Robust Tracking via Unifying Pretrain-Finetuning and Visual Prompt Tuning

Guangtong Zhang*
zhangguangtong@stu.gxnu.edu.cn
Guangxi Normal University
Guilin, China

Qihua Liang†*
qhliang@gxnu.edu.cn
Guangxi Normal University
Guilin, China

Ning Li*
ningli65536@mailbox.gxnu.edu.cn
Guangxi Normal University
Guilin, China

Zhiyi Mo*‡
zhiyim@gxuwz.edu.cn
Guangxi Normal University
Wuzhou University
Guilin, Wuzhou, China

Bineng Zhong*
bnzhong@gxnu.edu.cn
Guangxi Normal University
Guilin, China

ABSTRACT

The finetuning paradigm has been a widely used methodology for the supervised training of top-performing trackers. However, the finetuning paradigm faces one key issue: it is unclear how best to perform the finetuning method to adapt a pretrained model to tracking tasks while alleviating the catastrophic forgetting problem. To address this problem, we propose a novel partial finetuning paradigm for visual tracking via unifying pretrain-finetuning and visual prompt tuning (named UPVPT), which can not only efficiently learn knowledge from the tracking task but also reuse the prior knowledge learned by the pre-trained model for effectively handling various challenges in tracking task. Firstly, to maintain the pre-trained prior knowledge, we design a Prompt-style method to freeze some parameters of the pretrained network. Then, to learn knowledge from the tracking task, we update the parameters of the prompt and MLP layers. As a result, we cannot only retain useful prior knowledge of the pre-trained model by freezing the backbone network but also effectively learn target domain knowledge by updating the Prompt and MLP layer. Furthermore, the proposed UPVPT can easily be embedded into existing Transformer trackers (e.g., OSTracker and SwinTracker) by adding only a small number of model parameters (less than 1% of a Backbone network). Extensive experiments on five tracking benchmarks (i.e., UAV123, GOT-10k, LaSOT, TNL2K, and TrackingNet) demonstrate that the proposed UPVPT can improve the robustness and effectiveness of the model, especially in complex scenarios.

*The Guangxi Key Lab of Multi-Source Information Mining & Security, Guangxi Normal University, Guilin 541004, China.
†Corresponding author.
‡Guangxi Key Laboratory of Machine Vision and Intelligent Control, Wuzhou University, Wuzhou 543002, China.

MMAsia '23, December 6–8, 2023, Tainan, Taiwan

ACM ISBN 979-8-4007-0205-1/23/12…$15.00
https://doi.org/10.1145/3595916.3626410

CCS CONCEPTS

• **Computing methodologies** → **Tracking**.

KEYWORDS

Object Tracking, Prompt, Pretrain-finetuning

ACM Reference Format:
Guangtong Zhang, Qihua Liang, Ning Li, Zhiyi Mo, and Bineng Zhong. 2023. Robust Tracking via Unifying Pretrain-Finetuning and Visual Prompt Tuning. In *ACM Multimedia Asia 2023 (MMAsia '23), December 6–8, 2023, Tainan, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3595916.3626410

1 INTRODUCTION

Object tracking is well-known as a fundamental problem in computer vision and has a variety of real-life applications, such as video surveillance, autonomous vehicles, etc. Recently, with the rapid development of deep learning, various powerful trackers[1, 33] have been proposed to effectively handle the complex challenges that may occur in natural scenes, such as target appearance changes, illumination variations, occlusion, background clutters, etc. Despite their success, we observe that a critical problem that remains unsolved is the model's robustness against untrained target categories. To address this problem, one popular paradigm used in existing approaches[5, 14, 39] introduces pre-trained models to improve the model robustness against more categories by using the prior knowledge obtained from the pre-trained model on the object classification task. As shown in Tab.1, the datasets of the object classification tasks (i.e., COCO[24] and ImageNet-21K[10]) have more category information than that of the object tracking tasks (i.e., GOT-10k[18], LaSOT[12] and TrackingNet[28]). The effectiveness of the above approaches has been verified in many trackers[8, 39]. For example, in OSTrack[39], 10.9% AO performance gain is gained by using a pre-trained MAE than no pre-trained model in the GOT-10k dataset. We can observe that a standard training method for these trackers is to load the backbone networks with a pre-trained model and then update all parameters of the whole networks during their training process. However, the above overall fine-tuned paradigms ignore the difference between an object tracking task and an object classification task. In contrast to object classification, there are two critical challenges in the visual tracking task: unknown categories

Table 1: Comparison of dataset between object tracking dataset and pre-trained dataset based on object classification.

Dataset	Domain	Year	Size	Category
GOT-10k[18]	Tracking	2019	66G	563
LaSOT[12]	Tracking	2018	227G	70
TrackingNet[28]	Tracking	2018	1061G	27
COCO [24]	Detection/Segmentation	2015	25G	80
ImageNet-1k [10]	Classification	2012	138G	1000
ImageNet-21k [10]	Classification	2021	1.1T	21000

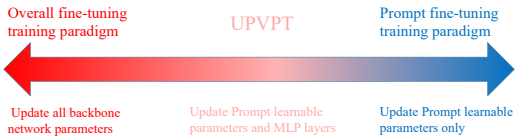

Figure 1: Our proposed partial fine-tuning method (i.e., UPVPT) is between the Prompt and overall fine-tuning methods. Our proposed UPVPT can balance the overall fine-tuning and Prompt fine-tuning methods well.

and the discrepancy between different objects. Classifiers usually only need to distinguish pre-defined target categories. However, in the task of object tracking, the target is specified by the first frame, and thus the categories of the target are unknown before tracking. Furthermore, unlike the classification task, where classifiers aim to distinguish different categories of targets, classifiers in tracking focus on determining whether two targets are the same. Due to the overall fine-tuning training paradigm ignoring these differences, the pre-trained model is directly applied to the tracking task for parameter updating, which destroys the original pre-trained model structure and catastrophically forgets the prior knowledge gained by the pre-trained model. With the rapid development of Transformer in computer vision, the publication of VPT[19] has brought increasing attention to the application of the Prompt method in computer vision. We found that using the Prompt method can retain the prior knowledge obtained by loading pre-trained models. However, the standard Prompt has two limitations: (1) The common Prompt method adds a small number of additional trainable parameters and freezes all parameters of the original network. Although, these operations significantly reduce trainable parameters, increase heavy dependence on pre-trained models, and have poor adaptability to target domain data. (2) Most Prompt methods are effective when the upstream and downstream tasks are consistent or close to each other. When the upstream and downstream tasks are significantly different, achieving better results than overall fine-tuning is not easy. Therefore, a question is raised: Is there a way to prevent the pretrained models from catastrophic forgetting while being well adapted to the visual object tracking task during the fine-tuning training phase not to limit the learning capability?

Based on the above observations, we propose a novel partial fine-tuning training method (UPVPT), which focuses on parameter tuning of the backbone network in the training phase after loading the pre-trained model. We introduce the Prompt method to construct the connection between the classification and tracking tasks. We freeze some parameters of the backbone and update the remaining parameters to ensure that the model can learn more knowledge. Our main contributions are summarized as follows:

- To the best of our knowledge, we are the first to study the problem of catastrophic forgetting of a pre-trained model for supervised tracker training.
- We propose a novel partially fine-tuned training paradigm, as shown in Fig.1 to balance the overall fine-tuning approach and the Prompt fine-tuning approach, which can alleviate the catastrophic forgetting problem of the pre-trained model during training without limiting the ability to learn new knowledge from new data.
- We apply the UPVPT approach to existing high-performing trackers, and extensive experimental comparisons demonstrate that our method can significantly enhance the robustness of the model and achieve better tracking performance.

2 RELATED WORK

2.1 Visual Object Tracking

With the rapid development of deep learning, most deep learning-based trackers use a pre-trained model from object classification as the initial feature representation. Early trackers[2, 7, 16, 21] used ResNet-50 as the backbone network, using a model trained based on ImageNet-1K[10] dataset of ResNet-50 in object classification task as the pre-trained model. Subsequently, many excellent Transformer-based trackers[5, 6, 34, 38] have emerged in the field of object tracking, which uses different backbone networks, e.g., CvT[37] for MixFormer[8]. Not only is the feature extraction ability of the backbone model constantly improving, but the performance of the pre-trained model is also making progress in visual object tracking. From the early phases of training models for object classification tasks based on the ImageNet-1K dataset as pre-training to the later phases of training models for object classification based on the ImageNet-22K dataset as pre-training. Recent OSTrack[39] using the superior MAE[17] as the pre-trained model can get better results than ImageNet-1K based pre-trained model. However, Bhat *et al.*[4]. demonstrated that combining pre-trained deep features may only sometimes improve performance due to target invisibility, resolution incompatibility, and dimensionality increase. Li *et al.*[22] also shows that in visual tracking, the target of interest can be any object class with any form. Therefore, pre-trained deep features are less effective in modeling these arbitrary forms of targets and distinguishing them from the background. Moreover, Li *et al.*[22] proposed a target-aware module based on this problem to effectively guide network learning through Loss to expand inter- and intra-class discrepancy. However, this approach only considers the differences between the object tracking task and the object classification task, ignores their similarities, and makes the model more dependent on the training data, affecting the model's robustness. In our present work, we maximize the feature extraction capability of the pre-trained model while allowing the model to learn the discrepancy between different tasks to adapt it to the object tracking domain.

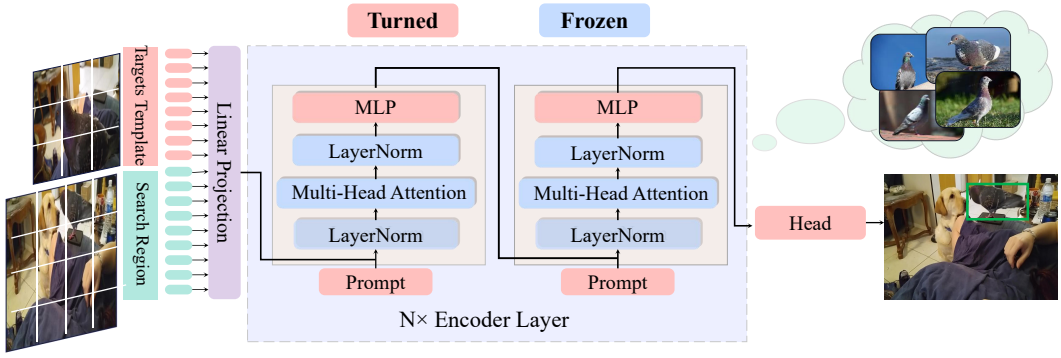

Figure 2: The overall structure of our proposed UPVPT. The network layer parameters marked in red are updated during the training phase, while the network layer parameters marked in blue are frozen and will not be updated during the training phase.

2.2 Prompt Turning

Prompt Tuning[25] was initially proposed in natural language processing with the GPT family[13]. This approach uses a model already trained in the upstream task as pre-training and adapts the model to the downstream task faster by adding downstream task hints to the model for the downstream task. However, with the rapid development of Transformer for computer vision tasks, the training parameters of the models are increasing, and the demand for data is becoming increasingly severe. Inspired by the Prompt idea in the field of natural language processing, Jia *et al.*[19] proposed VPT, which injects a small number of trainable parameters into VIT[11] and freezes all the original backbone network parameters, and trains only the additional added trainable parameters in the backbone network, achieving better results than full fine-tuning in most downstream tasks. Most of the subsequent proposed partial fine-tuning training paradigms[20, 30, 40] are based on the above two ideas. However, these approaches have two limitations: (1) The substantial reduction of learnable parameters results in very little learning in the training phase. (2) Most of these approaches fit for less discrepant in upstream and downstream tasks, while Prompt has yet to be significantly effective in areas where the discrepancy is significant. In our present work, we can successfully adapt Prompt to downstream tasks when the upstream and downstream tasks are of more significant difference.

3 METHOD

3.1 Overview

This section presents our proposed partially fine-tuned training paradigm applicable to the object tracking task. The overall architecture of our proposed UPVPT is depicted in Fig.2. It comprises the injection of the learnable parameter Prompt and freezing of the backbone network parameters. This new training paradigm builds a connection between the object tracking task and the pre-trained model task while safeguarding the prior knowledge obtained by loading the pre-trained model from being corrupted or forgotten by parameter updates. Moreover, it preserves the ability to learn from the object tracking task without excessive reliance on the pre-trained model.

3.2 Additional Parameters: Prompt

Due to the great success of Transformer[11] in computer vision, recently, most of the trackers[8, 14, 23, 39] have adopted Transformer networks as backbone networks. These transformer-based trackers are divided into two main types: one-stream methods[8, 39] and Siamese structures[14, 23]. To demonstrate the effectiveness of our proposed partial fine-tuning paradigm, we adopt OSTrack[39] as our baseline, while in the ablation experiments, we also adopt Siamese structure-based tracker SwinTrack[23] as the baseline to demonstrate the effectiveness of our proposed method in the ablation experiments.

For the backbone network with 12 layers, using the existing vision Transformer structure, the inputs to the model are a pair of images, i.e. template frames $z \in \mathbb{R}^{3\times H_z\times W_z}$ and search frames $x \in \mathbb{R}^{3\times H_x\times W_x}$. Next, we follow the original network architecture and embed learnable 1D position p_z and p_x for template tokens and search tokens to generate template tokens embeddings $\boldsymbol{H}_z^0 \in \mathbb{R}^{N_z\times D}$ and search tokens embeddings $\boldsymbol{H}_x^0 \in \mathbb{R}^{N_x\times D}$. The formula is as follows:

$$H_z^0 = \text{Embed}[Z_p] + p_z, \tag{1}$$

$$H_x^0 = \text{Embed}[X_p] + p_x, \tag{2}$$

We introduce a set of d-dimensional continuous learnable parameters in the input space before the Transformer layer, called Prompt. Each Prompt token is treated as a learnable input parameter, and we denote them as $\textbf{Prompt}_i = \left\{\mathbf{p}_i^k \in \mathbb{R}^d \mid k \in \mathbb{N}, 1 \le k \le m\right\}$. After that, we introduce Prompt in the input space of each Transformer layer of OSTrack, and the Prompt of this layer only serves as a hint to this Transformer layer. The Prompt of the previous layer is discarded before it enters the next Transformer layer, and then a completely new Prompt is introduced at the next Transformer layer. The input and output of each layer of Transformer are formulated as follows:

$$\left[\ldots, \mathbf{H_z}^i, \mathbf{H_x}_i\right] = L_i\left(\left[\textbf{Prompt}^{i-1}, \mathbf{H_z}^{i-1}, \mathbf{H_x}^{i-1}\right]\right) \quad i = 1, 2, \ldots, N. \tag{3}$$

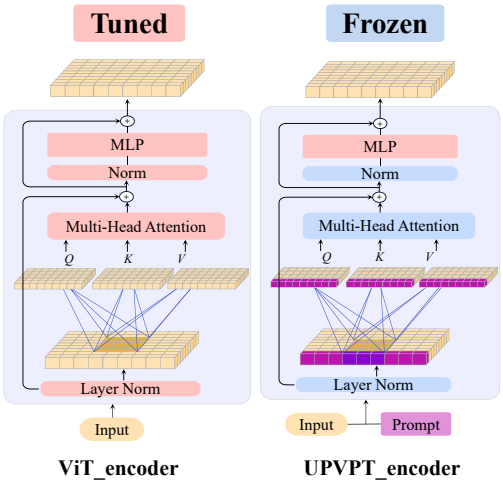

Figure 3: Illustration of the difference between the encoder layer in our proposed UPVPT with the encoder layer in the VIT models.

In addition, we observe that OSTrack proposes a candidate elimination module that speeds up the model's training with a slight performance reduction. To reflect the advantages of our approach and its openness, we keep the early elimination module of OSTrack.

3.3 Freeze or Tune

Currently, there are two mainstream fine-tuning approaches: One mainstream approach is adding a learnable Prompt, which helps the pre-trained model adapt to downstream tasks by guiding pre-training with additional cues through learning the cues. The other is the Adapeter tuning paradigm, where a new datapath is added around the MLP in the Transformer's structure, retaining the original MLP parameters. The new datapath is trained to achieve similar or even better results than the overall fine-tuning of the model. However, most conventional fine-tuning methods can only be adopted when the upstream and downstream tasks are consistent or close to each other. When facing significant task differences, achieving similar or even better results is more complex than overall fine-tuning.

In the object classification task, the model pays more attention to the differences between different classes and will suppress the differences within the class. This is significantly different from the target tracking task. Through experiments, we find that extracting features with intra-class differences is the key to learning the tracker loaded with the pre-trained model. There is no need to update all parameters, but only the parameters of the MLP layer can make up for this difference. The MLP adjustment will also allow the model to learn more knowledge from the object tracking data. We further explored the MLP layer in the Transformer structure. Transformer's MLP layer comprises two fully connected layers and the GELU layer, which strengthens the expressiveness of the features extracted from the previous layers and will enhance the critical features and suppress the unimportant ones. The formula of MLP can be formulated as follows:

$$T^{\ell} = \mathrm{MLP}\left(\mathrm{LN}\left(T^{\ell'}\right)\right) + T^{\ell'}, \ell = 1 \ldots L \tag{4}$$

4 EXPERIMENTS

We use the OSTrack-256 version as our experimental benchmark. We first describe our experimental setup and parameter settings. Next, we demonstrate the effectiveness of UPVPT by comparing experimental results in various aspects. Finally, we also do a large number of data experiments to improve the understanding of our method.

4.1 Experimental Details

Benchmark. To verify the effectiveness of our proposed UPVPT, we use the current OSTrack with excellent results as a benchmark and select the OSTrack-256 version for comparison.

Train. We follow the OSTrack training method and use training splits of GOT-10K[18], TrackingNet[28], COCO[24] and LaSOT[12] as the training dataset. The input to the network is an image pair consisting of a template patch of size 128×128 and a search patch of size 256×256. Our model was implemented in the Pytorch framework on a server with 1 NVIDIA V100 GPU. Our model is trained with 300 epochs, each epoch with 60,000 image pairs and each mini-batch with 64 sample pairs. We also train the model use the AdamW[26] optimizer, set the weight decay to 10^{-4}, the initial learning rate of the backbone to 1×10^{-5} and other parameters to 1×10^{-4}. After 240 epochs, the learning rate is decreased by a factor of 10.

Inference. We follow the original inference method, multiply the final obtained classification graph P with the Hanning window of the same size, and finally select the box with the highest score as the final tracking result.

4.2 Comparison Results

To demonstrate the effectiveness of our proposed approach, we evaluated the model on five challenging tracking benchmarks and compared it with the existing state-of-the-art trackers.

GOT-10k. The GOT-10k[18] dataset is important to validate the effectiveness of our approach. The test set of GOT-10K introduces a one-time tracker evaluation protocol, and the target classes in the training set are different from those in the test set, enabling avoidance bias in the evaluation of familiar objects. In our experiments, we follow this protocol to train our models and submit the results to an official evaluation server to assess the results. Benefiting from our new fine-tuning paradigm, our model has obtained a new and most advanced model on the GOT-10K data set and has achieved better results, getting an AO score of 0.727, $\mathrm{SR}_{0.5}$ score of 0.824 and $\mathrm{SR}_{0.75}$ score of 0.695, respectively.

LaSOT. In our experiments, we follow this protocol to train our models and submit the results to an official evaluation server to assess the results. As shown in Table.2, our result is very close to that of the original OSTrack, and the AUC score is 0.8% lower than that of OSTrack. However, it is still better than most existing trackers.

TrackingNet. The performance of our method is very close to that of OSTrack-256. Our AUC score is 82.8%, which is higher than most existing trackers. This result fully demonstrates the ability of our

Table 2: Comparison on the GOT-10k[18], LaSOT[12],TrackingNet[28] test set with the state-of-the-art. The best two results are highlighted in red and blue.

Method	GOT-10k			LaSOT			TrackingNet		
	AO	$SR_{0.5}$	$SR_{0.75}$	AUC	P_{Norm}	P	AUC	P_{Norm}	P
SiamFC[2]	34.8	35.3	9.8	33.6	42.0	33.9	57.1	66.3	53.3
MDNet[29]	29.9	30.3	9.9	39.7	46.0	37.3	60.6	70.5	56.5
siamRPN++[21]	51.7	61.6	32.5	49.6	56.9	49.1	73.3	80.0	69.4
SiamBAN[7]	-	-	-	59.4	-	-	71.6	68.5	79.4
SiamR-CNN[31]	64.9	72.8	59.7	64.8	72.2	-	81.2	85.4	80.0
DiMP[3]	61.1	71.7	49.2	56.9	65.0	56.7	74.0	80.1	68.7
ATOM[9]	55.6	63.4	40.2	51.5	57.6	50.5	70.3	77.1	64.8
Ocean[42]	61.1	72.1	47.3	56.0	65.1	56.6	-	-	-
MAMLTrack[32]	-	-	-	52.3	-	-	75.7	82.2	72.5
AutoMatch[41]	65.2	76.6	54.3	58.3	-	59.9	76.0	-	72.6
TrDiMP[35]	67.1	77.7	58.3	63.9	-	61.4	78.4	83.3	73.1
SparseTT[14]	69.3	79.1	63.8	66.0	74.8	70.1	81.7	86.6	79.5
STARK[38]	68.8	78.1	64.1	67.1	77.0	-	82.0	86.9	-
TransT[6]	67.1	76.8	60.9	64.9	73.8	69.0	81.4	86.7	80.3
OSTrack[39]	71.0	80.4	68.2	69.1	78.7	75.2	83.1	87.8	82.0
UPVPT	72.7	82.4	69.5	68.3	77.9	73.8	82.8	87.7	81.5

Table 3: Comparison on the UAV123[27] test set with the state-of-the-art. The best two results are highlighted in red and blue.

	Ocean[42]	ATOM[9]	STMTrack[15]	TransT[6]	STARK[38]	OSTrack-256[39]	Ours
UAV123	57.4	63.2	64.7	68.1	68.2	68.3	68.7

Table 4: Comparison on the TNL2K test set[36] with the state-of-the-art. The best two results are highlighted in red and blue.

	Ocean[42]	SiamBAN[7]	AutoMath[41]	ATOM[9]	TransT[6]	OSTracK-256[39]	SwinTrack-B/16[23]	Ours
AUC	38.4	41.0	47.2	49.1	50.7	54.3	54.8	55.8
P	37.7	41.7	43.5	39.2	51.7	-	53.8	56.7

method that learn the difference between tracking and classification tasks from the tracking task dataset.

UAV123. As shown in Table.3, benefiting from our new fine-tuning paradigm, our model has achieved better results on the UAV dataset, the AUC score of our method is 68.7, which is 0.4% higher than the OSTrack-256 version.

TNL2K. Benefiting from our new fine-tuning paradigm, our model has obtained a new and most advanced score on the TNL2K dataset, getting an AUC score of 0.558 and a P score of 0.567, respectively. Our experimental results are shown in Table.4, where we outperform OSTrack-256 by 1.5% in the AUC score.

4.3 Ablation Study and Analysis

Effect of expanding input sequence length. The effectiveness of our method has been demonstrated in many experiments. To prove whether our method's effectiveness is due to its extended input sequence length, we design a completely new experiment, where we freeze the Prompt parameters during the training process and update the rest of the model parameters. The results are shown in Table.5, we can see that updating the Prompt parameter gives a significant advantage to our method, and freezing the Prompt also gives better results than the original OSTrack-256 method.

Table 5: Comparison on the GOT-10k test set[18] with OSTrack. The best result is highlighted in red.

Tracker	AO	$SR_{0.5}$	$SR_{0.75}$
UPVPT	72.7	82.4	69.5
$UPVPT_{nofreeze}$	71.9	81.1	68.8
OSTrack	71.0	80.4	68.2

Applicability of the method. At this stage, we apply our UPVPT method to a tracker based on the Siamese framework to demonstrate the applicability of our method. We choose SwinTrack[23] as our benchmark for testing on two representative datasets (GOT-10k, LaSOT).

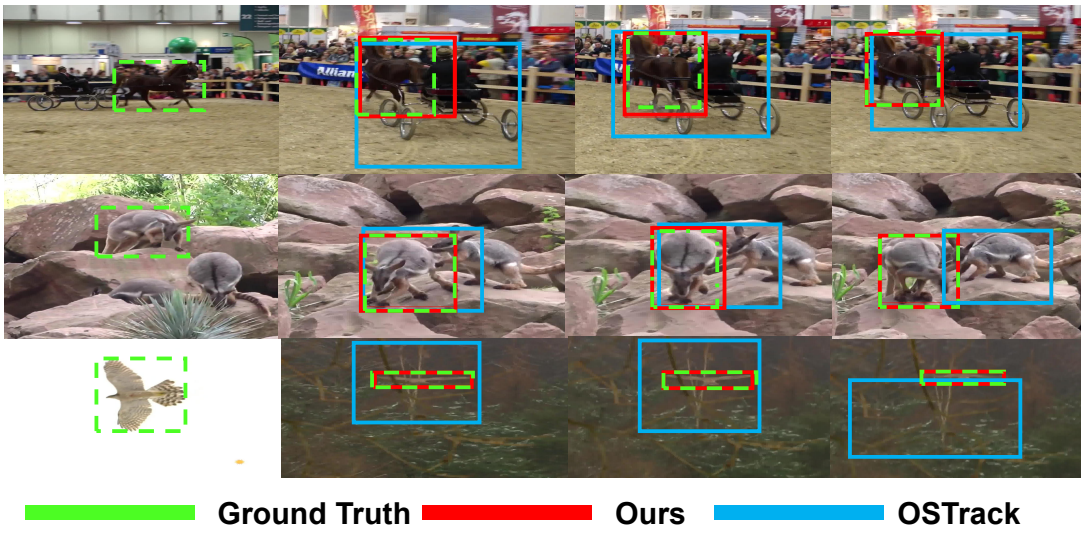

Figure 4: We present the qualitative comparison results of UPVPT with OSTrack for three challenging sequences from the GOT-10k test set. (It is worth noting that there is no open annotation information in the GOT-10k test set. For clarity, we use the green dashed boxes to indicate target annotations.)

Table 6: Comparison on the GOT-10k test set[18] with SwinTrack. The best one result is highlighted in red.

Tracker	AO	$SR_{0.5}$	$SR_{0.75}$
SwinTrack-T+UPVPT	69.5	79.4	62.4
SwinTrack-T[23]	69.0	78.1	62.1
SwinTrack-B[23]	69.4	78.0	64.3

Table 7: Comparison on the LaSOT test set[12] with SwinTrack. The best one results are highlighted in red.

Tracker	SUC	PRE	NPRE
SwinTrack-T+UPVPT	67.8	71.8	76.8
SwinTrack-T[23]	66.7	70.6	75.8

The results of the GOT-10k dataset are shown in Table.6. Our method scores 0.5% higher than SwinTrack-Tiny in AO, which is already higher than the SwinTrack-Base version. The $SR_{0.5}$ and $SR_{0.75}$ scores are also significantly higher than those of the SwinTrack-Tiny version. We also test our method on another representative LaSOT dataset, and the results are shown in Table.7. The SUC score of our method is 1.1% higher than that of the original SwinTrack-Tiny version, and the PRE score and NPRE score are also higher than those of SwinTrack-Tiny.

4.4 Visualization Analysis

To further clearly demonstrate the robustness of our approach, we provide some qualitative comparison results in Fig.4. Due to the prior knowledge of the pre-trained model, we show the performance of our model and OSTrack when facing sequences with target boundary challenges, similar object interference challenges, and background complexity challenges, respectively. For example, in the first sequence, the horse is the target object and OSTrack treats the whole carriage as the target object, while our UPVPT still accurately locates the horse.

5 CONCLUSION

In this work, we have explored an efficient training paradigm called UPVPT for partial fine-tuning of the backbone network to achieve robust tracking. In contrast with the traditional training paradigms of overall fine-tuning and the existing Prompt methods that freeze all network layers, our proposed UPVPT only updates a small number of learnable parameters and maintains prior information contained in a pre-trained model. Consequently, our proposed UPVPT can effectively mitigate catastrophic forgetting of pretrained models and achieve promising performance while reducing the number of parameters for learning the backbone network. We have implemented two variants of our method based on OSTrack with a one-stream structure and SwinTrack with a Siamese structure, respectively. The extensive experiments on five challenging data sets have validated the robustness and effectiveness of our method.

ACKNOWLEDGMENTS

This work was supported by the Project of Guangxi Science and Technology(No.2022GXNSFDA035079), the National Natural Science Foundation of China (No.61972167 and U21A20474), the Guangxi" Bagui Scholar" Teams for Innovation and Research Project, the Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing, the Guangxi Talent Highland Project of Big Data Intelligence and Application, and the Research Project of Guangxi Normal University (No.2022TD002).

REFERENCES

[1] Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassne. 2022. Robust Visual Tracking by Segmentation. *European Conference on Computer Vision,ECCV* (2022).

[2] Luca Bertinetto, Jack Valmadre, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. 2016. Fully-Convolutional Siamese Networks for Object Tracking. *European Conference on Computer Vision,ECCV* (2016).

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. 2019. Learning Discriminative Model Prediction for Tracking. *International Conference on Computer Vision, ICCV* (2019).

[4] Goutam Bhat, Joakim Johnander, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. 2018. Unveiling the Power of Deep Tracking. *European Conference on Computer Vision,ECCV* (2018).

[5] Boyu Chen, Peixia Li, Lei Bai, Lei Qiao, Qiuhong Shen, Bo Li, Weihao Gan, Wei Wu, and Wanli Ouyang. 2022. Backbone is All Your Need: A Simplified Architecture for Visual Object Tracking. *European Conference on Computer Vision,ECCV* (2022).

[6] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. 2021. Transformer Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2021).

[7] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, Rongrong Ji, Zhenjun Tang, and Xianxian Li. 2022. SiamBAN: Target-Aware Tracking With Siamese Box Adaptive Network. *IEEE Transactions on Pattern Analysis and Machine Intelligence,TPAMI* (2022).

[8] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. 2022. MixFormer: End-to-End Tracking with Iterative Mixed Attention. *Conference on Computer Vision and Pattern Recognition, CVPR* (2022).

[9] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. 2018. ATOM: Accurate Tracking by Overlap Maximization. *Conference on Computer Vision and Pattern Recognition, CVPR* (2018).

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. *Conference on Computer Vision and Pattern Recognition, CVPR* (2009).

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations,ICLR* (2021).

[12] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. 2018. LaSOT: A High-quality Benchmark for Large-scale Single Object Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2018).

[13] Luciano Floridi and Massimo Chiriatti. 2020. GPT-3: Its Nature, Scope, Limits, and Consequences. *Minds and Machines* (2020).

[14] Zhihong Fu, Zehua Fu, Qingjie Liu, Wenrui Cai, and Yunhong Wang. 2022. SparseTT: Visual Tracking with Sparse Transformers. *International Joint Conference on Artificial Intelligence,IJCAI* (2022).

[15] Zhihong Fu, Qingjie Liu, Zehua Fu, and Yunhong Wang. 2021. STMTrack: Template-free Visual Tracking with Space-time Memory Networks. *Conference on Computer Vision and Pattern Recognition, CVPR* (2021).

[16] Dongyan Guo, Jun Wang, Ying Cui, Zhenhua Wang, and Shengyong Chen. 2020. SiamCAR: Siamese Fully Convolutional Classification and Regression for Visual Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2020).

[17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked Autoencoders Are Scalable Vision Learners. *Conference on Computer Vision and Pattern Recognition, CVPR* (2022).

[18] Lianghua Huang, Xin Zhao, and Kaiqi Huang. 2022. GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[19] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual Prompt Tuning. *European Conference on Computer Vision,ECCV* (2022).

[20] Shibo Jie and Zhi-Hong Deng. 2022. Convolutional Bypasses Are Better Vision Transformer Adapters. *arXiv preprint arXiv:2207.07039* (2022).

[21] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. 2019. SiamRPN++: Evolution of Siamese Visual Tracking With Very Deep Networks. *Conference on Computer Vision and Pattern Recognition, CVPR* (2019).

[22] Xin Li, Chao Ma, Baoyuan Wu, Zhenyu He, and Ming-Hsuan Yang. 2019. Target-Aware Deep Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2019).

[23] Liting Lin, Heng Fan, Yong Xu, and Haibin Ling. 2021. SwinTrack: A Simple and Strong Baseline for Transformer Tracking. *arXiv: preprint arXiv:2112.00995* (2021).

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. *European Conference on Computer Vision,ECCV* (2014).

[25] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *arXiv: Computation and Language* (2021).

[26] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *International Conference on Learning Representations,ICLR* (2019).

[27] Matthias Mueller, Neil Smith, and Bernard Ghanem. 2016. A Benchmark and Simulator for UAV Tracking. *European Conference on Computer Vision,ECCV* (2016).

[28] Matthias A. Müller, Adel Bibi, Silvio Giancola, Salman Al-Subaihi, and Bernard Ghanem. 2018. TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild. *European Conference on Computer Vision,ECCV* (2018).

[29] Hyeonseob Nam and Bohyung Han. 2015. Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2015).

[30] Chunhong Pan, Qi Tian, Shiming Xiang, Zhaoxiang Zhang, Bolin Ni, Xing Nie, Jianlong Chang, Gaomeng Meng, and Chunlei Huo. 2022. Pro-tuning: Unified Prompt Tuning for Vision Tasks. *arXiv preprint arXiv:2207.14381* (2022).

[31] Paul Voigtlaender, Jonathon Luiten, Philip H. S. Torr, and Bastian Leibe. 2020. Siam R-CNN: Visual Tracking by Re-Detection. *Conference on Computer Vision and Pattern Recognition, CVPR* (2020).

[32] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. 2020. Tracking by Instance Detection: A Meta-Learning Approach. *Conference on Computer Vision and Pattern Recognition, CVPR* (2020).

[33] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. 2021. Transformer Meets Tracker: Exploiting Temporal Context for Robust Visual Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2021).

[34] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. 2021. Transformer Meets Tracker: Exploiting Temporal Context for Robust Visual Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2021).

[35] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. 2021. Transformer Meets Tracker: Exploiting Temporal Context for Robust Visual Tracking. *Conference on Computer Vision and Pattern Recognition, CVPR* (2021).

[36] Xiao Wang, Xiujun Shu, Zhipeng Zhang, Bo Jiang, Yaowei Wang, Yonghong Tian, and Feng Wu. 2021. Towards More Flexible and Accurate Object Tracking With Natural Language: Algorithms and Benchmark. *Conference on Computer Vision and Pattern Recognition, CVPR* (2021).

[37] Haiping Wu, Bin Xiao, Noel C. F. Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. 2021. CvT: Introducing Convolutions to Vision Transformers. *International Conference on Computer Vision, ICCV* (2021).

[38] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. 2021. Learning Spatio-Temporal Transformer for Visual Tracking. *International Conference on Computer Vision, ICCV* (2021).

[39] Botao Ye, Hong Chang, Bingpeng Ma, and Shiguang Shan. 2022. Joint Feature Learning and Relation Modeling for Tracking: A One-Stream Framework. *European Conference on Computer Vision,ECCV* (2022).

[40] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. 2022. Neural Prompt Search. *arXiv preprint arXiv:2206.04673* (2022).

[41] Zhipeng Zhang, Yihao Liu, Xiao Wang, Bing Li, and Weiming Hu. 2021. Learn to Match: Automatic Matching Network Design for Visual Tracking. *International Conference on Computer Vision, ICCV* (2021).

[42] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. 2020. Ocean: Object-aware Anchor-free Tracking. *European Conference on Computer Vision,ECCV* (2020).