Informatics and Health 1 (2024) 82–92

Contents lists available at ScienceDirect

Informatics and Health

journal homepage: www.keaipublishing.com/en/journals/informatics-and-health/

Full length article

Towards real-time interest point detection and description for mobile and robotic devices

Patrick Rowsome [a], Muhammad Adil Raja [b,*], R. Muhammad Atif Azad [c]

[a] Protex AI, Ireland
[b] Regulated Software Research Center (RSRC), Dundalk Institute of Technology, Country Louth, Ireland
[c] School of Computing and Digital Technology, Birmingham City University, Birmingham, UK

ARTICLE INFO

Keywords:
Interest point detection
Computer vision
Convolutional neural networks
Deep learning
Reliable detector and descriptor

ABSTRACT

Convolutional Neural Networks (CNNs) have been successfully adopted by state-of-the-art feature point detection and description networks for the past number of years. The focus of these systems has been predominately on the accuracy of the system, rather than on its efficiency or ability to be implemented in real-time on embedded robotic devices. This paper demonstrates how techniques, developed for other CNN use cases, can be integrated into interest point detection and description systems to compress their network size and reduce the computational complexity; this reduces the barrier to their uptake in computationally challenged environments. This paper documents the integration of these techniques into the popular Reliable Detector and Descriptor (R2D2) network. Along with the integration details, a comprehensive Key Performance Indicator (KPI) framework is developed to test all aspects of the networks. As a result, this paper presents a lightweight variant of the R2D2 network that significantly reduces parameters and computational complexity while crucially maintaining an acceptable level of accuracy. Consequently, this new compressed network is more appropriate for use in real world systems and advances the efforts to implement such CNN based system for mobile devices.

1. Introduction

Interest point detection and description is an area of Computer Vision (CV) that has been researched for decades [1–3]. It is a critical component of many CV applications in robotics such as camera calibration, Simultaneous Localisation and Mapping (SLAM) (see Fig. 1), and obstacle avoidance. Given the nature of these applications, there is a requirement for systems to run robustly and in real-time. For a system to run robustly it must provide not only invariance to many factors like motion, illumination changes, and image blur but also accurately locate features with associated distinctive descriptors. However, the obstacle to using the system as a real-time robotic agent is the limited availability of the computing and power resources. This is because such systems are battery-powered. In addition, interest point detection and description must run alongside other higher-level components that use the outputs of the feature detection and description, thus further straining the resources.

Interest point detection was first researched in the 80s [4,5]; those works demonstrated strong results that still stand up to current state-of-the-art systems. Since then attempts have been made to improve their accuracy [6]; importantly, also, several attempts have been made to improve their efficiency [4,5] in embedded devices [7].

Another seminal paper in interest point detection and description came in 2001 that introduced Scale Invariant Feature Transform (SIFT) [8] that outperformed earlier work with its accurate localisation and invariant descriptors demonstrating robust performance. While the accuracy of SIFT was state-of-the-art, the efficiency was poor because it required computationally expensive multi-scale convolutional operations. The following years saw variants of SIFT that tried to address this inefficiency, namely, SURF [9] and FREAK [10]. Given these trends, it can be seen that a pattern in the research direction initially concerns itself with accuracy improvements and then laterally efficient implementations where trade-offs are made between accuracy and efficiency. However, a common thread across this line of work was that they all needed a priori knowledge of the tasks at hand to *handcraft* parts of their systems.

More recently, CNN based methods have surpassed the accuracy performance of SIFT and other handcrafted methods. These CNNs can leverage learning from large datasets to create interest point detectors and descriptors that are robust to many variations. Crucially, however, a similar effort to optimize these CNNs has not gained traction yet and typically a CNN is not suitable for implementing interest point detection

* Corresponding author.
E-mail address: adil.raja@dkit.ie (M.A. Raja).

https://doi.org/10.1016/j.infoh.2024.06.002
Received 15 March 2024; Received in revised form 12 June 2024; Accepted 15 June 2024
Available online 2 July 2024

Nomenclature

AE	Autoencoder
CV	Computer Vision
CNN	Convolutional Neural Network
FPS	Frame Per Second
GPU	Graphics Processing Unit
SLAM	Simultaneous Localisation and Mapping
KPI	Key Performance Indicator
mAP	mean Average Precision
MLE	Mean Localisation Error
NNmAP	Nearest Neighbour mean Average Precision
ROC	Receiver Operating Characteristic
L	Lightweight
HP	High-Performance
MAC	Multiply-Accumulate Operation
R2D2	Reliable Detector and Descriptor
SIFT	Scale Invariant Feature Transform
SGD	Stochastic Gradient Descent
VGA	Video Graphics Array

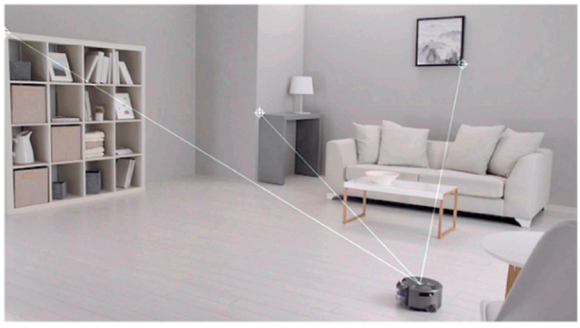

Fig. 1. Representation of interest point detection and description methods used in the Dyson 360 hoover system. The hoover uses a visual SLAM system to navigate and learn the topology of rooms it wishes to clean. Credit: Dyson: https://www.kurzweilai.net/a-robot-vacuum-cleaner-with-360-vision.

and description on mobile devices. For instance [11], which is considered one of the more efficient solutions, reports a frame-rate of 70fps using Video Graphics Array (VGA) input on an Nvidia Titan X Graphics Processing Unit (GPU). Such a device is not a viable option for many robotic applications because of the price and power requirements.

In contrast, other areas of CV (apart from keypoint localisation and description) that use CNN based methods have seen significant attention in terms of optimisation. For example, a state-of-the-art object detection network [12] was optimised using the MobileNetV2 [13] network that significantly improved efficiency while retaining the accuracy. However, this improvement is not domain agnostic and does not work for interest point detection and feature description in embedded devices.

The aim of this paper is to integrate these efficient architectural components developed in [13] into an efficient but accurate interest point detection and description network, while also providing metrics to analyse the accuracy and efficiency trade off for various network configurations. This effort is undertaken to develop a better understanding of how interest point detection and description can be optimised and to discover the viability of such networks on embedded devices. To that end, the results show that the proposed enhancements significantly reduce computational complexity of the state-of-the-art in this problem domain while maintaining acceptable accuracy.

The remaining paper is organised as follows. Section 2 details related work in interest point detection and description methods and similar efforts to compress CNN based systems. Section 2.1 describes the proposed methods used in this paper for compressing the CNNs. Section 4 details the experiments carried out. Section 5 provides a discussion on the results. Finally, Section 6 concludes with the main findings from this work.

2. Related work

In literature varying terminology has been used when referring to image keypoints (similar points of interest across images). For the purpose of clarity, this paper adopts similar jargon to that used in [14], where they have identified the process as two distinct categories: detection and description. *Detection* captures the process where salient regions are localised in an image. These regions are estimated to indicate the location of an important feature, usually a real world 3D landmark. These salient image regions are ill-defined in a strict mathematical sense but in practice they are image regions that are distinctive in their local context. An important attribute of these features is the accuracy of the location for the feature over time. *Description* is a step where a numerical or binary representation for the appearance of the feature is calculated. The aim of the description process is to estimate a representation that will distinguish a feature from other features within an image while also allowing for matching of descriptors from the same 3D landmark over time (Fig. 3). A simple example of such a descriptor would be a vector of the direction and magnitude of gradients in the locality of a feature. An example of image features and their associated matches are shown in Fig. 2.

Image keypoints have a broad application to CV systems and are an integral part of SLAM, stereo reconstruction, image retrieval, visual odometry, camera calibration, panoramic image generation, and tracking, to name a few. Image features are a critical part of many CV applications. They appear in three broad application areas in modern CV. The first, is to support image retrieval and description. These systems are mainly concerned with learning a compact statistical representation of an image from image keypoints with the goal of searching a database to find similar images. Image keypoints can also be designed to detect semantic keypoints for applications in video inspection. An example of this would be a vision defect detection system that is designed to identify cracks in a product. Finally, image features can be used to track points temporally to estimate structure in the scene and the motion of the camera. This final area is the concern of this paper. The goal in this case is to accurately locate and reliably detect these features over time, while also trying to develop a system which is appropriate for real-time implementation.

Accurately tracking and matching features over time requires the detection and description process to be invariant to environmental, photometric, and geometric changes. Environmental factors may include scene lighting changes or moving objects in the scene, whereas

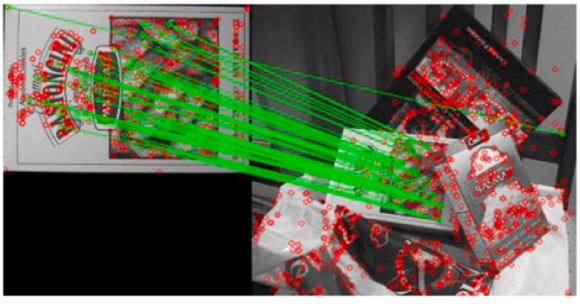

Fig. 2. Image Feature example. SIFT features estimated for two views. Features are indicated by the red circle in both images. The matches made by comparing descriptors from both views are indicated by the green lines. Credit: https://docs.opencv.org/3.4/dc/dc3/tutorial_py_matcher.html.

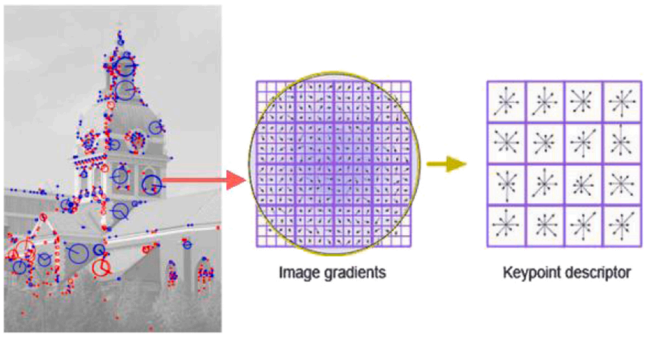

Fig. 3. Example of SIFT descriptor calculation, on the left the input image is overlaid with the detected keypoints. Here the circles indicate the scale at which the keypoint was detected. The line from the centre of the circle indicates the dominant gradient for that feature which acts to normalise the feature. In the middle and the right the pipeline to calculate the descriptor by taking a histogram of the local gradients is shown [15].

photometric factors are internal camera noise sources like image noise and motion blur. Finally, geometric factors are concerned with changes that occur when objects are occluded from the camera or projective transformation of features when the camera is under motion. In other words, while viewing the same 3D world point from various perspectives, we must accurately locate the same point for all images that have viewed this point.

A common workflow in calculating descriptors is to first transform the image patch to a normalised space that removes the perspective distortion of the feature. To fully model the transformation from one image to such a normalised space we need to use a projective transformation. Often we do not have the necessary information at the time of detection to model this projection to normalise the descriptors, so an affine projection is often used as an approximation. This affine projection is generally accurate enough when small camera motions are concerned. This means that generally descriptors are robust to scale and rotations only.

The remainder of this section is structured as follows. A brief outline of the classical methods used for image features is detailed. Understanding the motivations for the design and methods used should provide an insight into the problem and help develop domain knowledge. Then a more detailed description of the modern deep learning techniques is given.

2.1. Deep learning methods

Over the past ten years, deep learning methods have been revolutionising CV systems. Since [16] was published, a new era of CV has begun, which exploits the technology of deep neural networks. More specifically, CV usually employs CNNs which cascade convolutional layers along with pooling, spatial sub-sampling and non-linear activation functions to model complex CV problems. The general form of the multi-layer architectures performs by modeling low level features in the early layers while progressively capturing more complex features the data progresses through the network. In essence, the value added using such systems is that the data can be used to infer these image features, rather than classical methods outlined above which rely on an expert to "hand design" the desired appearance. The process of learning is mainly concerned with estimating a set of weights for the connections and the convolutions within the network. This process is performed by a regression of a provided cost function, in the context of supervised learning; this cost is estimated as the error between the putative output and the ground truth. The regression is performed by back propagation, a method that exploits the chain rule to allow efficient iterative refinement of the network weights.

In addition to the overview on detection and description, a subsection on end-to-end systems is also now provided. An advantage of CNNs is that they can receive low level pixel information and output high level data, allowing it to skip intermediate steps, and to jointly model both description and detection at once.

2.1.1. Detection

Detection is formulated as learning a response function that assigns a score for image patches. After all patches are processed in an image the scores are thresholded and minimum and maximum values are assigned as keypoints. This method provides a simple network which shows good invariance to rotation and scale changes [17]. These networks are trained using a loss function which jointly considers positive and negative correspondence at once in an attempt to create small intra and large inter class variance. This has shown positive results for providing reliable detections over time.

2.1.2. Description

The method of learning descriptors is a process of using a number of patches which are observations of the same 3D landmark and transforming these patches into a latent space that encodes the distinguishing features of the patches. The ultimate goal of such networks is to reduce the intra-class variance (observations of the same 3D landmark) and increase the inter-class variance (observations of different 3D landmarks). The innovations within these networks are realised in the loss functions used to train the networks. The simplest formulation for this loss function is referred to as *pairwise* loss, where it uses two patches at once to compute errors. Adding another image patch to the loss function is referred to as *triplet* loss. While *triplet* loss uses more information to learn descriptors and in turn should be more robust it also increases complexity and has been shown to provide a more unstable solution space [18]. Interesting work recently published [18] has shown improvements by selecting a subset of data from the training dataset, to perform back propagation on. This provides a method to identify specific weaknesses in the network during training and to dynamically select data to overcome these weaknesses.

2.1.3. End-to-end

The term *end-to-end* is used here to describe a system that outputs a complete set of image keypoints and descriptors. [14] reports that relative to the number of detection and description network papers published over the past number of years, end-to-end systems have received little attention. The first system of this kind was [19], which demonstrated how three sub networks, each trained for a distinct component of the image keypoint and description pipeline could be used to perform as an end-to-end system. The three networks included a keypoint detection network, a rotation normalisation network and finally a descriptor calculation network. The training process uses four image patches of the same 3D point viewed from different perspectives and trains each sub-network sequentially.

More recently, the work in [11] reports a deep CNN that uses novel techniques to generate ground truth data for training. The training involved in their method termed *SuperPoint* uses what the authors call "pseudo ground truth", which is comprised of simple geometric synthetic images with associated ground truth key points. This data is augmented with a method of homographic adaption, which warps each ground truth image a number of times with a randomly sampled homography. This has the effect of building affine transformation invariance into the network as well as providing more data for the network to train. They use this simple data to bootstrap a network and use this as initialisation on training for a real world dataset.

Likewise, the work in [20] has built on techniques from [11] but added the technique of dilation to both increase the receptive field of the network and remove the need for max pooling in the network. Removing the need for max pooling means that no spatial resolution is lost within the network, i.e. the input and output have the same spatial resolution, meaning that the resulting keypoint localisation is more accurate.

Increasing the dilation factor also results in increasing the receptive field, which has a positive effect on the descriptors.

2.2. CNN optimisation

Considerable attention has been given to making CNN based systems more efficient and possible to run on mobile devices over the past numbers of years. Many papers attempted to reduce the number of parameters by various techniques and still maintain state-of-the-art accuracy for CV tasks [21–24]. The most notable work [25], proved successful at compressing the network size and running on a mobile device while providing robust performance. The work showed that the preliminary layers of a network which are primarily responsible for extracting low level image features could be replaced by an efficient network. The authors demonstrated this by implementing the network on a Google Pixel phone and performing object detection and semantic segmentation in real-time. The technique used in this method was called depthwise separable convolution. This method is an alternative to the 2D convolutional operator normally found in CNNs. When applied to a network it significantly reduces the number of parameters and enhances the efficiency of each convolutional operation. In essence, it separates the normal convolutional operation into two distinct convolutions, namely depthwise and pointwise; the theory used to create these new methods can be thought of as similar to separable convolution filters in image processing. The results showed that despite significant reduction in network complexity and run time costs, the accuracy was acceptably similar to the state-of-the-art methods.

In later work from the same authors [13] they showed further improvements by using a technique called *inverted residuals* and *linear bottlenecks*. Again, they showed the results on a Google Pixel mobile device such that while network parameters decreased even further the accuracy remained comparable (see Table 1). Results taken from [13] show similar *mean Average Precision (mAP)* but with significantly less complexity. It is shown that the newly proposed MobileNetV2 sustains accuracy but reduces parameters and operations executed. Note MobilenetV1 was proposed in the authors' earlier work in [25].

While these works have shown positive results when applied to CV tasks such as object detection, little work has been carried out about applying these methods to an interest point detection and description network. The only notable work was [26] which applied a bespoke convolutional operation (which was a combination of normal convolution and separable depthwise convolution) to interest point detection. In addition, the experimental results provided were high level image retrieval metrics, whereas, in this paper low level metrics are used to further highlight the viability of methods. The aim of this paper is to investigate the optimisation techniques employed for other tasks in CV to the specific use case of interest point detection and description.

3. Model optimisation

In this section further details will be given about the techniques used to compress the model, along with information of improvements in terms of complexity. Then the interest point detection and description network selected to be optimised will be detailed to give an understanding of the original unmodified architecture. Finally, the steps taken to integrate these efficient design features will be outlined and the resulting network architectures will be defined.

Table 1
Precision and Complexity of Different Models.

Network	mAP	Params	MAdd	CPU
YOLOv2	21.6	50.7 M	17.5B	-
MobilenetV1	22.2	5.1 M	1.3B	270 ms
MobilenetV2	22.1	4.3 M	0.8B	200 ms

3.1. Depthwise separable convolution

Convolutional layers in CNNs normally operate by transforming an input tensor $F \in \mathbb{R}^{W_i \times H_i \times N}$ to an output tensor $G \in \mathbb{R}^{W_o \times H_o \times M}$, where W_i and H_i are the input width and height dimensions respectively. N and M are the input and output channel depths respectively, and W_o and H_o are the output width and height dimensions respectively. The transformation is parameterised by the convolution kernel K which is of size $K \times K \times M \times N \times W_i \times H_i$ where K is the kernel size.This operation of convolution, assuming padding and using a stride of one, is represented as:

$$G_{k,l,n} = \sum_{i,j,m} K_{i,j,m,n} \cdot F_{k+i-1,l+j-1,m}$$

From this equation we know the computational cost of this operation as:

$$K \times K \times M \times N \times W_i \times H_i$$

Within this operation each layer of the input tensor is convolved with a kernel and summed to produce the final output. In depthwise separable convolution, each of these steps are factorised into a separate operation. This results in two forms of convolution being applied, depth-wise and point-wise convolution. Depthwise convolves a kernel with each input layer. This operation can be written as:

$$\widehat{G}_{k,l,m} = \sum_{i,j,m} \widehat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m}$$

where $\widehat{K}$ is the depthwise convolutional kernel of size $k \times k \times M$ and each channel in F is convolved with a kernel in $\widehat{K}$.

Then pointwise convolution takes the output of depthwise convolution and sums values using a 1×1 convolution applied in the depth direction. Combining these two operations is known as depthwise separable convolution and has a computational cost as follows:

$$(K \times K \times M \times W_i \times H_i) + M \times N \times W_i \times H_i$$

Comparing the computational cost of the depthwise separable convolution over the standard convolution, we get:

$$\frac{(K \times K \times M \times W_i \times H_i) + M \times N \times W_i \times H_i}{K \times K \times M \times N \times W_i \times H_i}$$
$$= \frac{1}{N} + \frac{1}{K^2}$$

This shows that the improvement of the computational cost is a direct function of the number of input layers and the kernel size. This equates to an 8 – 9 times improvement for a kernel size of 3×3. While the computational complexity of depth-wise separable convolution is less, the expressibility of the network will also be affected. Since there will be fewer parameters used in the network using depth-wise separable convolution this may negatively affect the ability of the network to learn.

3.2. Linear bottlenecks

The premise of Linear Bottlenecks, documented in [13], is that it allows the information, normally flattened in non-linear activation layers, to be preserved. The advantage of using non-linear activation functions is that they increase the expressibility of the network. The goal for using linear bottlenecks is to mitigate this loss of information while also preserving the expressibility created. For this to be possible, the dimensionality of the activation space is increased to be sufficiently larger than the input space so it is possible for information, which is normally flattened in non-linear activation functions, to be preserved in other channels. Practically, this is achieved by performing a 1×1 convolution as the first operation in a residual block, which will increase the channel depth of the activation space by a factor. This parameter is known as the width

multiplier.

3.3. Inverted residual

Inverted residuals are designed primarily to help propagate residuals calculated in back propagation during the training process. For networks which are deep, often issues relating to vanishing gradients mean that the early layers of a network are either under-utilised or the training process will be slow. The principle behind residuals, first detailed in [27], is that the input is subtracted from the output for a discrete CNN block and the input to the next block is this residual. This results in the block being trained on the residual of the output minus the input. When preforming back propagation on this network it will now be more effective and allows residuals in the optimisation training procedure to be more stable and propagate through the network. This technique also has the added benefit of creating more separation between the scale of features learned in each CNN block.

3.4. Combining optimisations

The importance for each of the above optimisations have been extensively studied, residuals connections in [28], and linear bottlenecks [13]. It is well established that each technique contributes to optimising the network and an ablation study to prove this is provided in [13]. The experiments in this paper combine these techniques into a single convolutional block.

3.5. R2D2

Repeatable and Reliable Detector and Descriptor (R2D2) is a state-of-the-art interest point detection and description method. The method uses a CNN to jointly solve for interest points and associated descriptors. The network is constructed of a common feature extraction segment, based closely on the L2-Net description network [29], with two distinct heads. One head is for feature description and the other is to produce repeatability and reliability heatmaps; see Fig. 4. R2D2 offers two innovations: Reliability (over and above repeatability), and the use of kernel dilation. The reliability heatmap serves as a proxy to measure how reliable the descriptors are. R2D2 exploits this heatmap to provide an additional check when extracting the interest points so the results are not only repeatable but also have strong descriptors. Using kernel dilation allows the spatial resolution of the input to be preserved. This allows for accurate keypoint localisation accuracy as the resulting heatmap does not need to be up sampled to return to the original spatial dimensions. The common portion of the network is based on [29]; this was first designed to be a description network in a compact manner, which also allows Euclidean distance to be used as a correspondence metric. These features mean that [20] is a suitable starting point to optimise a feature point and description network.

3.6. Network architecture

This sub-section describes how the original R2D2 network is restructured and how the techniques above are integrated. Combining the techniques from above gives the basic building blocks of the network (see Table 2) called inverted residual block. The structure of an inverted residual block compared to a normal convolutional block can be seen in Fig. 5. The original architecture of the R2D2 network can be seen in Fig. 4. The section of the network indicated by the dashed line in Fig. 4 is replaced with a number of inverted residual blocks. Two configurations are proposed, both with a different number of blocks, to give insight into the effect of the number of blocks within the network on the efficiency and performance.

Table 3. The two variants of the network are named Lightweight (L) and High-Performance (HP). L is designed to reduce the number of parameters within the network to allow a considerable compression of the network. HP is designed to demonstrate any gains in accuracy which may be realised by a relatively larger network when compared to the L variants, but still employing the optimized techniques. The high level details of the parameter counts and complexity for each network variant are shown in Table 4. It shows Multiply-Accumulate Operations (MACs), parameter sizes, and inference Frame Per Second (FPS) which are common methods to compare relative performance of networks in deep learning. The inference FPS is measured by averaging execution times on an Nvidia Jetson Xavier NX device using the TensorRT inference library. Details for each layer in L network are shown in Table 5 and the HP network are shown in Table 6. Notice that as the layers get deeper in the original network, the parameter count and complexity grow exponentially in the original R2D2 network while the optimized variants increase linearly at a relatively small rate. This graph is a good demonstration of the savings in terms of complexity and size that the separable convolution provides. In normal convolutional layers, as the channel depth increases, the size and complexity of the layers increase exponentially. In other networks, where max pooling or striding are used some savings can be realised but in this case no max pooling layers are used and the stride is always equal to one meaning the input spatial resolution is preserved.

Normally max pooling also has the effect of increasing the receptive field within the network. This increase in receptive field has the effect of considering a larger image region when creating an image feature. To account for this, another method called kernel dilation is used. Kernel dilation allows the system to preserve the image resolution and increase the receptive field. The principle employed in kernel dilation is to use a sparse kernel sampling scheme where a number of pixels are skipped. A more detailed explanation of kernel dilation can be found in [30].

Table 2
Architecture of an inverted residual block, as designed in [13].

Input	Operator	Output
$h \times w \times k$	1×1 conv2d, ReLU6	$h \times w \times tk$
$h \times w \times tk$	3×3 dwise s = s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times tk$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1×1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

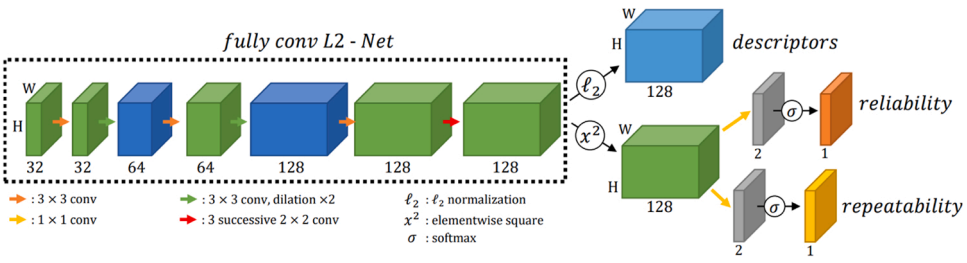

Fig. 4. R2D2 NetworkL: overview of the architecture is presented. The dashed part of a CNN can be replaced by the two example configurations given in the two runs underneath it [20]. Figure has been reproduced after permission by the corresponding author of [20].

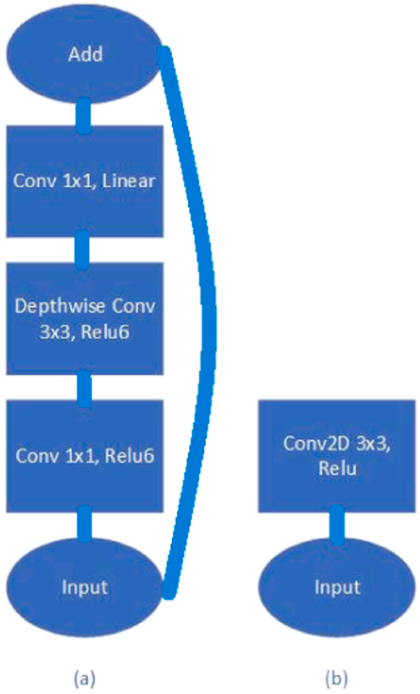

Fig. 5. Comparison of convolutional blocks. (a) shows the inverted residual block structure which combines the separable convolution, linear bottleneck and the inverted residual connection. (b) shows a normal convolutional block.

4. Experimental details

The following section gives implementation details for how the modified CNNs were trained, and tested. The definition of each KPI is provided. Details are given on the training of the networks. The datasets used for training and testing are outlined.

4.1. Metrics

Each metric is calculated for a pair of images, both images view the same scene either from a different perspective or under different lighting conditions. The location of each pixel in the other image is known by homographic transformation. During the metric calculations this homography is used to warp image keypoints between corresponding images. Let I^1 and I^2 be a pair of images and $KP^i = (kp^i_j)_{j<N_i}$ the set of N_i keypoints in image I^i.

4.1.1. Repeatability

Repeatability measures the percentage of keypoints common to both images. To calculate this metric we first need to warp KP^1 to I^2 using the homography transformation and let the result be $KP^{1,w}$. The repeatability is then calculated as the number of pairs which satisfy $||kp^{1,w} - kp^2||_2 < e$ with e set as the image distance threshold. The resulting matches are M and the final score is calculated as:

$$Repeatability = \frac{|M|}{min(|KP^1|, |KP^2|)}$$

4.1.2. Mean localisation error

Given the set of matches M, Mean Localisation Error (MLE) is the average error of the associated keypoints.

$$MLE = \frac{\sum ||kp^{1,w} - kp^2||_2 < e}{|M|}$$

4.1.3. Nearest neighbour mean average precision (NNmAP)

The NNmAP can be calculated by varying the descriptor distance

Table 4
Network Complexity. MACs, parameter sizes, and inference FPS.

Network	MACs [G]	Parameters [K]	Compression Ratio	FPS
R2D2	149	484	× 1	17.6
R2D2 - L	18	58	× 8.2	145
R2D2 - HP	137	449	× 1.1	19.5

Table 5
Dimensions of each layer in the L network, the number of parameters for each layer along with the associated total for the network.

Layer Type	K	R2D2 Output Shape			Params
		C	H	W	
Inverted Residual	3	32	480	640	480
Inverted Residual	3	32	480	640	5136
Inverted Residual	3	64	480	640	8832
Inverted Residual	3	64	480	640	10,000
Inverted Residual	3	128	480	640	14,848
Inverted Residual	3	128	480	640	14,848
Inverted Residual	2	128	480	640	21,056
Conv2d-BN	2	128	480	640	8448
Conv2d	1	2	480	640	258
Conv2d	1	1	480	640	129
Total					84,035

Table 3
Performance of networks. The key performance indicators for each network showing the performance difference with the efficient techniques integrated.

Network	MLE	Repeatability	NNmAP	MScore	Homog Est Err (HEE)= 1	HEE= 3	HEE= 5
				Overall			
SIFT	1.368	0.371	0.767	0.299	0.312	0.659	0.781
SuperPoint	1.158	0.581	0.821	0.470	0.310	0.684	0.829
R2D2	1.503	0.556	0.823	0.380	0.407	0.764	0.852
R2D2 - L	1.827	0.535	0.718	0.288	0.386	0.710	0.795
R2D2 - HP	1.609	0.643	0.675	0.303	0.448	0.750	0.821
				Viewpoint			
SIFT	1.324	0.383	0.792	0.331	0.319	0.678	0.797
SuperPoint	-	0.484	-	-	-	-	-
R2D2	1.639	0.527	0.721	0.302	0.237	0.597	0.739
R2D2 - L	1.909	0.504	0.595	0.217	0.176	0.519	0.651
R2D2 - HP	1.778	0.609	0.547	0.233	0.220	0.573	0.685
				Illumination			
SIFT	1.413	0.358	0.740	0.267	0.305	0.639	0.765
SuperPoint	-	0.631	-	-	-	-	-
R2D2	1.362	0.585	0.929	0.461	0.582	0.937	0.968
R2D2 - L	1.742	0.567	0.846	0.362	0.604	0.909	0.944
R2D2 - HP	1.434	0.679	0.807	0.376	0.684	0.933	0.961

Table 6
Dimensions of each layer in the high performance network are shown. The number of parameters for each layer along with the associated total for the network.

Layer Type	K	R2D2 Output Shape			Params
		C	H	W	
Inverted Residual	3	16	480	640	480
Inverted Residual	3	24	480	640	5136
Inverted Residual	3	24	480	640	8832
Inverted Residual	3	32	480	640	10,000
Inverted Residual	3	32	480	640	14,848
Inverted Residual	3	32	480	640	14,848
Inverted Residual	2	64	480	640	21,056
Inverted Residual	2	64	480	640	21,056
Inverted Residual	2	64	480	640	54,272
Inverted Residual	2	64	480	640	54,272
Inverted Residual	2	96	480	640	54,272
Inverted Residual	2	96	480	640	66,624
Inverted Residual	2	96	480	640	118,272
Conv2d-BN	2	128	480	640	25088
Conv2d	1	2	480	640	258
Conv2d	1	1	480	640	129
Total					448,771

threshold; as a result the Receiver Operating Characteristic (ROC) curve can be calculated, which shows the recall and precision function. The area under this curve is used as a measure of the average precision. The metric is an indicator of how successful the descriptors are at being matched correctly.

4.1.4. Matching score

Matching score captures the ratio of points which are both nearest neighbours in terms of image and descriptor distance M_d, while also being within their respective error thresholds.

$$MatchingScore = \frac{M \cap M_d}{min(|KP^1|, |KP^2|)}$$

This metric is an indicator of the joint performance of the interest points and the descriptors.

4.1.5. Homographic estimation error

Using the findHomography function from OpenCV a homography for an image pair is calculated and compared to the known ground truth homography. Since elements in the 3×3 homography matrix are scaled differently, comparison is not trivial. Following a similar approach as [11] where the corners of image I^1 projected into I^2 are calculated using both the estimated homography and the ground truth homography, the error is calculated as the average error of the four corners between the two projections. To allow for more granularity three thresholds are used when calculating. The resulting metric is the percentage of the computed homographies, which are within the stated error threshold.

4.2. Training

Each training script was developed using the PyTorch [31] library. The training was carried out on an Nvidia Titan V GPU. The datasets used for the training process were the same as in original R2D2 paper [20].

Following [13], the optimiser used to train was Stochastic Gradient Descent (SGD), with a learning rate of $1e-3$, momentum of $9e-1$, weight decay of $1e-1$, and the nestorov method activated. Each training was performed for 25 epochs and the batch size was 10. Ideally a larger batch size would produce more accurate results but in this case the batch size was limited by the memory available on the GPU. A result of using these values of hyper-parameters in this paper means that networks become easier to train.

It should also be noted that L2 regularization was used on the descriptor head layer to help improve its generalisation. This is an important step to prevent over-fitting of the descriptor layer and allows for better performance on previously unseen data.

4.3. Datasets

The dataset used for evaluation of the networks is the HPatches dataset [32]. This dataset contains 108 sequences, each with 6 images. One image is referred to as the reference image and the associated homography matrix to project each of the remaining images to the reference image is provided. This allows exact correspondence between each pixel in the reference image to the other images. 54 of the sequences are of a static scene with no camera motion and the only variable is scene illumination. The other 54 sequences contain camera motion.

To standardise the size of the images a scaling preserving resize is used which enforces that the maximum size is 640×480. The homography transformations are also scaled accordingly to account for the resize. The networks are limited to a maximum number of 300 interest points. This number was selected as it is a reasonable number of interest points to run higher level algorithms with, e.g. pose estimation, while also providing a relatively low memory requirement.

5. Results

The comparisons of the results are broken into two sections, KPIs, and Network Sizes and Complexity. KPIs capture the accuracy and robustness of the algorithms over the test dataset, while network sizes and complexity compare the computational cost and memory requirements for each network. In each case the baseline for comparison is the original R2D2 network. In addition, to demonstrate the performance of this network against state-of-the-art hand crafted and deep learning methods, SIFT and SuperPoint [11] are included. It is worth noting that the only KPI reported for the viewpoint and illumination changes in [11] is repeatability and for the overall results each KPI is reported.

5.1. KPIs

The results, shown in Table 3, detail the performance of the systems over all of the KPIs detailed above. The results are separated into three sections: the viewpoint section contains results from the sequences with viewpoint point changes only. The illumination section contains results for sequences with illumination changes only. And the overall section shows results over the complete dataset. It is valuable to show separate results for illumination and viewpoint changes because they are both distinct conditions which test the robustness of the system. The results are also shown in Fig. 6. It is noteworthy, that MLE is better as it approaches 0 and the remaining KPIs get better as they approach 1.

For the viewpoint results we can see that, apart from the repeatability results, SIFT performs best. These results demonstrate SIFT's ability to provide invariance to scale and rotation. Within SIFT a multi scale method is used to encode the size of features as well as rotation normalisation to remove the effect of a moving camera. Also the MLE for SIFT is lowest mainly due to the sub-pixel refinement step, which increases the accuracy of the output localisation. All these steps are valuable in terms of performance but they do come with a computational overhead related to the multiple scale processing. R2D2 does allow for multiple scale processing, where the input image is used to create a scale space and each scaled image is passed through the network and image points are intelligently combined to produce the output. This will have the effect of creating some scale invariance but given the focus of this work, to implement an efficient solution the single scale configuration is used. SuperPoint is repeatedly the worst performing method and shows its poor robustness to viewpoint changes. Robustness to viewpoint changes is vital for methods used to support a pose estimation system, where the viewpoint is expected to regularly change over time.

In terms of the relative R2D2 variants performance, we see that the

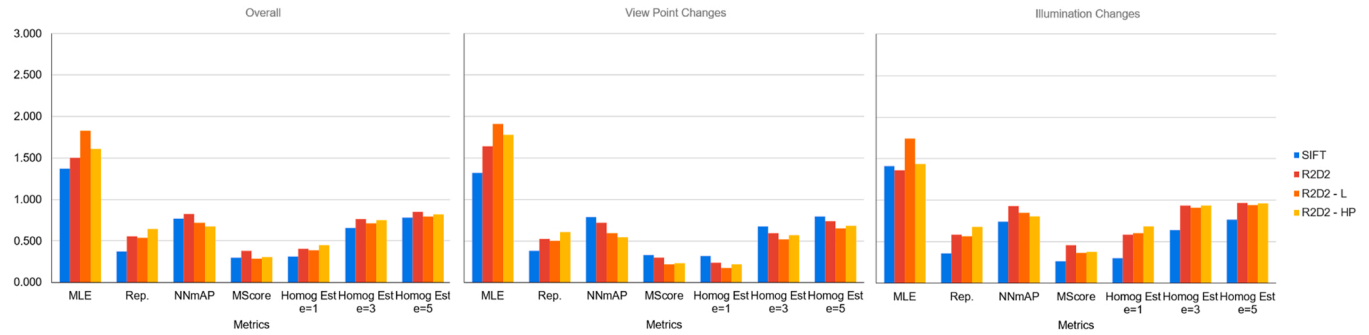

Fig. 6. KPI performance of systems. The performance for each system is graphed together to allow relative evaluation. The three graphs show separate results for illumination change, viewpoints changes and overall results.

original network performs best for the metrics. That being said, the other variants do provide comparable performance. It is worth mentioning that there is no reason for the networks to outperform the original, rather the aim here is to replicate the performance as closely as possible.

The illumination results show a different trend when compared to the viewpoint results. The original R2D2 performs better than SIFT in all of the KPIs. This demonstrates that R2D2 provides better invariance to scene illumination. It is difficult to attribute this to one aspect or technique used within the network but it does give proof of the ability of the CNN to generalise complex features. The HP variant performs slightly better when compared to L. The homography estimation results give a good indication for the performance within a pose estimation system. For these KPIs we see strong results for all R2D2 variants when compared to SIFT. This is due to the better repeatability, NNmAP, and matching score which indicates more accurate interest points and discriminate descriptors. SuperPoint, in terms of repeatability, is competitive with R2D2 and only the HP variant outperforms it.

Separating the results into illumination and viewpoint demonstrates the attributes of each system when it comes to robustness. As a whole, the overall results weigh up the combined performance and show that R2D2 is a state of the art system according to the KPIs used. The MLE and repeatability show that the interest points are accurate and reliable. The NNmAP shows that descriptors are distinctive and do well at encoding the appearance of the interest points compactly, while the MScore and Homography Estimation show that the interest point and descriptors combine well to provide higher level systems with good data. For the HP and L variants we do see a slight performance drop off when looking at the low level interest point and descriptor KPIs but the results for the Homograph are comparable, and better in some cases. The HP network does perform better when compared to the L network, but this was expected since there was a larger number of convolutional layers within the HP network. SuperPoint does perform well in the low level KPI but the performance on the homography estimation metrics is not as accurate as R2D2.

5.2. *Network sizes and complexity*

A comparison of the network sizes and complexity is shown in Fig. 2 and also in Fig. 7 where the numbers are presented for each layer. It is clear that the L variant of the network provides considerable compression of the network with a compression factor of 8.2 and the number of MACs is also an order of magnitude less than the original network. Taking into account the KPI performance of this network it proves to be a viable solution at providing a more compact and efficient network with only a slight trade-off in accuracy to be made. The HP variant proves to have similar requirements to the original network in terms of parameter count and MACs. Coupled with the fact the KPI performance is slightly worse than the original indicates that this variant does not provide any advantages. It is worth mentioning that the HP variant contains 6 extra convolutional layers when compared to the original network. These extra layers could allow for the network to learn more complex features but ultimately this is not realised within this testing. It should however be considered as future work to investigate the potential for these extra layers.

Fig. 8 shows the results of the L Network. The top left image shows the input image. The top right image shows the resulting keypoints from processing the output heatmaps. The bottom left image is the repeatability heatmap and the bottom right image is the reliability heatmap. The same kind of results for the HP Network can be seen in 9.

6. Conclusions

Inferring accurate models with low complexity is becoming a cross-cutting concern in Machine Learning [33,34] and Machine Learning

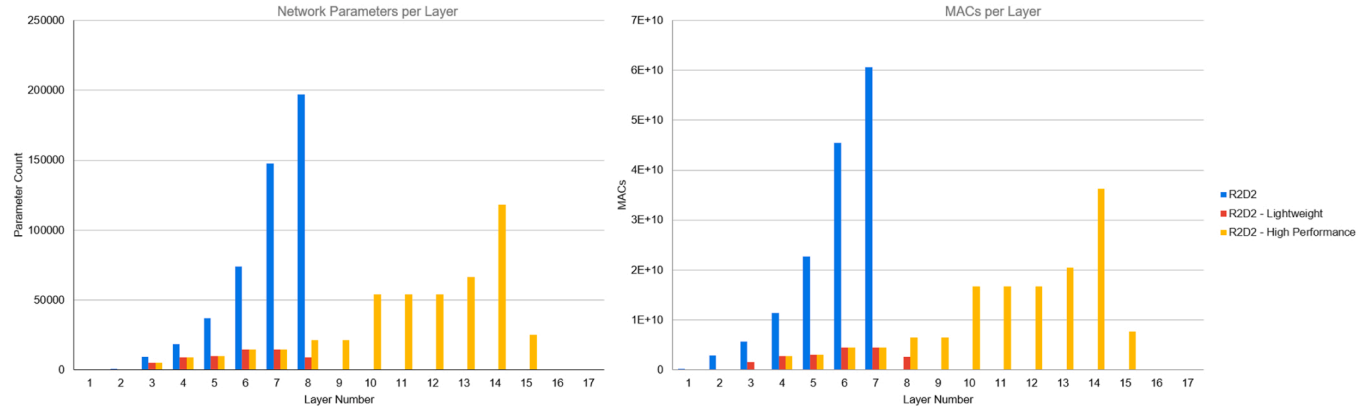

Fig. 7. Complexity of Networks. On the left the parameter count per layer of the original and optimised networks is shown. On the right are the number of MACs per layer of the original and optimised networks is shown.

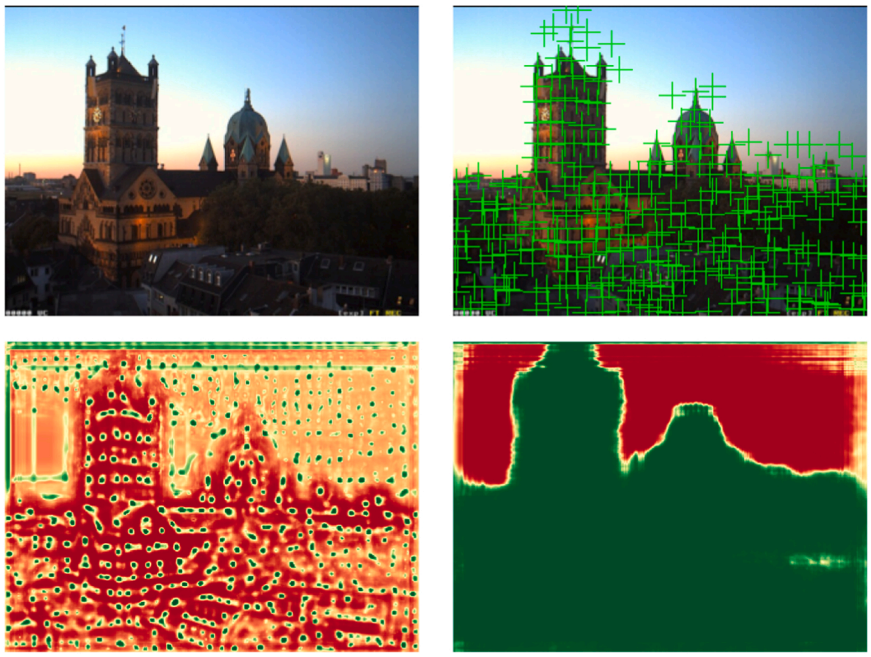

Fig. 8. R2D2 - L Network inputs and outputs. The top left image shows the input image, the top right image shows the resulting keypoints from processing the output heatmaps, the bottom left image is the repeatability heatmap and the bottom right image is the reliability heatmap.

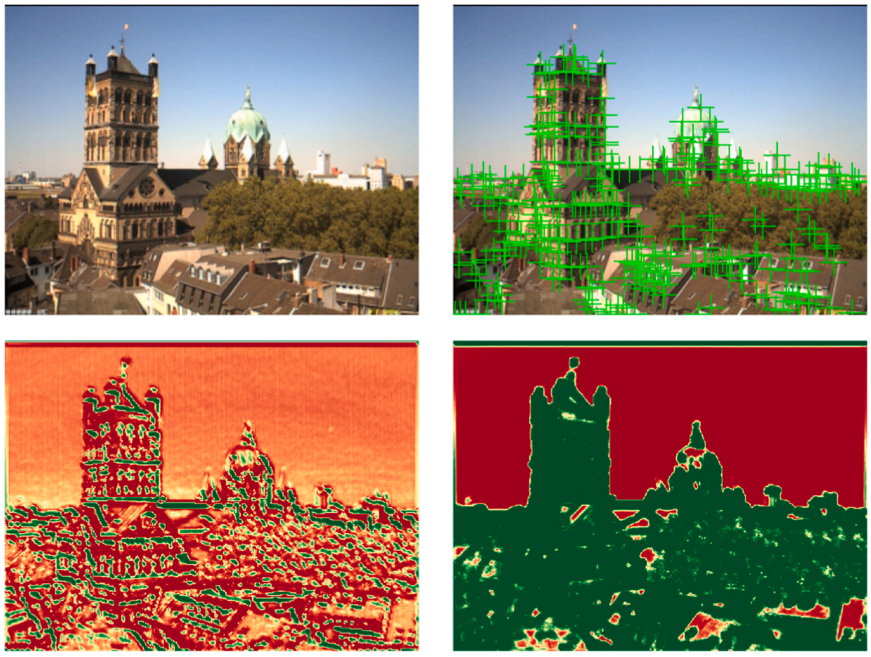

Fig. 9. R2D2 - HP Network inputs and outputs. The top left image shows the input image, the top right image shows the resulting keypoints from processing the output heatmaps, the bottom left image is the repeatability heatmap and the bottom right image is the reliability heatmap.

inspired Computing [35,36]. This work has demonstrated the successful integration of efficient techniques that help to reduce the complexity and size of CNNs used in interest point detection and description systems. The complete KPI framework has adopted a state-of-the-art dataset and KPIs to measure the relative accuracy performance of the various systems. It was found that only a slight accuracy trade-off afforded a considerable compression of network size, and lower computational complexity was achieved with the L variant. As a matter of fact, we have shown that the L network performed substantially better while it ran on an Nvidia Jetson Xavier NX GPU. On the other hand, the results reported for the SuperPoint network were accumulated using an Nvidia Titan X GPU. It is worth noting that the later GPU is much superior to the erstwhile in terms of specifications, throughput, and speed. The HP variant did not prove to have any advantages in terms of accuracy, size, or computational complexity.

The motivation of the paper was to enable a state-of-the-art network to operate in a constrained computing environment, such as a battery-powered robot. Examples of the applications that interest point networks can support are trajectory planning for robotic vacuum cleaners, head pose estimation for virtual reality headsets, and camera calibration for a wide variety of camera-based systems. The network selected to be studied was the best-performing network, in terms of accuracy, which

allowed the integration of the studied optimization techniques in [13]. The other state-of-the-art network, called SuperPoint [11], uses an Encoder-Decoder (Autoencoders (AEs)) architecture which is not appropriate for these optimizations since the skip connections are not within each convolutional block rather they are connected between each corresponding decoder and encoder blocks. AEs are not appropriate here because supervised learning is required to teach the network what interest points are.

The training hyper-parameters details are given in section 4.2. A result of using the optimization techniques in this paper means that networks become easier to train in the sense that they are less sensitive to hyper-parameters when compared to the original networks. The skip connections in each convolutional block allow gradients during back-propagation to flow into deeper layers thus allowing each layer to learn efficiently. However, in a future extension of this research, we also aspire to employ a hyper-parameter tuning scheme based on the traditional grid search or the more sophisticated and advanced evolutionary algorithms.

In terms of future work, it would be valuable to implement such efficient networks to better understand the practical challenges of running on embedded devices. In addition, it would be interesting to understand if and how the extra layers in the HP variant could be exploited to provide accuracy improvements.

Although the focus of this work has been to improve efficiency while retaining accuracy, future work can also look to further enhance the generalization ability of the models produced by the proposed methodology. Although we have already employed L2 regularization and shown the generalization ability of our models on unseen data, we also aim to leverage certain novel and promising regularization techniques. Regularization techniques have come a long way as compared with the traditional addition of a punitive term to the loss function. [37] have proposed a novel generalization algorithm that initially performs anomaly detection and eventually regularizes the model based on the distribution of the data. The technique has shown nice results in CV and deep learning. In a recent work manifold regularization was applied to train auto encoders successfully and was found to be quite beneficial [38]. We also plan to benefit from this approach in the future.

CRediT authorship contribution statement

Patrick Rowsome: Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Muhammad Adil Raja:** Writing – review & editing, Supervision, Conceptualization. **R. Muhammad Atif Azad:** Writing – review & editing, Supervision.

Declaration of Competing Interest

This is to state that this work has no conflict of interests.

Acknowledgments

This research was partly funded by the Technological University Transfer Fund (TUTF) of the Higher Education Authority (HEA) of Ireland.

References

1. J.N. Kundu, R.M. V, A. Ganeshan, R.V. Babu, Object pose estimation from monocular image using multi-view keypoint correspondence, CoRR abs/1809.00553 (2018). arXiv:1809.00553.⟨http://arxiv.org/abs/1809.00553⟩.
2. Torii A, Taira H, Sivic J, Pollefeys M, Okutomi M, Pajdla T, Sattler T. Are large-scale 3d models really necessary for accurate visual localization? *IEEE Trans Pattern Anal Mach Intell*. 2021;43:814–829.
3. G. Csurka, C.R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, In: In Workshop on Statistical Learning in Computer Vision, ECCV, 2004, 1–22.
4. C.G. Harris, M. Stephens, et al., A combined corner and edge detector., In: Alvey vision conference, Vol. 15, Citeseer, 1988, 10–5244.
5. Smith SM, Brady JM. Susan—a new approach to low level image processing. *Int J Comput Vis*. 1997;23(1):45–78.
6.. J. Shi, et al., Good features to track, In: 1994 Proceedings of IEEE conference on computer vision and pattern recognition, IEEE, 1994, 593–600.
7. Trajković M, Hedley M. Fast corner detection. *Image Vis Comput*. 1998;16(2):75–87.
8. Lowe DG. Distinctive image features from scale-invariant keypoints. *Int J Comput Vis*. 2004;60(2):91–110.
9. Bay H, Ess A, Tuytelaars T, Van Gool L. Speeded-up robust features (surf). *Comput Vis Image Underst*. 2008;110(3):346–359.
10. A. Alahi, R. Ortiz, P. Vandergheynst, Freak: Fast retina keypoint, In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2012, 510–517.
11. D. DeTone, T. Malisiewicz, A. Rabinovich, Superpoint: Self-supervised interest point detection and description, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, 224–236.
12. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. *Proc IEEE Conf Comput Vis Pattern Recognit*. 2016:779–788.
13. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, 4510–4520.
14. G. Csurka, M. Humenberger, From handcrafted to deep local invariant features, arXiv preprint arXiv:1807.10254 2 (2018).
15. B. Mahaseni, N.D. Salih, Asian stamps identification and classification system, arXiv preprint arXiv:1709.05065 (2017).
16. A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, In: Advances in neural information processing systems, 2012, 1097–1105.
17. N. Savinov, A. Seki, L. Ladicky, T. Sattler, M. Pollefeys, Quad-networks: unsupervised learning to rank for interest point detection, In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, 1822–1830.
18. E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, F. Moreno-Noguer, Discriminative learning of deep convolutional feature point descriptors, In: Proceedings of the IEEE International Conference on Computer Vision, 2015, 118–126.
19. Yi KM, Trulls E, Lepetit V, Fua P. *Lift: learned invariant feature transform. European Conference on Computer Vision*. Springer; 2016:467–483.
20. Revaud J, Weinzaepfel P, de Souza CR, Humenberger M. *R2D2: repeatable and reliable detector and descriptor. NeurIPS*. 2019.
21. L. Sifre, S. Mallat, Rigid-motion scattering for image classification, Ph. D. thesis (2014).
22. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167 (2015).
23. J. Jin, A. Dundar, E. Culurciello, Flattened convolutional neural networks for feedforward acceleration, arXiv preprint arXiv:1412.5474 (2014).
24. M. Wang, B. Liu, H. Foroosh, Factorized convolutional neural networks, In: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, 545–553.
25. A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).
26. R. Miles, K. Mikolajczyk, Compression of convolutional neural networks for high performance imagematching tasks on mobile devices, arXiv preprint arXiv: 2001.03102 (2020).
27. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, 770–778.
28. Shafiq M, Gu Z. Deep residual learning for image recognition: a survey. *Appl Sci*. 2022;12(18). https://doi.org/10.3390/app12188972. ⟨https://www.mdpi.com/2076-3417/12/18/8972⟩.
29. Y. Tian, B. Fan, F. Wu, L2-net: Deep learning of discriminative patch descriptor in euclidean space, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, 661–669.
30. F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122 (2015).
31. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, In: Advances in neural information processing systems, 2019, 8026–8037.
32. V. Balntas, K. Lenc, A. Vedaldi, K. Mikolajczyk, Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors, In: CVPR, 2017.
33. Sambo AS, Azad RMA, Kovalchuk Y, Indramohan VP, Shah H. Evolving simple and accurate symbolic regression models via asynchronous parallel computing. *Appl Soft Comput*. 2021;104, 107198. https://doi.org/10.1016/j.asoc.2021.107198. ⟨https://www.sciencedirect.com/science/article/pii/S1568494621001216⟩.
34. Sambo AS, Azad RMA, Kovalchuk Y, Indramohan VP, Shah H. Time control or size control? Reducing complexity and improving accuracy of genetic programming models. In: Hu T, Lourenço N, Medvet E, Divina F, eds. *Genetic Programming*. Cham: Springer International Publishing; 2020:195–210.
35. Chennupati G, Azad RMA, Ryan C. Performance optimization of multi-core grammatical evolution generated parallel recursive programs. *Assoc Comput Mac*. 2015. https://doi.org/10.1145/2739480.2754746.

36. Ryan C, Azad RMA. Sensible initialisation in chorus. In: Ryan C, Soule T, Keijzer M, Tsang E, Poli R, Costa E, eds. *Genetic Programming, Springer Berlin Heidelberg, Berlin, Heidelberg*. 2003:394–403.
37. Zheng Q, Yang M, Yang J, Zhang Q, Zhang X. Improvement of generalization ability of deep cnn via implicit regularization in two-stage training process. *IEEE Access*. 2018;6:15844–15869.
38. Zheng Q, Zhao P, Zhang D, Wang H. Mr-dcae: manifold regularization-based deep convolutional autoencoder for unauthorized broadcasting identification. *Int J Intell Syst*. 2021;36(12):7204–7238.