mathematics

MDPI

Article

Enhancement: SiamFC Tracker Algorithm Performance Based on Convolutional Hyperparameters Optimization and Low Pass Filter

Rogeany Kanza *, Yu Zhao, Zhilin Huang, Chenyu Huang and Zhuoming Li *

Harbin Institute of Technology, School of Electronics and Information Engineering, Harbin 150001, China; happymelody1996@gmail.com (Y.Z.); zerinhwang03@pku.edu.cn (Z.H.); 21S105182@stu.hit.edu.cn (C.H.)
* Correspondence: rkanza@hit.edu.cn (R.K.); zhuoming@hit.edu.cn (Z.L.)

Abstract: Over the past few decades, convolutional neural networks (CNNs) have achieved outstanding results in addressing a broad scope of computer vision problems. Despite these improvements, fully convolutional Siamese neural networks (FCSNN) still hardly adapt to complex scenes, such as appearance change, scale change, similar objects interference, etc. The present study focuses on an enhanced FCSNN based on convolutional block hyperparameters optimization, a new activation function (ModReLU) and Gaussian low pass filter. The optimization of hyperparameters is an important task, as it has a crucial ascendancy on the tracking process performance, especially when it comes to the initialization of weights and bias. They have to work efficiently with the following activation function layer. Inadequate initialization can result in vanishing or exploding gradients. In the first method, we propose an optimization strategy for initializing weights and bias in the convolutional block to ameliorate the learning of features so that each neuron learns as much as possible. Next, the activation function normalizes the output. We implement the convolutional block hyperparameters optimization by setting the convolutional weights initialization to constant, the bias initialization to zero and the Leaky ReLU activation function at the output. In the second method, we propose a new activation, ModReLU, in the activation layer of CNN. Additionally, we also introduce a Gaussian low pass filter to minimize image noise and improve the structures of images at distinct scales. Moreover, we add a pixel-domain-based color adjustment implementation to enhance the capacity of the proposed strategies. The proposed implementations handle better rotation, moving, occlusion and appearance change problems and improve tracking speed. Our experimental results clearly show a significant improvement in the overall performance compared to the original SiamFC tracker. The first proposed technique of this work surpasses the original fully convolutional Siamese networks (SiamFC) on the VOT 2016 dataset with an increase of 15.42% in precision, 16.79% in AUPC and 15.93% in IOU compared to the original SiamFC. Our second proposed technique also reveals remarkable advances over the original SiamFC with 18.07% precision increment, 17.01% AUPC improvement and an increase of 15.87% in IOU. We evaluate our methods on the Visual Object Tracking (VOT) Challenge 2016 dataset, and they both outperform the original SiamFC tracker performance and many other top performers.

Keywords: activation function; fully convolutional Siamese neural network; Gaussian low pass filter; hyperparameters; initialization; optimization; tracking performance

MSC: 68T07

check for updates

Citation: Kanza, R.; Zhao, Y.; Huang, Z.; Huang, C.; Li, Z. Enhancement: SiamFC Tracker Algorithm Performance Based on Convolutional Hyperparameters Optimization and Low Pass Filter. *Mathematics* **2022**, *10*, 1527. https://doi.org/10.3390/math10091527

Academic Editor: Marina Alexandra Pedro Andrade

Received: 11 March 2022
Accepted: 12 April 2022
Published: 3 May 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

In recent years, multiple improvements have been realized in deep learning (DL) in recent benchmarks [1,2]. Computer vision is one of the DL applications wherein a tremendous amount of work has been accomplished, leading to great progress in the computer science area. Object tracking is a key topic in machine vision that aims to

detect the position of a moving object in a video scene [3–5]. The target tracking process accurately follows a target of interest throughout a video scene. In the first frame, a quadrilateral with four right angles illustrates the target. Given the target's position on the first frame (e.g., via a bounding box) and a video scene, the algorithm must steadily designate the target's position in the following sequential images [5]. For so many years, the most suitable pattern archetype for this task has been to learn an online model of the target's appearance using patterns extracted from the processed video [6]. Unfortunately, the disadvantage is that only basic patterns can be learned in this manner. In addition, performing well in this archetype requires a large volume of data. The lack of enough supervised data sets is a limitation to this model. Fortunately, transfer learning has come to overcome this limitation. It consists of using a model trained on large datasets for one task in another related task for which datasets are relatively smaller. This approach uses pretrained models in new tasks as the first or starting part of the network. Among the existing networks are Siamese networks. Siamese neural networks have become popular in object tracking owing to their unique two inputs and correlation measurement. The first input is a preselected exemplar image. The second input is a search image, in which Siamese network's task is to find the exemplar image inside the search image. In general, Siamese architectures are employed to address similarity learning [7–9]. They can generate a map of similarity scores by measuring the similarity between an exemplar image and each region of the search image. Additionally, because SiamFC algorithms have a simple network and can be trained offline on a large dataset, they have piqued the interest of many researchers. [10–15]. Siamese networks can use different CNN backbone networks, such as AlexNet, ResNet, PyramidNet, etc. These architectures can be modified and fine-tuned according to the desired network configuration. For example, Bruno et al. [16] consider a method based on the integration of scale-invariant feature transform key points and transfer learning with pretrained CNNs, such as AlexNet and PyramidNet, for the detection of suspicious regions in mammograms. They first fine-tuned the pretrained AlexNet and PyramidNet CNN architectures, then compared their performances to choose the best CNN for their study. Subsequently, the performance comparison revealed the superiority of PyramidNet over AlexNet in the specific task. Therefore, PyramidNet was further used as the CNN for their solution. Some recent studies utilized even deeper CNN architectures to improve the tracking performance of Siamese networks, as in Ref. [17]. Despite their robustness, there are still relevant issues to which existing algorithms are unable to adapt well. Similar background interference, appearance changes, object shape, light conditions (scene contrast, weather, etc.), object cover, scene occlusion, scene clutter, motion smoothness and motion coherence figure among them. In the tracking process, the above-mentioned issues have a considerable impact on the performance of the trackers. For example, one of the algorithm's performance metrics that is often affected by the change of object scale is the intersection over union metric (IoU). The idea of implementing hyperparameters optimization or tuning in the convolutional layer is motivated by the importance and the impact the initialization parameters can have on the output of the neural network and, more generally, on the overall performance. As we know, various initializations lead to different results, and a poor initialization may yield unsatisfying gradients, which also slows down the optimization algorithm. So, setting a powerful and suitable initialization is a determining step in a tracking algorithm. Pixel-domain algorithms are distinguished by their high accuracy and computational complexity. However, they may be limited in use in real-time processing scenarios of several video bitstreams because of their high computational complexity [18–21]. As a result, the idea of introducing an appropriate initialization becomes relevant to achieve satisfactory results in real-time processing cases. Furthermore, for a convolutional neural network, the activation function is a fundamental element, as it plays an important role in normalizing the output of neurons based on the prior initialization. The activation function activates the features of neurons to address nonlinear problems. It is used to improve a convolutional neural network's expression ability, allowing the neural network to accurately possess the significance of

artificial intelligence [22]. An appropriate activation function can map data in dimensions more effectively [23,24]. If a CNN has linear characteristics, the function's linear equation can only be expressed linearly, rendering the multilayer perceptron meaningless. The ReLU function is one of the best activation functions available. Unfortunately, there are some disadvantages to it as well. The ReLU function is non-differentiable at zero, and this results in dead neurons. Therefore, in order to address this issue, add more diversity to the architecture and optimize the CNN block activation layer, we propose a new activation function based on ReLU function. While enforcing non-saturated nonlinearity, our proposed activation function keeps positive and negative information. Since we are in a pixel domain, the image structures and color are also important parameters to consider. Images can contain undesired details and variations in brightness or color. These random imperfections affect the image quality. Thus, this motivates the initiation of another approach that consists of applying a Gaussian low pass filter to reduce noise and details in images in the dataset. Low pass filters are commonly used to remove high-frequency noise from images. They use a moving window operator that moves over the images, affecting all the pixels inside them. During the process, the pixels are affected one by one.

The main contributions of this paper are the following:

- We propose an optimization strategy for the CNN block through the use of a simple but effective initialization and activation function in the first method.
- A new activation function (ModReLU) based on ReLU function is proposed to optimize the outputs of the CNN and improve the tracking precision in the second method.
- Introducing a low pass filter for noise and details reduction in the second method.
- The first proposed method surpasses the original fully convolutional Siamese networks (SiamFC) tracker performance with an increase of 15.42% in precision, 16.79% in AUPC, 15.93% in IOU.
- The second proposed technique also reveals remarkable advances over the original SiamFC with 18.07% precision increment, 17.01% AUPC improvement and an increase of 15.87% in IOU.
- Furthermore, both proposed techniques surpass other popular algorithms and top performers in relation to precision and speed.

The outline of our study is structured as follows. Section 2 introduces a summary of related work, while the proposed methods are described in two sections in Section 3. Section 4 focuses on the experimental results and the discussion of our methods. Next, we compare the obtained results with the original SiamFC [25] and other popular top performers. Finally, the conclusion is reported in Section 7.

2. Related Work

Over the past years, owing to their strong abstract feature representation, CNNs [26–30] have drawn much attention in the field of computer vision. For the last few years, target tracking has been one of the crucial areas in machine vision to accomplish impressive progress. A great amount of work has been accomplished in the sphere of visual object tracking. As an illustration, Hossein Kashiani et al. [31] introduced a new approach that outperforms avant-garde trackers in the context of performance. In their work, they introduced a powerful tracking algorithm that tackles both the motion and observation models at the same time. The motion estimation network (MEN) is used in the motion model to sample the most likely candidates. Next, the Siamese network is trained offline to detect the best candidates amid all the patterns. Each pattern is evaluated using an adaptable buffer with the best prior selected patterns due to the fact that the object appearance changes during the tracking process. Limiting the buffer updating to a formerly defined function allows effectively dealing with occlusion. Furthermore, a weighting CNN (WCNN) is used to enhance the tracker's robustness as to address the coexistence of similar objects and substantial appearance changes. This weighting network uses sequence-specific information to down-weight the confusing patterns.

On the other side, SiamFC is a kind of algorithm that is more balanced in tracking robustness and real time [32]. It opens up a new branch of target tracking algorithm. Many subsequent papers [33,34] have been improved on the basis of SiamFC algorithm. CFNet [35] converts the convolution layer of SiamFC for similarity matching into a differentiable layer, which can be updated by back propagation in the training stage. Guo et al. put forward DSiam [36]. Its starting point is that the objective in real life, especially the appearance of non-rigid objects, will change significantly over time, while the original SiamFC [25] algorithm does not update feature templates online, which may lead to the subsequent loss of targets. DSiam updates the feature template of the target and suppresses the interference from the background by learning the changes of the appearance and background of the target, thus improving the tracking accuracy. SA-Siam [37] trained two twin networks for tracking to extract the semantic and external features of the object separately and merged both branches in tracking, which improved the generalization capability of SiamFC algorithm. SiamAttn [38] introduces a novel attention mechanism and a region fine-tuning module for better tracking precision. Yan et al. [39] designed an effective anti-interference module to optimize the algorithm's discrimination capability. In their study, to extract information from the candidate target files provided by the main network of the SiamFC, the anti-interference module employs another feature extraction network. Furthermore, the feature representation set saves the feature vectors of the template image and the tracking object. Then, the tracking object is estimated by calculating the cosine distance between the candidate target's feature vector and the vector in the feature vector set. SCRPN-CISA [40] uses three attention mechanisms and a cascaded region proposal networks architecture to boost flexibility, discrimination capability and feature embeddings. VGG-Net-D network serves as an anchor to improve the feature extraction ability. Next, the authors developed a channel-interconnection-spatial attention module to enhance the discrimination capability and the flexibility of the algorithm, and a deconvolution adjustment block to combine cross-layer attributes. Following that, the authors propose a three-layer cascaded region proposal network to obtain the foreground-background classification and a screening method to improve the tracking accuracy. Bertinetto et al. [25,35] advocate an alternative approach on which our work relies, except for the training part. They propose a fully convolutional Siamese architecture relative to the pattern image x. They consider a function to be fully convolutional if it commutes beside translation. To give a more precise explanation, introducing L_τ to denote the translation operator in Equation (1)

$$(L_\tau x)[u] = x[u - \tau] \tag{1}$$

a function h that maps signals to signals is fully convolutional with stride k if

$$h(L_{k\tau} x) = L_\tau h(x) \tag{2}$$

for a translation τ. When x is a finite signal, this only needs a hold over the valid region of the output. They also emphasize the benefit of a fully convolutional neural network (FCNN) that allows a wider search image as input to the FCNN instead of a pattern image with an identical size. Additionally, the analogy with all translated sub-windows will be computed on a dense grid in one evaluation. In order to accomplish this, they employ a convolutional embedding function and merge the consequent feature maps with a cross-correlation layer, such as

$$f(z, x) = \varphi(z) \times \varphi(x) + b \tag{3}$$

where b represents a signal. Their paper's main contribution is that it shows that their method yields a state-of-the-art performance in recent tracking benchmarks at very high speeds, which exceed the requirement of the frame rate. They train a neural network based on Siamese architecture to find an exemplar image in a wider search image. The Bertinetto et al. [25] fully convolutional Siamese architecture is depicted in Figure 1 in relation to the candidate image x. An efficient and dense sliding-window assessment is

accomplished through the use of a bilinear layer, which calculates the cross-correlation of both inherent inputs.

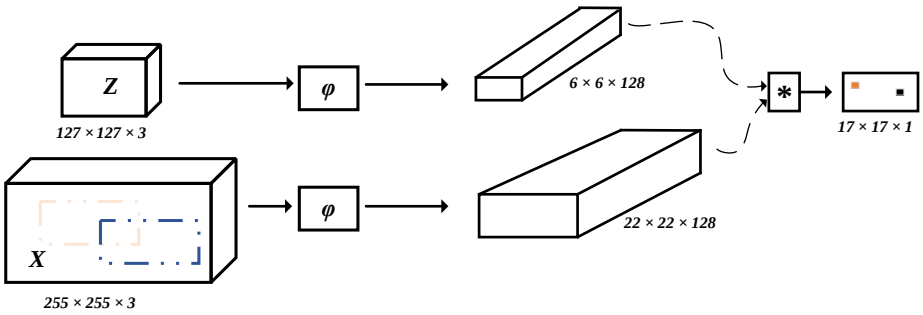


Figure 1. Fully convolutional Siamese architecture.

The resulting score map is scalar valued, the size of which is determined by the size of the search image. This allows the computation of the analogy function in a single evaluation for every translated sub-window in the search image. Finally, Bertinetto et al. show that for tracking process applications, Siamese FCNN deep networks can use available data more efficiently.

3. Implementation

This paper presents two techniques founded on the fully convolutional Siamese baseline architecture method described in Ref. [25]. The implemented algorithm uses only the SiamFC tracker with a pretrained network in a forward mode whose architecture is the same as in Ref. [25] and shown in Table 1. We used VOT 2016 dataset from the VOT benchmark for tracking during the experiments.

Table 1. Convolutional embedding function's architecture of the network used in this work.

				Size of Activation		
Layer	**Support**	**Channel Map**	**Stride**	**Exemplar**	**Search**	**Channels**
				127 × 127	255 × 255	×3
Conv1	11 × 11	96 × 3	2	59 × 59	123 × 123	×96
Max Pool1	3 × 3		2	29 × 29	61 × 61	×96
Conv2	5 × 5	256 × 48	1	25 × 25	57 × 57	×256
Max Pool2	3 × 3		2	12 × 12	28 × 28	×256
Conv3	3 × 3	384 × 256	1	10 × 10	26 × 26	×192
Conv4	3 × 3	384 × 192	1	8 × 8	24 × 24	×192
Conv5	3 × 3	256 × 192	1	6 × 6	22 × 22	×192

3.1. Siamese Network Model

A Siamese network contains two branches of shared network parameters. The network weakens the label of the data and adjusts its model by the similarity of the two branch features. The structure of the network is shown in Figure 2. The main idea is to find a function that can map the input to the target space, so that the "semantic" distance of the input space can be approximately represented by a simple distance in the target space. More precisely, to give a function family $G_w(x)$ parameterized by w, we need to find a value so that the parameter w can make the similarity measure, such as Equation (4):

$$E_w(x_1, x_2) = G_w(x_1) - G_w(x_2) \quad (4)$$

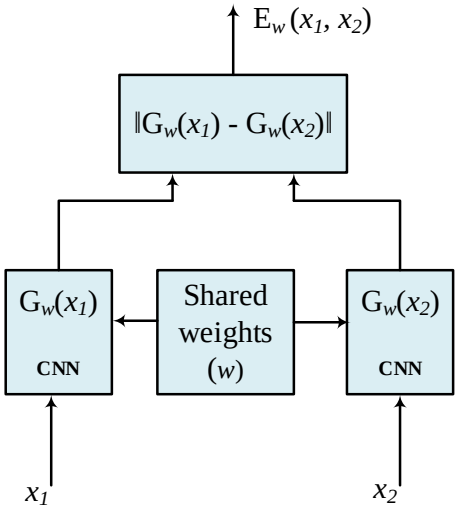


Figure 2. Schematic diagram of Siamese network.

$E_w(x_1, x_2)$ is smaller when x_1 and x_2 belong to the same kind, and larger when they are different. This system is trained by the combination input of the training set. The loss function minimized after training can minimize $E_w(x_1, x_2)$ when x_1 and x_2 belong to the same class, and maximize $E_w(x_1, x_2)$ when they belong to different classes. For the properties of $G_w(x)$, no hypothesis is made, except for the distinguishing ability of w. Because the same function $G_w(x)$ with the same parameter w acts on two inputs, the similarity measure is symmetric.

Taking face recognition as an example, if the input two branches of face images come from the same person, the extracted features should be as close as possible; otherwise, if the face images of different people are input into the network, the more different the feature vectors of the two branches, the better. When a convolutional neural network's extracted features match the properties listed above, the extracted features can be relatively general. Twin networks can make the extracted features have generalization and discrimination ability. It can map the samples that it has not seen, or seen a few times, to the target space. For example, the network has never seen an object, but after feature mapping, it is found that it is very close to the feature vector of a known object, so it can be inferred that the probability of the two being the same kind of object is very high.

This kind of network architecture needs a similarity measure function to calculate the characteristic distance of two branches. Euclidean distance and cosine distance are the two most basic and commonly used functions. The specific distance and similarity calculations formula are shown in Equations (5) and (6), respectively. In general, cosine distance is more robust because in many cases the images of the two branches differ greatly in the background and intensity of the illumination.

$$dist(X, Y) = \sqrt{\sum_{n=1}^{m} (x_n - y_n)^2} \tag{5}$$

$$sim(X, Y) = \cos \frac{x \times y}{x \times y} \tag{6}$$

One of the benefits of using a FCNN is that we can input a considerably bigger search image to the network rather than inputting a candidate image of equal size. Then, it will calculate the similarity at all translated sub-windows on a dense grid in one assessment. In Siamese neural networks, the same transformation is applied to the template image and the search image. The resulting combined representation of both inputs is a function g defined as

$$f(x,y) = g(\varphi(x), \varphi(y)) \tag{7}$$

where φ is the transformation applied to both inputs.

3.2. Methods

The main aim of this work is to optimize and enhance the overall performance of SiamFC tracker by introducing two methods. The first method implies the convolutional block initialization's parameters optimization [41] and the related activation function. The second method introduces a new activation function (ModReLU activation function) and Gaussian low pass filter to denoise the images and reduce details. The proposed methods help address rotation, moving, occlusion and appearance change problems. Figure 3 depicts the scheme and the sequence of the overall research procedure.

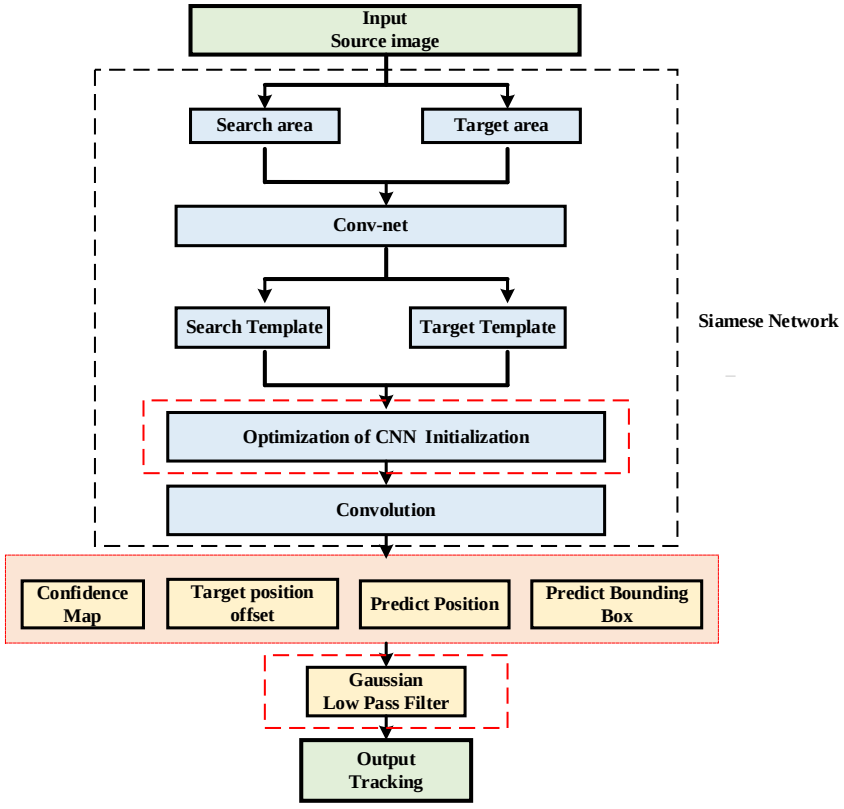


Figure 3. Scheme and sequence of the overall research procedure.

The specific implementation process is as follows:

(1) Initialize the target area image. According to the file that contains the correct annotation of each frame image in the video clip, the target area in the first frame image is determined, and the target area image is acquired, so as to replace the process of manually selecting the tracking target.

(2) Capture the image corresponding to the search area. Based on the location of the object in the present image predicted in the previous frame of the algorithm, the search area in the source image is determined according to the designed search area size, and the search area image is intercepted.

(3) Recover the convolutional neural network. The same convolutional neural network Conv-Net used in pretraining is rebuilt, and the trained network parameters are imported from the model file to recover Conv-Net.

(4) Construct a full convolution twin network structure. The acquired image and search area are sent to Conv-Net recovered in step (3), and the target template and search area template obtained after feature extraction are output. The two templates are convoluted to get the similarity response graph.

(5) With regard to the first method, in the convolutional neural network (CNN), we set the weights to constants and the bias to zero to implement a simple but very effective initialization. A Leaky ReLU activation is introduced in the activation layer according to the proposed initialization to accomplish the optimization of the CNN initialization.
(6) With respect to the second method, we realize the optimization of the CNN initialization by proposing and introducing a new activation function, ModReLU.
(7) Predict the target position in the next image. Bicubic interpolation is used to adjust the size of the similarity response graph to the size of the search area and find out the maximum response position in the similarity response graph. The image block corresponding to the position is the image block with the largest similarity to the image in the search area. Since the search area is determined by taking the target position in the current frame image predicted by the algorithm in the previous frame as the center, the maximum response position obtained at this time is the offset relative to the position of the target in the present frame image. This offset is used as the update parameter of the target position predicted by the algorithm, so as to update the target position predicted by the algorithm and obtain the target position in the next image predicted by the algorithm. Finally, according to the boundary frame parameters of the first frame, the image area of the target in the next frame predicted by the algorithm in the current frame is obtained. In the process of implementation, since the convolution neural network used for two inputs in the twin network is identical, only one Conv-Net is built for the common use of two input image data.
(8) Introduce Gaussian low pass filter for noise and details reduction.
(9) Tracking process.

3.2.1. Initialization Optimization and Activation Layer

Initialization Optimization

The two parameters involved at this point are the weights (w) and bias (b). There are different types of initialization techniques, among which the "Kaiming He initialization" [42] and "Xavier initialization" [43] are the methods often mentioned in the literature. A poor initialization can result in gradients that are either very small or very large. This causes the optimization algorithm to slow down. Normally, the weights should be randomly initialized to break the symmetry (because initializing to zeros can lead to a failure of the network to break the symmetry, meaning that each neuron of the neural network will learn the same features, and this would be the same as training a neural network with one unique neuron $n[l] = 1$ for each layer, so the neural network becomes equivalent to a linear classifier, such as logistic regression). Moreover, it ensures that distinct hidden neurons learn different features. However, it is acceptable to set the bias to zeros; even so, the symmetry will still be broken as long as the weights are not set to zeros. A random initialization is usually used to break the symmetry. So, various initializations result in different results. In the original configuration of the weights and bias initialization, both parameters were initialized as constants. In our work, we carefully chose the initialization, e.g., weights remained constants, but the biases were initialized to zeros. This leads to better results than the original algorithm. The considered weights and bias are indicated in Figure 4 as an illustration between the input and the output layers.

Bias is added to the product of the weights by the inputs as a constant or a vector. It is applied to balance the results; it shifts the activation function's result to the negative or positive side. The addition of bias reduces variance, allowing the neural network to be more flexible and generalizable. Because bias is essentially the inverse of the threshold, its value determines when the activation function is activated. The above-mentioned are depicted in the Figure 5 below for greater clarity.

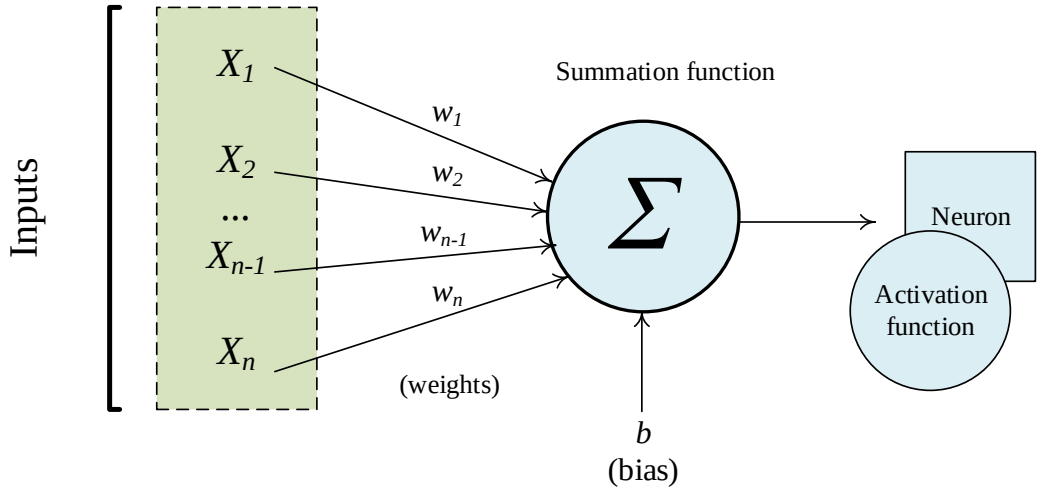


Figure 4. Weights and bias.

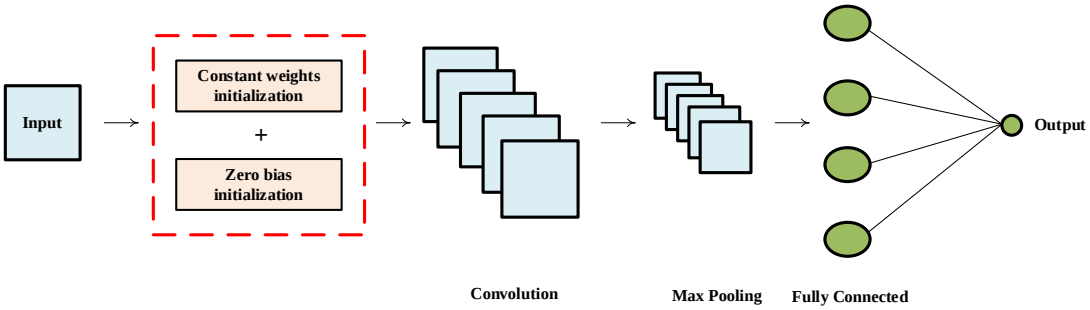


Figure 5. The architecture of proposed convolutional block optimization.

Activation Layer

The most common activation function in deep learning is the ReLU activation function due to its properties. It usually gives better results than the other activation functions. However, despite the encouraging results yielded by the ReLU activation function, it is not always the best suitable activation function for all the algorithms. In this particular work, we implemented a very similar function to the ReLU, which is the Leaky ReLU [44] activation function shown in Figure 6 and expressed as follows:

$$y_i = \begin{cases} \varphi x_i, x < 0 \\ x_i, x > 0 \end{cases} \quad (8)$$

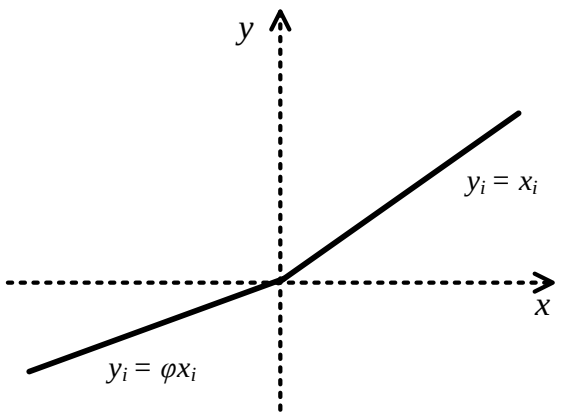


Figure 6. Leaky ReLU function graph.

Unlike ReLU, Leaky ReLU is more balanced. The leak contributes to expanding the scope of the ReLU function. Usually, the value of φ is set around 0.01.

Figure 7 illustrates the implementation of the proposed convolutional block initialization optimization and activation function. It includes an input layer, convolutional layers, maximum pooling layers, a fully connected layer and an output layer.

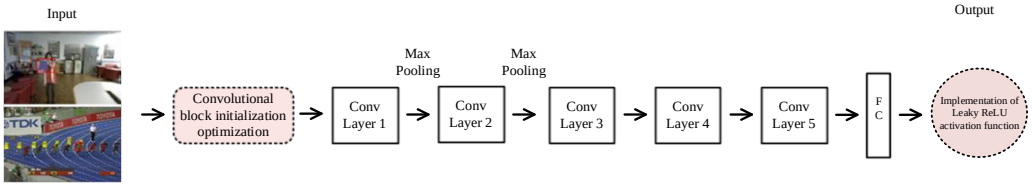


Figure 7. Proposed convolutional block initialization optimization and activation function.

3.2.2. New Activation Function and Gaussian Low Pass Filter

New Activation Function

In terms of the complexity of the CNN model's hierarchical structure, the activation is at the heart of it because the nonlinearity property of the activation function is what gives the neural network genuine AI. The ReLU activation has gained popularity in the deep-learning domain and is one of the best activation functions available. However, it also has some drawbacks. For instance, when the input is negative, the ReLU function does not produce activation, resulting in dead neurons. To address the above issue and to add more variety to the architecture and improve the tracking precision, we propose a new activation function called ModReLU in the activation layer as presented in Figure 8. The equation of ModReLU activation is expressed as

$$f(x) = max(a(e^x - 1), x)) + max(0, x), \text{ where } a > 0 \quad (9)$$

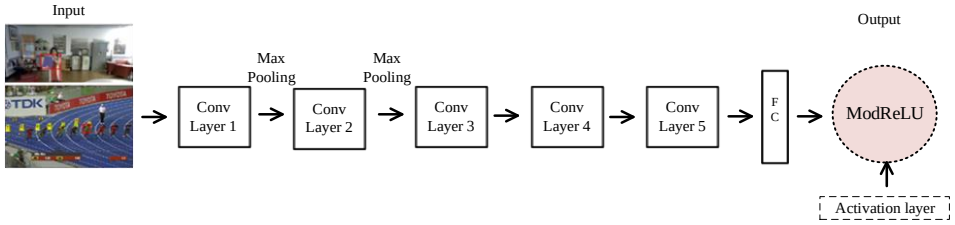


Figure 8. Illustration of our proposed ModReLU activation.

The ELU function controls the first half curve of the new activation (ModReLU). The slope of the curve of ELU near zero is large on the left axis, and it will be close to the best possible output rapidly. This allows ELU to produce negative outputs. The ReLU function is in charge of the second half curve of ModReLU.

The new activation function presents the following advantages:

(1) The new activation function can aid in avoiding the problem of dying ReLU because it selectively activates a large number of negative values, which further assist the network in squeezing weights and bias in the proper direction.

(2) The new activation function has some attributes of ReLU. It does not simultaneously activate all of the neurons.

Another way to allow negative activation is to employ Leaky ReLU activation function or other functions similar to Leaky ReLU, as we did in the first section. These activation functions have the same motivation with our ModReLU, in that they all address the two problems caused by the zero threshold of the ReLU activation. Nevertheless, ModReLU differentiates itself from the aforementioned activations, as it can generate a wider range of negative outputs. In other words, ModReLU is more balanced.

Gaussian Low Pass Filter

In order to remove details and imperfections from images before the tracking process, we apply a Gaussian low pass filter. To accomplish this, we propose the framework depicted in Figure 9. Compared to others, such as median or mean filters, in practice, it has proven to yield more effective results. The Gaussian filter keeps low spatial frequency while reducing noise and trivial elements in images, and it attenuates high frequency. The Gaussian filter is better at separating frequencies, which leads to nearby pixels having a bigger influence on the smoothed rather than the more distant ones; this appears to be relevant for the final results of our algorithm. It is implemented using a Gaussian function. Typically, it is accomplished by convolving a Gaussian kernel with an image [45–49].

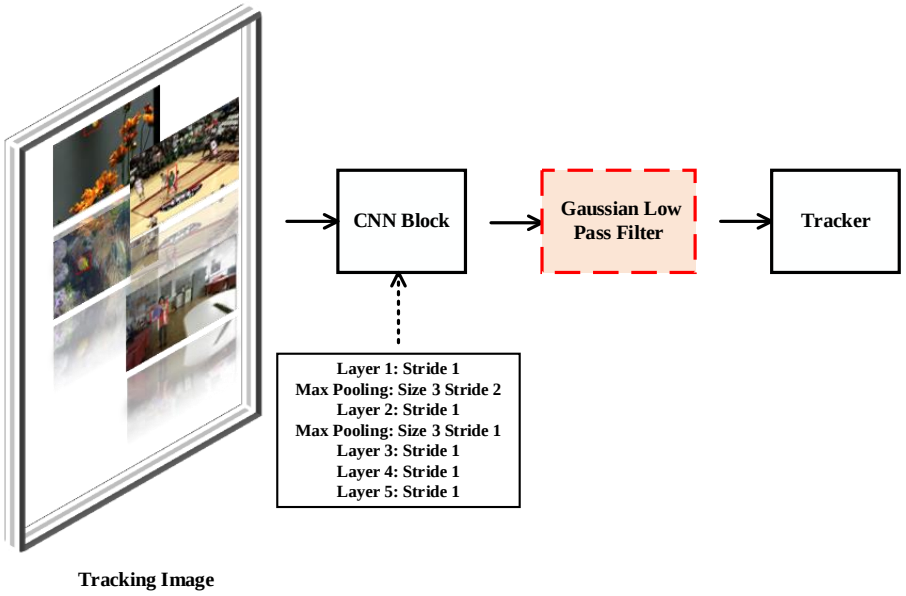


Figure 9. Overview of the proposed architecture with Gaussian low pass filter.

The Gaussian low pass filter is expressed as below:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{10}$$

σ is the Gaussian distribution's standard deviation, and x, y are the location indices. The standard deviation governs the variance of the Gaussian distribution around a mean value, which defines the blurring effect's range around a pixel. The standard deviation is set to 1 in our work.

4. Experimental Results and Discussion

To evaluate our approaches, the experiments were conducted with the Visual Object Tracking (VOT) Challenge 2016 dataset [47]. This dataset includes 60 sequences of images in JPEG format for the evaluation and the classification of cutting-edge pixel-based tracking algorithms. This benchmark is very representative of the issues of tracking algorithms, as it takes into account target motion, appearance change, rotation, occlusion, light condition, blurring, etc.

The network architecture we used is similar to the baseline conv5 from Ref. [41], except for some details. It has five convolutional layers, a maximum pooling of size 3 and a stride of size 2 following the first two convolutional layers. In the activation layer, a Leaky ReLU activation is used, and we also used SAME padding in both implementations.

The experiments are conducted and implemented based on TensorFlow library, and the evaluation is on a computer terminal with 3.20 GHz Intel(R) Core (TM) i7-8700 CPU and NVIDIA GeForce GTX 1070.

4.1. Convolutional Block Optimization Results

In this implementation related to Section 1, we carry out a tuning on the convolutional block initialization parameters, succeeded by a unique Leaky ReLU activation at the output layer. Another step comprises the application of brightness adjustment. The overall results are brought to the following table.

As is indicated in Table 2, during the experiments, it was noticed that the improved algorithms handle better the rotation, moving, occlusion and appearance change problems. This distinctly reflects on the results below with an increase in precision, AUPC and IOU.

Table 2. Overall results of the proposed convolutional block initialization and activation-based implementation.

Algorithm	Precision (%)	AUPC (%)	IOU (%)	Speed (FPS)
Optimized bias initialization-based SiamFC	53.39	18.85	35.43	46.46
Convolutional block optimization-based SiamFC	59.34	21.14	39.29	45.23

4.2. ModReLU Activation and Gaussian Low Pass Filter Implementation Results

Table 3 summarizes the outcomes of the second proposed method. This incorporates the implementation of the Gaussian low pass filter with the optimized convolutional block initialization parameters associated with our proposed ModReLU activation at the output layer. In addition, we added an adjustment of color brightness. From the table below, we clearly notice that the denoising and details reduction operation yielded encouraging results. The default sigma value used in the experiment is set to 1.

Table 3. Overall results of the proposed ModReLU–Gaussian low pass filter-based implementation.

Algorithm	Precision (%)	AUPC (%)	IOU (%)	Speed (FPS)
ModReLU–Gaussian low pass filter-based SiamFC	60.69	21.18	39.27	44.20

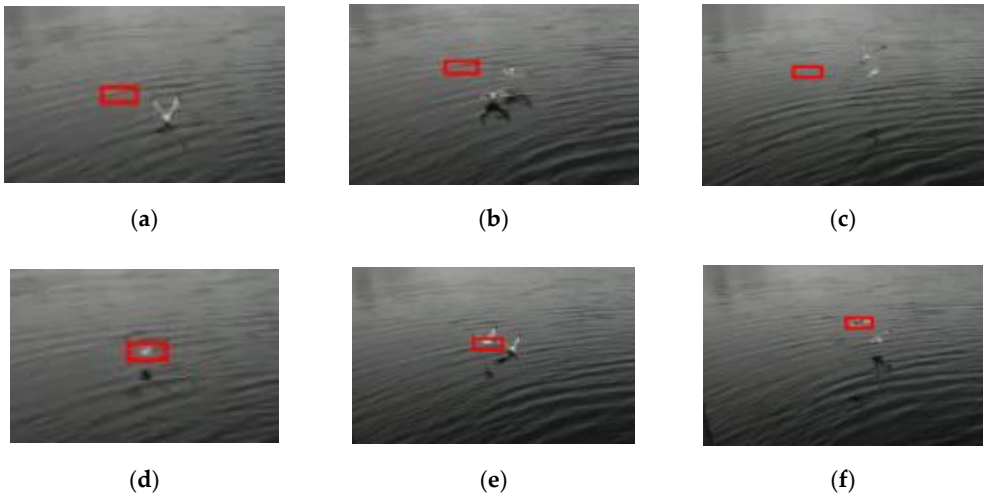
The literature divides tracking techniques into two types: single object tracking (SOT) and multiple object tracking (MOT). In single object tracking approaches, the appearance of the object of interest is known in advance, whereas the goal of MOT techniques is to evaluate the trajectories of multiple targets of one or more categories with no previous knowledge of their appearance. Object detection across the frames is necessary for MOT. Among the existing tracking methods that also address motion and appearance issues to enhance tracking performance, DeepSORT [49] is actually one of the most widely utilized ones that achieves state-of-the-art performance. DeepSORT is a tracking-by-detection algorithm that considers both the bounding box parameters of the detection results and the information about the appearance of the tracked objects to associate the detections in a new frame with previously tracked objects. It is an improvement of the SORT algorithm that integrates additional apparent feature information matching based on pretrained CNNs, allowing re-identification of tracks within a longer period of occlusion to improve tracking performance. However, this approach differs from ours because it is mostly used in multiple object tracking scenarios and is evaluated in MOT datasets, while our method is suitable for single object tracking scenes. Several research works have shown that applying a single object tracking method to perform a multiple object tracking task usually yields low performance.

5. Comparison of Implemented Approaches with Original SiamFC Tracker Performance on VOT 2016 Dataset

After the evaluation of our proposed approaches, we show the comparison of the results including the aforementioned implementations and the original SiamFC, and the related illustrations. First, we present the overall performance comparison between the original SiamFC tracker and convolutional block optimization-based SiamFC in Table 4, followed by pictures illustration in Figures 10 and 11. Then, the performance of ModReLU–Gaussian low pass filter-based SiamFC is compared to the original implementation in Table 5 with illustration in Figure 12. Finally, Table 6 provides a summary comparison of the proposed techniques and the original SiamFC tracker performance.

Table 4. Overall results comparison of convolutional block optimization-based SiamFC with original SiamFC.

Algorithm	Precision (%)	AUPC (%)	IOU (%)	Speed (FPS)
Original SiamFC	51.41	18.10	33.89	49.39
Convolutional block optimization-based SiamFC	59.34	21.14	39.29	45.23



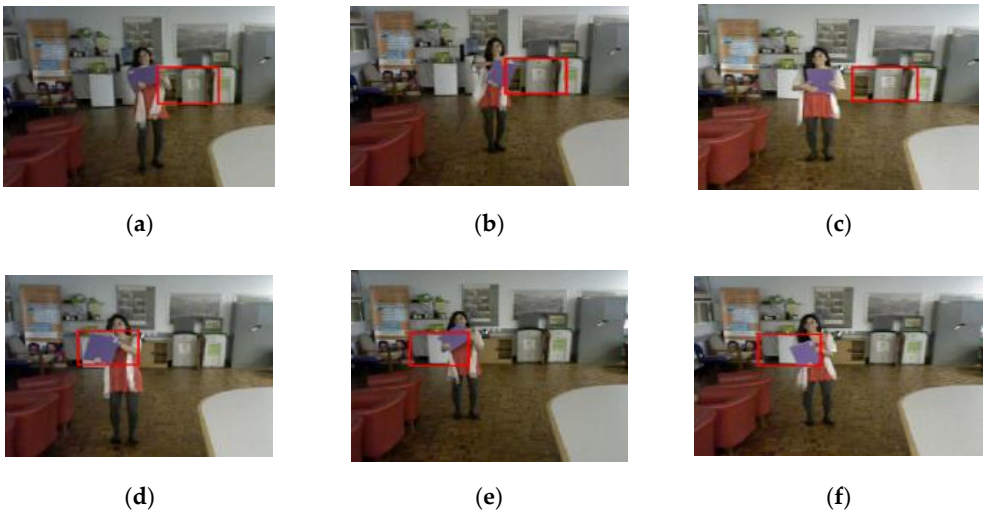
(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 10. Comparison of original SiamFC and our convolutional block optimization-based SiamFC in scene "Birds 1": (**a**–**c**) Snapshots from original SiamFC (Precision: 7.67, AUPC: 3.33, IOU: 5.55, Speed (FPS): 32.35); (**d**–**f**) Snapshots from our convolutional block optimization-based SiamFC (Precision: 90.56, AUPC: 32.31, IOU: 52.40, Speed (FPS): 30.47).

Table 5. Overall results comparison of the proposed ModReLU–Gaussian low pass filter-based SiamFC with original SiamFC.

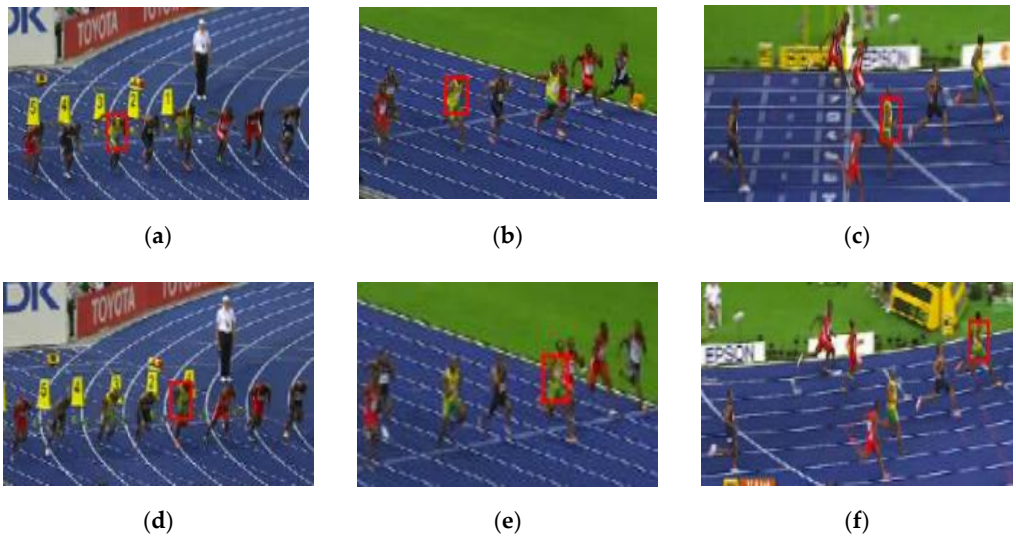
Algorithm	Precision (%)	AUPC (%)	IOU (%)	Speed (FPS)
Original SiamFC	51.41	18.10	33.89	49.39
ModReLU–Gaussian low pass filter-based SiamFC	60.69	21.18	39.27	44.20



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 11. Comparison of original SiamFC and our convolutional block optimization-based SiamFC in scene "Book": (**a**–**c**) Snapshots from original SiamFC (Precision: 9.14, AUPC: 2.85, IOU: 12.12, Speed: 50.32); (**d**–**f**) Snapshots from our convolutional block optimization-based SiamFC (Precision: 26.86, AUPC: 8.70, IOU: 18.23, Speed: 44.12).



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 12. Comparison of original SiamFC and our ModReLU–Gaussian low pass filter-based SiamFC in scene "Bolt 1": (**a**–**c**) Snapshots from original SiamFC (Precision: 3.43, AUPC: 1.12, IOU: 2.16, Speed: 45.65); (**d**–**f**) Snapshots from our ModReLU–Gaussian low pass filter-based SiamFC (Precision: 99.71, AUPC: 30.99, IOU: 47.27, Speed: 43.00).

Table 6. Overall results comparison of convolutional block optimization-based SiamFC and ModReLU–Gaussian low pass filter-based SiamFC with original SiamFC.

Algorithm	Precision (%)	AUPC (%)	IOU (%)	Speed (FPS)
Original SiamFC	51.41	18.10	33.89	49.39
Convolutional block optimization-based SiamFC	59.34	21.14	39.29	45.23
ModReLU–Gaussian low pass filter-based SiamFC	60.69	21.18	39.27	44.20

The above Table clearly shows a remarkable improvement with the first proposed approach compared to the original SiamFC. The related illustration of the foregoing results is shown next.

On the above illustrations, it is clearly shown in both VOT 2016 dataset scenes ("Birds 1" and "Book") that our convolutional block optimization-based SiamFC overcomes the original SiamFC in different scenarios with rotation and occlusion. The comparative metrics indicate a significant improvement in the aforesaid scenes, as reflected in the overall performance in Table 4.

Table 5 shows that our ModReLU–Gaussian low pass filter-based SiamFC proved to be effective in this work due to its characteristics. After brightness adjustment, the overall performance of the precision increases considerably, and the IOU also shows a notable improvement, while only a slight increase can be observed for the area under the curve precision metric. To highlight the given results, we show an illustration of some selected frames for both ModReLU–Gaussian low pass filter SiamFC and original SiamFC below.

In this particular scene of the VOT 2016 dataset named "Bolt 1", the original SiamFC tracking algorithm fails by completely losing the target of interest and tracking a similar neighbor target with the same uniformity until the end of the video sequence, whereas our ModReLU–Gaussian low pass filter-based SiamFC performs better by keeping track of the moving target during the entire video sequence without switching to a neighbor moving object.

To summarize all of the above, we arranged all the results into a histogram as depicted in Figure 13.

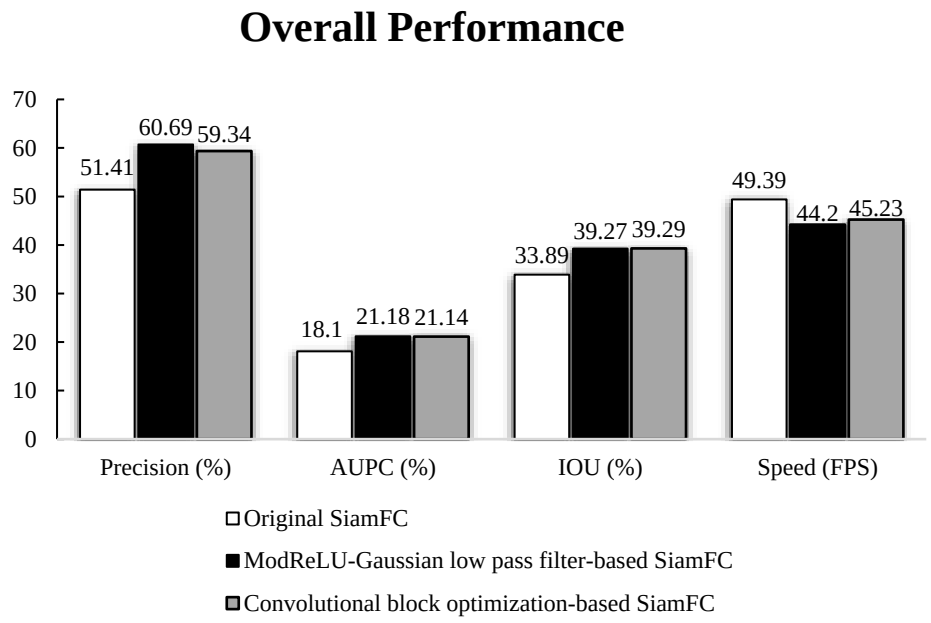


Figure 13. Results comparison histogram of our proposed methods and original SiamFC tracker performances.

6. Comparison with the State-of-the-Art Trackers

We compare the improved trackers to several top performers in VOT 2016. As shown in Table 7, our trackers show remarkable performance and outperform many trackers in terms of precision and speed. In particular, our ModReLU–Gaussian low pass filter-based SiamFC tracker achieves the best precision among all the compared trackers, and our convolutional block optimization-based SiamFC tracker realizes the second best performance in tracking speed.

Table 7. Comparison results under precision and speed (FPS) with top performers on the VOT2016. The top two outcomes are highlighted in red and blue.

Tracker	Precision	Speed (FPS)
STAPLEp	0.557	44.8
CCOT	0.539	0.5
TCNN	0.554	1.1
SSKCF	0.54	>25
DPT	0.49	>25
Staple	0.544	11
DNT	0.515	1.1
DeepSRDCF	0.528	0.38
MDNet_N	0.541	0.989
KCF	0.48	172
SiamRPN	0.560	23.0
SSAT	0.577	0.5
MLDF	0.490	1.2
SRBT	0.496	3.7
FlowTrack	0.58	-
ECO	0.55	-
Convolutional block optimization-based SiamFC (ours)	0.593	45.23
ModReLU–Gaussian low pass filter-based SiamFC (ours)	0.606	44.20

7. Conclusions

This work introduces an enhanced deep-learning fully convolutional Siamese neural network based on convolutional hyperparameters optimization, new activation function (ModReLU) and Gaussian low pass filter. The proposed methods are divided into two parts, where two different approaches are proposed to enhance the overall performance of the SiamFC tracker. Both tackle rotation, moving, occlusion and appearance change issues. They are evaluated on the Visual Object Tracking (VOT) Challenge 2016 dataset and outperform the original SiamFC and many well-known existing algorithms in terms of precision and speed. This paper does not include the training process but only the tracking. All experiments are conducted offline. Our convolutional block optimization-based SiamFC achieves remarkable results and surpasses the original SiamFC performance and other popular algorithms as shown in Table 7. This is another demonstration that less computational complexity can achieve high performance. ModReLU–Gaussian low pass filter-based SiamFC tracker yields even a higher performance and more consistent results, especially in terms of precision, compared to the results of the previous method. It also outclasses other popular algorithms. The results illustrate the influence and effect of proper optimization, fitting dataset processing and a suitable activation function on the overall performance of a neural network. Future work could consider taking into account the development of fine-tuning approaches combined with our proposed techniques so as to achieve the highest level of effectiveness.

Author Contributions: Conceptualization, R.K.; methodology, R.K.; software, R.K. and Z.H.; validation, Y.Z. and R.K.; formal analysis, Y.Z.; investigation, Z.H. and C.H.; resources, Z.L.; data curation, R.K.; writing—original draft preparation, R.K.; writing—review and editing, R.K., Z.L. and Z.H.; visualization, Y.Z.; supervision, Z.L.; project administration, Z.L. and Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available at https://github.com/torrvision/siamfc-tf (accessed on 9 December 2021).

Acknowledgments: We wish to express our gratitude to Sha Xuejun and Mei Lin for their support and contributions to this work. Additionally, we would like to thank Dong Heng, Lu Zirui, Gao Xinbo, Li Jiazhe, Li Huaqing, Li Yongjian, Wang Houlei, and Li Siyi for their availability and valuable discussions, as well as AR JUNEJO for his precious help preparing the manuscript.

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

References

1. Wu, Y.; Lim, J.; Yang, M.H. Online object tracking: A benchmark. In Proceedings of the Computer Vision and Pattern Recognition, Portland, OR, USA, 25–27 June 2013; pp. 2411–2418.
2. Marvasti-Zadeh, S.M.; Cheng, L.; Ghanei-Yakhdan, H.; Kasaei, S. Deep learning for visual tracking: A comprehensive survey. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–26. [CrossRef]
3. Guo, D.; Wang, J.; Cui, Y.; Wang, Z.; Chen, S. SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 6268–6276.
4. Wu, F.; Zhang, J.; Xu, Z. Stably adaptive anti-occlusion Siamese region proposal network for realtime object tracking. *IEEE Access* **2020**, *8*, 161349–161360. [CrossRef]
5. Sosnovik, I.; Moskalev, A.; Smeulders, A. Scale equivariance improves Siamese tracking. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 5–9 January 2021; pp. 2764–2773.
6. Abbass, M.Y.; Kwon, K.-C.; Kim, N.; Abdelwahab, S.A.; El-Samie, F.E.A.; Khalaf, A.A.M. A survey on online learning for visual tracking. *Vis. Comput.* **2021**, *37*, 1–22. [CrossRef]
7. Li, D.; Yu, Y. Foreground information guidance for Siamese visual tracking. *IEEE Access* **2020**, *8*, 55905–55914. [CrossRef]
8. Rao, Y.; Cheng, Y.; Xue, J.; Pu, J.; Wang, Q.; Jin, R.; Wang, Q. FPSiamRPN: Feature pyramid Siamese network with region proposal network for target tracking. *IEEE Access* **2020**, *8*, 176158–176169. [CrossRef]
9. Luo, Y.; Cai, Y.; Wang, B.; Wang, J.; Wang, Y. SiamFF: Visual tracking with a Siamese network combining information fusion with rectangular window filtering. *IEEE Access* **2020**, *8*, 119899–119910. [CrossRef]
10. Zhao, F.; Zhang, T.; Wu, Y.; Tang, M.; Wang, J. Antidecay LSTM for Siamese tracking with adversarial learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4475–4489. [CrossRef]
11. Zhao, F.; Zhang, T.; Song, Y.; Tang, M.; Wang, X.; Wang, J. Siamese regression tracking with reinforced template updating. *IEEE Trans. Image Process.* **2021**, *30*, 628–640. [CrossRef]
12. Wu, Y.; Lim, J.; Yang, M.H. Object tracking benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1834–1848. [CrossRef]
13. Lu, L.; Fei, M.; Wang, H.; Hu, H. A new meanshift target tracking algorithm by combining feature points from gray and depth images. In *Advanced Computational Methods in Life System Modeling and Simulation*; Fei, M., Ma, S., Li, X., Sun, X., Jia, L., Su, Z., Eds.; Springer: Singapore, 2017; pp. 545–555.
14. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.T.; Upcroft, B. Simple online and realtime tracking. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
15. Brookner, E. *Tracking and Kalman Filtering Made Easy*; Wiley-Blackwell: New York, NY, USA, 1998.
16. Bruno, A.; Ardizzone, E.; Vitabile, S.; Midiri, M. A Novel Solution Based on Scale Invariant Feature Transform Descriptors and Deep Learning for the Detection of Suspicious Regions in Mammogram Images. *J. Med. Signals Sens.* **2020**, *10*, 158–173.
17. Zhao, Y.; Yu, L.; Zheng, X. A Deep Hyper Siamese Network for Real-Time Object Tracking. *Trans. Mach. Learn. Artif. Intell.* **2020**, *8*, 35–46. [CrossRef]
18. Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-Learning-Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 1409–1422. [CrossRef] [PubMed]
19. Bromley, J.; Bentz, J.W.; Bottou, L.; Guyon, I.; LeCun, Y.; Moore, C.; Säckinger, E.; Shah, R. Signature verification using a "Siamese" time delay neural network. In Proceedings of the 6th International Conference on Neural Information Processing Systems, Denver, CO, USA, 29 November–2 December 1993; pp. 737–744.
20. Zagoruyko, S.; Komodakis, N. Learning to compare image patches via convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 4353–4361.
21. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. DeepFace: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 1701–1708.
22. Wang, Y.; Li, Y.; Song, Y.; Rong, X. The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition. *Appl. Sci.* **2020**, *10*, 1897. [CrossRef]
23. Maguolo, G.; Nanni, L.; Ghidoni, S. Ensemble of Convolutional Neural Networks Trained with Different Activation Functions. *Expert Syst. Appl.* **2019**, *166*, 114048. [CrossRef]
24. Dubey, A.K.; Jain, V. Comparative study of convolution neural network's ReLu and Leaky-ReLu activation functions. In *Applications of Computing, Automation and Wireless Systems in Electrical Engineering*; Springer: Singapore, 2019; pp. 873–880.

25. Bertinetto, L.; Valmadre, J.; Henriques, J.F.; Vedaldi, A.; Torr, P.H.S. Fully-convolutional Siamese networks for object tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 850–865.
26. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, USA, 15–17 June 2010; pp. 2544–2550.
27. Nam, H.; Han, B. Learning multi-domain convolutional neural networks for visual tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 4293–4302.
28. Ma, C.; Huang, J.-B.; Yang, X.; Yang, M.-H. Hierarchical convolutional features for visual tracking. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 3074–3082.
29. Danelljan, M.; Hager, G.; Khan, F.S.; Felsberg, M. Convolutional features for correlation filter based visual tracking. In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW), Santiago, Chile, 7–13 December 2015; pp. 58–66.
30. Wang, N.; Li, S.; Gupta, A.; Yeung, D.-Y. Transferring rich feature hierarchies for robust visual tracking. *arXiv* **2015**, arXiv:1501.04587.
31. Kashiani, H.; Shokouhi, S.B. Visual object tracking based on adaptive Siamese and motion estimation network. *Image Vis. Comput.* **2019**, *83–84*, 17–28. [CrossRef]
32. Zhai, M.; Roshtkhari, M.J.; Mori, G. Deep learning of appearance models for online object tracking. In Proceedings of the European Conference on Computer Vision Workshops (ECCVW), Munich, Germany, 8–14 September 2018.
33. Danelljan, M.; Robinson, A.; Khan, F.S.; Felsberg, M. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 472–488.
34. Held, D.; Thrun, S.; Savarese, S. Learning to track at 100 fps with deep regression networks. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 749–765.
35. Valmadre, J.; Bertinetto, L.; Henriques, J.F.; Vedaldi, A.; Torr, P.H.S. End-to-end representation learning for correlation filter-based tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2805–2813.
36. Guo, Q.; Feng, W.; Zhou, C.; Huang, R.; Wan, L.; Wang, S. Learning dynamic siamese network for visual object tracking. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1763–1771.
37. He, A.; Luo, C.; Tian, X.; Zeng, W. A twofold siamese network for real-time object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4834–4843.
38. Yu, Y.; Xiong, Y.; Huang, W.; Scott, M.R. Deformable Siamese attention networks for visual object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, Online, USA, 13–19 June 2020; pp. 6727–6736.
39. Yan, Y.; Huo, W.; Ou, J.; Liu, Z.; Li, T. Improved SiamFC Target Tracking Algorithm Based on Anti-Interference Module. *J. Sens.* **2022**, *2022*, 2804114. [CrossRef]
40. Cui, Z.; An, J.; Ye, Q.; Cui, T. Siamese Cascaded Region Proposal Networks with Channel-Interconnection-Spatial Attention for Visual Tracking. *IEEE Access* **2020**, *8*, 154800–154815. [CrossRef]
41. Madrigal, F.; Maurice, C.; Lerasle, F. Hyper-parameter optimization tools comparison for multiple object tracking applications. *Mach. Vis. Appl.* **2019**, *30*, 269–289. [CrossRef]
42. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1026–1034.
43. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. *J. Mach. Learn. Res.* **2010**, *9*, 249–256.
44. Ding, B.; Qian, H.; Zhou, J. Activation functions and their characteristics in deep neural networks. In Proceedings of the Chinese Control and Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 1836–1841.
45. Li, B.; Yan, J.; Wu, W.; Zhu, Z.; Hu, X. High performance visual tracking with siamese region proposal network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 8971–8980.
46. Misra, S.; Wu, Y. Machine learning assisted segmentation of scanning electron microscopy images of organic-rich shales with feature extraction and feature ranking. In *Machine Learning for Subsurface Characterization*; Elsevier: Cambridge, MA, USA, 2020; pp. 289–314.
47. Kristan, M.; Leonardis, A.; Matas, J.; Felsberg, M.; Pflugfelder, R.; Čehovin Zajc, L.; Vojir, T.; Hager, G.; Lukezic, A.; Eldesokey, A.; et al. The visual object tracking VOT 2016 challenge results. In Proceedings of the European Conference on Computer Vision Workshops (ECCVW), Amsterdam, The Netherlands, 8–16 October 2016; pp. 191–217.
48. Zhou, L.; Zhang, J. Combined kalman filter and multifeature fusion siamese network for real-time visual tracking. *Sensors* **2019**, *19*, 2201. [CrossRef] [PubMed]
49. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.