References

1. Baldock, R., Maennel, H., Neyshabur, B.: Deep learning through the lens of example difficulty. Advances in Neural Information Processing Systems **34**, 10876–10889 (2021) 6
2. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple Online and Realtime Tracking. In: Proceedings of the 23$^{\text{th}}$ IEEE International Conference on Image Processing (ICIP). pp. 3464–3468 (2016) 4, 10, 18, 23
3. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934 (2020) 1, 3
4. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001) 10
5. Chang, M.F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al.: Argoverse: 3D Tracking and Forecasting with Rich Maps (2019) 3, 4, 5, 9
6. Courdier, E., Fleuret, F.: Real-time segmentation networks should be latency aware. In: Proceedings of the Asian Conference on Computer Vision (2020) 3
7. Dendorfer, P., Rezatofighi, H., Milan, A., Shi, J., Cremers, D., Reid, I., Roth, S., Schindler, K., Leal-Taixé, L.: Mot20: A benchmark for multi object tracking in crowded scenes. arXiv preprint arXiv:2003.09003 (2020) 20
8. Gog, I., Kalra, S., Schafhalter, P., Wright, M.A., Gonzalez, J.E., Stoica, I.: Pylot: A Modular Platform for Exploring Latency-Accuracy Tradeoffs in Autonomous Vehicles. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA) (2021) 3, 4
9. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both Weights and Connections for Efficient Neural Networks. In: Proceedings of the 28$^{\text{th}}$ International Conference on Neural Information Processing (NeurIPS). pp. 1135–1143 (2015) 3
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 24, 25
11. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 3
12. Jiang, J., Ananthanarayanan, G., Bodik, P., Sen, S., Stoica, I.: Chameleon: Scalable Adaptation of Video Analytics. In: Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM). pp. 253–266 (2018) 3, 4
13. Leal-Taixé, L., Milan, A., Schindler, K., Cremers, D., Reid, I., Roth, S.: Tracking the trackers: an analysis of the state of the art in multiple object tracking. arXiv preprint arXiv:1704.02781 (2017) 2, 20
14. Li, M., Wang, Y., Ramanan, D.: Towards Streaming Perception. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020) 1, 2, 3, 4, 6, 9, 10
15. Lin, J., Rao, Y., Lu, J., Zhou, J.: Runtime Neural Pruning. In: Proceedings of the 31$^{\text{st}}$ International Conference on Neural Information Processing Systems (NeurIPS). pp. 2178–2188 (2017) 3
16. Lin, S.C., Zhang, Y., Hsu, C.H., Skach, M., Haque, M.E., Tang, L., Mars, J.: The Architectural Implications of Autonomous Driving: Constraints and Acceleration. In: Proceedings of the 23$^{\text{rd}}$ International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS). pp. 751–766 (2018) 4

17. Liu, T.Y., et al.: Learning to rank for information retrieval. Foundations and Trends® in Information Retrieval **3**(3), 225–331 (2009) 6
18. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 21–37. Springer (2016) 1
19. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017) 25
20. Luiten, J., Osep, A., Dendorfer, P., Torr, P., Geiger, A., Leal-Taixé, L., Leibe, B.: Hota: A higher order metric for evaluating multi-object tracking. International journal of computer vision **129**(2), 548–578 (2021) 20
21. Luo, J.H., Wu, J., Lin, W.: Thinet: A Filter Level Pruning Method for Deep Neural Network Compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5058–5066 (2017) 3
22. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K.: Mot16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831 (2016) 2, 19, 20
23. Mittal, G., Liu, C., Karianakis, N., Fragoso, V., Chen, M., Fu, Y.: HyperSTAR: Task-Aware Hyperparameters for Deep Networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8736–8745 (2020) 3
24. Nishi, K., Shimosaka, M.: Fine-Grained Driving Behavior Prediction via Context-Aware Multi-Task Inverse Reinforcement Learning. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 2281–2287. IEEE (2020) 8
25. NVIDIA: Tensor RT. https://developer.nvidia.com/tensorrt 10
26. Padilla, R., Passos, W.L., Dias, T.L., Netto, S.L., da Silva, E.A.: A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics **10**(3), 279 (2021) 20
27. Pang, J., Qiu, L., Li, X., Chen, H., Li, Q., Darrell, T., Yu, F.: Quasi-dense similarity learning for multiple object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 164–173 (2021) 9, 25
28. Park, E., Ahn, J., Yoo, S.: Weighted-entropy-based Quantization for Deep Neural Networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5456–5464 (2017) 3
29. Pleskac, T.J., Busemeyer, J.R.: Two-stage dynamic signal detection: a theory of choice, decision time, and confidence. Psychological review **117**(3), 864 (2010) 3
30. Shen, H., Han, S., Philipose, M., Krishnamurthy, A.: Fast Video Classification via Adaptive Cascading of Deep Models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3646–3654 (2017) 3
31. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and Efficient Object Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 1, 3, 4, 10
32. Wang, X., Yu, F., Dou, Z.Y., Darrell, T., Gonzalez, J.E.: Skipnet: Learning Dynamic Routing in Convolutional Networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 409–424 (2018) 3
33. Waymo Inc.: Waymo Open Dataset. https://waymo.com/open/ 2, 3, 4, 5, 9, 12
34. Xu, Y., Wang, Y., Zhou, A., Lin, W., Xiong, H.: Deep Neural Network Compression with Single and Multiple Level Quantization. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018) 3
35. Yogatama, D., Mann, G.: Efficient transfer learning method for automatic hyperparameter tuning. In: Artificial intelligence and statistics. pp. 1077–1085. PMLR (2014) 6, 24

36. Yu, F., Xian, W., Chen, Y., Liu, F., Liao, M., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving video database with scalable annotation tooling. arXiv preprint arXiv:1805.04687 **2**(5), 6 (2018) 25
37. Zhao, R., Hu, Y., Dotzel, J., De Sa, C., Zhang, Z.: Improving Neural Network Quantization without Retraining using Outlier Channel Splitting. In: International Conference on Machine Learning (ICML). pp. 7543–7552 (2019) 3
38. Zhou, X., Koltun, V., Krähenbühl, P.: Tracking objects as points. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 474–490. Springer (2020) 6, 24

Supplementary Material: Table of contents

A Metaparameter choice

In this study we use two metaparameters (detection model and tracking maximum age) out of several that we considered initially. These two metaparameters were chosen because (*i*) they represent 83% of the opportunity gap conferred by all the metaparameters considered together (see below) and (*ii*) a limited number of metaparameters allows computing the Octopus dataset on more scenarios.

I. Metaparameters we did not use:

• **Detection confidence threshold**: sets the minimum confidence score of bounding-box predictions to be used by the tracker.

• **Minimum matching tracker IOU**: sets the minimum bounding-box IOU (Intersection-Over-Union) that the SORT tracker [2] requires to perform obstacle association across frames.

• **Tracking re-initialization frequency**: specifies every how many frames detection is run to update the tracker. When detection is not run, the SORT tracker falls back on a Kalman filter to linearly extrapolate existing bounding box movement. This option bypasses the detection inference latency, but quickly incurs significant error.

The additional metaparameter ranges are:

```
# Additional Metaparameters
detection-confidence = {0.4, 0.6, 0.7}
minimum-matching-tracker-IOU = {0.1, 0.2, 0.3}
Tracker-re-initialization-frequency = {1, 2, 3}
```

Listing 1.2: Values for the other metaparameters.

II. The relative contribution of the two metaparameters we use. The new metaparameter cartesian product (Listing 1.2) is of size $6 * 3 * 3 * 3 * 3 = 486$ configurations. Due to computational constraints, we create a version of the Octopus dataset using 49 training and 23 test scenarios from the Waymo dataset, with $\tau = 1s$.

Table 5: Metaparameter contribution to the opportunity gap. The top row shows the score of the global-best policy where no metaparameter changes. Each row below shows the score of the optimal dynamic policy as one additional metaparameter is allowed to change. The incremental score increase is shown on the right.

Dynamic metaparameters	S-MOTA score	Score increase
None (global best policy)	24.0	-
+Detection Model	28.3	4.3
+Tracking maximum age	30.1	1.8
+Tracking re-init frequency	30.5	0.4
+Tracking minimum IOU	31.0	0.5
+Detection confidence threshold	31.3	0.3

Table 5 shows the relative contribution of each metaparameter to the score opportunity gap (§3.3) between the baseline global best policy and the optimal dynamic policy. The first two metaparameters (detection model and tracking maximum age) are used in our study and achieve 83% $((30.1-24.0)/(31.3-24.0))$ of the opportunity gap conferred by using all 5 metaparameters. We leave investigation into optimizing the rest of the metaparameters for future work.

B Training details

During training, we exclude video segments with no ground truth labels, where the MOTA score [22] is undefined. Then, at test time, we impute the global static configuration computed over the training dataset. We clip the regression targets to [-100, 100] (i.e., $\epsilon = 100$ in Eq. (2)). In our evaluation setup we consider an IOU of 0.4 between a prediction and ground truth to be a true positive. We train the models and evaluate using this schema. We use a single GPU (instead of 3 used in Waymo) for evaluation on the Argoverse dataset, because the high framerate would require over 7 GPUs to support executing detection in parallel on every frame.

C Training hyperparameters

The training hyperparameters used in training the regression and classification models are shown in Table 6.

Table 6: Training Hyperparameters.

Method	Max Tree Depth	Max # of Features	# of Estimators	Min Impurity Decrease
Regression	20	18	400	0.000186
Classification (Joint)	8	3	400	0.000285
Classification (Independent)	7	4	200	0.000529

The rest of the hyperparameters are the default used in Scikit-Learn v0.23.2.

D S-MOTA vs S-MOTP

In this work, we demonstrate that environment context can be leveraged to perform test-time optimization of tracking in streaming settings. Unlike other perception tasks (e.g. detection [26]), where a single optimization metric is commonly used, in multi-object tracking there is no consensus on the best metric [7,20]. Therefore, in this study we chose to optimize MOTA [22], as it is the metric that most closely aligns with human perception of tracking quality [13].

Nevertheless, a battery of other tracking metrics is presented in the evaluation §5.2. The main results (Table 3) show that the S-MOTA-optimal policy deteriorates in S-MOTP, and that our learned policy does the same in the Argoverse dataset. This occurs because MOTA and MOTP are designed to describe fundamentally different properties in tracking [22]: Whereas MOTA equally balances precision, recall and identification, MOTP only focuses on precise obstacle localization. Indeed, Table 7 empirically confirms the conflict between these metrics. The S-MOTP-optimal policy achieves a far lower S-MOTA score (23.0) than a policy that optimizes directly for S-MOTA (31.2), and vice versa for S-MOTP (75.6 and 71.0, respectively). The conflict between the S-MOTA and S-MOTP scores can be illustrated as a pareto frontier in Figure 7 (generated by linearly interpolating these metrics).

To further visualize the different strategies needed to maximize each metric, we compare the decision frequency of the S-MOTA-optimal and S-MOTP-optimal policies (Figure 8). As shown, the S-MOTP-optimal policy concentrates on the fastest-models (D3, D4) and on a max-age value of 1, whereas the S-MOTA-optimal policy's decisions are much more spread out. This takes place because these models maximize predicted obstacle localization precision by minimizing prediction lag after the moving ground-truth. The low max-age minimizes error from SORT's Kalman filter (data not shown). Using weaker models with a low max-age, however, incurs a higher rate of false-negatives and ID-switches, which reduces S-MOTA (Table 7). We leave further investigation for future work.

Table 7: Evaluating dynamic policies that optimize S-MOTA and S-MOTP.

Method	S-MOTA↑	S-MOTP↑	S-FP↓	S-FN↓	S-ID_{sw}↓
S-MOTA-optimal	31.2	71.0	28907	590847	6997
S-MOTP-optimal	23.0	75.6	31468	640261	9424

E Study-case video

A video of the study case presented in §5.3 is attached with the supplementary material (`study-case.mp4`). Bounding box and ID annotations are added,

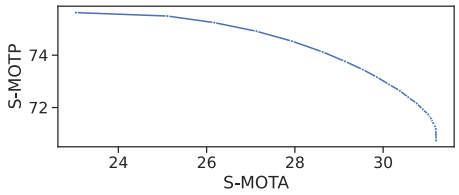

Fig. 7: The pareto frontier of optimal S-MOTA vs. S-MOTP optimizing policies. The S-MOTA (x-axis) and S-MOTP (y-axis) of optimal policies with gradually varying weights (blue curve) from S-MOTP to S-MOTA.

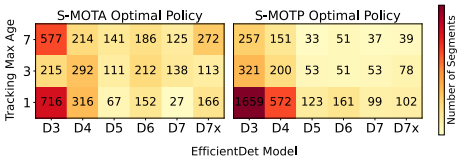

Fig. 8: Different policy strategies are needed to optimize S-MOTA and S-MOTP. The configuration choice frequency (color intensity) of S-MOTA-optimal policy (left) and of the S-MOTP-optimal policy (right).

matching the color schema of the line-plot that shows the S-MOTA scores of the different policies.

The 10-12 second mark in the video illustrates a large difference between the Octopus policy (green) and the global best policy (black). This difference occurs because the Octopus policy successfully tracks obstacles adjacent to the road on the left and even more on the right that the global best policy does not.

F t-SNE visualization

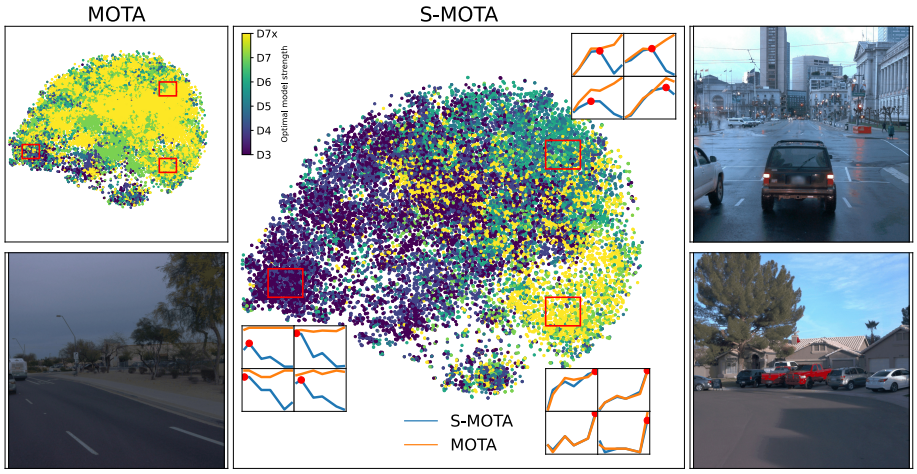

Fig. 9: MOTA and S-MOTA responses in different regions of the t-SNE plot, with video-segment visualization. This figure illustrates the following: (*i*) The t-SNE figures presented in Fig. 5 (*ii*) Three line-plot quartets, illustrating the MOTA and S-MOTA response (y-axis) of nearby points in the figure to detection model size (x-axis), (*iii*) A visualization of a representative scenario for each quartet.

We present an expanded analysis of the t-SNE plot in §5.3. The figure is presented again in Figure 9 with more annotations.

Proximity in score space. Three distinct regions in the t-SNE plot are selected and marked with red rectangles in Figure 9. Four points, representing four 1-second video segments, are selected in each region. The MOTA and S-MOTA scores (y-axis) of increasing detection model size (x-axis) are plotted for these scenarios, following the same methodology as Figure 1b and Figure 2 in §1. The y-axis ticks and scale are omitted to emphasize that nearby video segments have similar, *normalized*, MOTA and S-MOTA response to the metaparameters. These scenarios, therefore, also tend to be optimized by the same metaparameter values.

Case studies. We perform an in-depth analysis of a video segment selected from each of the three regions indicated in red in Figure 9.

Bottom-left: Turning in difficult conditions. The ego-vehicle turns left into a highway. Larger models cannot detect the parked cars in the background, and therefore do not boost offline MOTA over smaller models. At the same time, vehicle turning induces high obstacle-displacement in the frame, causing accuracy deterioration from higher inference latency. This scene is both very difficult (minimal offline MOTA boost from bigger models) and fast (high accuracy deterioration).

Bottom-right: Slow movement towards partially-occluded vehicles. Stronger models better detect the partially-occluded parked vehicles. The slow ego-vehicle movement towards non-moving obstacles induces minimal obstacle-displacement and therefore negligible accuracy degradation.

Top-right: A mix of still and moving obstacles. The vehicle is standing at an intersection. Still or parked cars on the road are mixed with fast-moving pedestrians. The intermediate-size models, EfficientDet-D5 and D6, detect most of the still obstacles and are able to keep up with the fast-moving pedestrians. Larger models (EfficientDet-D7 and D7x), however, introduce too much inference runtime delay in tracking the moving pedestrians, and sustain more severe accuracy deterioration as a result.

Taken together, these examples illustrate that the environment context can be used to infer MOTA and S-MOTA response to the metaparameters. This then enables test-time S-MOTA optimization by dynamically tuning the metaparameters.

G Full Centroid Visualization

Full score space clustering analysis. The clustering analysis discussed in §5.3 (Figure 6) is extended to all eight centroids, shown in Figure 10. Several clusters present similar, though slightly shifted behavior, to the ones presented in the body of the paper. For example, cluster centroids 1, 5, and 8 represent environments with varying S-MOTA response to increasing latency. This demonstrates that the performance penalty (degradation) may occur at different stages as the model size increases. Clusters 3, 4, and 6 show similar improvement in S-MOTA with larger models, though in cluster 6, the performance drops for the two largest

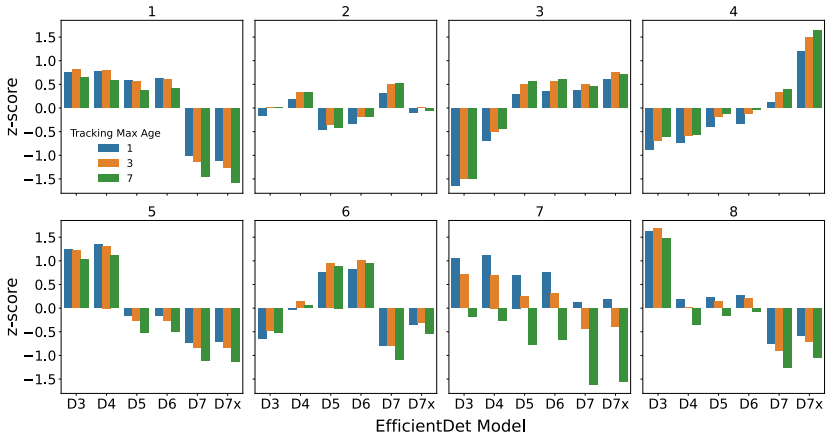

Fig. 10: All eight centroids in the clustering analysis performed in §5.3.

models, i.e., EfficientDet-D7 and D7x. These models' predictions are evaluated 3 frames away from the ground truth, whereas the rest of the models are evaluated up to 2 frames away. This may raise the possibility that the increase in frame gap from the ground truth causes a more significant degradation to S-MOTA in cluster 4 if it contains faster moving scenes than cluster 6.

Cluster scenario visualization. Curated videos of several of the clusters' video segments are attached to the supplementary material (`cluster-{4, 7, 8}.mp4`) The cluster numbers correspond to the centroid number plotted in Fig. 10.

When comparing clusters 4 and 8, we observe a clear trend: (*i*) video segments with a preference for larger models (cluster 4) show more subdued obstacle displacement from the ego vehicle camera's perspective and (*ii*) the opposite in video segments with preference for smaller models (cluster 8).

In cluster 7, a lower tracking maximum age has better performance. Here, we observe video segments with two primary modes of obstacle behavior: (*i*) occlusions or obstacles leaving the scene, and (*ii*) very fast obstacle movement, where the IoU of the obstacle to its previous location is small, causing SORT [2] to do an ID switch. In both cases, maintaining the old tracklets for a shorter amount of time reduces false positives and improves performance.

These observations illustrate the idea that visible properties of the environment context may be leveraged to discern the mode of metaparameter score behavior. These properties are used as features in Octopus to optimize S-MOTA at test time.

H Ranking Implementation Ablations

We conduct an ablation study of the experimental setup and the design choices.

Baseline subtraction. As a variance reduction technique, we consider regressing over the relative score improvement from h^{global} instead of absolute scores as described in 4.1. Thus, we avoid dedicating model capacity to learn environment

"difficulty" properties, which apply to all configurations in a manner that does not change their score order. As this distinction is not relevant for predicting configuration order, it only introduces noise to the downstream ranking task [35]. Accordingly, in Table 8, we observe a 0.3 point decrease in S-MOTA score on the Waymo dataset when baseline subtraction is ablated.

Classification *vs.* regression. We compare the regress-then-rank approach described in §4.1 against the classification formulation (using Random Forests). This approach directly predicts the metaparameter values of the best configuration from the environment features using m classifiers for each of the m metaparameters. In Table 8 we show the resulting score in "classification (joint)", indicating a reduced score compared to regression with baseline subtraction.

Independent *vs.* joint metaparameter optimization. We examine the degree of independence between metaparameter choices in the optimization process. To this end, we consider a new setting "classification (independent)" in Table 8. This is done by separately learning to predict the value of each metaparameter, while holding the other metaparameter values constant (in essence assuming convexity). We show that joint classification performs 0.4 S-MOTA points better than independent.

Table 8: Comparing configuration ranking approaches

Method	S-MOTA↑	S-MOTP↑	S-FN↓	S-FP↓	S-ID$_{sw}$↓
Regression w/ baseline subtraction	27.9	72.3	31489	608870	8966
Regression w/o baseline subtraction	27.6	72.5	31405	610617	8982
Classification (independent)	27.2	72.9	32725	615537	9326
Classification (joint)	27.6	72.7	35107	611196	8829

I Policy Design Using Neural Networks

The key idea in this study is that the environment context can be leveraged to optimize streaming tracking accuracy at test time. The Octopus policy model presented is implemented using engineered features and random forest regression. A natural question is whether the policy performance can be improved using deep neural networks. To this end, we evaluated policy design that incorporates a conventional, convolutional neural network (CNN) [10] as well. We found that performance is on par with the global best policy, much worse than the solution using engineered features (see Table 9). In order to get good performance, we believe that the model needs to predict features at the granularity of instance-level motion (e.g. instance-flow [38]) because the tracking score that is being predicted is defined at this granularity. We believe that the model does not get good score because it does not capture these features well. Further investigation is needed.

I.1 Methodology

We use a ResNet50 backbone [10] that takes as input the middle frame of each 1-second video segment and classifies the best configuration using a new MLP. The backbone is pretrained using QDTrack [27] on the BDD100k dataset [36] (QDTrack's zero-shot accuracy on Waymo is 40.3, on par with the optimal policy score in offline settings). The backbone output is average-pooled and fed to an MLP of width of 256 with one hidden layer. The MLP output is fed into separate linear layers that generate the logits for each metaparameter in order to separately classify the values of the optimal configuration. This follows the methodology described in Appendix H. The model is trained using the cross-entropy loss on the Waymo dataset, using the methodology described in §5.1. The logits are initialized with a small bias so that the model behaves like the global best policy at the start of training. The model is trained using AdamW [19] with a learning rate of 1e-4 and weight decay of 0.01 for 10 epochs.

I.2 Results

The results are shown in Table 9. The CNN achieves negligible performance improvement over the global best policy.

Table 9: Neural-network based policy performance

Method	S-MOTA↑	S-MOTP↑	S-FP↓	S-FN↓	S-ID$_{sw}$↓
Global best	25.1	72.2	33616	633159	11212
Neural network based policy	25.2	72.3	45658	611827	8056

J Faster hardware simulation

Hardware performance is expected to improve over time. We therefore evaluate Octopus on faster hardware execution by simulating 50% faster inference of the detection model measured on the V100 GPU. We repeat the evaluation on the Argoverse dataset described in §5.2, showing the results in Table 10. The results show that the Octopus policy with closed-loop prediction outperforms the global best static policy by 2.8 S-MOTA, up from the 1.7 S-MOTA result using the V100 GPU latency readings.

Table 10: Performance on Argoverse with 50% faster GPU inference

Method	**S-MOTA↑**	**S-MOTP↑**	**S-FP↓**	**S-FN↓**	**S-ID$_{sw}$↓**
Global best	54.5	77.3	8508	44965	1173
Optimal	63.7	75.5	6897	36634	649
Optimal from the prev. segment	57.6	75.5	9655	40800	776
Octopus with:					
Ground truth from current segment	59.3	75.9	7951	40405	815
Ground truth from prev. segment	57.9	76.3	7988	41553	958
Prediction from prev. segment	**57.3**	**76.4**	**7911**	**42489**	**958**