Real Time Object Tracking For Autonomous Driving Using Model Compression

Unmesh Patil Bharghav Srikhakollu Aniket Poojari
Oregon State University
{patilun, srikhakb, poojaria}@oregonstate.edu

Abstract

In the realm of autonomous driving, efficient real-time object tracking is a crucial challenge that needs to be addressed. Numerous approaches have been suggested in both industry and academia to integrate object detection and tracking. However, these approaches often face limitations in balancing trade-off between achieving high accuracy and fast processing. Our objective is to enhance the processing speed of a high-accuracy end-to-end multi-object tracking model while minimizing the impact on the model's accuracy. In our study, we employed model compression techniques such as pruning and knowledge distillation to improve the performance of our model. Through the combination of these compression methods, we observed superior outcomes compared to the baseline approaches. We have provided an experimental evaluation to show the effectiveness of pruned teacher for the knowledge distillation and theoretical reasoning for the same.

1. Introduction

A crucial aspect of ensuring safe and efficient autonomous driving is the ability to accurately and reliably track multiple objects. Being able to identify and monitor multiple objects in real-time is vital for autonomous vehicles to make well-informed decisions and successfully navigate complex surroundings. This motive gave the insights about the importance of detection and tracking of objects in autonomous driving.

In the field of autonomous driving, Multi Object Tracking (MOT) [19] involves the detection and tracking of vehicles, objects, pedestrians, and other elements. Despite appearing as a relatively straightforward task, MOT presents various challenges, including occlusions and complex motion patterns. Fortunately, advancements in computer vision, machine learning, and deep learning have played a crucial role in addressing these challenges [16]. Several techniques have been suggested to address the challenge of multi-object tracking, including traditional methods that rely on feature extraction and matching, as well as newer approaches that utilize deep neural networks for comprehensive tracking from start to finish.

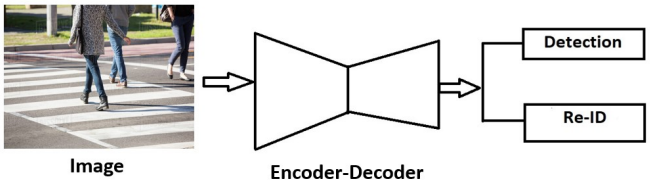

Figure 1. Detection and tracking of Multiple objects: FairMOT Tracker

In the domain of Multi Object Tracking, a commonly used approach involves a two-step process consisting of object detection using a CNN-based detector, followed by a separate component for Re-identification (Re-ID) [8, 10]. However, this pipeline tends to be slow for real-time applications, and it also has a tendency to produce biased object detection proposals. To address these limitations, the FairMOT algorithm was developed, which combines object detection and Re-ID tasks into a unified framework [21]. FairMOT is a multi-object tracker that jointly performs object detection and Re-ID tasks Figure (Figure 1). It employs the Resnet-34 architecture as the backbone and utilizes deep layer aggregation to fuse features from different layers of the network. In this approach, deformable convolutions are employed to dynamically adjust the receptive fields, replacing the traditional convolution layers. This allows for more flexible and adaptable processing.

FairMOT has a limitation due to its complex algorithm, which involves multiple tasks such as object detection, feature extraction, matching, and data association. These tasks increase the computational and resource demands, especially when dealing with large-scale tracking scenarios.

To address this issue, our focus is on reducing the algorithm's complexity through model compression. Model compression techniques aim to decrease the computational requirements or size of deep neural network models without compromising performance. The techniques include:

- Pruning [6], which eliminates unnecessary parameters

and connections in a neural network while maintaining the model's functionality.

- Knowledge Distillation [5], which involves transferring knowledge from a larger and more complex teacher model to a smaller student model by training the student to imitate the teacher's outputs.

- There is a third and slightly less explored method known called Quantization [18], which replaces floating point parameters of a Neural Network with low bit width numbers like integers or even binary numbers.

2. Related Work

The initial research efforts in multi-object tracking focused on separate models for detection and tracking. Prominent detection methods included DPM (Deformable Parts Model) and Faster R-CNN (Region-based Convolutional Neural Networks). However, subsequent advancements introduced more effective detection approaches, such as using VGG-16 [15] as a backbone for Faster R-CNN, which outperformed earlier methods in terms of detection performance. To further enhance detection capabilities, more powerful detectors like Cascade R-CNN [3] were employed.

In conjunction with the aforementioned detection techniques, tracking methods such as SORT (Simple Online and Realtime Tracking) [1] and IOU-Tracker [2] were developed. SORT employs the Kalman Filter, a mathematical model that estimates an object's state based on noisy measurements. This filter predicts an object's future state and updates its estimates with new measurements. The Hungarian algorithm is then utilized to associate these predicted tracks from the Kalman Filter with object detections in the current frame. While these approaches demonstrated effectiveness in scenarios with limited object motion, they faced challenges in crowded scenes and high-speed motion.

Several methods have been developed to handle fast motion and occlusion in detection and tracking tasks [7, 9]. However, many existing approaches that use separate models for detection and tracking are slow and struggle to achieve real-time inference, which is often required in various applications. To address this issue, researchers have explored the concept of using a single model for joint detection and tracking, specifically focusing on joint detection and Re-ID (identification) as well as joint detection and motion prediction. These methods leverage deep learning and multi-task learning techniques to learn both detection and motion features within a single network. However, joint detection and motion prediction methods [13, 22] face challenges in handling scenarios involving occlusion since they only associate objects in adjacent frames without re-initializing lost tracks.

Another approach, known as joint detection and Re-ID, combines object detection and re-ID feature extraction in a single network to reduce inference time. However, these methods [11, 17] tend to achieve lower accuracy compared to the traditional two-step process of separate detection and tracking. The FairMOT model [21] belongs to the joint detection and Re-ID class. It not only inherits the capabilities of existing single-model methods but also focuses on improving tracking accuracy without introducing complex modifications. FairMOT achieves this by performing long-range association using appearance features, thereby effectively addressing occlusion scenarios. Considering the enhanced end-to-end architecture for detection and tracking, we have selected FairMOT as our model for this research. Our work involves compressing the model to enhance its speed while preserving its accuracy.

In previous research, model compression techniques such as pruning and knowledge distillation have been employed separately. In our study, we propose an approach where we utilize FairMOT as our base model and apply a combination of pruning and knowledge distillation. Our objective is to assess the impact of integrating these two compression strategies on both processing efficiency and model accuracy. The concept of end-to-end model compression through knowledge distillation is inspired by the work referenced in [20]. However, our approach differs in that we combine pruning and knowledge distillation specifically on the FairMOT model.

To validate our approach before proceeding with actual model compression on the FairMOT model, we conducted preliminary experiments on a simpler problem. We utilized a convolutional neural network from our coursework, trained on the 3-class CIFAR dataset. Initially, we applied global pruning to this model and observed improved results compared to the base model. Subsequently, we further compressed the model using a teacher-student network, implementing knowledge distillation on top of the pruned model. This iterative process resulted in a lightweight model without sacrificing accuracy. Encouraged by these outcomes, we proceeded to apply our approach to the FairMOT model. The architecture, results, and detailed steps are elaborated in the subsequent sections.

3. Methodology

In this section we explain our approach to the problem statement. Firstly, we explain different model compression methods and how to apply them to a given problem. Next, we discuss about employing these methods on a toy problem. Finally, we elaborate on the architecture of FairMOT, dataset generation, training procedures and pruning of FairMOT.

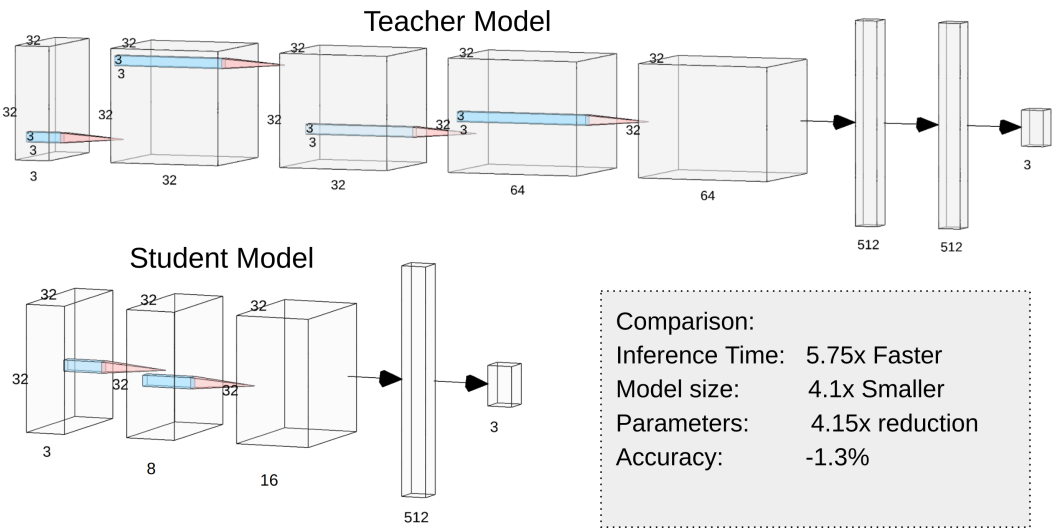

Figure 2. knowledge distillation on 3 class image classification task on CIFAR-10

3.1. Model Compression

Model compression is a general class of techniques employed to reduce model size and inference time. In the subsequent sections, we talk about principles and application of these methods.

3.1.1 Pruning

Pruning is a technique utilized in neural networks to decrease model complexity and enhance the speed of the network without compromising accuracy. It involves the removal of unnecessary weights connecting neurons. There are two types of pruning: structured pruning and unstructured pruning. For our research, we focused on unstructured pruning due to its sparsity and ability to perform precise pruning. Specifically, we employed L1 global pruning, which applies L1 regularization to eliminate weights. Through L1 regularization, weights are minimized based on their absolute values (L1 norm), resulting in numerous weights being set to zero and effectively removed from the network.

The process of L1 global pruning typically entails training the neural network, calculating weight importance, establishing a prune threshold, pruning the network, and fine-tuning the pruned network. In our initial experimentation (toy problem), we pruned both convolutional and fully connected layers. However, in our larger model approach, we focused on pruning only the convolutional layers.

3.1.2 Knowledge Distillation

The process involves initially training a large model and then utilizing it as a teacher to educate a smaller model. Instead of employing hard class labels while training the student model, the fundamental concept of model distillation is to utilize the soft probabilities or logits generated by the teacher. These probabilities carry more information about the input. Although there has been extensive research on distillation, a significant challenge lies in achieving a high compression ratio solely through distillation.

Since we utilize logits generated by the teacher, we created a separate loss function for student network. It is a Mean Squared Error (MSE) loss function that computes error between teacher network's logit values and student network's logit values. So the student network learns to match logits with that of the teacher network. We first test this method on a previously mentioned toy problem by creating a model with much fewer convolution filters. After testing and understanding the method, we deploy it to improve the accuracy of a lite MobileNetV2 [14] model using a complex VGG-16 network as a teacher. The loss function in this case was adopted from [4].

Instead of MSE loss function, we are using the below loss function for knowledge distillation for the image classification task.

$$\frac{1}{m}\sum_{j=0}^{m}\left\{2T^2\alpha D_{KL}(P^{(j)},Q^{(j)}) - (1-\alpha)\sum_{i=1}^{c} y_i^{(j)}\log(1-\hat{y}_i^{(j)})\right\} \tag{1}$$

The knowledge distillation loss is calculated using the KL Divergence loss between the softmax outputs of the students and teacher models. The KL Divergence is between the log softmax of the student model's outputs divided by the temparature (T) and the softmax of the teacher model's outputs divided by the temparature. These softmax functions are used to convert the raw logits into probability distributions. The loss function also includes the original en-

tropy loss which measures the discrepancy between the predicted logits of the student model and the ground truth labels.

3.1.3 Quantization

Quantization is reported to be very useful when memory and/or computational resources are severely restricted. Moving from memory and computation consuming floating-point representations to fixed integer values represented in four bits or less is proven to reduce memory footprint and latency by a factor of 16x in some applications [18]. PyTorch 2.0 library provides a feature to implement quantization (still in beta). We implemented quantization in the Eager mode provided by PyTorch 2.0, which allows us much flexibility on implementation level in terms of placement of quant and dequant operators. In simple terms, these operators allow us to convert between floats and integers. There are two ways of employing quantization, the first one is to do a post training quantization and the second one is called as Quantization Aware Training (QAT). We implemented post training quantization and during implementation, we had to remove the BatchNorm operation due to issues in fusing modules for Quantization. We have reported results of our Quantization experiments on the a image classification toy problem, but due to limited time, we don't have quantization results for the MOT problem.

3.1.4 Pruning + Knowledge Distillation

As previously mentioned, knowledge distillation is a process that involves transferring knowledge from a larger teacher network to a smaller student network. In certain studies, a pruning technique has been proposed where the teacher network is pruned prior to performing the teacher-student knowledge distillation, resulting in more effective knowledge transfer [12]. This approach typically consists of three steps: 1) training the teacher network, 2) pruning the teacher network, and 3) distilling the pruned network into the smaller student network. The paper [12] presents a particular outcome showcasing the success of pruning the teacher network before distilling it to a smaller student network using models such as ResNet18, VGG16, and MobileNetV2. This concept served as inspiration for our research, where we aimed to combine pruning and knowledge distillation for an efficient model compression technique. By testing our pruning task with the optimal pruning threshold in conjunction with knowledge distillation, we observed improved results.

4. Results

This section starts with the experimental and implementation level details from our work and then it continues to report results from the image classification toy problem and the MOT problem.

4.1. Experiments

All the experiment listed below were carried out on a NVIDIA GTX 1650 GPU (4GB) with Cuda 12 and PyTorch 2.0. Evaluations were conducted on Intel Core i7-9750H CPU @ 2.60GHz processor (16 GB RAM).

4.1.1 The Toy Problem

The toy problem consists of a simple convolutional neural network with the following structure: conv1, relu1, conv2, pool1, relu2, conv3, relu3, conv4, relu4, pool2, fc1, Batch Norm, relu5, and fc2. We employed the Adam optimizer with a learning rate of 1e-4 and weight decay of 0.001. The model was trained for 50 epochs.We conducted training and evaluation using a batch size of 256. The PyTorch implementation of this convolutional neural network was employed for the CIFAR dataset, which consists of three classes.

Initially, we explored unstructured L1 global pruning with empirically chosen pruning rates 0.1, 0.2, 0.45 and 0.6. Based on the results, we determined that a pruning rate of 0.2 or 20% yielded better test and validation accuracy for the model while maintaining the sparsity trade-off. Pruning results are detailed in the table (Table 1).

Following the pruning process, we proceeded with knowledge distillation. In this context, the main neural network served as the teacher network, while we designed a smaller network to act as the student network. We empirically designed a smaller network with following structure: conv1, relu1, conv2, pool1, relu2, fc1, Batch Norm, relu5, and fc2. Both student and teacher architectures are shown in the Figure (Figure 2) as well. In this work, we also addressed a novel direction of studying distillation performance of pruned networks. Since pruning encourages sparse representations and promotes better generalization, we expect it to be a good teacher as compared to a complex over-fitted model.

4.1.2 The MOT problem

The Multi-Object Tracking problem is addressed in two different set of experiments in our work.

1. We implemented pruning method on the End-To-End method FairMOT [21].]

2. We tested knowledge distillation on the VGG-16 feature extractor.

The FairMOT method shows results using multiple feature extractors. These feature extractors are major bottlenecks in the MOT problem. So, we later decided to implement

Model	Train Acc.	Val. Acc.	Test Acc.	Inf. Time	Model size	Parameters
Base model	98.46	95.55	94.53	1.15s	26 MB	2.16 M
40% pruned	**99.1**	95.7	95.06	0.91s	26 MB	2.16 M
20% pruned	98.98	**95.77**	**95.07**	0.94s	26 MB	2.16 M
Student	93.22	92.4	91.66	0.31s	6.4 MB	**0.52 M**
Base + KD	95.92	93.75	92.0	0.21s	6.4 MB	**0.52 M**
Pruning + KD	97.25	94.65	93.3	**0.2s**	6.4 MB	**0.52 M**
Quantization (without BN)	87.5	86.1	85.85	0.8s	**2.2 MB**	2.16 M

Table 1. Results obtained on the toy problem (Image Classification on CIFAR-10) with different model compression techniques

knowledge distillation only on the feature extractor part of the network. Mainly used feature extractors are VGG-16, DLA, HRNet-18 etc. We also compare performance of one of these high accuracy feature extractor with YOLOv5s and our pruned version of it. For the dataset and relevant details please refer to (Link: https://github.com/ifzhang/FairMOT). The global pruning pruning was applied during training to all available convolutions in the model structure. The amount of pruning was kept at 20%.

In the second experiment, we tested the effect of knowledge distillation by fine-tuning a student network with MobileNetV2(SSDLite) architecture and a teacher network with VGG-16 architecture. The knowledge distillation loss function for this experiment is explained in detail in [4]. We used VOC2007 dataset and a pretrained VGG-16 model (available online) for this task. We employ mAP (Mean Average Precision) as a metric to evaluate the performance.

4.2. Results

This section is divided between the image classification problem and the MOT problem. In this section we discuss the results and reasoning behind the results that led us to a set of conclusions.

4.3. Image Classification Results

This particular task allowed us to understand and compare different model compression methods in the literature.

4.3.1 Pruning and Distillation

In Table 1, we have presented comparative study of different compression techniques. Comparing the base model (explained in the experiments section) with pruned models, we observed improved training, validation and testing accuracy. We attribute this increase to the induced sparsity which has regularizing effect which helps the model to learn better representations. However, the size of the model remains same after pruning. This is due to the lack of support for sparse tensors in convolutions in the PyTorch library. This also explains why the inference time is not much improved (exposing the underlying zero-operations). We also observed that with increasing sparsity in the network, the model starts to struggle to capture fine-grained details and nuances in the data resulting in reduced accuracy.

Knowledge distillation experiments were conducted with a previously mentioned student architecture. The student model showed fairly good independent performance considering the small architecture. With the base model as a teacher network, the student network was able to achieve improved accuracy but the improvement was not significant. The reason for this is in the complexity of the base model which can easily lead to overfitting and hence it is not the best teacher. For this reason, we used pruned model as a teacher as it is sparse and well generalized. This change in teacher showed significant progress in the performance of the student network. In summary, we found that pruned networks are better teachers for knowledge distillation.

Quantization results show huge reduction in the size of the model, however due to float to integer conversions at the time of inference hinder the improvement in the inference time. In addition to that, as mentioned before, we had to remove the BatchNorm operation which affects the performance of this method. Many functions related to this module in PyTorch are still under active development but our preliminary results definitely show the potential of this method.

4.4. MOT Results

4.4.1 Pruning

In this section we explain the results shown in table 2. For these results FairMOT was trained on MOT-15 dataset which has images and object tracking labels. This dataset is considered particularly difficult to achieve higher MOTA scores. From our literature survey, we found that the tracking results are mostly limited by detection heads and that's why we decided to focus on feature extractor networks. The DLA-34 network is one of the best performers on feature extraction but it is computationally very expensive. So we

FairMOT + model	mAP	FPS	Model size
DLA-34	**0.47**	0.2	259.4 MB
YOLOv5s	0.42	3.45	68 MB
YOLOv5s + Pruning	0.39	**3.47**	**60 MB**

Table 2. Results obtained on the FairMOT model with different feature extractors. Effect of pruning is not significant here because of the lack of sparse tensor support for convolutions. The FPS performances are calculated on a intel i7 CPU and are limited by compute.

Model	Teacher	mAP	Model size
VGG-16	None	0.75	107.3 MB
MobileNetV2	None	0.68	14.2 MB
MobileNetV2	VGG-16	0.72	14.2 MB
Pruned VGG-16	None	0.75	107.3 MB
MobileNetV2	Pruned VGG-16	0.69	14.2 MB

Table 3. Results obtained by training VGG-16 and MobileNetv2 (SSDLite) on VOC2007 datset. Effect of pruning is not significant but the knowledge distillation technique shows promising improvement in the performance of MobileNetv2

replaced it with YOLOv5s and found that we could achieve many folds better FPS rate with a minute difference in mAP score. The 20% pruned network, however does not achieve a good mAP score and shows a meager improvement in the FPS and the model size.

4.4.2 Knowledge Distillation + Pruning

Motivated from the results of distillation of pruned teacher networks, we implemented the similar approach for two popular feature extractors used in many MOT solutions. Our aim here was to improve the precision of MobileNet by using the expertise of VGG-16 network. It is promising to see that the performance of MobilNet improved by a considerable amount after distillation compared to the standalone performance. This clearly shows an agreement between student and teacher network. However, the pruned method did not show any significant improvement in the performance. We postulate that this is a result of our poor pruning strategy for VGG-16. A good iterative pruning strategy is needed to create a sparse and more generalized teacher model.

5. Conclusion

We proposed a potential solution to address a need of faster and more accurate object detection and tracking module in the self-driving car industry. Firstly, we learned about the effectiveness of various model compression methods available in the literature by applying them on a relatively simple image classification problem. From these experiments we draw a significant hypothesis about the effectiveness of pruned teacher models for knowledge distillation. We provide a theoretical reasoning for this increased distillation efficiency by the induction of pruned teacher. We attribute this improvement to the regularizing effect and sparse representations resulting from pruning.

However, the simpler pruning strategies that we employed were not able to succesfully prune complex networks like VGG-16. Therefore, the next logical step would be to come up with better pruning strategies for complex and large networks, so that they can be used for knowledge distillation. In turn, this will end up creating more energy-efficient and memory efficient deep neural networks.

References

[1] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016. 2

[2] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2017. 2

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6154–6162, 2018. 2

[4] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems*, 30, 2017. 3, 5

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2

[6] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989. 1

[7] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. Bag of tricks and a strong baseline for deep person re-identification, 2019. 2

[8] Nima Mahmoudi, Seyed Mohammad Ahadi, and Mohammad Rahmati. Multi-target tracking using cnn-based features: Cnnmtt. *Multimedia Tools Appl.*, 78(6):7077–7096, mar 2019. 1

[9] Nima Mahmoudi, Seyed Mohammad Ahadi, and Mohammad Rahmati. Multi-target tracking using cnn-based features: Cnnmtt. *Multimedia Tools Appl.*, 78(6):7077–7096, mar 2019. 2

[10] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking, 2016. 1

[11] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking, 2021. 2

[12] Jinhyuk Park and Albert No. Prune your model before distill it, 2022. 4

[13] Jinlong Peng, Changan Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking, 2020. 2

[14] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 3

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 2

[16] Ganglin Tian, Xinyu Zhang, Shichun Guo, Yuchao Liu, Xiaonan Liu, and Kun Wang. Occlusion handling based on motion estimation for multi-object tracking. In *2021 IEEE International Conference on Unmanned Systems (ICUS)*, pages 1031–1036, 2021. 1

[17] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition, 2020. 2

[18] Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen, and Kailash Gopalakrishnan. Training deep neural networks with 8-bit floating point numbers. *Advances in neural information processing systems*, 31, 2018. 2, 4

[19] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric, 2017. 1

[20] Wei Zhang, Lingxiao He, Peng Chen, Xingyu Liao, Wu Liu, Qi Li, and Zhenan Sun. Boosting end-to-end multi-object tracking and person search via knowledge distillation. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, page 1192–1201, New York, NY, USA, 2021. Association for Computing Machinery. 2

[21] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. FairMOT: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision*, 129(11):3069–3087, sep 2021. 1, 2, 4

[22] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points, 2020. 2