

Article

Shape–Texture Debiased Training for Robust Template Matching [†]

Bo Gao *  and Michael W. Spratling 

Department of Informatics, King's College London, London WC2R 2LS, UK

* Correspondence: bo.gao@kcl.ac.uk

[†] This paper is an extended version of our paper published in the International Conference on Image, Vision, and Intelligent Systems (ICIVIS 2021), Changsha, China, 15–17 June 2021.

Abstract: Finding a template in a search image is an important task underlying many computer vision applications. This is typically solved by calculating a similarity map using features extracted from the separate images. Recent approaches perform template matching in a deep feature space, produced by a convolutional neural network (CNN), which is found to provide more tolerance to changes in appearance. Inspired by these findings, in this article we investigate whether enhancing the CNN's encoding of shape information can produce more distinguishable features that improve the performance of template matching. By comparing features from the same CNN trained using different shape–texture training methods, we determined a feature space which improves the performance of most template matching algorithms. When combining the proposed method with the Divisive Input Modulation (DIM) template matching algorithm, its performance is greatly improved, and the resulting method produces state-of-the-art results on a standard benchmark. To confirm these results, we create a new benchmark and show that the proposed method outperforms existing techniques on this new dataset.

Keywords: template matching; convolutional neural networks; VGG19

Citation: Gao, B.; Spratling, M.W. Shape–Texture Debiased Training for Robust Template Matching. *Sensors* **2022**, *22*, 6658. <https://doi.org/10.3390/s22176658>

Academic Editor: Sylvain Girard

Received: 12 July 2022

Accepted: 31 August 2022

Published: 2 September 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Template matching is a technique for finding a rectangular region of an image that contains a certain object or image feature. It is widely used in many computer vision applications, including object tracking [1,2], object detection [3,4], and 3D reconstruction [5,6]. A similarity map is generally used to quantify how well a template matches each location in an image, typically generated by sliding the template through the search image, then the matching position is determined by finding the location of maximum value of the similarity map. Traditional template matching generates the similarity map based on pixel intensity values, and is not robust to hard matching scenarios such as significant non-rigid deformations of the object, changes in the illumination and size of the target, and occlusion [7]. To address this issue, more distinctive hand-crafted features such as scale-invariant feature transform (SIFT) [8] and histogram of oriented gradients (HOG) [9] can be used instead of the intensity values for robust template matching [10–13]. However, these features must be extracted by certain manually predefined algorithms based on expert knowledge, and therefore have limited description capabilities [14].

With the help of deep features learned from convolutional neural networks (CNNs), vision tasks such as image classification [15,16], object recognition [17,18], and object tracking [19,20] have recently achieved great success. In order to succeed in such tasks, CNNs need to be trained with big data and automatically build internal representations that are less effected by changes in the appearance of objects in different images. Therefore, CNNs have strong description capability far exceeding that of hand-crafted features; recent methods have been successfully applied to a feature space produced by the convolutional layers of a CNN, achieving impressive performance [7,21–26].

The higher layers of CNNs are believed to learn representations of shapes from low-level features [27]. However, recent studies [28,29] have demonstrated that ImageNet-trained CNNs are biased toward making categorisation decisions based on texture rather than shape. The same works showed that CNNs could be trained to increase sensitivity to shape, resulting in improved accuracy and robustness of object classification and detection. Assuming that shape information is useful for template matching, these results suggest that the performance of template-matching methods applied to CNN-generated feature spaces could potentially be improved by training the CNN to be more sensitive to shape.

In this article, we verified this assumption by comparing features from five CNN models that had the same network structure while differing in shape sensitivity. Our results show that training a CNN to learn about texture while biasing it to be more sensitive to shape information can improve template matching performance. Furthermore, by comparing template-matching performance when using feature spaces created from all possible combinations of one, two, and three convolutional layers of the CNN, we found that the best results were produced by combining features from both early and late layers. Early layers of a CNN encode lower-level information such as texture, while later layers encode more abstract information such as object identity. Hence, both sets of results (the need to train the CNN to be more sensitive to shape and the need to combine information for early and late layers) suggest that a combination of texture and shape information is beneficial for template matching.

Our main contributions are summarised as follows:

- We created a new benchmark; compared to the existing standard benchmark, it is more challenging, provides a far larger number of image pairs, and is better able to discriminate between the performance of different template matching methods.
- By training a CNN to be more sensitive to shape information and combining features from both early and late layers, we created a feature space in which the performance of most template matching algorithms is improved.
- Using this feature space together with an existing template matching method, DIM [30], we obtained state-of-art results on both the standard and new datasets.

This paper is an extension of work originally presented at ICIVS2021 [31]. The conference paper reported the template matching results of the DIM algorithm using features extracted from four VGG19 models with different shape sensitivities in order to determine the best deep feature space for template matching, then compared the performance of many template-matching algorithms in that feature space. The current work adds a reviews of the latest literature in Section 2, details of the DIM algorithm in Section 3.2, new results using features from a new VGG19 model (Model_E) trained by the latest shape–texture debiased training method [29] along with related discussion in Section 4, visualisation of the results of different template matching algorithms in Section 4.4, and a concluding discussion in Section 5.

2. Related Work

2.1. Template Matching

Traditional template matching methods calculate the similarity map using a range of metrics such as the normalised cross-correlation (NCC), sum of squared differences (SSD), and zero-mean normalised cross-correlation (ZNCC), which are applied to the pixel intensity or colour values. However, because these methods rely on comparing the values in the template with those at corresponding locations in the image patch, they are sensitive to changes in lighting conditions, non-rigid deformations of the target object, or partial occlusions, which can result in a low similarity score when one or multiple of these situations occur. To overcome the limitations of classic template matching methods, many approaches [7,21,22,24–26,32] have been developed. These methods can be classified into two main categories.

One category attempts to increase tolerance to changes in appearance by changing the computation that is performed to compare the template to the image. For example,

Best-Buddies Similarity (BBS) counts the proportion of sub-regions in the template and the image patch that are Nearest-Neighbour (NN) matches [7]. Deformable Diversity Similarity (DDIS) explicitly considers possible template deformation using the diversity of NN feature matches between a template and a potential matching region in the search image [24]. Annulus Projection Transformation and Neighbour Similarity (APT-MNS) [26] builds the global spatial structure of the target object using a novel annulus projection transformation (APT) vector to filter out the incorrectly matched NN candidates, then estimating the best matched candidates using the MNS measurement. Weighted Smallest Deformation Similarity (WSDS) [25] calculates the smallest deformation between each point in the template and its NN matches to explicitly penalise the deformation. In addition, weights are defined for points in the template based on their likelihood of belonging to the background calculated through NN matching with the points around the target window. This reduces the negative effect of background pixels contained in the template box. The Divisive Input Modulation (DIM) algorithm [30] extracts additional templates from the background and lets the templates compete with each other to match the image. Specifically, this competition is implemented as a form of probabilistic inference known as explaining away [33,34], which causes each image element to only provide support for the template that is the most likely match. Previous work has demonstrated that DIM, when applied to colour feature-space, is more accurate in identifying features in an image compared to both traditional and recent state-of-the-art matching methods [30].

The second category of approaches changes the feature space in which the comparison between the template and the image is performed. The aim is for this new feature space to allow better discrimination in template matching while increasing tolerance to changes in appearance. Co-occurrence based Template Matching (CoTM) transforms the points in the image and template to a new feature space defined by the co-occurrence statistics to quantify the dissimilarity between a template and an image [22]. Quality-Aware Template Matching (QATM) is a method that uses a pretrained CNN model as a feature extractor. It learns a similarity score that reflects the (softness) repeatness of a pattern using an algorithmic CNN layer [21]. Occlusion Aware Template Matching (OATM) [32] searches neighbours among two sets of vectors and uses a hashing scheme based on consensus set maximisation, and is hence able to efficiently handle high levels of deformation and occlusion.

2.2. Deep Features

Many template matching algorithms from the first category above can be applied both to deep features and directly to colour images. The deep features used by BBS, CoTM, and QATM are extracted from two specific layers of a pre-trained VGG19 CNN [35], *conv1_2* and *conv3_4*. Following the suggestion in [20] for object tracking, DDIS takes features from a deeper layer, fusing features from layers *conv1_2*, *conv3_4*, and *conv4_4*. In [23], the authors proposed a scale-adaptive strategy to select a particular individual layer of a VGG19 to use as the feature space according to the size of template. In each case, using deep features was found to significantly improve template matching performance compared to using colour features.

A recent study has shown that ImageNet-trained CNNs are strongly biased towards recognising textures rather than shapes [28]. The same study demonstrated that the same standard architecture (ResNet-50) [36] that learns a texture-based representation on ImageNet is able to learn a shape-based representation when trained on ‘Stylised-ImageNet’, a version of ImageNet that replaces the texture in the original image with the style of a randomly selected painting through AdaIN style transfer [37]. This new shape-sensitive model was found to be more accurate and robust in both object classification and detection tasks. However, the stylised dataset needs to be generated before the training process using a pre-defined set of texture source images. Due to computation and resource limitations, each image in the stylised dataset is only transferred by one random artistic image, which results in lack of diversity for each sample. Furthermore, the training process is complicated and involves first training the network on both standard and stylised training

datasets, then fine-tuning on the standard dataset. In contrast, [29] proposes a shape–texture debiased training method which provides the corresponding supervisions from shape and texture simultaneously. Similarly, this method is based on AdaIN style transfer, with the difference in implementation being that it replaces the original texture information with uninformative texture patterns from another randomly selected image from the training mini-batch rather than with the style of randomly selected artistic paintings. This results in increments of diversity for each image; hence, this method achieves higher accuracy and robustness than [28] for image classification with the ResNet-50 architecture. Inspired by these findings, in this paper we investigate whether enhancing the shape sensitivity of a CNN can produce more distinguishable features that improve the performance of template matching.

3. Methods

3.1. Training CNN with Stylised Data

Previous work on template matching in deep feature space (see Section 2) has employed a VGG19 CNN. To enable a fair comparison with those previous results, we used the VGG19 architecture as well. However, we used five VGG19 models that differed in terms of the way they were trained to encode different degrees of shape selectivity, as summarised in Table 1.

Table 1. The five different VGG19 CNN models used in this paper. IN and SIN are the abbreviations of ImageNet and Stylised-ImageNet, respectively.

Name	Training	Fine-Tuning	Rank of Shape Sensitivity
Model_A	IN	-	4
Model_B	SIN	-	1
Model_C	IN + SIN	-	2
Model_D	IN + SIN	IN	3
Model_E	IN	-	-

Model_A to Model_D were trained using the same approach as in [28], with a stylised dataset generated before the training process; the ranking shape-sensitivity of these models was controlled by setting the different training datasets manually. Model_A was trained using the standard ImageNet dataset [35]; we used the pretrained VGG19 model from the PyTorch torchvision library. This model has the least shape bias. Model_B was trained on the Stylised-ImageNet dataset, and thus has the most shape bias. Model_C was trained on a dataset containing the images from both ImageNet and Stylised-ImageNet. Model_D was initialised with the weights of Model_C followed by fine-tuning on ImageNet for 60 epochs using a learning rate of 0.001 multiplied by 0.1 after 30 epochs. Therefore, Model_C and Model_D have intermediate levels of shape bias, with model_D being less selective to shape than Model_C. The stylised data samples used in Model_E were generated during training, and the training process provided supervisions from shape and texture simultaneously [29]. Hence, Model_E has an intermediate level of shape bias, although where it should rank relative to Model_C and Model_D was impossible to quantify. The learning rate was 0.01 multiplied by 0.1 after every 30 epochs for Model_B and Model_E, and after every 15 epochs for Model_C. The number of epochs was 90 for Model_B and Model_E and 45 for Model_C; as the dataset used to train Model_C was twice as large as that used to train Model_B, the number of weight updates was the same for both models. The other training hyperparameters used for each model were a batch size of 256, momentum 0.9, and weight decay 1×10^{-4} . The optimiser was SGD.

3.2. DIM Template Matching Algorithm

The DIM algorithm has previously been found to produce the best performance for template matching in colour feature space [30]. Hence, it was selected as the underlying

algorithm to determine the best CNN feature space to use for template matching. A detailed description of the DIM algorithm can be found in [30]; for the convenience of the reader, a brief introduction is provided below.

In contrast to other template matching methods that only use the appearance of the target, DIM considers potential distractors, that is, regions that are similar to the matching target. These distractors are represented as additional templates that are cropped from the same image as the given template. All of the templates, representing both the target and the distractors, compete with each other to be matched with the search image. This inference process performs by explaining away [33,38,39]: possible causes (i.e., templates) compete to explain the sensory evidence (i.e., the search image), and if one cause explains part of the evidence (i.e., a part of the image), then support from this evidence for alternative explanations (i.e., other templates) is reduced, or explained away. An example is shown in Figure 1.



Figure 1. An illustration of template competition. The red rectangle area in the left image shows the template used for finding the matching location in the right image. The four same-sized green rectangle areas are the additional templates taken from the left image. These templates compete with each other to be matched to the right image. This competition means that only one template can be the best matching one at each location. The locations matching each template are shown on the right.

DIM minimises the Kullback–Leibler (KL) divergence between the input and a reconstruction of the input created by the additive combination of the templates. This requires the input to be non-negative [30]. Therefore, a pre-processing step is required to separate the positive and rectified negative values of the features directly into two parts, which are then concatenated along the channel dimension:

$$\mathbf{I}_{pre} = \text{concatenate}(\text{ReLU}(\phi(\mathbf{I})), \text{ReLU}(-\phi(\mathbf{I}))) \quad (1)$$

where \mathbf{I} is a colour or grayscale image, $\phi(\mathbf{I})$ are features extracted from the image, and ReLU is the function that, if positive, outputs an element of the input directly, and otherwise outputs zero.

To apply DIM directly to the image feature space, feature extraction was performed as follows:

$$\phi(\mathbf{I}) = \gamma \sum_{c=1}^k (\mathbf{I}_c - \mathbf{I}_c \star f) \quad (2)$$

where c is the index over the number of image channels, k has a value of one for a grayscale image and three for a colour image, γ is a gain factor that was set here to a value of 2, and f is a Gaussian filter with a standard deviation equal to half of the smaller value of the template width or height [30]. This operation results in each channel of $\phi(\mathbf{I})$ being represented by the deviations between the pixel intensity values and the local mean intensity. In this paper, the five VGG19 models were used as feature extractors; hence, $\phi(\mathbf{I})$ represents deep features of \mathbf{I} extracted by the CNNs.

Both the template and the search image were pre-processed as described in the previous paragraph. For the template image, additional templates were extracted around locations where the correlation between the target template and the image was strongest, excluding locations where the additional templates would overlap with each other or

where the bounding box defined the target [30]. Five additional templates were used in the experiments described in this paper. DIM requires the templates to have dimensions that are odd numbers, otherwise the reconstruction of the input does not align with the actual input; see Equation (3) for details. Therefore, if one side of the target template is even, it is padded by one row on the right or one column on the bottom with zeros, then the new size of target template is used to generate the additional templates.

DIM was implemented using the following equations:

$$\mathbf{R}_i = \sum_{j=1}^p (\mathbf{v}_{ji} \star \mathbf{S}_j) \quad (3)$$

$$\mathbf{E}_i = \mathbf{I}pre_i \oslash [\mathbf{R}_i]_{\epsilon_2} \quad (4)$$

$$\mathbf{S}_j \leftarrow [\mathbf{S}_j]_{\epsilon_1} \odot \sum_{i=1}^k (\mathbf{w}_{ji} \ast \mathbf{E}_i) \quad (5)$$

where i is an index over the number of input channels (the maximum index k is twice the channel number of the extracted features); j is an index over p , which is the number of different templates being compared to the image; \mathbf{R}_i is a two-dimensional array representing a reconstruction of $\mathbf{I}pre_i$ (\mathbf{I} , pre-processed using Equation (1)); \mathbf{E}_i is a two-dimensional array representing the discrepancy (or residual error) between $\mathbf{I}pre_i$ and \mathbf{R}_i ; \mathbf{S}_j is a two-dimensional array that represents the similarity between template feature j and the image feature at each pixel; \mathbf{w}_{ji} is a two-dimensional array representing channel i of template j , with the sum of the values in each template \mathbf{w}_j being normalised in order to sum to one; \mathbf{v}_{ji} is another two-dimensional array representing template values (where the values of \mathbf{v}_j were made equal to the corresponding values of \mathbf{w}_j , except that they were normalised to have a maximum value of one); $[\cdot]_{\epsilon} = \max(\cdot, \epsilon)$; ϵ_1 and ϵ_2 are parameters with their values set to $\frac{\epsilon_2}{\max_j(\sum_i v_{ji})}$ and 1×10^{-2} , respectively; \oslash and \odot indicate element-wise division and

multiplication, respectively; and \star and \ast represent the cross-correlation and convolution operations, respectively. All elements of \mathbf{S} were initially set to zero, and Equations (3)–(5) were iteratively updated and terminated after ten iterations for all of the experiments reported in this paper.

For a search image \mathbf{I} , in order to avoid a poor estimate of $\phi(\mathbf{I})$ and edge effects during template matching, when DIM was directly applied to the image feature-space, \mathbf{I} was first padded on all sides with intensity values that were mirror reflections of the image pixel values near the borders of \mathbf{I} . The width of the padding was equal to the width of the template on the left and right borders and equal to the height of the template on the top and bottom borders. The final similarity maps \mathbf{S} were cropped to be the same size as the original image once the template matching method had been applied [30]. When applying DIM to deep feature space, $\phi(\mathbf{I})$ was padded using the same method corresponding to the width and height of the template in deep space, and \mathbf{S} was cropped to be the same size as $\phi(\mathbf{I})$ after application of DIM.

The best matching location can be represented by a single element with the largest value of the similarity map \mathbf{S}_j for template j , as in other template matching methods. However, the best matching location is often represented by a small population of neighbouring elements with high values [30]. Therefore, post-processing was performed to sum the similarity values within neighbourhoods:

$$\mathbf{S}_j = \mathbf{S}_j \star \mathbf{K}_e(\max(1, \alpha \ast w), \max(1, \alpha \ast h)) \quad (6)$$

where \mathbf{K}_e is a binary-valued kernel containing ones within an elliptically shaped region, and has a width and height equal to α times the width w and height h of the template; α was set to 0.025.

4. Results

4.1. Dataset Preparation

The BBS dataset [7] has been widely used for the quantitative evaluation of template matching algorithms [7,21–24]. This dataset contains 105 template–image pairs sampled from 35 videos (three pairs per video) from a tracking dataset [40]. Each template–image pair is taken from frames of the video that are 20 frames apart. To evaluate the performance of a template matching algorithm, the intersection-over-union (IoU) is calculated between the predicted bounding box and the ground truth box for the second image in the pair. The overall accuracy is then determined by calculating the area under the curve (AUC) of a success curve produced by varying the threshold of IoU that counts as success.

Although the BBS data is widely used, it is not particularly good at discriminating the performance of different template matching methods. To illustrate this issue, we applied one baseline method (ZNCC) and three state-of-art methods (BBS, DDIS, and DIM) to the BBS dataset in colour space. The results show that there are 52 template–image pairs where all methods generate very similar results; these can be sub-divided into seven template–image pairs for which all methods fail to match (IoU less than 0.1 for all four methods), 13 template–image pairs for which all methods succeed (IoU greater than 0.8 for all four methods), and 32 template–image pairs for which all methods produce similar intermediate IoU values within 0.1 of each other. This means that only 53 template–image pairs in the BBS dataset help to discriminate the performance of these four template matching methods. These results are summarised in Figure 2.

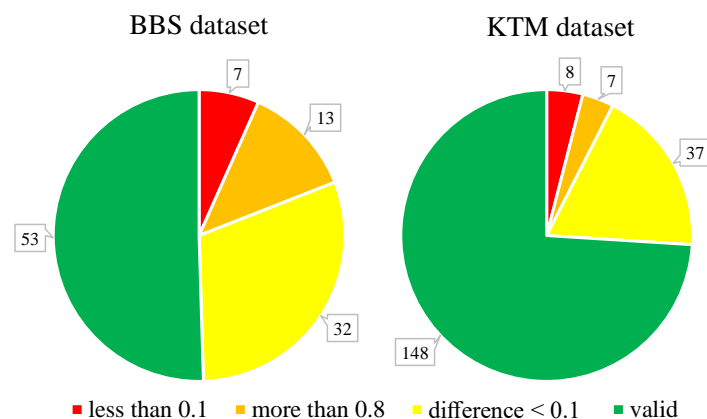


Figure 2. Discriminative ability of two datasets evaluated by comparing the IoU scores produced by ZNCC, BBS, DDIS, and DIM.

We therefore created a new dataset, the King’s Template Matching (KTM) dataset, following a similar procedure to that used to generate the BBS dataset. The new dataset contains 200 template–image pairs sampled from 40 new videos (five pairs per video) selected from a different tracking dataset [41]. In contrast to the BBS dataset, the template and the image were chosen manually in order to avoid pairs that contain significant occlusions and non-rigid deformations of the target (which no method is likely to match successfully), and the image pairs were separated by 30 (rather than 20) frames in order to reduce the number of pairs for which matching would be easy for all methods. These changes make the new data more challenging and provide a far larger number of image pairs that can discriminate the performance of different methods, as shown in Figure 2. Both the new dataset and the BBS dataset were used in the following experiments.

4.2. Template Matching Using Features from Individual Convolutional Layers

To reveal how the shape bias affects template matching, we calculated the AUC using DIM with features from every single convolutional layer of the five models. As the features from the later convolutional layers are down-sampled using max-pooling (by a factor of

$\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, and $\frac{1}{16}$ compared to the original image), the bounding box of the template was multiplied by the same scaling factor and the resulting similarity map is resized back to the original image size in order to make the prediction. The AUC scores across the BBS and KTM datasets are summarised in Figure 3, and the mean and standard deviation of these AUC scores are summarised in Figure 4.

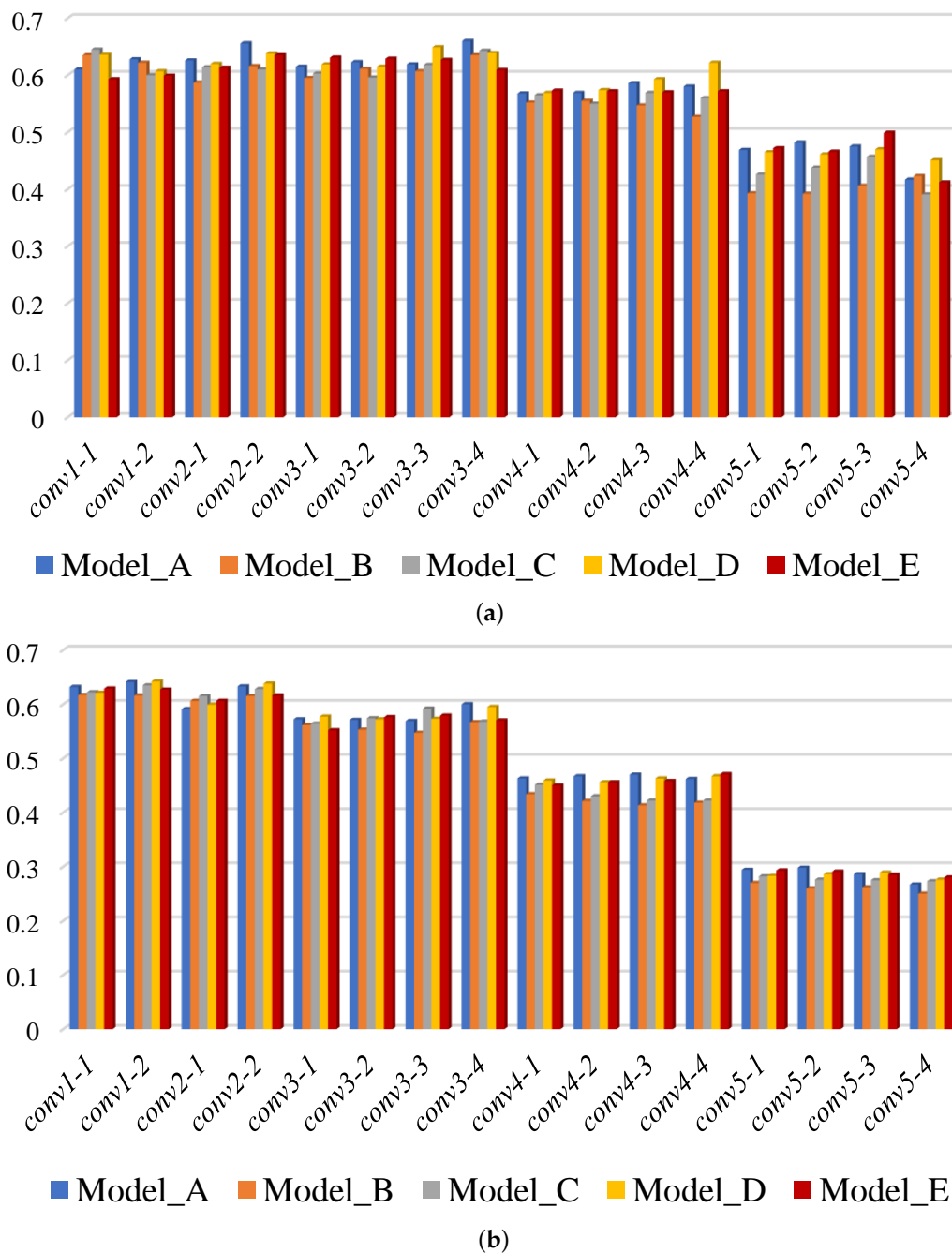


Figure 3. The AUC scores of DIM using features from different convolutional layers of five models: (a) evaluation on BBS dataset and (b) evaluation on KTM dataset.

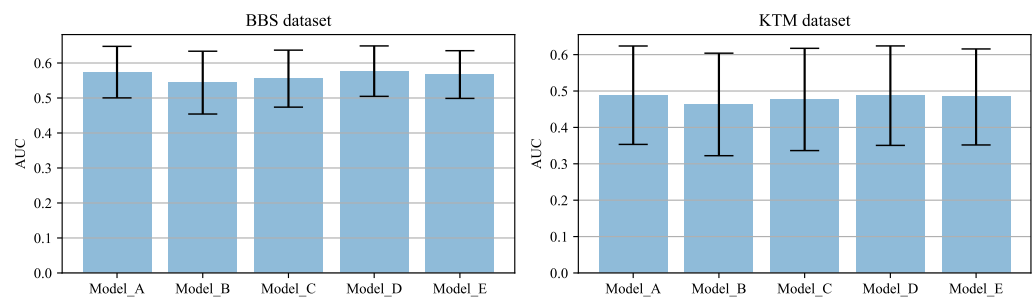


Figure 4. Mean and standard deviation of the AUC scores using features from different convolutional layers of five models.

For all five models, there is a tendency for the AUC to be higher when template matching is performed using lower layers of the CNN compared to later layers. This suggests that template matching relies more on low-level visual attributes such as texture, rather than higher-level ones such as shape. Among the four models trained with stylised samples, the AUC score for most CNN layers is greater for Model_D than Model_E, greater for Model_E than Model_C, and greater for Model_C than Model_B. This tendency, which can be clearly seen in Figure 4, suggests that template matching relies more on texture features than shape features. Comparing Model_A and Model_D, it is hard to say which is better. However, the AUC score calculated on the BBS dataset using features from *conv4_4* of Model_D is noticeably better than that for Model_A. This suggests that increasing the shape bias of later layers of the CNN could potentially lead to better template matching. However, this result is not reflected by the results for the KTM dataset. One possible explanation is that, in general, the templates in the KTM dataset are smaller than those in the BBS dataset (with the template size defined in terms of area, i.e., as the product of its width and height; the mean template size of for the KTM dataset is 1603 pixels², whereas it is 3442 pixels² for the BBS dataset). Smaller templates tend to be less discriminative. The sub-sampling that occurs in later levels of the CNN results in templates that are even smaller and less discriminative. This may account for the worse performance of the later layers of each CNN when tested using the KTM dataset rather than the BBS dataset. This represents a confounding factor in attributing the better performance of the early layers to a reliance on texture information.

In order to illustrate the differences in the features learned by Model_A and Model_D, the first three principal components of *conv4_4* were converted to RGB values. As shown in Figure 5, the features from Model_D contain more information about edges (shape) than those from Model_A. However, it is hard to distinguish the small object in fourth row, as it is represented by a very small region of the feature space.

4.3. Template Matching Using Features from Multiple Convolutional Layers

We compared Model_A, Model_D, and Model_E by applying the DIM template matching algorithm to features extracted from multiple convolutional layers of each CNN. In order to combine feature maps with different sizes, bilinear interpolation was used to make them the same size. If the template was small (height times width less than 4000) the feature maps from the later layer(s) were scaled to be the same size as those in the earlier layer(s). If the template was large, the feature maps from the earlier layer(s) were reduced in size to be the same size as those in the later layer(s). To maintain a balance between low and high level features, the dimension of the feature maps from the later layer(s) was reduced by PCA to the same number as in the earlier layer.

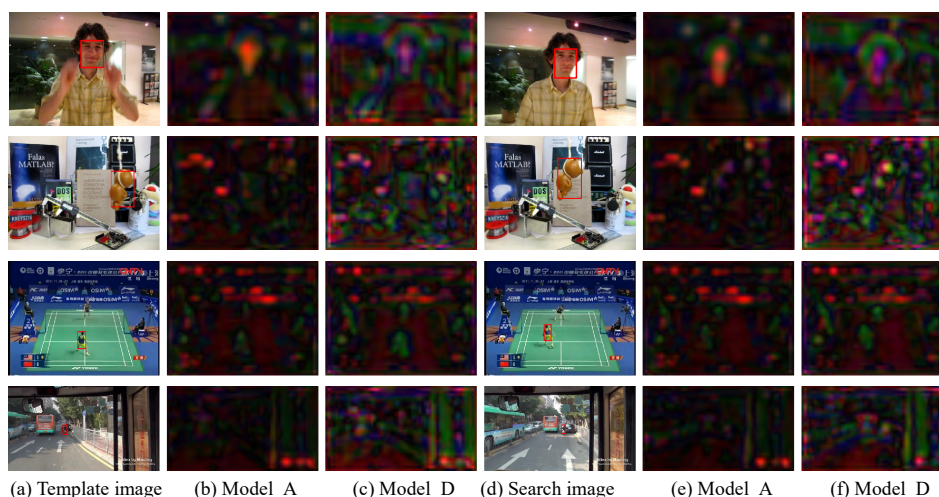


Figure 5. Visualisation of *conv4_4* layers for image pairs with different template sizes: (a) the first image in the pair shows the bounding box defining the template; (b,c) visualisations of the corresponding features produced in layer *conv4_4* of (b) Model_A and (c) Model_D; (d) the second image in the pair shows the bounding box of the ground truth location of the target object; (e,f) visualisations of the corresponding features produced in layer *conv4_4* of Model_A and Model_D, respectively.

Table 2 shows the AUC scores produced by DIM using features from two convolutional layers of Model_A, Model_D, and Model_E. All possible combinations of two layers were tested; the table shows only selected results with the best performance. It can be seen from Table 2 that for the 24-layer combinations for which results are shown, 21 results for both BBS and KTM dataset are better for Model_D than for Model_A, and 14 results for BBS dataset and 13 results for KTM dataset are better for Model_E for Model_A. Hence, both networks with more shape bias perform better than the network with the least shape bias. These results thus support the conclusion that more discriminative features can be obtained by increasing the shape bias of the VGG19 model, which increases the performance of template matching.

The results for Model_D are better than those for Model_E for 17 of the 24 layer combinations for the BBS dataset and for 18 of the 24 layer combinations for the KTM dataset. Furthermore, the best result for each dataset (indicated in bold) is generated using the features from Model_D. Hence, among the three models, Model_D produced best performance. To determine whether fusing features from more layers would further improve template matching performance, DIM was applied to all combinations of three layers from Model_D, resulting in a total of 560 different combinations using three layers. As it is impossible to show all these results in this paper, the highest ten AUC scores are shown in Table 3. For both datasets, using three layers produced an improvement in the best AUC score (around 0.01) compared to using two layers.

Table 2. Partial AUC scores of DIM using features from two convolutional layers of Model_A (upper value in each cell), Model_D (middle value in each cell), and Model_E (lower value in each cell). The up and down arrows indicate whether the AUC score of Model_D or Model_E is better or worse than that of Model_A.

(a) Evaluation on BBS dataset.						
AUC \ Layer	conv3_3	conv3_4	conv4_1	conv4_2	conv4_3	conv4_4
conv1_1	0.710	0.705	0.713	0.697	0.698	0.711
	0.707↓	0.714↑	0.704↓	0.718 ↑	0.710↑	0.708↓
	0.692↓	0.673↓	0.700↓	0.711↑	0.697↓	0.694↓
conv1_2	0.686	0.686	0.674	0.655	0.680	0.683
	0.686	0.687↑	0.707↑	0.696↑	0.690↑	0.710↑
	0.658↓	0.659↓	0.675↑	0.695↑	0.683↑	0.677↓
conv2_1	0.658	0.670	0.664	0.653	0.662	0.667
	0.659↑	0.669↓	0.665↑	0.671↑	0.683↑	0.693↑
	0.686↑	0.663↓	0.666↑	0.690↑	0.687↑	0.687↑
conv2_2	0.659	0.661	0.653	0.641	0.659	0.663
	0.665↑	0.667↑	0.676↑	0.679↑	0.676↑	0.682↑
	0.682↑	0.653↓	0.688↑	0.684↑	0.672↑	0.691↑

(b) Evaluation on KTM dataset.						
AUC \ Layer	conv3_3	conv3_4	conv4_1	conv4_2	conv4_3	conv4_4
conv1_1	0.687	0.684	0.677	0.668	0.670	0.678
	0.689↑	0.691↑	0.682↑	0.695↑	0.684↑	0.687↑
	0.672↓	0.676↓	0.664↓	0.672↑	0.675↑	0.678
conv1_2	0.687	0.689	0.682	0.685	0.675	0.682
	0.680↓	0.694↑	0.695↑	0.697 ↑	0.691↑	0.690↑
	0.668↑	0.672↓	0.670↓	0.669↓	0.669↓	0.673↓
conv2_1	0.634	0.633	0.647	0.645	0.655	0.639
	0.642↑	0.651↑	0.665↑	0.671↑	0.668↑	0.666↑
	0.658↑	0.673↑	0.643↓	0.648↑	0.665↑	0.660↑
conv2_2	0.642	0.651	0.664	0.661	0.669	0.669
	0.657↑	0.664↑	0.670↑	0.673↑	0.669	0.669
	0.657↑	0.665↑	0.640↑	0.651↓	0.665↓	0.670↑

Table 3. Best ten results when using combinations of features from three convolutional layers of Model_D; here, for instance, C12₄₄⁴¹ means that features from conv1_2, conv4_1, and conv4_4 were fused.

(a) Evaluation on BBS dataset.										
Layers	C12 ₄₄ ⁴¹	C11 ₄₃ ³⁴	C11 ₄₄ ⁴²	C12 ₅₂ ⁴³	C11 ₄₃ ²²	C11 ₄₄ ⁴¹	C11 ₄₃ ⁴¹	C11 ₄₂ ³⁴	C12 ₄₄ ⁴³	C11 ₄₄ ³⁴
AUC	0.728	0.727	0.724	0.724	0.723	0.722	0.720	0.720	0.720	0.720

(b) Evaluation on KTM dataset.										
Layers	C11 ₄₂ ³⁴	C12 ₄₃ ³²	C11 ₄₃ ³⁴	C12 ₄₂ ²²	C12 ₄₂ ³¹	C12 ₄₂ ³⁴	C12 ₄₃ ³⁴	C12 ₄₃ ³¹	C12 ₄₃ ³³	C11 ₄₂ ³³
AUC	0.711	0.709	0.708	0.706	0.706	0.705	0.705	0.705	0.705	0.704

4.4. Comparison with Other Methods

This section compares our results with those produced by other template matching methods in both colour and deep feature space. When evaluated on the BBS dataset, the deep features used by each template matching algorithm were the features from layers conv1_2, conv4_1, and conv4_4 of Model_D. When evaluated on the KTM dataset, the deep features used as the input to each algorithm were those from layers conv1_1, conv3_4,

and *conv4_2* of Model_D. BBS, CoTM, and QATM have been tested on BBS data by their authors using different deep features, and thus we compared our results to these earlier published results as well.

The comparison results are summarised in Table 4, and examples of the results for particular images are shown in Figure 6. All methods except QATM and BBS produce improved results using the proposed deep features than when using colour features. This is true for both datasets. Of the methods that have previously been applied to deep features, the performance of two (NCC and QATM) are improved, while that of two others (BBS and CoTM) is made worse when using our proposed method to define the deep feature space. Potential further improvements to the performance of these methods could be achieved by optimising the feature extraction method for the individual template matching algorithm, as has been done here for DIM. However, it should be noted that simple metrics for comparing image patches such as NCC and ZNCC produce close to state-of-the-art performance when applied to our proposed deep feature space, outperforming much more complex methods of template matching such as BBS, CoTM, and QATM when these methods are applied to any of the tested feature spaces, including those proposed by the authors of these algorithms.



Figure 6. Comparison performance of different methods using the proposed feature extraction method. The colour boxes indicate the template location defined by: — Ground truth — BBS — CoTM — DDIS — DIM — NCC — QATM — SSD — ZNCC. Note that the colour images have been converted to greyscale in order to make the bounding boxes more visible. The boxes predicted by ZNCC and NCC are remarkably similar, and thus overlap.

ZNCC and NCC produce very similar scores on both datasets. ZNCC is similar to NCC, with the only difference being the subtraction of the local mean value from the feature vectors being compared. This operation makes ZNCC more robust to changes of lighting conditions when applied directly to colour images, and for this reason the AUC score of ZNCC on both datasets is higher than that of NCC in colour space. The features extracted by the CNN appear to be insensitive to lighting changes; therefore, the results of NCC and ZNCC are remarkably similar when applied to these features.

Table 4. Quantitative comparison of the performance of different template matching algorithms using different input features.

AUC \ Method	Feature	BBS Dataset			KTM Dataset	
		Colour	Deep	Deep (Proposed)	Colour	Deep (Proposed)
SSD		0.46	-	0.54	0.42	0.54
NCC		0.48	0.63 [23]	0.67	0.42	0.67
ZNCC		0.54	-	0.67	0.48	0.67
BBS		0.55	0.60 [7]	0.54	0.44	0.55
CoTM		0.54 ¹	0.67 [22]	0.64	0.51	0.56
DDIS		0.64	-	0.66	0.63	0.68
QATM		-	0.62 ²	0.66	-	0.64
DIM		0.69	-	0.73	0.60	0.71

¹ We were unable to reproduce this result using the code provided by the authors of CoTM. Our different result is shown in the table. ² The authors of QATM report an AUC score of 0.69 when this method is applied to the BBS dataset [21]. However, examining their source code, we note that this result is produced by setting the size of the predicted bounding box as equal in size to the width and height of the ground truth bounding box. Other methods are evaluated by setting the size of the predicted bounding box equal to the size of the template (i.e., without using knowledge of the ground truth that the algorithm is attempting to predict). We have re-tested QATM using the standard evaluation protocol and our result for the original version of QATM is 0.62. As QATM is designed to work specifically with a CNN, it was not applied directly to colour images.

One known weakness of BBS is that it may fail when the template is very small compared to target image [7]. This may explain the particularly poor results of this method when applied to the KTM dataset.

DIM achieves the best results on both datasets when applied to deep features. DIM performs particularly well on the BBS dataset, producing an AUC of 0.73, which, as far as we are aware, makes it the only method to have scored more than 0.7 on this dataset. The DIM algorithm produces state-of-the-art performance on the KTM dataset when applied to deep features. When applied to colour features, the results are good, although not as good as DDIS on the KTM dataset. This is because small templates in the KTM dataset may contain insufficient detail for the DIM algorithm to successfully distinguish the target object. Using deep features enhances the discriminatory ability of small templates enough that the performance of DIM increases significantly. These results demonstrate that the proposed approach is effective at extracting distinguishable features, which lead to robust and accurate template matching.

5. Discussion

The experiments described above demonstrate that template matching relies more on low-level visual attributes such as texture than higher-level attributes such as shape. However, it is clear that slightly increasing the shape bias of a CNN by changing the method of training the network and then combining the outputs of a range of convolutional layers produces a feature space in which template matching can be achieved with greater accuracy. This is because the combination of low-level features that can accurately locate the target with high-level features that are more tolerant to appearance changes enables more robust recognition and localisation of the target object.

6. Conclusions

Our results demonstrate that slightly increasing the shape bias of a CNN by changing the method used to training the network can produce more distinguishable features, allowing template matching to be achieved with greater accuracy. By running a large number of experiments using shape-biased VGG19 architectures, we determined the best

combination of convolutional features on which to perform template matching with the DIM algorithm. This same feature space was shown to improve the performance of most other template matching algorithms as well. When applied to our new feature space, the DIM algorithm was able to produce state-of-art results on two benchmark datasets.

Author Contributions: Conceptualization, B.G. and M.W.S.; methodology, M.W.S.; software, B.G.; validation, B.G.; formal analysis, B.G.; investigation, B.G.; resources, B.G.; data curation, B.G.; Writing—Original draft preparation, B.G.; Writing—Review and editing, B.G. and M.W.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the China Scholarship Council (CSC).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: As part of this study we created a new benchmark, KTM, for template matching. This dataset, along with the code used in the proposed method, can be found at: <https://github.com/iminfine/Deep-DIM> (accessed on 1 June 2022).

Acknowledgments: The authors acknowledge use of the research computing facility at King’s College London, Rosalind (<https://rosalind.kcl.ac.uk> (accessed on 1 June 2022)) as well as the Joint Academic Data science Endeavour (JADE) facility. Reprinted by permission from Springer with license number 534007092188 for the paper: Gao, B.; Spratling, M.W., Robust Template Matching via Hierarchical Convolutional Features from a Shape-Biased CNN, in Proceedings of the International Conference on Image, Vision, and Intelligent Systems (ICIVIS 2021), Changsha, China, 15–17 June 2021, pp. 333–344.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gao, B.; Spratling, M.W. Explaining away results in more robust visual tracking. *Vis. Comput.* **2022**, 1–15. [\[CrossRef\]](#)
2. Gao, B.; Spratling, M.W. More Robust Object Tracking via Shape and Motion Cue Integration. *Signal Process.* **2022**, *22*, 108628. [\[CrossRef\]](#)
3. Ahuja, K.; Tuli, P. Object recognition by template matching using correlations and phase angle method. *Int. J. Adv. Res. Comput. Commun. Eng.* **2013**, *2*, 1368–1373.
4. Dai, J.; Li, Y.; He, K.; Sun, J. R-fcn: Object detection via region-based fully convolutional networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 379–387.
5. Scharstein, D.; Szeliski, R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vis.* **2002**, *47*, 7–42. [\[CrossRef\]](#)
6. Chhatkuli, A.; Pizarro, D.; Bartoli, A. Stable template-based isometric 3D reconstruction in all imaging conditions by linear least-squares. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 23–28 June 2014; pp. 708–715.
7. Oron, S.; Dekel, T.; Xue, T.; Freeman, W.T.; Avidan, S. Best-buddies similarity—Robust template matching using mutual nearest neighbors. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 1799–1813. [\[CrossRef\]](#)
8. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [\[CrossRef\]](#)
9. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.
10. Dou, J.; Qin, Q.; Tu, Z. Robust image matching based on the information of SIFT. *Optik* **2018**, *171*, 850–861. [\[CrossRef\]](#)
11. Lee, H.; Kwon, H.; Robinson, R.M.; Nothwang, W.D. DTM: Deformable template matching. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016; pp. 1966–1970.
12. Sibiryakov, A. Fast and high-performance template matching method. In Proceedings of the CVPR 2011, Washington, DC, USA, 20–25 June 2011; pp. 1417–1424.
13. Arslan, O.; Demirci, B.; Altun, H.; Tunaboylu, N.S. A novel rotation-invariant template matching based on HOG and AMDF for industrial laser cutting applications. In Proceedings of the 2013 9th International Symposium on Mechatronics and Its Applications (ISMA), Amman, Jordan, 9–11 April 2013; pp. 1–5.
14. Antipov, G.; Berrani, S.A.; Ruchaud, N.; Dugelay, J.L. Learned vs. hand-crafted features for pedestrian gender recognition. In Proceedings of the 23rd ACM international Conference on Multimedia, Brisbane, Australia, 26–30 October 2015; pp. 1263–1266.
15. Chan, T.H.; Jia, K.; Gao, S.; Lu, J.; Zeng, Z.; Ma, Y. PCANet: A simple deep learning baseline for image classification? *IEEE Trans. Image Process.* **2015**, *24*, 5017–5032. [\[CrossRef\]](#)

16. Wang, F.; Jiang, M.; Qian, C.; Yang, S.; Li, C.; Zhang, H.; Wang, X.; Tang, X. Residual attention network for image classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3156–3164.
17. Liang, M.; Hu, X. Recurrent convolutional neural network for object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3367–3375.
18. Wohlhart, P.; Lepetit, V. Learning descriptors for object recognition and 3d pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3109–3118.
19. Bertinetto, L.; Valmadre, J.; Henriques, J.F.; Vedaldi, A.; Torr, P.H. Fully-convolutional siamese networks for object tracking. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Amsterdam, The Netherlands, 2016; pp. 850–865.
20. Ma, C.; Huang, J.B.; Yang, X.; Yang, M.H. Robust visual tracking via hierarchical convolutional features. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 2709–2723. [[CrossRef](#)]
21. Cheng, J.; Wu, Y.; AbdAlmageed, W.; Natarajan, P. QATM: Quality-Aware Template Matching For Deep Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 11553–11562.
22. Kat, R.; Jevnisek, R.; Avidan, S. Matching pixels using co-occurrence statistics. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1751–1759.
23. Kim, J.; Kim, J.; Choi, S.; Hasan, M.A.; Kim, C. Robust template matching using scale-adaptive deep convolutional features. In Proceedings of the 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Kuala Lumpur, Malaysia, 12–15 December 2017; pp. 708–711.
24. Talmi, I.; Mechrez, R.; Zelnik-Manor, L. Template matching with deformable diversity similarity. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 175–183.
25. Zhang, Z.; Yang, X.; Gao, H. Weighted smallest deformation similarity for NN-based template matching. *IEEE Trans. Ind. Inform.* **2020**, *16*, 6787–6795. [[CrossRef](#)]
26. Lai, J.; Lei, L.; Deng, K.; Yan, R.; Ruan, Y.; Jinyun, Z. Fast and robust template matching with majority neighbour similarity and annulus projection transformation. *Pattern Recognit.* **2020**, *98*, 107029. [[CrossRef](#)]
27. Kriegeskorte, N. Deep neural networks: A new framework for modeling biological vision and brain information processing. *Annu. Rev. Vis. Sci.* **2015**, *1*, 417–446. [[CrossRef](#)] [[PubMed](#)]
28. Geirhos, R.; Rubisch, P.; Michaelis, C.; Bethge, M.; Wichmann, F.A.; Brendel, W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv* **2018**, arXiv:1811.12231.
29. Li, Y.; Yu, Q.; Tan, M.; Mei, J.; Tang, P.; Shen, W.; Yuille, A.; Xie, C. Shape-Texture Debiased Neural Network Training. *arXiv* **2020**, arXiv:2010.05981.
30. Spratling, M.W. Explaining away results in accurate and tolerant template matching. *Pattern Recognit.* **2020**, *104*, 107337. [[CrossRef](#)]
31. Gao, B.; Spratling, M.W. Robust Template Matching via Hierarchical Convolutional Features from a Shape Biased CNN. In Proceedings of the The International Conference on Image, Vision and Intelligent Systems (ICIVIS 2021), Changsha, China, 15–17 June 2021; pp. 333–344.
32. Korman, S.; Milam, M.; Soatto, S. OATM: Occlusion aware template matching by consensus set maximization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2675–2683.
33. Kersten, D.; Mamassian, P.; Yuille, A. Object perception as Bayesian inference. *Annu. Rev. Psychol.* **2004**, *55*, 271–304. [[CrossRef](#)]
34. Spratling, M.W. Unsupervised learning of generative and discriminative weights encoding elementary image components in a predictive coding model of cortical function. *Neural Comput.* **2012**, *24*, 60–103. [[CrossRef](#)]
35. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
36. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
37. Huang, X.; Belongie, S. Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE International Conference on Computer Vision, Honolulu, HI, USA, 21–26 July 2017; pp. 1501–1510.
38. Lochmann, T.; Deneve, S. Neural processing as causal inference. *Curr. Opin. Neurobiol.* **2011**, *21*, 774–781. [[CrossRef](#)]
39. Lochmann, T.; Ernst, U.A.; Deneve, S. Perceptual inference predicts contextual modulations of sensory responses. *J. Neurosci.* **2012**, *32*, 4179–4195. [[CrossRef](#)]
40. Wu, Y.; Lim, J.; Yang, M.H. Object tracking benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1834–1848. [[CrossRef](#)] [[PubMed](#)]
41. Liang, P.; Blasch, E.; Ling, H. Encoding color information for visual tracking: Algorithms and benchmark. *IEEE Trans. Image Process.* **2015**, *24*, 5630–5644. [[CrossRef](#)] [[PubMed](#)]