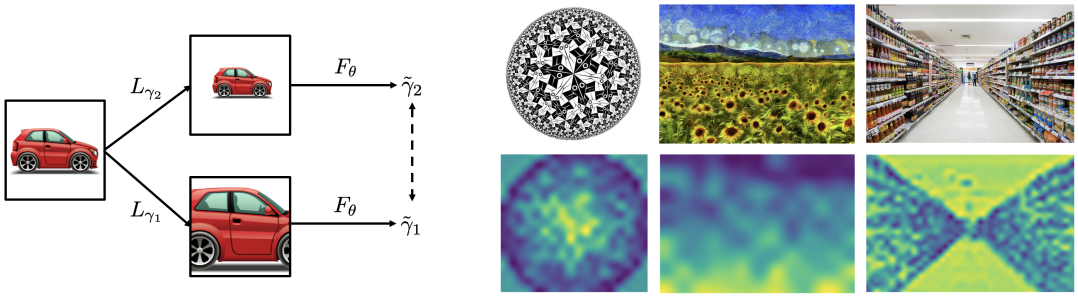


Figure 13: Left: the network is trained to predict the scale difference between an object and its resized version. Right: images and their scale fields produced by the DISCO model trained to contrast scales.

For clarity we provide a PyTorch pseudo-code for DISCO scene geometry estimation (Listing 4.1). We utilize scale-equivariant ResNet as a backbone feature extractor. The produced feature map is reduced in a spatial domain. Then `argmax` along the scale dimension is extracted and passed to the scale MLP regressor to produce a scale estimate. Additional qualitative results are presented in Figure 12.

Listing 4.1: PyTorch pseudo-code for DISCO scene geometry estimation.

```
import torch.nn as nn
import SE_ResNet

class ScaleEstimator(nn.Module):
    def __init__(self):
        super().__init__()
        self.backbone = SE_ResNet(pretrained=True)
        self.regressor = nn.Sequetial(
            nn.Linear(512, 256),
            nn.ReLU(),
            nn.Linear(256, 1),
            nn.ReLU()
        )

    def forward(self, x):
        # x.shape = B, 3, 64, 64
        y = self.backbone(x)
        # y.shape = B, 512, 9, 1, 1
        y = y.mean(-1).mean(-1)
        # y.shape = B, 512, 9
        y = y.argmax(-1)
        scale = self.regressor(y)
        return scale
```

4.5 DISCUSSION

In this work, we demonstrate that the equivariance error affects the performance of equivariant networks. We introduce DISCO, a new class of kernels for scale-convolution, so the equivariance error is minimized. We develop a theory to derive an optimal rescaling to be used in DISCO and analyze under what conditions an optimal rescaling is possible and how to find a good approximation if these conditions do not hold. We also demonstrate how to efficiently incorporate DISCO into an existing scale-equivariant network.

We experimentally demonstrate that DISCO scale-equivariant networks outperform conventional and other scale-equivariant models, setting the new state-of-the-art on the MNIST-Scale and STL-10 datasets. In the visual object tracking experiment, DISCO matches the state-of-the-art performance of SE-SiamFC+ on OTB-13, however, works 2 times faster.

We suppose that the DISCO would be the most useful in problems, where an accurate scale analysis is required, such as multi-object tracking for autonomous vehicles, where the scale of objects can rapidly change due to the relative motion. We additionally want to highlight that the approach presented in this paper can be used to construct scale-equivariant self-attention models with reduced complexity [130].

5

FIRST ORDER FUNCTIONAL VARIATIONS NETWORKS

5.1 INTRODUCTION

Consider a network represented by a function Φ, taking the input image f to produce $\Phi(f) = \Phi(\mathbf{K} \times \mathbf{f})$, where the weights collected in $\mathbf{K}$ are the main parameters of the classification function. In this paper, we aim to make Φ more reliable by introducing small, linear, stochastic variations of Φ, the First order Functional variations, into the function. Including the variations is like considering the Taylor expansion in the functional space where Φ resides. In this way, we aim to improve the stability of the network without changing the data: we do not use new data nor do we modify existing data.

Variations in the functional space of Φ have been used in [25, 30, 47, 81] with great success. These papers have built-in transformations into the network to achieve geometrical equivariance. As a consequence, these expanded networks are equivariant against the global transformations of rotation and scaling. For our purpose, we also apply multiple variations in the network. Rather than implementing global transformation of rotation and scale as the references do, we opt for stochastic, local, linear transformations. These stochastic local transformations are independent of the global geometric variations cited above and could be implemented on top of them, but in this paper, we have not explored that possibility. Our transformations aim to make the network more reliable in general.

We evaluate the reliability of the enhanced network on its classification performance. We do so for a low-resolution CIFAR-10, a small STL-10 [22] and the fine-grained Stanford dog dataset (subset of ImageNet) [86]. We choose the ResNet-network architecture to evaluate the new cell architecture of the network. In addition, we evaluate the reliability of the network by testing it on data with perturbations (rotation-scaling, occlusion, snow, Gaussian noise, Gaussian blur) as presented in [70] benchmark. We evaluate them both on classification while the perturbations were seen during training as well as for the more important case when perturbations are not seen during the training. In summary

- We propose a new cell architecture, First-order Functional variations, to make the resulting network more reliable and solid in its classification performance.
- We demonstrate state-of-the-art performance on a small STL-10 dataset, 95.45%, on CIFAR-10, 94.97% and on the fine-grained Stanford dogs dataset 56.46%.
- We show robustness on *seen* but also general robustness on perturbations *unseen* during training, our network also shows better performance than data augmentation.

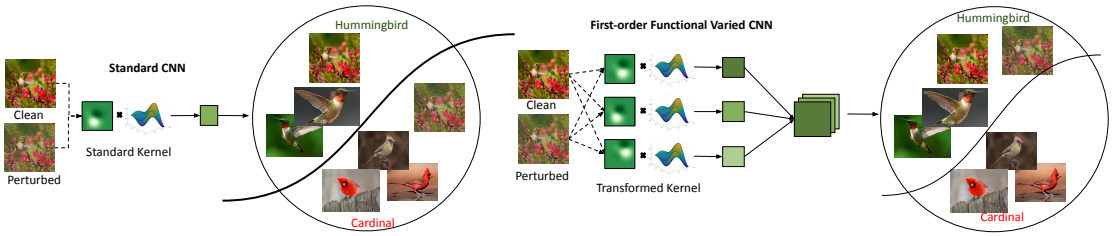


Figure 14: A standard neural network (left) and our First-order Functional varied network (right) at test time for classification. The FoF-network integrates small, stochastic, linear variations in the functional space of classifiers to achieve a more reliable classification without expanding or touching the data.

5.2 RELATED WORK

BUILT-IN IMAGE TRANSFORMS. One of the first methods, suggesting transformations as small units in the network that locally transform their inputs for estimating geometric changes, is in the capsule network architecture [73]. In contrast, in [136] the network is not modified, but rather the update rule of the gradient descent is adapted to learn transformation-invariant weights. Later, both directions evolved to [14, 25, 30, 140, 141, 144, 156, 162, 163], where neural networks are equipped with a rotation or scale-equivariance. In [81, 84, 96] neural network modifications are proposed to make them invariant under input transformations. While these methods consider specific geometric transformations, we focus on small, stochastic, and linear perturbations of the network. We demonstrate how they can be incorporated into a CNN for improved reliability.

THE RELIABILITY OF CLASSIFIERS. Recent work focusing on the reliability of classifiers shows that neural networks are not robust to translations and rotations [46, 85]. The performance of networks [51] drops when the signal-to-noise ratio in the image increases. They fail to perform well in the presence of Gaussian noise or blur, which humans can easily handle [41]. We aim to improve the reliability and solidity of networks in general by considering the network as a function and including small functional variations in the network.

A few carefully composed benchmarks are available to evaluate the performance of classifiers under less ideal circumstances [51, 69, 70]. We select six of perturbations covering the breadth of styles from [70]. In [70] the authors have defined five levels of visual severity for each type of perturbation. In this work, we tune the parameters of perturbation such that the drop in the performance for each perturbation is the same for quantitative comparison of the ability of networks to handle perturbations. Table 17 shows the significance of the standardization for quantitative comparison of robustness compared to standardizing the visual effect therein by the mean square error among images. The MSE shows a large variation in classification performance for different types of perturbations.

DATA-DRIVEN APPROACHES To improve the robustness against natural perturbations, [134] and [147] propose to use batch normalization performed on perturbed images

Metric	Adversarial	Elastic	Occlusion	Gaussian Noise	Blur
Standardized Drop, %	10.22	10.60	10.24	10.10	10.51
MSE	0.02	54.31	199.73	11.79	18.20

Table 17: ***Motivation for the method of standardization.*** *To permit a quantitative comparison, the perturbation is iteratively applied to images until the classification performance by the classifier has dropped by* 10% *with a margin of 0.5%. Data shown on the CIFAR-10, but the iterative procedure was applied to all datasets. In comparison, the standardized drop in accuracy is fair to compare classification performance in contrast to comparison on the basis of the mean square error between the clean and the perturbed image, bottom row, which is suited for visual closeness but unsuited for comparing classification performances.*

instead of clean ones. [9] propose to rectify batch-normalization statistics for enhancing the robustness of neural networks against perturbations. Simultaneously, [133] introduces a noise generator that learns uncorrelated noise distributions, demonstrating that training on noisy images enhances the performance against natural perturbations. [60] trains on images with natural perturbations like occlusions or elastic deformations while achieving good generalization for many unseen perturbations. [127] and [159] note the impossibility to capture all possible natural perturbations in one expression. Therefore, they use generative models to generate images with perturbations to train the network. Adversarial training [15,43,57,57,94] is yet another direction for improving the reliability of networks.

OUR APPROACH. In this work, rather than modifying the data we focus on the reliability of the classification function itself. We do not employ synthetic or real perturbations of images, which is sensible to employ when available for a specific application, we first focus on the stability of the network as a function. Therefore we include the functional variations of the classifier as a general mechanism to achieve robustness against small, stochastic, linear variation of any kind. The new network improves the performance of standard classifiers especially when the dataset size and image resolution are small. For perturbed data, our approach is independent of the application of data augmentation, which could be added on top of our method, as we will demonstrate in an experiment.

5.3 METHOD

5.3.1 *Image Transformations*

Consider an image f as a vector $\mathbf{f}$. For small deviations from the classifier, a Taylor expansion in functional space can be used to cover many small variations by the principle

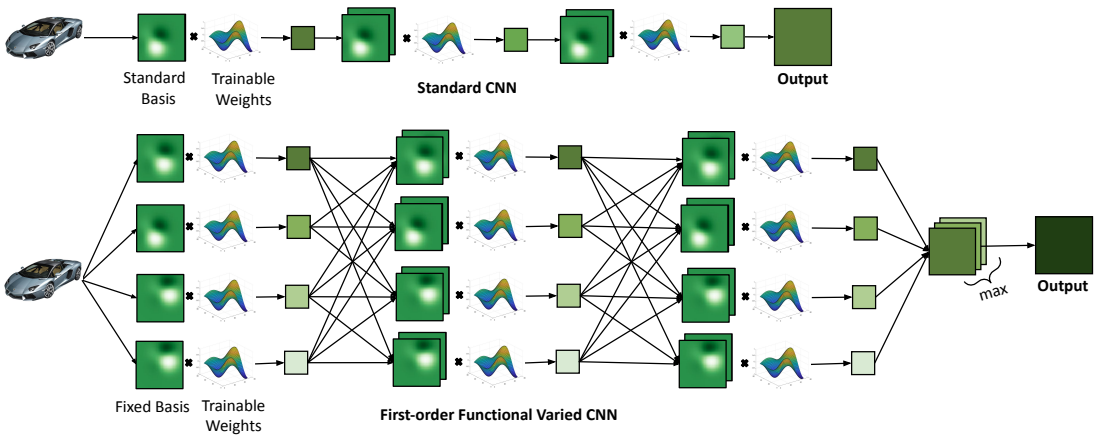


Figure 15: First-order Functional cell architecture. Top: A standard CNN with three convolutional layers. Bottom: The first-order functional variation. By multiplying the fixed basis with trainable weights, a single network is transformed into a network with multiple paths, each path with a slightly different basis. In the end, the maximum is selected. This is aimed to provide relaiability to the local variation of any kind.

of coherence: when data are close they should be classified in the same class. The small linear deviation likely covers many image transformations of practical use. Consider

$$\begin{aligned} T[f](\epsilon) &\approx T[f](0) + \epsilon \left.\frac{\partial T[f]}{\partial \epsilon}\right|_{\epsilon=0} \\ &= \mathbf{f} + \epsilon \mathbf{L}_T \times \mathbf{f} = (\mathbf{I} + \epsilon \mathbf{L}_T) \times \mathbf{f} = \mathbf{T} \times \mathbf{f} \end{aligned} \tag{5.1}$$

where T is a transformation, ϵ is the parameter of the transformation and $\mathbf{T}$ is a linear approximation of T for small values of the parameter and $\mathbf{L}_T$ is a matrix representation of an infinitesimal generator of T.

An image f can be viewed as a real-value function of its coordinates $f : x \rightarrow f(x)$. We focus here on smooth linear displacements τ in the space of coordinates. Equation 5.1 can then be rewritten as follows:

$$T[f(x)](\epsilon) \approx f(x + \epsilon\tau(x)) \tag{5.2}$$

We will refer to such transformations as elastic transformations. We will consider them as a linear approximation of a wide range of complex (camera) transformations. All other perturbations could be derived similarly, up to an additive noise.

5.3.2 *First-order Functional Convolutions*

Let us consider a convolutional layer Φ parameterized by a filter κ. It takes input image f. The output is:

$$\Phi(f, \kappa) = f \star \kappa = \mathbf{K} \times \mathbf{f} \tag{5.3}$$

where $\mathbf{K}$ is a matrix representation of the filter.

$$\Phi(T[f], \kappa) = T[f] \star \kappa = \mathbf{K} \times (\mathbf{T} \times \mathbf{f}) = (\mathbf{K} \times \mathbf{T}) \times \mathbf{f} = \Phi(f, T'[\kappa]) \tag{5.4}$$

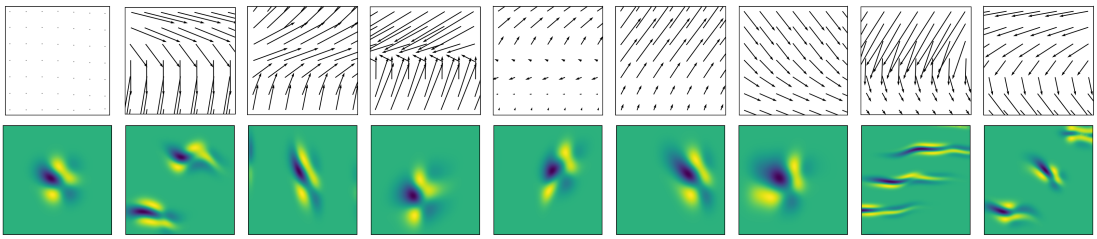


*Figure 16: Row 1: Smooth perturbations for the **local FoF elastic transforms**. Row 2: an original filter and its transformed versions.*

In the most general case, **KT** is a matrix representation of a zero padding, followed by a convolution with a kernel and a cropping afterwards. The size of the kernel $T'[\kappa]$ depends on the nature of the transformation T. If the transformation is global, the kernel can be of a size bigger than the input image. We will consider only the cases when $T'[\kappa]$ is of the same or of a slightly bigger size than the original one.

We propose *First-order Functional convolutions*, shortly FoFConv, as follows:

$$\text{FoFConv} = \max \begin{bmatrix} \beta_0 \Phi(f, \kappa) \\ \beta_1 \Phi(f, T_1[\kappa]) \\ \vdots \\ \beta_n \Phi(f, T_n[\kappa]) \end{bmatrix} \tag{5.5}$$

where β_i are trainable coefficients. We initialize them such that $\beta_0 = 1$, and the rest are zeros. The maximum is calculated per pixel among different transformations of the kernel. At the beginning of training, the operation is thus identical to the original convolution with the same filter. If it is required during training, the other coefficients will activate the corresponding transformations.

5.3.3 *Transformations of a Complete Basis*

In order to apply transformations to filters, we parameterize each filter as a linear combination of basis functions:

$$\kappa = \sum_i w_i \psi_i \tag{5.6}$$

where ψ_i are functions of a complete fixed basis and w_i are trainable parameters. We follow [80] and choose a basis of 2-dimensional Gaussian derivatives.

The transformations when applied to the basis form a transformed basis. Thus, for every transformation from the set, there is a corresponding transform basis. Weights w_i are shared among all bases. We propose elastic transformation here, and test it on global rotation-scaling, local occlusions, local snow, Gaussian noise, and Gaussian blur.

Let us assume that the center of a filter is a point with coordinates $(0, 0)$. For every function from the basis, we first generate a grid of coordinates (x, y). Then we evaluate the value of the function in the coordinates when projected on the pixel grid.

FOF ELASTIC TRANSFORM. In order to transform functions, we add a small displacement to the coordinates. Given a grid of coordinates (x, y), α the elasticity

coefficient and σ be the scaling factor we define the elastically transformed filter as following (See Figure 16), i) we take a 2D-affine transform A_θ and map the coordinates (x, y) to the target coordinates $(x^{\mathrm{T}}, y^{\mathrm{T}})$:

$$\begin{pmatrix} x^{\mathrm{T}} \\ y^{\mathrm{T}} \\ 1 \end{pmatrix} = A_\theta \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{23} & \theta_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{5.7}$$

In order to find the θ-parameters we select three points in the input grid (x, y) and map them to the output $(x^{\mathrm{T}}, y^{\mathrm{T}}) = (x + \delta_x, y + \delta_y)$, where, $\delta_x, \delta_y \sim U[-\alpha, \alpha]$. ii) We get another set of displaced coordinates (x', y') by mapping the coordinates of the kernel as follows:

$$x' = x + \alpha \cdot \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{x^2}{2\sigma^2} \right] \right) \qquad y' = y + \alpha \cdot \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{y^2}{2\sigma^2} \right] \right) \tag{5.8}$$

iii) Finally, we map the target coordinates $(x^{\mathrm{T}}, y^{\mathrm{T}})$ to (x', y') using bi-linear interpolation.

We follow [144] and use a basis of 2D Hermite polynomials with the Gaussian envelope for the transform:

$$\psi_\sigma(x', y') = A \frac{1}{\sigma^2} H_n\left(\frac{x'}{\sigma}\right) H_m\left(\frac{y'}{\sigma}\right) \exp\left[-\frac{x'^2 + y'^2}{2\sigma^2} \right] \tag{5.9}$$

where, A is the normalization constant, H_n is the Hermite polynomial of $n-$th order and σ is the scaling factor. We iterate over n, m-pairs to generate functions.

5.3.4 *First-order Functional Residual Blocks*

In order to transform residual networks, we propose a straightforward generalization of the proposed convolution. The standard residual block can be formulated as follows:

$$\text{ResBlock} = f + G(f, \kappa_1, \kappa_2, \dots) \tag{5.10}$$

The FoF block is then formulated as follows:

$$\text{FoFResBlock} = f + \max \begin{bmatrix} \beta_0 G(f, \kappa_1, \kappa_2, \dots) \\ \beta_1 G(f, T_1[\kappa_1], T_1[\kappa_2], \dots) \\ \vdots \\ \beta_n G(f, T_n[\kappa_1], T_n[\kappa_2], \dots) \end{bmatrix} \tag{5.11}$$

Transformed kernels in the network architecture are shown in the Figure 15.

5.3.5 *Weight Transfer*

To train neural networks successfully, initializing neural networks with Imagenet pre-trained model weights is common. In our case, it is not straight forward to transfer the weights of a standard network to our FoFconv-network because the network is composed

Network	CIFAR-10	STL-10	Stanford Dogs
Baseline	92.53	84.40	48.66
FoF-network	**94.97**	**95.45**	**56.46**

Table 18: ***Classification Accuracy.*** *The FoF-network improves the performance for CIFAR-10 and significantly for fine-grained Stanford dogs* $\approx 8\%$*, and STL-10* $\approx 11\%$*.*

of fixed bases and trainable weights, i.e. multiple parallel networks connected to one another, see Figure 15. Inspired by [143] we assume that in FoFConv there is a sub-network which is identical to the standard network, permitting the transfer of weights from the standard to the FoFConv sub-network. We initialize all weights responsible for inter correlations to zero. Now, the FoF-network until the FoFConv max pooling layer (equation 5.11) consists of several parallel networks disconnected to one another. As the filter sizes of the convolutional layers of FoFConv match with the sizes of the standard network, we initialize them with the corresponding Imagenet weights in the standard network. 1×1 convolutions of the standard network and the FoF-network are identical, and therefore, we copy the weights from the standard to the FoF-network.

5.4 EXPERIMENTS AND RESULTS

DATA. Three datasets, CIFAR-10 [92] containing 10 classes (50,000 training and 10,000 test images of size 32×32), small STL-10 [22] containing 10 categories (5,000 training and 8,000 test images of size 96×96) and a fine-grained subset of ImageNet, the Stanford Dogs datset [86], consisting of 120 categories (12,000 training, 8,580 test images of size 224×224) are used in our experiments.

IMPLEMENTATION DETAILS. We use Resnet-152 as the baseline network, trained with SGD-optimizer and cyclic learning rate scheduler at a rate of 0.05. For all datasets, we experimented with the first-order stochastic variation of multiple Resnet blocks. Restricting the variation to the first block of multiple non-linear layers [67] delivers good results. We begin our training by pre-trained Imagenet initialization and fine tune them on each dataset.

BASELINE PERFORMANCE. Our baseline model achieves 92.53%, 84.40%, 48.66% for CIFAR-10, STL-10 and Stanford dogs test sets respectively, see Table 18. An FoF modification of the baseline significantly improves these results.

5.4.1 *Evaluating the FoF-Network for Reliability*

STANDARDIZING NETWORK ROBUSTNESS. While considering standard networks as the baseline, we standardize the comparison in robustness of different networks by tuning the parameters of perturbations which lead to a fixed drop, i.e., 10% in our experiments, shown in Table 19, within a maximum deviation of 0.44. Hence, the stan-

Network	Roto-Scaling	Elastic	Occlusion	Blur	Noise	Snow
			STL-10			
Baseline	73.98	73.88	73.28	73.86	73.60	73.49
FoF-network	**92.29**	**90.94**	**81.90**	**91.01**	**92.60**	**85.63**
			Stanford Dogs			
Baseline	39.20	38.81	38.75	39.87	38.62	38.51
FoF-network	**48.62**	**47.61**	**46.76**	**47.02**	**43.80**	**47.40**
			CIFAR-10			
Baseline	82.81	82.61	81.90	82.60	81.47	82.81
FoF-network	**86.59**	87.12	80.18	90.03	**89.38**	**85.23**
Elastic Aug.	81.61	**90.79**	**83.62**	90.10	2.51	82.51
FoF + Aug.	88.04	89.68	75.32	**91.61**	85.74	83.41

Table 19: ***Classification Accuracy on Perturbed Images.*** *FoFConv for seen (elastic) and unseen naturally perturbed images. For a standard network, we drop the performance to a standardized level by tuning the perturbations. FoFConv recovers the drop in the performance for all the perturbations on CIFAR-10 except Occlusion. For STL-10 and Stanford dogs, we recover the classification accuracy on all perturbations. Hence, FoFConv significantly enhances the robustness against natural perturbations.*

dardization enables quantitative comparison of the classification performance among the various versions of the network.

TRAINING. We initialize the weights of our FoFConv Resnet-152 with Imagenet weights and fine-tune it. We evaluate our method by adding FoFConv with four stochastic versions of the transform, as shown in Figure 15.

Clean Images

On clean CIFAR-10 test set, FoFConv showed an improvement in the performance of 2.44%, see Table 18. For STL-10, the improvement in the performance with FoF convolutions is significant, leading to an improvement of 11.05%, see Table 18. We contend that the reason behind the significant improvement in the performance for STL-10 dataset is that STL-10 is a small dataset, and our FoF convolutions provide variations in the network, which leads to an improvement in the performance, especially for small datasets. Similarly, for Stanford dogs dataset, our FoFConv showed an improvement of 7.80%.

Seen Naturally Perturbed Images.

Table 19 compares the performance of the standard versus the FoF-network on perturbed images. We observe in the elastic column in Table 19, that for CIFAR-10 and Stanford dogs, our FoF-convolutions recover the drop due to elastic perturbations. On the other hand, on STL-10, our modified convolutions recover the drop even leading to an improvement in the original performance. Hence, the FoF-convolutions are generally robust against natural perturbations.

Unseen Naturally Perturbed Images.

We consider the occlusion, Gaussian blur, Gaussian noise, rotation-scaling and snow perturbations not explicitly covered by an elastic transformation, hence "unseen during training", for evaluating our new model.

On the CIFAR-10, we observe that the FoF-network recovers the drop in the performance for all the perturbed unseen inputs except occlusions, see Table 19. For instance, for Gaussian Blur it recovers the drop of 9.43%, for Gaussian noise 7.91% and for snow occlusions 2.24%. Thus, the FoF convolutions show robustness against unseen natural perturbations, with the one exception for occlusions for the CIFAR-10 dataset. The lack in recovery due to occlusions is ascribed to the size of the CIFAR-10 images, making it difficult for the networks to recover the information lost in occlusion.

On the STL-10, in Table 19 we also test the FoF-network on five different natural unseen perturbations. We observe that the network recovers the induced drop in the performance on unseen perturbations. Dissimilar to CIFAR-10, the network shows a better recovery on unseen occlusions (8.62%), Gaussian blur (17.15%) and Gaussian noise (19.00%) perturbations. Similarly, for Stanford dogs dataset with a large input size, the new network shows recovery in the drop. In contrast with CIFAR-10, on STL-10 the proposed model shows significant recovery in the drop for occlusion perturbations. We conclude that FoF convolutions show a much better general robustness on unseen naturally perturbed images.

The last two rows of Table 19 contrast FoFConv with data augmentation and data augmentation combined with FoFConv. We observe that FoFConv shows better general robustness and when combined with data augmentation enhances further generalization.

5.4.2 *Comparing Computational Resources*

In Table 20 we compare the computational complexity of FoF-network with a standard network. While delivering a much better robustness, our network requires no processing of the data for augmentation, the same disk space as the baseline network, only double the time to train, and GPU memory proportional to the number of transforms.

5.5 CONCLUSION

We formulate a method to enhance the robustness of networks for classification against common perturbations such as occlusion, Gaussian noise, Gaussian blur, and snow. The method transforms the network's weights by four different stochastic instantiations

Network	Training Time, s	GPU Usage, Gb	Network Size, Gb
ResNet-152	0.14	1.9	221.9
FoFConv-ResNet-152	0.32	9.1	221.9

Table 20: ***Comparing resources on an Nvidia 1080Ti.*** *Our network requires the same disk space as the baseline network and only double the time to train per batch.*

of small, stochastic, linear transformation to cover the local neighborhood by Taylor expansion in the functional space of all classifiers. In the evaluation, we note that the new method enhances the network classification accuracy on CIFAR-10, and especially for the fine-grained Stanford dogs and small STL-10 dataset by a considerable margin.

To permit a quantitative comparison in the performance of perturbed images, we first tune the perturbation parameters to the same drop in performance. In this standardized setting, we demonstrate the effectiveness of our method by improving the reliability of the network by considering classification in spite of perturbations in comparison to how baseline networks perform. FoF convolutions generally show robustness. The results show improved network reliability for perturbations *unseen* during training. The improvement in robustness is usually by a large margin, even compared to training with data augmentation by the same transform, *seen* perturbations. FoFConv is independent of data techniques like data augmentation, as application of both results in a further improvement. We conclude that our First order Functional approach improves the reliability of the network especially when the dataset size is small, STL-10 or when it faces perturbations in the data not seen during training, while the costs of implementing it into the network and the extra compute time remain modest.

6

LEARNING TO SUMMARIZE VIDEOS BY CONTRASTING CLIPS

6.1 INTRODUCTION

Video summarization aims at choosing parts of a video that narrate a story as close as possible to the original one. In this day and age, video streaming without personalized recommendations is almost gone. Current recommendations select a fixed preview provided by the distributor on the basis of past preferences. We aim to go one step further and to provide personalized previews. Apart from better video selection for streaming, it also opens possibilities for better video editing, ad creation, and edge-device software development.

Existing approaches for video summarization focus on supervised summarization [55, 101, 128, 175, 178, 180]. With a growing number of videos, supervised summarization may still be somewhat affordable for the distributor. When the number of videos grows exponentially, the supervised model is not sustainable. And, from the standpoint of the user labeling can be applied only in very moderate amounts. In this paper, we aim to maximally exploit unsupervised video summarization while concentrating the supervision on a few personalized labels as an add-on.

Unsupervised video summarization techniques were developed in the pre-deep-learning era, when no large labeled datasets were available [56]. They are still being used these days as labeling the full spectrum of possible videos is no longer possible [108]. Regardless of the method of video analysis, with deep learning or not, the main reasoning for selecting a good summary is left unchanged. The summary must be a compressed representation of the original video while being closer in content to the source than to other videos [52, 115]. Two core questions remain: how to summarize a video while preserving most information in the video, and how to measure distances between two videos on the basis of their content? Formulating the video summarization in this way, in this paper, we propose contrastive learning [19] as the answer to both questions. Contrastive learning was designed to handle compression and metric learning in one go. We propose that neural networks can be trained to optimize a contrastive loss which represents the core requirements of a good summary.

Contrastive learning has been successfully applied in image and video classification [19, 34, 132]. In classification, the contrastive loss is evaluated by comparing descriptive feature vectors of equal size [83]. In video summarization, the comparison is between a vector describing the full video and a vector describing the summary. As a consequence, the vectors will not be equal in size, and hence cannot be compared directly by common contrastive losses. To overcome the inequality in size, the common

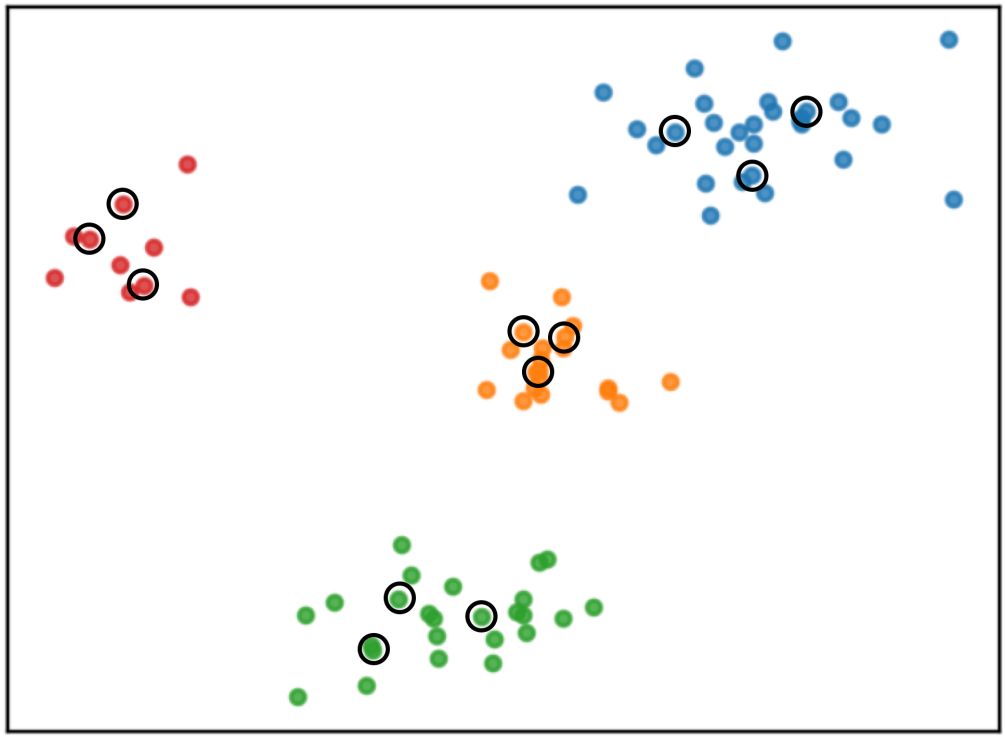


Figure 17: The t-SNE of the features space of example clips and the learned summaries (black circles) for these clips. Finding a good summary in the feature space does not boil down to the centroids of corresponding feature distributions, but rather consists of finding samples that informatively describe the whole input sequence.

approach for comparing vector representations of videos and their summaries is to use their time-averaged representations [4]. In this work, we note that such an approach is invariant to a wide range of transformations and does not account for important moments of high information in the video. When taking an average, the summary is adequate when the video develops slowly like a game of snooker or a sit-com interview but expected to be less adequate when there are short moments of great significance. While various architectural solutions were proposed to improve over quality by average [128], we propose that a combination of a well-chosen loss function and training approach suffices to avoid the unwanted invariances for a wide range of backbones.

In this paper, we focus on developing a simple, flexible, yet efficient recipe for contrastive training of deep-learning-based video summarizers. We start from the principle of maximum information preservation. We demonstrate that the ensuing loss function and maximization process preserves important information while avoiding undesired invariances. Our main contributions are the following:

- From the requirements of video summaries we propose a method for contrastive learning of video summaries with no need for labeling.
- We propose implementations of the main building blocks which are required to convert any video-analysis network into a summarizer.
- We demonstrate the advantage of contrastive video summarizers on popular benchmarks for a set of backbone architectures over their original training methods. We

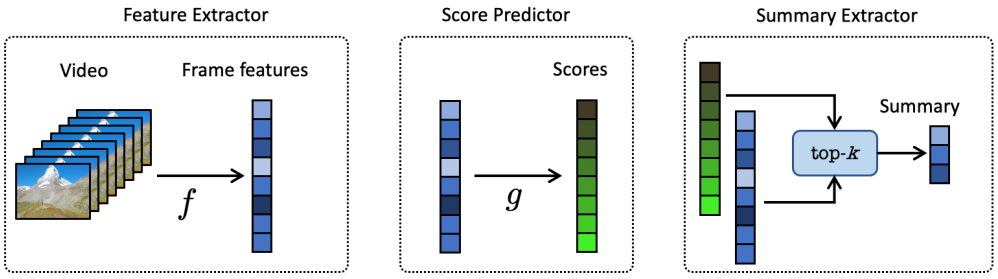


Figure 18: The main building blocks of our framework for video summarization. Feature extractor f transforms an arbitrary-length sequence of frames into a sequence of features. They are then used to predict a set of scores by using a function g. The summary extractor block uses both sequences to extract a set of k frames with the highest scores.

also demonstrate how it can be used for video highlight selection with a slight modification.

6.2 RELATED WORK

SUPERVISED METHODS With the rise of deep learning, a wide range of papers has considered video summarization as a regression problem. In such a paradigm a neural network is used to take frames of the video and predict their importance scores so that the top-scored frames form the summary. In [175] the authors use Recurrent Neural Networks (RNNs) to combine the temporal information from the video with the content of each frame to successfully predict frames' scores. Alternatively, in [128] and [45] convolutional and attention-based architectures were proposed to improve the quality of predictions. To effectively combine information about videos from multiple scales, hierarchical models were proposed [178, 179]. By using hierarchical RNNs, models benefit from considering the video as a whole, as a set of short clips and as a sequence of individual frames at the same time. It allows to create summaries of less-contract granularity than before. While these methods demonstrated the great success of deep neural networks for video summarization, manually labeled annotations are required for their training. It makes it impossible to scale such methods to long videos, movies and streams of videos that are constantly being uploaded on the major video services. For these reasons, we focus on methods that do not rely on human-annotated labels.

UNSUPERVISED METHODS Early-day methods for video summarization relied on heuristics designed by a human. The heuristics were designed to satisfy the main requirements for video summaries such as representativeness and diversity, justified in [35, 118]. In [35, 93, 115] the authors clustered frames and use the centroids to form a summary. The authors of [28, 111] formulate video summarization as a sparse dictionary selection problem. Later, in the deep-learning era, video summarization was approached from the perspective of adversarial training [68, 109] or in the reinforcement learning paradigm [180]. We draw inspiration from the pre-deep-learning era methods. By starting from the reasoning of video summarization, we demonstrate that we can satisfy the main requirements by formulating it as a contrastive learning problem that we can easily solve.

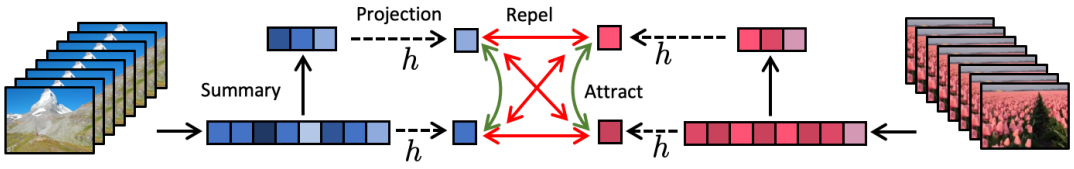


Figure 19: An illustration of the proposed contrastive training pipeline. Given two videos, the features are first calculated for them and then a summary is extracted. Both summaries and original videos are projected by a neural network h to a hidden space afterward. The whole pipeline is trained to attract the projections of summaries to the projections of the original videos and to repel them from other summaries and videos.

CONTRASTIVE LEARNING Contrastive learning is an approach for performing self-supervised pretraining of a model by using a pre-text task. The model learns to attract representations that are meant to be close, and are thus called positive, and repel them from negative representations which are meant to be distant enough to distinguish between different objects [19, 65, 114, 150]. Various methods have been proposed for learning image-level [16, 20] and spatio-temporal models [4, 48, 58]. The current application of contrastive learning methods for video summarization is rather limited due to special architectural solutions dictated by the domain. In this work, we demonstrate an approach for contrastive learning for video summarization that does not rely on any specific backbones and allows one to use any model and framework of their choice.

VIDEO HIGHLIGHT SELECTION Another popular approach for creating a compressed visual representation of videos is video highlight selection. While the summary has a fixed length, the highlights are not bounded in length but have a lower bound for the importance scores. Various methods have been proposed for solving this problem both from the supervised perspective [3, 75, 106, 145, 169, 170, 172], as well as in the unsupervised manner [4]. In this paper, we demonstrate that with a slight modification of our video summarization framework, we can outperform modern video highlight selection models without significant transformations of the original pipeline.

6.3 METHOD

6.3.1 *Summary Requirements*

Video summarization is a very subjective task, as a manually labeled summary is biased towards the personal preferences of the annotator, assessor [139]. However, it is possible to select several properties of a good summary that we would consider as summary requirements. They also give us hints on how to build an efficient model for video summarization.

REPRESENTATIVENESS The composed summary should deliver the same message as the original video. As the summary is a compressed representation of the source, the

loss of the original information is inevitable. However, we require a good summary to contain all the information necessary to distinguish between the original video and all other videos [109]. With no loss of generality, we can assume that each video contains a finite set of sub-videos each of which tells a separate narrative. Thus, the desired summary is a combination of sub-videos that is as close as possible to all of them at the same time, as well as distant enough from all sub-videos of other original videos. We suggest to learn summaries by selecting a set of sub-videos which we call clips which once they are projected to some hidden space minimize a variant of the triplet loss. As we want to develop a model for unsupervised summarization, it leads us to the framework of contrastive learning [19].

SPARSITY The original videos may come from various sources: be it video news, a video blog, several-hours-long online streams or a TV show. For all cases, the desired summary would be just several seconds long as it is the average amount of time a user can spend before deciding whether to watch it or to skip it. It leads us to the requirement of the sparsity of the resulting summary. While for short videos the desired summary may be around 15% of the length [64], this ratio may drop significantly for longer videos. Thus, our model should be capable of choosing the very top segments of the video with a significant distinction from the rest. The problem of ranking items and selecting the top of them cannot be overestimated as it has significant limitations in the realm of deep learning especially when it comes to a very sparse output [59, 122, 168]. In our approach, we should not directly rely on any heuristics for performing such an operation and should seek an as accurate as possible algorithmic implementation of it.

DIVERSITY Another important property of a video summary is the diversity among its frames. We may assume that for some videos and for some datasets it is possible to create a summary that will contain a lot of very similar frames and clips. Although it is possible for a user to understand what the video is about just by taking a look at one frame, it is still desired to have a summary with a higher diversity of visual information. If we consider two models which satisfy the above-mentioned requirements, we want to select the model which selects diverse summaries over uniform summaries. It shows us that the function which we will use to measure the distance between videos should not be invariant to the spread of the frames. In other words, it should take into account not only a single frame and the general content of videos but also the variations inside them.

6.3.2 *Contrastive Summarization*

A wide range of trainable video summarizers can be decomposed into the following three blocks: features extractor f, score predictor g and summary extractor (see Figure 18). From the summary requirement, we generated several requirements for the video summarization pipeline. And none of them are related to the feature extractor or the score predictor. Thus, we assume that these two blocks are the free parameters of our framework. Once a feature extractor and a score predictor are chosen, we train their parameters by performing a variant of contrastive learning.

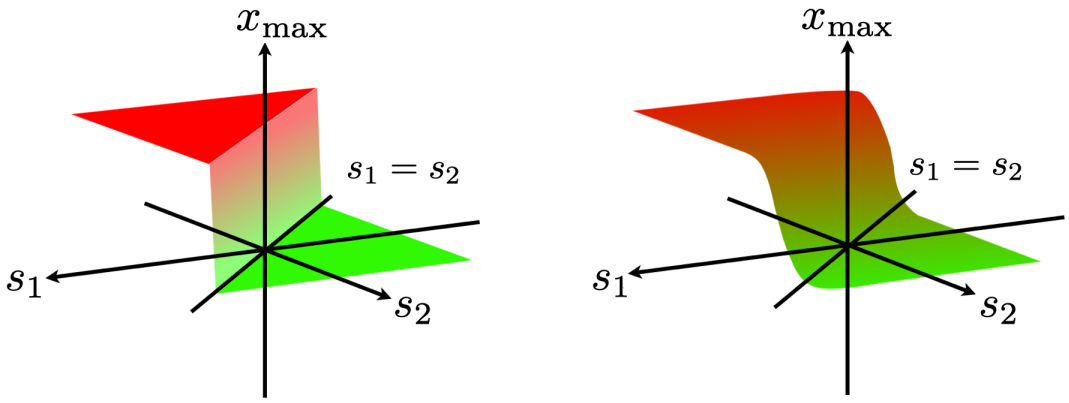


Figure 20: Illustration of frame selection based on the scores s_1 and s_2. Left: the original step function $x_{max} = x_1$ if $s_1 > s_2$ and $x_{max} = x_2$ otherwise. Right: a relaxed version with a smooth replacement for the step function.

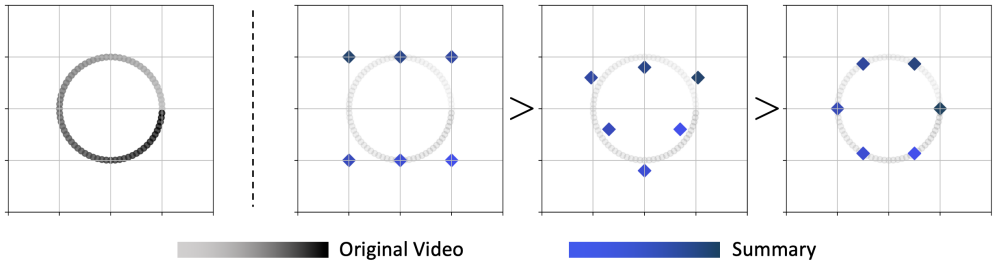


Figure 21: Left: a sample video as a set of short clips forms a circle in some hidden space. Right: Three sets of summaries that yield the same distance function when they are time-averaged. However, these summaries have different distances from the whole video if the distance function is given as in Equation 6.4 which leads to a uniform distribution of clips.

During training, we consider two videos (see Figure 19). Each of the videos is processed with the feature extractor and the score predictor functions. After that, for each of the videos a summary is generated. We choose the parameters of the networks f and g by training them to generate video and summary embeddings s.t. the summary attracted to its source video is repelled from any other videos and summaries. It is done by minimizing the following loss [19]:

$$\mathcal{L} = \sum_{z,z_+} -\log \frac{\exp(\text{dist}(z, z_+)/\tau)}{\sum_{z_-} \exp(\text{dist}(z, z_-)/\tau)} \tag{6.1}$$

where z, z_-, z_+ are embeddings for the anchor video, its negative and positive pairs. We calculate this loss by iterating over all possible sets of such videos and their summaries. The parameter τ is the smoothing factor of the loss functions. It is a hyperparameter of our approach. In order to perform the training of such a pipeline successfully, we need to define the distance function $\text{dist}(\cdot, \cdot)$

6.3.3 *Clip-Contrastive Distance Function*

Let us consider two sequences of D-dimensional vectors represented as matrices:

$$\boldsymbol{X} = \{X_{ij}\}_{i,j=1}^{D,t} \qquad \boldsymbol{Y} = \{Y_{ij}\}_{i,j=1}^{D,T} \tag{6.2}$$

where t and T and the lengths of the sequences. In our case, these matrices are clip features for the summary and the original video and thus we assume $t < T$. A common approach for calculating the distance between two sequences [4] is to compare their features averaged it time: $\bar{\boldsymbol{x}} = \sum_j X_{ij}/t'$ and $\bar{\boldsymbol{y}} = \sum_k Y_{ik}/T$. It can be done for example by calculating the scalar product of these vectors:

$$\text{dist}(\boldsymbol{X}, \boldsymbol{Y}) = \left\langle \sum_j X_{ij}/t', \sum_k Y_{ik}/T \right\rangle = \frac{1}{T \cdot t} \sum_{ijk} X_{ij} Y_{ik} \tag{6.3}$$

The main drawback of such a distance function is that it compares the videos just by calculating the discrepancy between the average value of their clips. Thus, it does not take into account the diversity of clips within the video.

We suggest the following procedure for calculating the distance between two videos. Let us consider a parameter n which we call the length of a sub-video. We consider each of the videos as a distribution of all possible sub-videos of the lengths n. And then we calculate the mathematical expectation of the distance calculated between a sub-video from the first video and a sub-video from the second video as follows:

$$\text{dist}_n(\boldsymbol{X}, \boldsymbol{Y}) = E_{\boldsymbol{x}' \sim q_n(\boldsymbol{X})} E_{\boldsymbol{y}' \sim q_n(\boldsymbol{Y})} < \boldsymbol{x}', \boldsymbol{y}' > = E_{\boldsymbol{x}' \sim q_n(\boldsymbol{X})} E_{\boldsymbol{y}' \sim q_n(\boldsymbol{Y})} \sum_{j=1}^{n \times D} x'_j y'_j \tag{6.4}$$

where $q(\boldsymbol{X})$ is distribution of all possible sub-videos from $\boldsymbol{X}$ which have n clips inside. Such a distance function will degrade to Equation 6.3 if we consider $n = 1$. For all other cases, it will take into account not only the difference between the mean values of the video embeddings but also their distributions.

6.3.4 *Differentiable Summary Selection*

In order to perform end-to-end training of the proposed pipeline, we must make all of the steps differentiable. The feature extractor, the score predictor and the projector are parametrized with neural networks and are thus differentiable. A more sophisticated part of the pipeline is the module, which selects clips with the highest scores, the top-k frame selector. Given a set of frames $\{x_1, x_2, \ldots, x_N\}$ and a set of corresponding scores $\{s_1, s_2, \ldots s_N, \}$ the top-k selector outputs a set of k frames which have the highest scores.

Ranking a set of frames according to their scores is equivalent to choosing the frame with the highest score, then removing it from the set and repeating the operation again and again. Choosing the maximum, or the top-1 frame can be formalized as follows:

$$x_{\max} = \sum_j x_j \mathbb{1}[s_j > s_i, \ \forall i \neq j] \tag{6.5}$$

Model	Source	Supervised	TVSum			SumMe		
			F_1	τ	ρ	F_1	τ	ρ
vsLSTM	[175]	✓	54.2	-	-	37.6	-	-
dppLSTM	[175]	✓	54.7	-	-	38.6	-	-
VASNet	[45]	✗	61.4	0.16	0.17	49.7	0.16	0.17
MSVA	[53]	✓	62.8	**0.19**	**0.21**	54.4	0.20	**0.23**
CSUM vsLSTM,	Ours	✗	59.0	-	-	41.0	-	-
CSUM dppLSTM	Ours	✗	60.5	-	-	44.2	-	-
CSUM VASNet	Ours	✗	62.7	0.17	0.17	52.1	0.16	0.17
CSUM MSVA	Ours	✗	**63.9**	**0.19**	0.20	**58.2**	**0.22**	**0.23**

*Table 21: Experimental results on the TVSum and SumMe dataset. The reported metrics are F1-score, Spearman and Kendall correlation coefficients. We compare various backbone models with default training regimes and the same model trained with our contrastive approach. The best results are in **bold**.*

where $\mathbb{1}[\dots]$ is the indicator function. The value of $x_{\max}$ changes with jumps from x_1 to x_2 and so on when the corresponding scores dominate the other scores. If we fix all the scores but just one, and then vary it from $-\infty$ to ∞, the value of $x_{\max}$ will change just once and this change will be a jump (see Figure 20). Thus, the gradient of $x_{\max}$ with respect to the varying score will remain 0 everywhere except for the point of the jump, where the gradient is undefined. Therefore, using this gradient value for back-propagation is not possible.

By following [59] we use a relaxation of this step function. Equation 6.5 can be approximated as follows

$$x_{\max} \approx \sum_j x_j \frac{\exp(\alpha s_j)}{\sum_i \exp(\alpha s_i)} = \sum_j x_j \cdot \mathrm{SoftMax}(\alpha s)_j \tag{6.6}$$

The parameter α can be interpreted as the inverse of the width of the transition region and if $\alpha \to \infty$, then $\mathrm{SoftMax}(\alpha s) \to \mathbb{1}[\dots]$.

To rank the set of frames, we step-by-step select the maximum element by using Equation 6.6 and then subtract from the maximum score a large number, so that the same frame will not be selected on the next step. In order to minimize the computational complexity of such an operation, we follow the approach proposed in [123] and compare pairs of frames. Thus, the processing time growth logarithmically with the number of frames.

6.4 EXPERIMENTS

In this section, we evaluate the quality of video summarizations learned with the proposed method. We conduct experiments on several datasets and with several backbone models to demonstrate that the proposed method generalizes well for various video summarization

setups. Next, we present qualitative examples of extracted video summarizations. Finally, we provide an ablation study on the hyper-parameters of our method.

DATASETS We conduct experiments with 3 datasets: TVSum [139], SumMe [64] and YouTube Highlights [145] datasets. The TVSum dataset consists of 50 videos from 10 categories from [137]. In TVSum each video has frame-level importance scores annotated by 20 users. Importance scores range from 1 to 5, where 5 denotes the highest importance. The SumMe dataset includes 25 short videos of various events such as cooking or sports. Each video is attributed with frame-level importance scores. The YouTube Highlights dataset contains videos divided into 6 categories with around 1000 videos of various lengths available for each domain. For each video, there is a ground truth highlight in a form of a sequence of consecutive frames summarizing the content of the video in the best way.

EVALUATION To quantitatively evaluate the quality of extracted summaries we employ 5-fold cross-validation with an average F1-score across the splits. The cross-validation splits are the same as in [53]. The average F1-score over videos in the dataset is reported. As noted in [120] F1-score has certain limitations. We thus also adopt Spearman's correlation (ρ) and Kendall correlation (τ) coefficients between the summaries predicted by the models and ground truth summaries. For the YouTube Highlights dataset we perform a summary evaluation as a task of highlight detection in time. We thus employ mean average precision (mAP) as a known detection metric. The final mAP score is computed over [0.5:0.05:0.95] IoU thresholds.

BACKBONE MODELS To demonstrate that our method generalizes for various setups, we conduct experiments with several known backbone models: Video-LSTM and bi-directional Video-LSTM [175], LSTM with attention [45], Multi-Source Visual Attention model [53] and multi-modal Transformers [106]. For our experiments, we leave the backbone architecture unchanged and only modify the training pipeline of the models.

6.4.1 *Summarization performance*

We start with summarization experiments on the TVSum and SumMe datasets. Here we evaluate the proposed contrastive learning approach with various feature extraction backbone models. We use the proposed differentiable top-k summary extractor during training. During the inference stage, we simply select N frames with the highest predicted scores. The results are reported in Table 21.

As can be seen from Table 21, using the proposed approach results in significant improvement for all of the baseline models. Notably, video LSTM (v-LSTM) enjoys a 4.8% improvement in F1-score on the TVSum dataset, given that the proposed contrastive training does not use labels compared to its default supervised training regime. Also, our approach outperforms the previous best-performing unsupervised method VASNet [45] by 1.1% on TVSum and by up to 2.4% on SumMe, when the performance is measured with the F1-score. In terms of Spearman and Kendall correlation coefficients, our unsupervised method performs on par with the supervised models.

Top-k method	F_1-score
Sinkhorn [168]	63.1 ± 0.4
Perturbed [29]	62.9 ± 0.5
Successive Halving [123]	63.4 ± 0.3

Table 22: F_1-score of the MSVA model [53] with different top-k selection mechanism on the TVSum dataset. A Successive Halving algorithm performs the best.

This experiment demonstrates that our contrastive learning approach generalizes well for various backbone architectures and for various datasets. Without using any labels, we are able either to match or to outperform existing supervised methods.

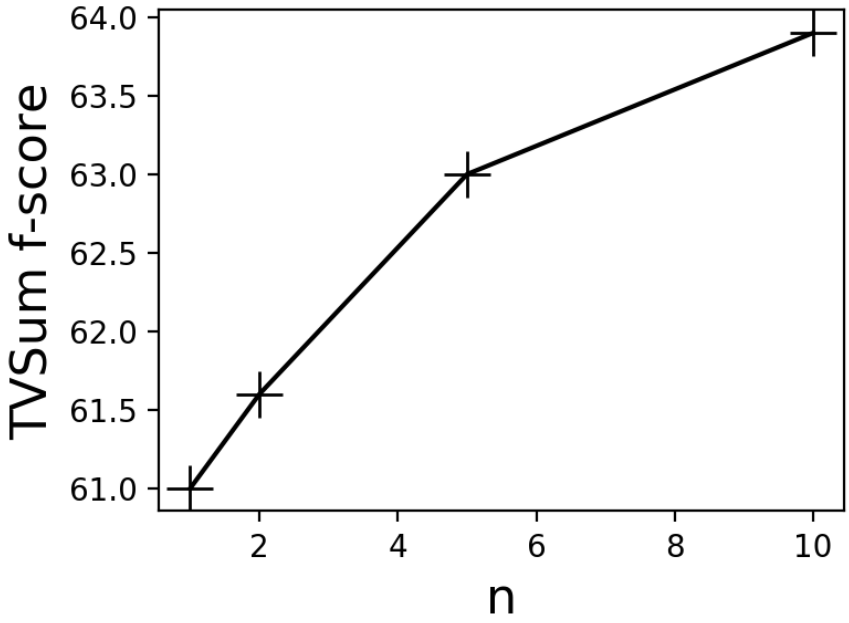


Figure 22: F_1-score of the MSVA backbone [53] on the TVSum dataset for various values of n used in Equation 6.4.

6.4.2 *Ablation studies*

In this section, we ablate the top-k selection algorithm and the window parameter n in Equation 6.4. We also investigate if our top-k selector can robustly distribute importance scores regardless of the number of input frames.

For top-k differentiable selection ablation, we compare Sinkhorn [168], Perturbed top-k [29] and Successive Halving [123] algorithms. As can be seen from Table 22 the choice of top-k influences the final performance with Successive Halving delivering the best results for the MSVA backbone [53] on the TVSum dataset. We thus chose to use Successive Halving in all of the experiments.

We next ablate the window parameter n in Equation 6.4. Intuitively, n is responsible for the granularity of the resulting video summarization, where lower values of n result

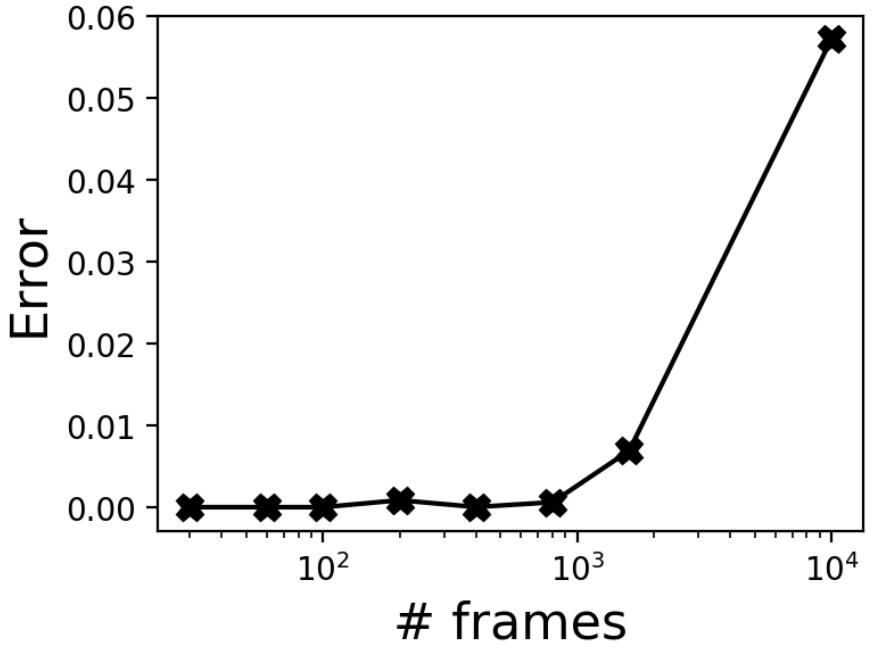


Figure 23: Normalized L2-error between the feature maps of top-10 frames selected with the proposed method and ground truth feature maps. On the x-axis is the total number of frames to extract the summary from.

in higher granularity. As can be seen in Figure 22, the summarization quality benefits from higher values of n. That indicates that good video summaries should not be of the highest granularity. Thus, we use $n = 10$ in all of the experiments.

Finally, we investigate if our differentiable top-k selector is robust with respect to the number of input frames. In Figure 23 we report how the normalized $L2$-error between the feature maps of top-10 frames selected with our method and ground truth feature maps depends on the number of input frames. The results suggest that even when the number of input frames is huge, the error does not exceed 0.06. It indicates that the used differentiable top-k selector is robust with respect to the number of input frames.

6.4.3 *Highlight detection*

We view highlight detection as a special case of the summarization task, i.e. the highlight is a top-1 summary extraction coupled with surrounding context frames. Practically, to detect a highlight from a full-video, we prepossess summarization scores with Gaussian smoothing to enforce temporal continuity. After that, we extract a top-score frame with the surrounding frames with high enough scores to serve as one highlight.

We conduct experiments in both supervised and unsupervised scenarios. For supervised highlight detection, we first pre-train the models with the proposed contrastive approach for 20 epochs and then fine-tune it for 50 epochs using the loss described in [106]. Evaluating fine-tuned representation is a standard procedure in contrastive learning [19, 20]. For unsupervised highlight detection, we directly use the scores after 20 epochs of contrastive training.

Model	Sup.	Audio	Dog	Gym.	Park.	Skat.	Ski.	Surf.	Avg
LSVM [145]	✓	✗	60.0	41.0	61.0	62.0	36.0	61.0	53.6
LIM-S [169]	✓	✗	57.9	41.7	67.0	57.8	48.6	65.1	56.4
SL-Module [170]	✓	✗	**70.8**	53.2	77.2	**72.5**	66.1	76.2	69.3
CHD [4]	✗	✗	60.6	71.1	74.2	49.8	68.2	68.5	65.4
CSUM UMT, Ours	✗	✗	60.9	70.2	73.8	63.2	70.0	71.4	68.3
CSUM UMT, Ours	✓	✗	64.8	**73.6**	**79.9**	70.5	**71.5**	**80.0**	**73.3**
MINI-Net [75]	✓	✓	58.2	61.7	70.2	72.2	58.7	65.1	64.4
TCG [172]	✓	✓	55.4	62.7	70.9	69.1	60.1	59.8	63.0
Joint-VA [3]	✓	✓	64.5	71.9	80.8	62.0	73.2	78.3	71.8
UMT [106]	✓	✓	65.9	**75.2**	**81.6**	71.8	72.3	82.7	74.9
CSUM UMT, Ours	✓	✓	**66.1**	75.1	**81.6**	**71.9**	**73.0**	**82.8**	**75.1**

*Table 23: Experimental results on the YouTube Highlights benchmark. The reported metric is mAP in percentages. We compare both the methods which use the audio information from the video and the methods which rely on the visual features only. We additionally compare supervised (Sup) and unsupervised methods. The best performing models are in **bold**.*

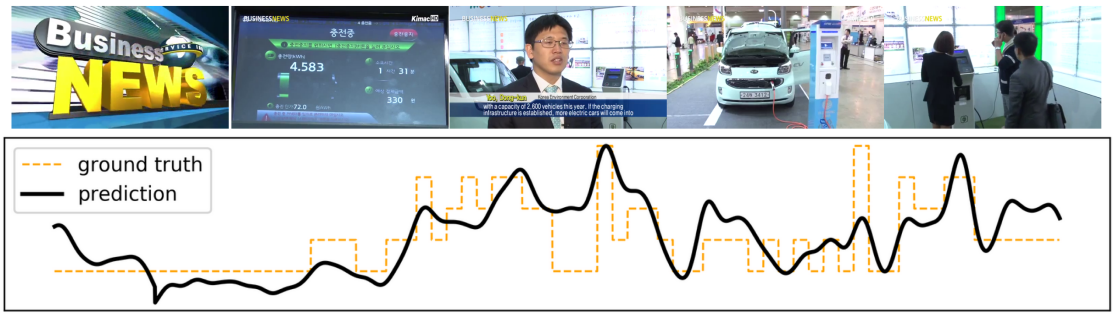


Figure 24: Top row: A visual example of the video summary extracted with our method. Bottom row: human-annotated ground truth importance scores and the importance scores trained with our contrastive learning method.

We present the results for the cases when audio features are available and when they are not. In the supervised scenario, as can be seen from Table 23, our method (CSUM UMT) outperforms the competitive approaches or performs on par. In particular, for the no-audio case our method delivers more than 5% improvement relative to the best-performing non-contrastive method [170]. With the audio information included, our method slightly outperforms the baseline UMT model, when the only modification being made is the contrastive pre-training used. In the unsupervised case, our method delivers more than 4% increase in mAP score with respect to the previous best performing unsupervised method from [4]. Also, the mAP score of our unsupervised model is only 1% behind [170], which fully relies on training with labels.

We conclude that our contrastive approach is very competitive with existing methods, even when comparing our unsupervised with previous supervised results.

6.4.4 *Qualitative evaluation*

In Figure 24 we present an example of the video summarization of a sequence from the SumMe dataset trained with our contrastive framework and differentiable top-k. We can see that the predicted importance score can detect the regions of both low and high significance.

6.5 DISCUSSION

In this work, we propose a novel approach for unsupervised video summarization. We start by formulating the requirements for a good video summary: representatives, sparsity, and diversity. We observe that the contrastive learning framework naturally includes representatives and diversity. For sparsity, we propose a differentiable top-k selector based on predicted frame-level scores, where the importance is inherently distributed only among top-k input frames. This allows stepping away from comparing mean feature vectors, which may result in sub-optimal solution space, during the contrastive learning stage. Our approach does not rely on a specific kind of backbone; we experimentally show that it generalizes well for various architectures and summarization scenarios.

7

CONCLUSION

In this thesis we considered the problem of learning from limited data by using the power of symmetries. We started with the following question:

Can we build neural networks which are aware of the principle geometric symmetries without learning them?

In Chapter 2 we developed a theory of scale-equivariant convolutional neural networks. We demonstrated the idea of filter reparametrization for the rescaling of convolutional kernels. We presented an algorithm for fast filter rescaling without using tensor interpolations. We demonstrated how the most common building blocks of neural networks such as convolution, batch normalization and nonlinearities can be extended to scale equivariance. We also demonstrated how one can transform a scale-equivariant neural network into a scale-invariant function. We presented a recipe for how to make an arbitrary CNN-based model scale equivariant, by simply drop-in replacing its building blocks with the proposed ones.

In Chapter 3 we considered Siamese trackers, for which principle transformations such as scale-variations are considered as significant disturbing factors. We proposed a recipe for how to extend any CNN-based Siamese visual object tracker to scale equivariance. We also proposed a method for model initialization to reuse weights from an already pre-trained model.

In our experiments in Chapter 2 and Chapter 3 we demonstrated that the proposed simple modifications of CNN-based models significantly outperform their conventional counterparts. By making the networks aware of scale variations by design we allowed them to benefit when less training data is available.

In order to improve the results even further, we considered the discrepancy between the symmetries we assume in the data and the symmetries the data actually contains. Thus, we posed the next question:

How to match the scale symmetries of the observed world with the scale symmetries of mathematical structures?

In Chapter 4 from the first principles, we derived general constraints, under which scale-equivariant neural networks are indeed equivariant under scale transformations of input. We found that for integer scale factors, which transform the original image into another one that lies on exactly the same grid, the exact solution exists. In this case, there is no discrepancy between symmetries. For all other cases, there is no exact solution. We demonstrated that it is possible to learn an approximate solution by minimizing the

discrepancy between the theoretical and real output of the layer when it's calculated on real data. In our experiments, we demonstrate that by simply replacing one set of filters with another set we improve the model. After comparing several models side-by-side we concluded that the lower the discrepancy between the real symmetries and the mathematical symmetries are the better model performs. And the best-performing model is the proposed model.

In Chapter 5 we focused on model robustness. Scale-equivariant convolutional neural networks are robust with respect to scale variations in the input. We sought a method to extend this property to small variations of the input, be it blur, partial occlusion, view-point variations or fog.

How to make models robust to subtle variations that go beyond affine transformations?

With this question in mind, we introduced a new class of convolutional neural networks. At the core of our method is the idea that for any transformation of the input, there exists a coupled transformation of the filter to produce the same result after convolution. We demonstrated that learned data augmentation can be implemented in every layer of a convolutional neural network by extending the filter bank with a set of filter transformations. We proposed an algorithm for an efficient implementation of these new models. We demonstrated that by using our algorithm, one can improve a wide range of convolutional models just by drop-in replacement of the standard convolutional layers with the proposed ones.

In Chapter 6 we considered unsupervised video summarization. The task of video summarization is select a part of the video which delivers the same message as the original video. How to summarize? What to highlight? What information seems more important than the other? As we sought to solve it in an unsupervised manner, we couldn't derive the answers from some previously-seen data. Thus, we posed the following question.

How to make models learn from unlabeled data by assuming symmetries?

To answer this question, we considered every video as a set of short clips. We proposed a method for learning how clips relate to each other by mapping them to a hidden space where closer clips organize symmetric structures. Thus, a summary is a subset of clips that after some symmetry transformations can reconstruct the rest. We introduced an algorithm for fast implementation of this approach. In our experiments, we demonstrated that a wide range of models significantly improve their results if they are trained by using our method.

SUMMARY

In this thesis, Symmetry-Based Learning from Limited Data, we introduced new approaches for training better machine learning models for computer vision tasks in the absence of large labeled datasets. Our approach is based on equipping neural networks with the notion of symmetry for the sake of better learning real-world constraints without observing all their realizations in the training data. We started with explicit mathematical structures such as the scale group, then we considered a wider group of non-affine transformations. And finally, we demonstrated that models can learn structures themselves by assuming symmetry in the input data.

Our contributions are the following:

- We introduced **Scale-Equivariant Steerable Networks** (Chapter 2), a class of convolutional neural networks that are equipped with an extra notion of scale variations. We presented a theory and an effective implementation for these networks. In our experiments, the new class of models outperformed all other models without introducing new trainable parameters.

- We introduced **Scale-Equivariant Siamese Trackers** (Chapter 3), a simple extension of well-known Siamese trackers, which uses the power of scale equivariance to improve visual object tracking in scenarios when scale variations are significant. We proposed a theory and an effective recipe for how to convert any convolutional Siamese tracker into a Scale-Equivariant counterpart. In our experiments, we demonstrated that after extending the tracker, it performs way better while retaining real-time performance.

- We introduced **Accurate Discrete Scale Convolutions** (Chapter 4), a class of scale-equivariant convolutional neural networks that are optimized for discrete signals. We started from the first principles and derived the constraints under which scale-equivariant convolutional neural networks demonstrate the lowest discrepancy from the theoretical properties. We proposed a method for building models which satisfy these constraints as close as possible. In our experiments, we demonstrate that the models with the lowest discrepancy demonstrate the best results in image classification and visual object tracking.

- We introduced **First-order Functional Convolutions** (Chapter 5), a modification of standard convolutions, which allows one for training more robust counterparts of the original models by simply replacing the convolutional layers. We presented a theory and an effective implementation for robust modifications. In our experiments, the proposed approach outperformed the conventional counterparts.

- We introduced **Contrastive Video Summarizers** (Chapter 6), an algorithm for video summarization that does not rely on annotations and is flexible with respect to the backbone choices. We present a theory and demonstrate that with just

several building blocks any video summarizer can be used in our framework. In our experiments, the proposed framework improves the results for several popular models on multiple benchmarks.

SAMENVATTING

In dit proefschrift, *Symmetry-Based Learning from Limited Data*, hebben we nieuwe benaderingen geïntroduceerd voor het trainen van betere machine learning modellen voor beeldherkenningstaken bij afwezigheid van grote gelabelde datasets. Onze aanpak is gebaseerd op het uitrusten van neurale netwerken met de notie van symmetrie om de beperkingen uit de echte wereld beter te leren kennen zonder al hun realisaties in de trainingsdata te observeren. We zijn begonnen met expliciete wiskundige structuren zoals de schaalgroep, vervolgens hebben we een bredere groep niet-affiene transformaties overwogen. En tot slot hebben we aangetoond dat modellen zelf structuren kunnen leren door uit te gaan van symmetrie in de invoerdata.

Onze bijdragen zijn als volgt:

- We introduceerden **Scale-Equivariant Steerable Networks** (Hoofdstuk 2), een klasse van convolutionele neurale netwerken die zijn uitgerust met een extra notie van schaalvariaties. We presenteerden een theorie en een effectieve implementatie voor deze netwerken. In onze experimenten presteerde de nieuwe klasse modellen beter dan alle andere modellen zonder nieuwe trainbare parameters te introduceren.

- We introduceerden **Scale-Equivariant Siamese Trackers** (Hoofdstuk 3), een eenvoudige uitbreiding van bekende Siamese trackers, die de kracht van schaalequivariantie gebruiken om het visueel volgen van objecten te verbeteren in scenario's waarin schaalvariaties aanzienlijk zijn. We hebben een theorie en een effectief recept voorgesteld voor het omzetten van elke convolutionele Siamese tracker in een schaal-equivariante tegenhanger. In onze experimenten hebben we aangetoond dat na uitbreiding van de tracker deze veel beter presteert met behoud van real-time prestaties

- We introduceerden **Accurate Discrete Scale Convolutions** (Hoofdstuk 4), een klasse van schaal-equivariante convolutionele neurale netwerken die zijn geoptimaliseerd voor discrete signalen. We zijn uitgegaan van de eerste principes en hebben de beperkingen afgeleid waaronder schaal-equivariante convolutionele neurale netwerken de laagste discrepantie vertonen met de theoretische eigenschappen. We hebben een methode voorgesteld om modellen te bouwen die zo goed mogelijk aan deze beperkingen voldoen. In onze experimenten laten we zien dat de modellen met de laagste discrepantie de beste resultaten laten zien op het gebied van beeldclassificatie en het volgen van visuele objecten.

- We introduceerden **First-order Functional Convolutions** (Hoofdstuk 5), een aanpassing van standaard convoluties, die het mogelijk maakt om robuustere tegenhangers van de originele modellen te trainen door simpelweg de convolutionele lagen te vervangen. We presenteerden een theorie en een effectieve implementatie voor robuuste aanpassingen. In onze experimenten presteerde de voorgestelde aanpak beter dan de conventionele tegenhangers.

- We introduceerden **Contrastive Video Summarizers** (Hoofdstuk 6), een algoritme voor videosamenvatting dat niet afhankelijk is van annotaties en flexibel is met betrekking tot de backbone-keuzes. We presenteren een theorie en demonstreren dat met slechts enkele componenten elke videosamenvatting in ons raamwerk kan worden gebruikt. In onze experimenten verbetert het voorgestelde raamwerk de resultaten voor verschillende populaire modellen op meerdere benchmarks.

ACKNOWLEDGMENTS

First of all, I would like to thank Arnold Smeulders, my PhD advisor, who once told me that I am accepted to this PhD position, which started one of the most interesting journeys of my life. Arnold has provided me with freedom to develop and to pursue my research interests, and provided encouragement when required. I am happy that Arnold always asked concrete questions when my thoughts went too abstract and provided me with abstract ideas when my thinking became too focused on details. Arnold has pushed me into the beautiful world of science and provided me with everything I could require: paper recommendations, ideas, networking, advises on how to develop various skills, funny stories, and jokes. Thank you for helping me with my defense ceremony in these not-so-easy times!

I would like to thank Virginie Mes. It seems like Virginie knows everything and can help with anything when it comes to the official part of the PhD program. I am happy for the friendly atmosphere and all the help that Virginie provided me with.

I am grateful to my PhD defense committee members: prof. dr. Marcel Worring, prof. dr. Max Welling, dr. Jörn-Henrik Jacobsen, dr. Efstratios Gavves, prof. dr. Tony Lindeberg and my copromotor prof. dr. Cees G.M. Snoek. It is a pleasure to have you all in my defense committee.

I am grateful to my friend and a great scientist, Artem Moskalev. Artem came from the same country, the same city, and the same university, but demonstrated an absolutely different way of thinking. It helped me to perceive the problems we worked on from a different perspective. I believe, that such a unique combination of great knowledge, curiosity and an outstanding sense of humour makes discussions with Artem so important. I have a feeling that our conversations shed light on many things that were and still are considered as the *terra incognita* of artificial intelligence and science in general.

I would like to thank Sadaf Gulshad, a great collaborator and a lab mate. Sadaf is one of the brightest and the most optimistic persons I have ever met. I am extremely happy that some randomly generated numbers made us neighbours both in the east and in the west of Amsterdam.

I feel fortunate to have met and worked with many bright people during this journey. The team of UvA-Bosch Delta Lab: Herke van Hoof, Zeynep Akata, Emile Hoogeboom, Jorn Peters, Elise van der Pol, Sadaf Gulshad, Thomas Andy Keller, Victor Garcia Satorras, Artem Moskalev, Wenling Shang, Sindy Lowe and Mevlana Gemici. All members of our great IvI team, who inspired me with their talks, conversations, achievements, papers, smiles: Daniel Worrall, Erik Bekkers, Patrick Forré, Taco Cohen, Gabriele Cesa, Pim de Haan, Evgenii Egorov, Maurice Weiler, Kirill Gavrilyuk, Sarah Ibrahimi, Tom Runia, Nanne van Noord, Gjorgji Strezoski, Anna Sepliarskaia, Mert Kilickaya, Maximilian Ilse, Thomas Kipf, Rianne van den Berg, Karen Ullrich, Anil Baslamisli, Thomas Mensink, Pascal Mettes, Dennis Koelma, Berkay Kicanaoglu, Jakub Tomczak, and many others. The great team of Bosch Center for Artificial Intelligence: Konrad Groh, Anna Khoreva, Volker Fischer, Edgar Schönfeld, and many others. Amazing scientists from other uni-

versities in The Netherlands: Anna Kuzina, David Romero, Robert-Jan Bruintjes and others. The students I have supervised: Cees Kaandorp, Lucas Meijer, Dario Shehni Abbaszadeh, Dave Meijdam, Jonne Goedhart, Daan Ferdinandusse, Gongze Cao, Michał Szmaja, Jan Jetze Beitler. The great scientists from Russia who now live all around the globe: Andrei Atanov, Alexandra Volokhova, Arsenii Ashukha, Dmitry Vetrov, Kirill Neklyudov and many many others.

I would like to express my gratitude to my parents for helping me to become the person I am. My life wouldn't be so bright and interesting without the love, support and different tasks my mom and dad gave me.

The very important role in this story is played by a talented scientist, who found this PhD position and recommended me to apply for it. I would like to thank my sister, Vera Sosnovik, for her constant help, recommendations, inspiration and lots of valuable conversations.

BIBLIOGRAPHY

[1] H. L. Anderson, R. Bajcsy, and M. Mintz. Adaptive image segmentation. 1988.

[2] A. Atanov, A. Volokhova, A. Ashukha, I. Sosnovik, and D. Vetrov. Semi-conditional normalizing flows for semi-supervised learning. *arXiv preprint arXiv:1905.00505*, 2019.

[3] T. Badamdorj, M. Rochan, Y. Wang, and L. Cheng. Joint visual and audio learning for video highlight detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8127–8137, 2021.

[4] T. Badamdorj, M. Rochan, Y. Wang, and L. Cheng. Contrastive learning for unsupervised video highlight detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14042–14052, 2022.

[5] B. Bamieh. Discovering transforms: A tutorial on circulant matrices, circular convolution, and the discrete fourier transform. *arXiv preprint arXiv:1805.05533*, 2018.

[6] E. Barnard and D. Casasent. Invariance and neural nets. *IEEE Transactions on neural networks*, 2(5):498–508, 1991.

[7] J. J. Beitler, I. Sosnovik, and A. Smeulders. Pie: Pseudo-invertible encoder. *arXiv preprint arXiv:2111.00619*, 2021.

[8] E. J. Bekkers. B-spline cnns on lie groups. *arXiv preprint arXiv:1909.12057*, 2019.

[9] P. Benz, C. Zhang, A. Karjauv, and I. S. Kweon. Revisiting batch normalization for improving corruption robustness. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 494–503, 2021.

[10] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016.

[11] G. Bhat, M. Danelljan, L. V. Gool, and R. Timofte. Learning discriminative model prediction for tracking. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[12] M. Born and L. Infeld. Foundations of the new field theory. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 144(852):425–451, 1934.

[13] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[14] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

[15] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

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

[17] C.-Y. Chang, A. A. Maciejewski, and V. Balakrishnan. Fast eigenspace decomposition of correlated images. *IEEE Transactions on Image Processing*, 9(11):1937–1949, 2000.

[18] J. Chang and J. W. Fisher. Analysis of orientation and scale in smoothly varying textures. In *2009 IEEE 12th International Conference on Computer Vision*, pages 881–888. IEEE, 2009.

[19] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[20] X. Chen and K. He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[21] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji. Siamese box adaptive network for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6668–6677, 2020.

[22] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[23] T. Cohen, M. Geiger, J. Köhler, and M. Welling. Convolutional networks for spherical signals. *arXiv preprint arXiv:1709.04893*, 2017.

[24] T. Cohen, M. Geiger, and M. Weiler. A general theory of equivariant cnns on homogeneous spaces. *arXiv preprint arXiv:1811.02017*, 2018.

[25] T. Cohen and M. Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

[26] T. S. Cohen, M. Weiler, B. Kicanaoglu, and M. Welling. Gauge equivariant convolutional networks and the icosahedral cnn. *arXiv preprint arXiv:1902.04615*, 2019.

[27] T. S. Cohen and M. Welling. Steerable cnns. *arXiv preprint arXiv:1612.08498*, 2016.

[28] Y. Cong, J. Yuan, and J. Luo. Towards scalable summarization of consumer videos via sparse dictionary selection. *IEEE Transactions on Multimedia*, 14(1):66–75, 2011.

[29] J.-B. Cordonnier, A. Mahendran, A. Dosovitskiy, D. Weissenborn, J. Uszkoreit, and T. Unterthiner. Differentiable patch selection for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2351–2360, 2021.

[30] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[31] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Atom: Accurate tracking by overlap maximization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[32] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017.

[33] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European conference on computer vision*, pages 472–488. Springer, 2016.

[34] I. Dave, R. Gupta, M. N. Rizve, and M. Shah. Tclr: Temporal contrastive learning for video representation. *Computer Vision and Image Understanding*, 219:103406, 2022.

[35] S. E. F. De Avila, A. P. B. Lopes, A. da Luz Jr, and A. de Albuquerque Araújo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern recognition letters*, 32(1):56–68, 2011.

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[37] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[38] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[39] N. Diaconu and D. Worrall. Learning to convolve: A generalized weight-tying approach. In *International Conference on Machine Learning*, pages 1586–1595. PMLR, 2019.

[40] D. Diderot and J. L. R. d'Alembert. *Encyclopédie, ou, Dictionnaire raisonné des sciences, des arts et des métiers*, volume 5. Pergamon Press, 1776.

[41] S. Dodge and L. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *2017 26th international conference on computer communication and networks (ICCCN)*, pages 1–7. IEEE, 2017.

[42] X. Dong and J. Shen. Triplet loss in siamese network for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 459–474, 2018.

[43] Y. Dong, Q.-A. Fu, X. Yang, T. Pang, H. Su, Z. Xiao, and J. Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 321–331, 2020.

[44] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–68, 2018.

[45] J. Fajtl, H. S. Sokeh, V. Argyriou, D. Monekosso, and P. Remagnino. Summarizing videos with attention. In *Asian Conference on Computer Vision*, pages 39–54. Springer, 2018.

[46] A. Fawzi and P. Frossard. Manitest: Are classifiers really invariant? *arXiv preprint arXiv:1507.06535*, 2015.

[47] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[48] B. Fernando, H. Bilen, E. Gavves, and S. Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017.

[49] G. B. Folland. *A course in abstract harmonic analysis*. Chapman and Hall/CRC, 2016.

[50] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):891–906, 1991.

[51] R. Geirhos, D. H. Janssen, H. H. Schütt, J. Rauber, M. Bethge, and F. A. Wichmann. Comparing deep neural networks against humans: object recognition when the signal gets weaker. *arXiv preprint arXiv:1706.06969*, 2017.

[52] H. Gharbi, S. Bahroun, and E. Zagrouba. Key frame extraction for video summarization using local description and repeatability graph clustering. *Signal, Image and Video Processing*, 13(3):507–515, 2019.

[53] J. A. Ghauri, S. Hakimov, and R. Ewerth. Supervised video summarization via multiple feature sets with parallel attention. 2021.

[54] R. Ghosh and A. K. Gupta. Scale steerable filters for locally scale-invariant convolutional neural networks. *arXiv preprint arXiv:1906.03861*, 2019.

[55] B. Gong, W.-L. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. *Advances in neural information processing systems*, 27, 2014.

[56] Y. Gong and X. Liu. Video summarization using singular value decomposition. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 174–180. IEEE, 2000.

[57] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[58] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun. Unsupervised learning of spatiotemporally coherent metrics. In *Proceedings of the IEEE international conference on computer vision*, pages 4086–4093, 2015.

[59] K. Goyal, G. Neubig, C. Dyer, and T. Berg-Kirkpatrick. A continuous relaxation of beam search for end-to-end training of neural sequence models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[60] S. Gulshad and A. Smeulders. Natural perturbed training for general robustness of neural network classifiers. *arXiv preprint arXiv:2103.11372*, 2021.

[61] S. Gulshad, I. Sosnovik, and A. Smeulders. Built-in elastic transformations for improved robustness. *arXiv preprint arXiv:2107.09391*, 2021.

[62] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang. Learning dynamic siamese network for visual object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1763–1771, 2017.

[63] D. K. Gupta, D. Arya, and E. Gavves. Rotation equivariant siamese networks for tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12362–12371, 2021.

[64] M. Gygli, H. Grabner, H. Riemenschneider, and L. V. Gool. Creating summaries from user videos. In *European conference on computer vision*, pages 505–520. Springer, 2014.

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

[66] D. Han, J. Kim, and J. Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5927–5935, 2017.

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

[68] X. He, Y. Hua, T. Song, Z. Zhang, Z. Xue, R. Ma, N. Robertson, and H. Guan. Unsupervised video summarization with attentive conditional generative adversarial networks. In *Proceedings of the 27th ACM International Conference on multimedia*, pages 2296–2304, 2019.

[69] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

[70] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[71] J. F. Henriques, P. Martins, R. F. Caseiro, and J. Batista. Fast training of pose detectors in the fourier domain. *Advances in neural information processing systems*, 27:3050–3058, 2014.

[72] J. F. Henriques and A. Vedaldi. Warped convolutions: Efficient invariance to spatial transformations. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1461–1469. JMLR. org, 2017.

[73] G. F. Hinton. A parallel computation that assigns canonical object-based frames of reference. In *Proceedings of the 7th international joint conference on Artificial intelligence-Volume 2*, pages 683–685, 1981.

[74] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*, pages 286–297. Springer, 1990.

[75] F.-T. Hong, X. Huang, W.-H. Li, and W.-S. Zheng. Mini-net: Multiple instance ranking network for video highlight detection. In *European Conference on Computer Vision*, pages 345–360. Springer, 2020.

[76] E. Hoogeboom, J. W. Peters, T. S. Cohen, and M. Welling. Hexaconv. *arXiv preprint arXiv:1803.02108*, 2018.

[77] L. Huang, X. Zhao, and K. Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2019.

[78] T. Iijima. Basic theory of pattern observation. *Technical Group on Automata and Automatic Control*, pages 3–32, 1959.

[79] J.-H. Jacobsen, B. De Brabandere, and A. W. Smeulders. Dynamic steerable blocks in deep residual networks. *arXiv preprint arXiv:1706.00598*, 2017.

[80] J.-H. Jacobsen, J. Van Gemert, Z. Lou, and A. W. Smeulders. Structured receptive fields in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2610–2619, 2016.

[81] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[82] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[83] L. Jing and Y. Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020.

[84] A. Kanazawa, A. Sharma, and D. Jacobs. Locally scale-invariant convolutional neural networks. *arXiv preprint arXiv:1412.5104*, 2014.

[85] C. Kanbak, S.-M. Moosavi-Dezfooli, and P. Frossard. Geometric robustness of deep networks: analysis and improvement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4441–4449, 2018.

[86] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

[87] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[88] J. J. Koenderink. The structure of images. *Biological cybernetics*, 50(5):363–370, 1984.

[89] R. Kondor. N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. *arXiv preprint arXiv:1803.01588*, 2018.

[90] R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *arXiv preprint arXiv:1802.03690*, 2018.

[91] M. Kristan, A. Leonardis, J. Matas, and M. F. et. The visual object tracking vot2017 challenge results. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 1949–1972, 2017.

[92] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[93] S. K. Kuanar, R. Panda, and A. S. Chowdhury. Video key frame extraction through dynamic delaunay clustering with a structural constraint. *Journal of Visual Communication and Image Representation*, 24(7):1212–1227, 2013.

[94] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[95] L. Lang and M. Weiler. A wigner-eckart theorem for group equivariant convolution kernels. *arXiv preprint arXiv:2010.10952*, 2020.

[96] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys. Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 289–297, 2016.

[97] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[98] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. *arXiv preprint arXiv:1812.11703*, 2018.

[99] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4282–4291, 2019.

[100] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, 2018.

[101] Y. Li, L. Wang, T. Yang, and B. Gong. How local is the local diversity? reinforcing sequential determinantal point processes with dynamic ground sets for supervised video summarization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 151–167, 2018.

[102] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[103] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[104] T. Lindeberg. Scale-space for discrete signals. *IEEE transactions on pattern analysis and machine intelligence*, 12(3):234–254, 1990.

[105] T. Lindeberg. *Scale-space theory in computer vision*, volume 256. Springer Science & Business Media, 2013.

[106] Y. Liu, S. Li, Y. Wu, C. W. Chen, Y. Shan, and X. Qie. Umt: Unified multi-modal transformers for joint video moment retrieval and highlight detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[107] N. Loehr. *Advanced linear algebra*. CRC Press, 2014.

[108] I. Mademlis, A. Tefas, and I. Pitas. Regularized svd-based video frame saliency for unsupervised activity video summarization. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2691–2695. IEEE, 2018.

[109] B. Mahasseni, M. Lam, and S. Todorovic. Unsupervised video summarization with adversarial lstm networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 202–211, 2017.

[110] D. Marcos, B. Kellenberger, S. Lobry, and D. Tuia. Scale equivariance in cnns with vector fields. *arXiv preprint arXiv:1807.11783*, 2018.

[111] S. Mei, G. Guan, Z. Wang, M. He, X.-S. Hua, and D. D. Feng. L 2, 0 constrained sparse dictionary selection for video summarization. In *2014 IEEE international conference on multimedia and expo (ICME)*, pages 1–6. IEEE, 2014.

[112] A. Moskalev, I. Sosnovik, and A. Smeulders. Relational prior for multi-object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1081–1085, 2021.

[113] A. Moskalev, I. Sosnovik, and A. Smeulders. Two is a crowd: tracking relations in videos. *arXiv preprint arXiv:2108.05331*, 2021.

[114] A. Moskalev, I. Sosnovik, F. Volker, and A. Smeulders. Contrasting quadratic assignments for set-based representation learning. In *European Conference on Computer Vision*, 2022.

[115] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, 6(2):219–232, 2006.

[116] M. Murugan, K. Subrahmanyam, and S. I. Park. So (2)-equivariance in neural networks using tensor nonlinearity.

[117] H. Naderi, L. Goli, and S. Kasaei. Scale equivariant cnns with scale steerable filters. In *2020 International Conference on Machine Vision and Image Processing (MVIP)*, pages 1–5. IEEE, 2020.

[118] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang. Automatic video summarization by graph modeling. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 104–109. IEEE, 2003.

[119] J. Ormond. Fathers of the deep learning revolution receive ACM A.M. Turing Award, 2018.

[120] M. Otani, Y. Nakashima, E. Rahtu, and J. Heikkila. Rethinking the evaluation of video summaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7596–7604, 2019.

[121] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[122] F. Petersen, H. Kuehne, C. Borgelt, and O. Deussen. Differentiable top-k classification learning. In *International Conference on Machine Learning*, pages 17656–17668. PMLR, 2022.

[123] M. Pietruszka, Ł. Borchmann, and F. Graliński. Successive halving top-k operator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 15869–15870, 2021.

[124] S. Qiao, L.-C. Chen, and A. Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *arXiv preprint arXiv:2006.02334*, 2020.

[125] 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 (NIPS)*, 2015.

[126] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[127] A. Robey, H. Hassani, and G. J. Pappas. Model-based robust deep learning. *arXiv preprint arXiv:2005.10247*, 2020.

[128] M. Rochan, L. Ye, and Y. Wang. Video summarization using fully convolutional sequence networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 347–363, 2018.

[129] D. W. Romero, E. J. Bekkers, J. M. Tomczak, and M. Hoogendoorn. Attentive group equivariant convolutional networks. *arXiv preprint arXiv:2002.03830*, 2020.

[130] D. W. Romero and J.-B. Cordonnier. Group equivariant stand-alone self-attention for vision. *arXiv preprint arXiv:2010.00977*, 2020.

[131] D. W. Romero and M. Hoogendoorn. Co-attentive equivariant neural networks: Focusing equivariance on transformations co-occurring in data. *arXiv preprint arXiv:1911.07849*, 2019.

[132] S. Roy and A. Etemad. Spatiotemporal contrastive learning of facial expressions in videos. In *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–8. IEEE, 2021.

[133] E. Rusak, L. Schott, R. S. Zimmermann, J. Bitterwolf, O. Bringmann, M. Bethge, and W. Brendel. A simple way to make neural networks robust against diverse image corruptions. In *European Conference on Computer Vision*, pages 53–69. Springer, 2020.

[134] S. Schneider, E. Rusak, L. Eck, O. Bringmann, W. Brendel, and M. Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33, 2020.

[135] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *International Journal of Computer Vision*, 56(3):151–177, 2004.

[136] P. Y. Simard, Y. A. LeCun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition—tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pages 239–274. Springer, 1998.

[137] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, MIR '06, page 321–330. Association for Computing Machinery, 2006.

[138] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1442–1468, 2013.

[139] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes. Tvsum: Summarizing web videos using titles. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5179–5187, 2015.

[140] I. Sosnovik, A. Moskalev, and A. Smeulders. Disco: accurate discrete scale convolutions. *arXiv preprint arXiv:2106.02733*, 2021.

[141] I. Sosnovik, A. Moskalev, and A. Smeulders. How to transform kernels for scale-convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1092–1097, 2021.

[142] I. Sosnovik, A. Moskalev, and A. W. Smeulders. Scale equivariance improves siamese tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2765–2774, January 2021.

[143] I. Sosnovik, A. Moskalev, and A. W. Smeulders. Scale equivariance improves siamese tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2765–2774, 2021.

[144] I. Sosnovik, M. Szmaja, and A. Smeulders. Scale-equivariant steerable networks. *arXiv preprint arXiv:1910.11093*, 2019.

[145] M. Sun, A. Farhadi, and S. Seitz. Ranking domain-specific highlights by analyzing edited videos. In *ECCV*, pages 787–802. Springer International Publishing, 2014.

[146] K. S. Tai, P. Bailis, and G. Valiant. Equivariant transformer networks. *arXiv preprint arXiv:1901.11399*, 2019.

[147] Z. Tang, Y. Gao, Y. Zhu, Z. Zhang, M. Li, and D. Metaxas. Selfnorm and crossnorm for out-of-distribution robustness. *arXiv preprint arXiv:2102.02811*, 2021.

[148] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1420–1429, 2016.

[149] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

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

[151] M. Ulicny, V. A. Krylov, and R. Dahyot. Harmonic networks with limited training samples. *arXiv preprint arXiv:1905.00135*, 2019.

[152] B. S. Veeling, J. Linmans, J. Winkens, T. Cohen, and M. Welling. Rotation equivariant cnns for digital pathology. In *International Conference on Medical image computing and computer-assisted intervention*, pages 210–218. Springer, 2018.

[153] X. Wang, S. Zhang, Z. Yu, L. Feng, and W. Zhang. Scale-equalizing pyramid convolution for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13359–13368, 2020.

[154] M. Weiler and G. Cesa. General e (2)-equivariant steerable cnns. In *Advances in Neural Information Processing Systems*, pages 14334–14345, 2019.

[155] M. Weiler, M. Geiger, M. Welling, W. Boomsma, and T. Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Advances in Neural Information Processing Systems*, pages 10381–10392, 2018.

[156] M. Weiler, F. A. Hamprecht, and M. Storath. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 849–858, 2018.

[157] H. Weyl. *Philosophy of mathematics and natural science*. Princeton University Press, 2021.

[158] A. P. Witkin. Scale-space filtering. In *Readings in Computer Vision*, pages 329–332. Elsevier, 1987.

[159] E. Wong and J. Z. Kolter. Learning perturbation sets for robust machine learning. *arXiv preprint arXiv:2007.08450*, 2020.

[160] D. Worrall and G. Brostow. Cubenet: Equivariance to 3d rotation and translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 567–584, 2018.

[161] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5028–5037, 2017.

[162] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[163] D. E. Worrall and M. Welling. Deep scale-spaces: Equivariance over scale. *arXiv preprint arXiv:1905.11697*, 2019.

[164] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[165] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[166] Y. Wu, S. Ying, and L. Zheng. Size-to-depth: a new perspective for single image depth estimation. *arXiv preprint arXiv:1801.04461*, 2018.

[167] Y. Xian, B. Schiele, and Z. Akata. Zero-shot learning-the good, the bad and the ugly. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4582–4591, 2017.

[168] Y. Xie, H. Dai, M. Chen, B. Dai, T. Zhao, H. Zha, W. Wei, and T. Pfister. Differentiable top-k with optimal transport. *Advances in Neural Information Processing Systems*, 33:20520–20531, 2020.

[169] B. Xiong, Y. Kalantidis, D. Ghadiyaram, and K. Grauman. Less is more: Learning highlight detection from video duration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1258–1267, 2019.

[170] M. Xu, H. Wang, B. Ni, R. Zhu, Z. Sun, and C. Wang. Cross-category video highlight detection via set-based learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7970–7979, 2021.

[171] Y. Xu, T. Xiao, J. Zhang, K. Yang, and Z. Zhang. Scale-invariant convolutional neural networks. *arXiv preprint arXiv:1411.6369*, 2014.

[172] Q. Ye, X. Shen, Y. Gao, Z. Wang, Q. Bi, P. Li, and G. Yang. Temporal cue guided video highlight detection with low-rank audio-visual fusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7950–7959, 2021.

[173] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[174] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016.

[175] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman. Video summarization with long short-term memory. In *European conference on computer vision*, pages 766–782. Springer, 2016.

[176] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu. Structured siamese network for real-time visual tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 351–366, 2018.

[177] Z. Zhang and H. Peng. Deeper and wider siamese networks for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4591–4600, 2019.

[178] B. Zhao, X. Li, and X. Lu. Hierarchical recurrent neural network for video summarization. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 863–871, 2017.

[179] B. Zhao, X. Li, and X. Lu. Hsa-rnn: Hierarchical structure-adaptive rnn for video summarization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7405–7414, 2018.

[180] K. Zhou, Y. Qiao, and T. Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[181] W. Zhu, Q. Qiu, R. Calderbank, G. Sapiro, and X. Cheng. Scale-equivariant neural networks with decomposed convolutional filters. *arXiv preprint arXiv:1909.11193*, 2019.