Hyperparameter Optimization using Grid Search for use in Monocular Depth Estimation

Calvin Barrett and Tomohide Bessho

August 12, 2020

Abstract

Autonomous vehicles uses a myriad of sensors to observe and monitor its surroundings. Current research incorporates static digital images that are used to train and optimize deep neural networks. These deep neural networks are used to identify and determine the distances of objects from the autonomous vehicle. Unfortunately, the hyperparameters used to train these networks were chosen arbitrarily. We explored the hyperparameter space through a two-step grid search in order to recognize the optimal combination of losses before continuing to train the rest of the system to understand the trade offs between constant parameters. Specifically, the research focuses on finding the optimal relation between the left-right consistency, appearance, disparity losses in generating accurate feature edges in our output images. The trade off between costs allows for the neural network an increased performance at edge detection.

1 Introduction

It is critical to consider the different parameters and hyperparameters needed for the convolutional neural networks (CNN) to operate when working with any machine learning problem utilizing neural networks. While parameters within our network are learned from features present among batches of images, hyperparameters are operator-defined at the onset of the training process [4]. Examples of hyperparameters for our CNN include the batch size, number of training epochs, and the weights for our loss function. These weights and their relation to network accuracy comprise the center of our research. The CNN is utilized to generate a color-coded filter over an image to aid in depth estimation. Image post-processing techniques are applied to the filtered image to reduce occlusion on objects. The resulting smooth edges allow for more accurate depth estimation. In application to autonomous vehicles, this edge-detection process allows for more accurate spatial analysis by the vehicle while relying only on static RGB images. Thus the safety and strategic ability of the vehicle are heightened without the traditional use of expensive laser arrays. In this work we show that, by tuning the network hyperparameters before training, we are able to achieve a greater degree of accuracy on driving data-set benchmarks with no additional operating cost.

2 Related Works

This work is centered around the VGG convolutional network with added Atrous Spatial Pyramid Pooling (ASPP) module [5] created to process monocular RGB images and reduce edge occlusion for accurate depth estimation. This derivative of the VGG network, VGGASPP, created by Peng et. al is a VGG network with an ASPP layer within the structure of the model [5] in order to capture multi-class features. This network utilizes a cost function that weighs different costs from other state-of-the-art network designs [3]. The researchers considered various techniques in order to approach this problem of optimization with focus placed on the speed of training each succession of optimization with such a large dataset. Others have used techniques such as simulated annealing or genetic algorithms [4] to gradually optimize the hyper-parameters in their networks. However these approaches come at the expense of computing resources and time. Genetic algorithms are modelled after biological evolutionary processes. Each hyperparameter is modelled as a gene and each gene is the fitted with a function that determines the accuracy and verification time with the given model. An analysis of the fitness is conducted on each gene and is determined whether the fitness calculated is sufficient through an auxiliary function. This process is repeated multiple times until each hyperparameter is successfully optimized for the given model [4].

A different approach to optimizing hyperparameters is a grid search. Grid search considers and tests discrete intervals from within a range of numbers, allowing for researchers to understand the time needed to

arrive at the best optimization afforded by the interval [1]. Furthermore, individual hyperparameters can be kept static to understand the impact of each individual weight on network accuracy.

3 Methodology

Researchers originally considered other optimization techniques like those highlighted in the previous section. Simulated annealing and genetic algorithms were considered due to to their history of high-accuracy optimization but ultimately were not chosen due to the intense computational resources needed for both techniques to converge. The high computational load on the computing environment required the optimization technique used to have a relatively low compute cost as well as the ability to train and test multiple instances at once. These criteria further invalidated optimization techniques that relied on a progressive approach to hyper-parameter optimization, including genetic algorithms and simulated annealing. Grid search offered the ability to train and test modified networks concurrently while not needing to converge on a value over many training iterations, and was ultimately chosen by the researchers as the most-applicable optimization method.

The objective function from Godard et. al [5, 3] was used to make sure that the results from this experiment could be compared with the previous work to evaluate the performance of the model with the different parameters. The neural network uses this objective function as a loss function to determine the error between the measured disparities and the ground truth. The objective function is defined as a sum of three terms: appearance matching (C_{ap}), disparity smoothness (C_{ds}), and left-right consistency (C_{cor}) [5]. The definitions of these three terms can be found in [1].

$$C_s = \alpha_{ap} \times C_{ap} + \alpha_{ds} \times C_{ds} + \alpha_{lr} \times C_{cor} \tag{1}$$

where the hyperparameters ($\alpha_{ap}, \alpha_{ds}, \alpha_{lr}$) correspond to each term in the function. Appearance matching considers the output of the autoencoder, and compares it to the input image. Disparity smoothness correlates to drastic local disparities within the image, which would suggest occlusion. Finally, left-right consistency is considered when inputting an image into the network. This image is flipped, and the network predicts a disparity map for the flipped image after convolution. This predicted, flipped map is compared to the original image's predicted map. [3] Left-right consistency and disparity smoothness are the hyperparameters that were varied in the experiment. The original weights of this function were (1, 0.5, 1) respectively [5]. We seek to find optimal values for these hyperparameters to minimize the cost function and thus increase network accuracy.

4 Experiment

Our work compared the tuned-hyperparameter VGGASPP network's accuracy to the previous network. Both networks were trained and evaluated with the same parameters used to benchmark the previous network [5]. Specifically, we trained our network using the same batch size of 8 and epoch count of 100 as the previous network. We also benchmarked hte networks on the same data-set, KITTI 2015. This was to ensure that only the intentional changes to the hyperparameters affected network accuracy.

4.1 Dataset

We used the KITTI benchmark [2] to evaluate the performances of the different hyperparameters on the model. Similar to Peng et. al, we used 29,000 image pairs for the training set [5], and the remaining images to evaluate the accuracies of the models with the updated hyperparameters. We trained the models with with the same batch size and epoch count as Peng et. al on the images from the KITTI 2015 dataset.

4.2 Metrics and Implementation

We continued to use the same metrics and network implementation as prior work [5]. To execute a grid search on the hyperparameters of the network, the disparity smoothness and the left-right consistency weights were altered. The two weights were evaluated at values .25, .5, .75, and 1, giving 16 total models. For each new

combination of these two parameters, the model was trained with the new parameters and then evaluated. During each evaluation, the model is evaluated without any post-processing, with post-processing, and edge-guided post-processing. The methods of post-processing and edge-guided post-processing are explained in the previous work [5].

The appearance weight was kept constant due to its connotation within an autoencoder network structure. This weight is kept constant since it is desired that the reconstruction of the image in our model should match the input, as this signals that the network has learned all the necessary features of the image. Decreasing the weight of this hyperparameter may have adverse affects on accuracy. It would potentially allow for less accurate internal reconstructions, thus decreasing the amount of necessary features learned.

4.3 Results

The results from the experiments are shown in Table 1. Each of the models were evaluated by calculating the absolute relative error between the model's predicted disparities and the ground truth, the square relative error, the root mean square error, and the log of the root mean square error. If the value of any of these errors are lower than the original values, they are bolded in the tables, as they show improvement. The models were also evaluated with the individual losses of each term. If the value of these losses are higher than the original values, then they are emphasized in the tables, as they show improvement as well. The baseline values are given on the top of the table (denoted by the default parameters in the table). These are the values that are used to compare the evaluated values of updated parameter's models.

As shown in section A of the appendix, when α_{lr} is set to 0.25 and α_{ds} is set to 0.5, every error metric is lower than the baseline (except for the log_rms). This would suggest that the network is able broadly attain greater accuracy when the left-right disparity weight is significantly decreased in relation to the original hyperparameter value. In addition, when α_{lr} is set to 0.75 and α_{ds} is 0.5, more than half of the error metrics are lower than the baseline. Specifically, the D1_all metric is significantly lower than the value yielded when training the network with the original hyperparameters. This metric is computed when evaluating the network on the KITTI Stereo Evaluation 2015 benchmark, and suggests that the network would also benefit in this benchmark by slightly decreasing the left-right consistency weight. However, this metric is actually even lower when significantly decreasing α_{lr}, such as when $\alpha_{lr} = 0.25$. This would suggest that a network would be able to generate even better benchmark scores on Stereo 2015 with a low α_{lr}, but this finding is still significant as it increases network accuracy without drastically reducing α_{lr}. Applications where the left-right consistency is of high importance would have more practical use for a solution such as when $\alpha_{lr} = 0.75$ as the left-right consistency is still given a prominent weight in the cost function.

5 Conclusion

Our evaluation has demonstrated that the hyperparameters in the VGGASPP network are able to be tuned to achieve even greater accuracy in monocular edge detection. However, the relatively large step size that the researchers used opens the possibility that hyperparameters can be tuned to an even greater degree to achieve similarly greater accuracy. The left-right consistency hyperparameter displayed the largest impact on accuracy in our tests. Decreasing the weight on this parameter would suggest that we now generate images with more occlusion, but the benchmark tests on KITTI 2015 prove that this is not the case. During experimentation, there was no immediate correlation between the disparity smoothness and left-right consistency hyperparamater. Considering the role of disparity smoothness in evaluating local disparities, it would be useful to address this in future work. Increasing the value of this hyperparameter may produce even more accurate results for the network. Tuning these parameters also demonstrated that, after applying edge-guided post-processing discussed in prior work [5], the network consistently evaluated as more accurate than the same network without the added post-processing. This supports prior results that maintain that reduced occlusion within the generated images results in a network more accurate to ground truth data [5].

References

[1] James Bergstra and Yoshua Bengio. "Random Search for Hyper-Parameter Optimization". In: *J. Mach. Learn. Res.* 13.null (Feb. 2012), pp. 281–305. ISSN: 1532-4435.

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? The KITTI Vision Benchmark Suite". In: May 2012, pp. 3354–3361. ISBN: 978-1-4673-1226-4. DOI: 10.1109/CVPR.2012.6248074.

[3] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised Monocular Depth Estimation with Left-Right Consistency". In: *CoRR* abs/1609.03677 (2016). arXiv: 1609.03677. URL: http://arxiv.org/abs/1609.03677.

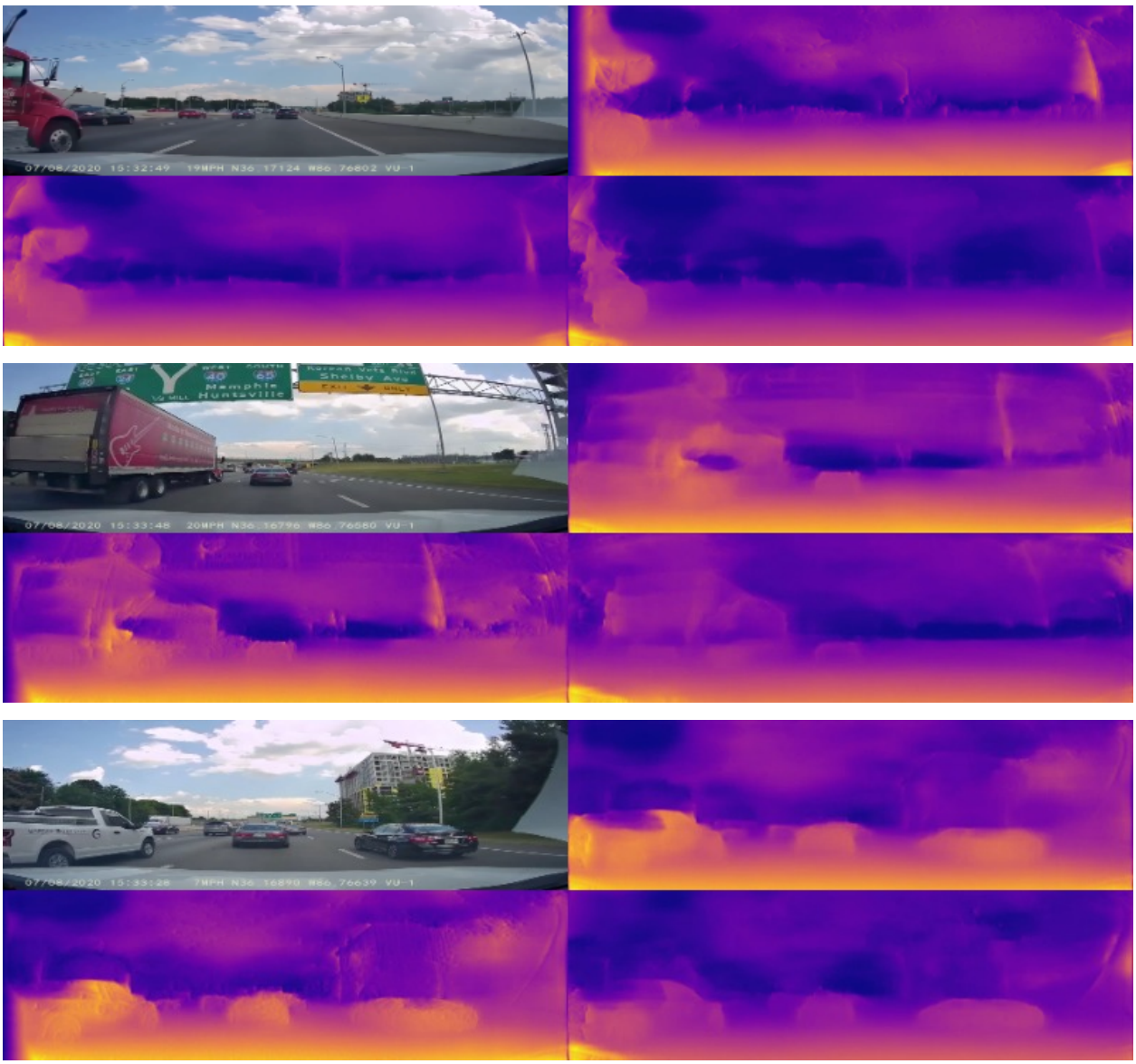
[4] Ji-Hoon Han et al. "Hyperparameter Optimization Using a Genetic Algorithm Considering Verification Time in a Convolutional Neural Network". In: *Journal of Electrical Engineering Technology* 15 (Jan. 2020). DOI: 10.1007/s42835-020-00343-7.

[5] J. Rozenblit K.-S. Peng G. Ditzler. *Edge-Guided Occlusion Fading Reduction for a Light-Weighted Self-Supervised Monocular Depth Estimation*. 2019. URL: https://arxiv.org/abs/1911.11705.

A Image Comparisons

We include qualitative results to better visualize the difference in performance of our updated neural network. The top left-corner image the original dash cam footage from the autonomous vehicle. The top-right image shows the previous neural network [5] without any edge guided post processing. The occlusion around the red truck and the light poles are abundant. The bottom-left image shows the previous neural network with edge guided post processing [5]. The occlusions decrease; however, there is still heavy occlusion around the nearby red truck on the left side of the image. Lastly, the bottom-right image shows our updated network when $\alpha_{lr} = 0.25$ and $\alpha_{ds} = 1.0$. There is less occlusion around the light poles and the red truck when comparing with the previous images.

Figure 1: Qualitative Results Comparing Previous Network [5] to Our Updated Network



B Evaluated Results of the Updated Parameters for the Model

α_{lr}	α_{ds}	abs_rel	sq_rel	rms	log_rms	d1_all	a1	a2	a3
1 (Default)	.5 (Default)	0.1128	1.1528	5.735	0.201	27.367	0.857	0.946	0.978
		0.1063	0.9913	5.4	0.191	26.021	0.864	0.951	0.982
		0.1057	0.9815	5.309	0.187	25.727	0.867	0.867	0.983
0.25	0.25	0.1176	1.1837	5.965	0.213	29.598	0.84	0.936	0.976
		0.1123	1.0585	5.75	0.205	28.707	0.842	0.938	0.978
		0.1118	1.0497	5.638	0.2	28.38	0.846	***0.941***	0.979
	0.5	***0.1116***	***1.0944***	***5.688***	0.202	***27.264***	0.856	0.944	0.978
		0.1047	***0.9482***	***5.397***	0.191	***25.505***	0.862	0.949	0.981
		0.1041	***0.9424***	***5.292***	***0.186***	***25.117***	0.866	***0.952***	0.983
	0.75	0.1174	***1.1263***	5.779	0.208	29.198	0.846	0.942	0.978
		0.1124	1.0106	5.537	0.198	28.095	0.849	0.946	0.98
		0.1122	1.0064	5.43	0.194	27.873	0.852	***0.948***	0.982
	1	0.1186	***1.148***	5.842	0.208	30.83	0.852	0.94	0.978
		0.1133	1.0249	5.647	0.2	30.35	0.847	0.944	0.981
		0.1126	1.0128	5.553	0.195	29.922	0.851	***0.947***	0.982
0.5	0.25	0.1205	1.2505	6.007	0.214	29.938	0.844	0.937	0.975
		0.1132	1.0812	5.701	0.203	28.64	0.849	0.943	0.978
		0.1125	1.0741	5.586	0.198	28.092	0.854	***0.947***	0.98
	0.5	0.1147	1.1978	5.825	0.203	***27.215***	0.855	0.946	0.978
		0.1068	1.0015	5.416	***0.19***	***25.592***	0.862	0.951	0.982
		0.1061	0.9987	5.321	***0.185***	***25.161***	0.866	***0.954***	***0.984***
	0.75	0.1198	1.2712	6.046	0.212	30.055	0.843	0.939	0.977
		0.1126	1.0924	5.711	0.2	28.391	0.849	0.945	0.98
		0.1124	1.1009	5.623	0.196	28.055	0.853	***0.948***	0.982
	1	0.1214	1.2296	5.894	0.21	30.301	0.841	0.94	0.978
		0.1134	1.0666	5.588	0.198	28.262	0.849	0.945	0.981
		0.1132	1.0726	5.534	0.195	28.049	0.852	***0.948***	0.982
0.75	0.25	0.118	1.1649	5.794	0.206	28.244	0.848	0.941	0.978
		0.112	1.0521	5.567	0.197	26.792	0.854	0.945	0.98
		0.1116	1.0441	5.49	0.193	26.535	0.856	***0.947***	0.981
	0.5	***0.1108***	***1.0981***	5.737	***0.2***	***26.805***	0.855	0.946	***0.979***
		0.1059	***0.9816***	5.461	0.192	***25.867***	0.86	0.949	0.982
		0.1055	***0.9768***	5.368	0.187	***25.509***	0.863	***0.952***	0.983
	0.75	0.1187	***1.1431***	5.864	0.212	29.719	0.843	0.938	0.975
		0.1126	1.0399	5.646	0.203	27.944	0.848	0.941	0.978
		0.1118	1.0226	5.528	0.198	27.563	0.852	***0.944***	0.98
	1	0.124	1.3573	6.038	0.21	28.797	0.847	0.941	0.976
		0.1169	1.2086	5.748	0.201	27.113	0.854	0.945	0.979
		0.1165	1.2187	5.693	0.197	26.775	0.857	***0.948***	0.98
1	0.25	0.1214	***1.1507***	5.886	0.213	32.534	0.832	0.937	0.977
		0.1158	1.0312	5.656	0.204	30.908	0.838	0.942	0.979
		0.1156	1.038	5.591	0.2	30.682	0.841	***0.944***	0.981
	0.75	0.1214	1.2535	5.946	0.209	28.864	0.846	0.941	0.977
		0.1145	1.1083	5.651	0.199	27.083	0.854	0.944	0.979
		0.1141	1.1064	5.542	0.194	26.61	0.858	***0.948***	0.98
	1	0.1282	1.234	5.977	0.215	35.599	0.837	0.937	0.976
		0.1219	1.0866	5.674	0.206	33.801	0.843	0.942	0.979
		0.1215	1.0799	5.575	0.202	33.528	0.846	***0.945***	0.98