Learning SfM, an Unsupervised Way

Abhijit Mahalle
University of Maryland
College Park, MD

I. Introduction

In this project, we explored an unsupervised learning approach to estimate depth and egomotion from the kitti dataset. SfmLearner published by Zhou et al [1] is an end to end unsupervised learning approach to regress depth and egomotion from image sequences. It has a depth network and a pose network from which the predictions are used to projective transform a target image view to source image view to compute loss in an unsupervised fashion. The aim of the project is to enhance the performance of sfmLearner by innovating with different loss functions, variable learning rate, architectural changes and data augmentations. We progressively attempted to improve the model's performance and conducted a various set of experiments that aid to analyse the successes and failures in our approaches .

II. Methodology

In this section, we describe the various improvisation techniques carried out to enhance the SfMLearner.

Loss functions

We first attempted to improve the model performance by bringing in a new loss function. SfMLearner has three types of losses. Photometric loss, smoothness loss/regularization and explainability loss. We mainly focused on the smoothness and photometric loss and discarded explainability loss which was even excluded by the authors of SfMLearner in the later revisions of the work.

To compute the Photometric loss in SfMLearner, we need to obtain the depth and pose predictions from the depth and pose networks in the pipeline, which can be used to projective transform a target image view I_t to the source image view $\hat{I}_s$. This is called view synthesis. The actual source image view is denoted as I_s. Let a pixel coordinate in source image is given as p_s and that of target image is given as p_t. The photometric loss measures the high-level similarity between a target image and the warped source image. In SfMLearner, it is a simple L1 norm computed between source and target view. However, this loss computed based on pixel intensities assumes that constant brightness and contrast between source and target frames, which can be often violated in practical settings. The structural information of a scene is independent of illumination and contrast. Inspired by [2], we computed the structural similarity metric (SSIM) between the target and source view. SSIM provides a robust metric for measuring perpetual differences between two images by considering the 3 factors of luminance, contrast and structure. The photometric loss is a weighted sum of the structural similarity based loss and the L1 photometric loss.

$$L_{pixel} = \alpha \sum_s \frac{1 - SSIM(I_t, \hat{I}_s)}{2} + (1-\alpha) \left\| I_t - \hat{I}_s \right\|_1 \quad (1)$$

α is the weighting parameter and was chosen to be 0.85.

The issue with the view synthesis, is that the gradients are mainly derived from the pixel intensity difference between $I(p_t)$ and the four neighbours of $I(p_s)$. This which would affect learning if the actual p_s (which can be projected using the ground-truth depth and pose) is located in a region of low-texture or far from the current estimation. One way to address this is to explicitly define a multi-scale, smoothness loss that allows gradients to be derived from larger spatial regions directly. In SfMLearner this loss was computed by minimizing the L1 norm of the second-order gradients for the predicted depth map. To improve smoothness loss, we utilise an edge-aware depth smoothness loss used by [2]. This computes a cost based on the gradients of the depth map and multiplies with a weighing factor based on the exponential inverse of the gradients of the actual image. This is given as

$$L_{smooth} = \sum_s |\Delta D(p_t)|.(e^{-|\Delta I(p_t)|})^T \quad (2)$$

The total loss is the sum of photometric and smoothness losses.

Architecture

As known before, SfMLearner uses a depth and pose network, and specifically the depth network uses VGG-16 architecture at the encoder stage. Inspired by [2], we tried to investigate the performance of Resnet architecture at the encoder stage of the depth network instead of VGG-16.

A. *Variable Learning Rate*

The learning rate was set to be 0.0002 in the SfMLearner. We decided to experiment with variable learning rate, by decreasing the learning rate by 35% after 25000 iterations.

B. Data Augmentation

SfmLearner by default uses random scaling and cropping as data augmentation procedures. Though, we could perform so many different types of data augmentation procedures image recognition problems, it intuitively did not make sense for us to perform operations like vertical flipping or rotation in dataset of road driving sequences as well as the fact that the loss is computed based on view synthesis from source and target image view. However, sudden changes in scene brightness are often occuring in practical settings and hence it was understandable to consider these as data augmentation procedures.

III. Training and Evaluation setting

We utilised the tensorflow code base of SfMLearner made modifications and improvements in this implementation. All the models were trained with the given dataset of 12000 images for 200,000 iterations with adam optimizer of $\beta_1 = 0.9$. The training was done in a laptop computer with i7 10750H CPU and 2070 MaxQ GPU which took about 7.5 hours for each model to train. To evaluate the depth models, we downloaded the raw kitti data and utilised eigen test data split corresponding to dates ”2011_9_26” and ”2011_9_28”. This comprised of about 547 images. To evaluate the pose models we downloaded the raw odometry image dataset and utilised sequence number 9 for our results. This sequence comprised of about 1587 images. Since we are training with significantly lesser amount of data compared to the actual implementation by the authors, we faced few many issues in figuring out the correct set of parameters suitable to provide successful results. This is further addressed in next section

IV. Metrics

For depth, the authors have used absolute relative error, squared relative error, root mean square error(RMS) and log RMS error. Lower the values of the first three quantities, better is the model performance. The evaluation is done on KITTI data set which has corresponding LiDaR(velodyne) data and the images from left and right camera which can be used for stereo depth. the depth predicted by the model is compared with depth map generated from the above mentioned quantities.

For pose, Absolute Trajectory Error (ATE) have been used. The ground truth data is provided by the authors using which the error metrics are obtained.

For measuring accuracy, we define three thresholds. If the ratio of predicted to ground truth values or vice versa is greater than this threshold, we reject those values.

V. Experimentation, Debugs and Discussion

A. Improving the model.

1) Default scores: First we tried to train a model of SfM Learner without making any tweaks to the default settings provided by the authors. The model had about 33 million trainable parameters. The loss over the course of training this model is shown in 4. We noticed that the loss has only slightly converged (from 1.12 to 0.9) and was highly fluctutating. For some strange reason, we were unable to obtain a good disparity map test result at the end of training and the results were extremely blurred out images. The pose and depth evaluation results of this model is denoted as ”default” in table I respectively.

2) Introducing SSIM: We tried to improve from whatever result that we obtained in the default case. So we altered the default photometric loss and incorporated SSIM as discussed in previous section. We did not include the smoothness loss in this model. The learning rate was fixed the same at 2×10^{-4} as the default SfMLearner. The pose and depth evaluation results of this model is denoted as ”ssim” in table I. We saw a marginal improvement in performance from the ”default” to ”ssim” case which signifies that inclusion of the SSIM loss is a step towards the right direction. Nevertheless, the results were still unfavourable. The depth maps obtained in the ”default” and ”ssim” case is shown in 1. The loss curve of ”ssim” starts off at 1.7 and converges further.

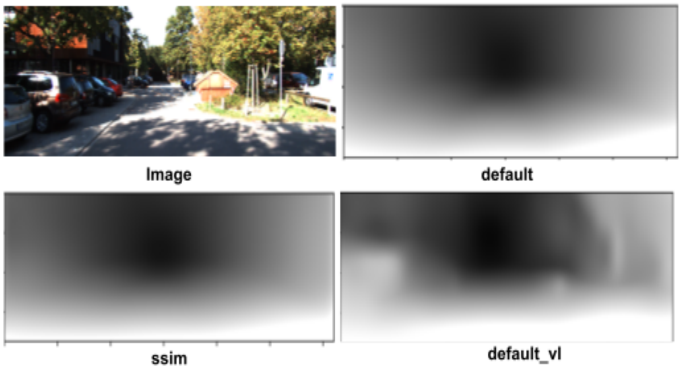

Fig. 1. depth maps of failed experiments

3) Learning the right Learning Rate: In both the cases, the model’s learning rate was fixed at 2×10^{-4}. Since we suspected that reason for the bad results could be due to low learning rate, we decided to run a subset of 1000 images for 50,000 iterations with a high learning rate of 2×10^{-3} and see if the loss converges. The loss converged very well the model was unable to make any meaningful predictions and returned a blank image. So we decided to train our models with all the 12,000 image sequences by starting at this learning rate 2×10^{-3} and decreasing it by 35% every 25,000 iterations. However, the depth maps that resulted on testing this model was just a plain blank image. We realised we made a wrong decision as the choice of starting learning rate at 2×10^{-3} was too high and hence reverted back to 2×10^{-4}. We suspected a bug in the depth network of the SfMLearner that we were using and hence, we decided to rewrite the depth network again with the same vgg16 encoder. Now, we again trained this updated ”default” model but this time with this approach of variable learning rate and the results of this model is denoted as ”default_vl” in I. The depth maps were still blurry for ”default_vl” model but considerably better looking.

TABLE I
POSE DEPTH RESULTS OF FAILED EXPERIMENTS

	Pose Error		Depth Error				Depth Accuracy		
	ATE mean	ATE std	abs_rel	sq_rel	rms	log_rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
default	0.0278	0.0146	0.2585	2.617	8.179	0.3436	0.5906	0.8416	0.926
default_vl	0.025	0.0161	0.2608	2.3498	7.8925	0.3439	0.5915	0.8176	0.9326
ssim	0.0242	0.0113	0.2479	2.4641	7.9469	0.3328	0.6262	0.8432	0.929

TABLE II
POSE DEPTH RESULTS OF SUCCESSFUL EXPERIMENTS

	Pose Error		Depth Error				Depth Accuracy		
	ATE mean	ATE std	abs_rel	sq_rel	rms	log_rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
sfm pretrained	0.0113	0.0068	0.1822	1.6368	6.9268	0.2747	0.7356	0.8991	0.9562
resnet	0.0183	0.0062	**0.1937**	2.9306	**7.1199**	**0.2764**	**0.7591**	**0.9137**	**0.9619**
resnet_aug	0.0187	0.0059	0.2371	**2.1001**	7.4368	0.392	0.6377	0.8376	0.908
vgg12	0.0177	0.0059	0.2171	2.8612	7.6426	0.3011	0.6951	0.8831	0.9491
vgg16	**0.0174**	**0.0058**	0.2172	2.7574	7.7575	0.2998	0.7107	0.8874	0.9501

4) Introducing Resnet: Having figured the right choice of learning rate as 2×10^{-4} and having solved the bug in the architecture, we wanted to solve the problem of blurred out depth images that resulted in the previous two models. We doubted that the VGG-16 encoder of the depth network was unable to learn sufficiently well with this limited set of 12,000 image sequences, which could've been due to the vanishing gradient problem and so, we decided perform three experiments to debug this issue.

- To train the same model with our improved SSIM, smoothness loss and variable learning rate.
- To reduce the number of layers in VGG-16 encoder to VGG-12. The deconvolution layers were also reduced as a result.
- To replace this VGG-16 encoder of the depth network with a ResNet encoder, since ResNet is known to address vanishing gradient issue with the presence of skip connections

For experimentation with all of these modified architectures, we used the SSIM loss and updated the smoothness loss with the ”edge-aware” smoothness loss that we discussed in the previous section. The training of all these models were done with variable learning rate.

We seemed to have finally debugged the problem to obtain significantly better and well detailed disparity map predictions. The depth map predictions of all the three models are shown in 2

B. Data Augmentation

We also performed data augmentation with 65% probability of changing the brightness and applying gamma contrast for the best performing resnet model. However, the results of the intensity augmented model only deteriorated. The pose and depth evaluation results of the model is denoted as ”resnet_aug”.

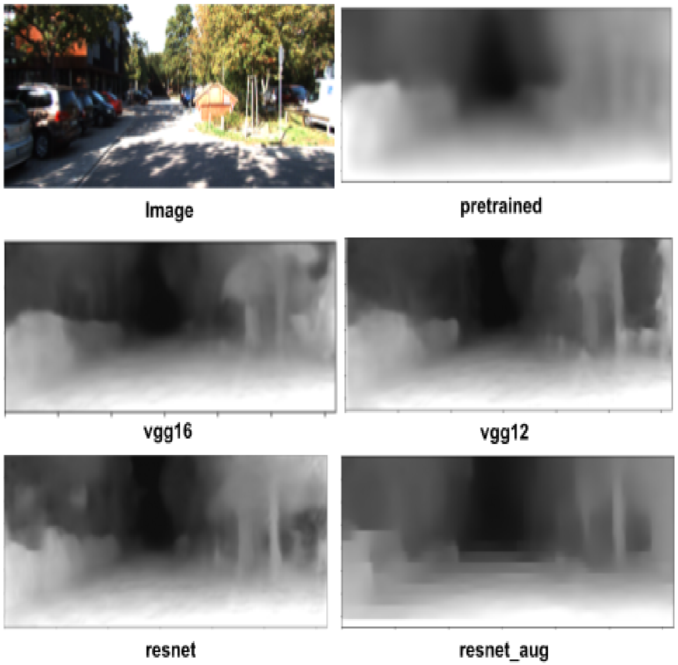

Fig. 2. Depth maps of successful experiments

VI. Results

The pose and depth evaluation results of the model with VGG-12 encoder is denoted as ”vgg-12”, VGG-16 encoder is denoted as ”vgg-16” and results corresponding to the ResNet encoder is denoted as ”resnet” in table II. To evaluate the efficacy from the models that we trained, we compared the results with pretrained model provided the authors of SfM-Learner is denoted as ”sfm pretrained”. Based on results from II, we consider that the model ”resnet” as the best performing model of all the models that we trained in our experiments. We can even notice that the ”resnet” model which was trained with just 12,000 images, shows better depth accuracy than the ”sfm pretrained” which was trained with 44,000 image sequences. This ”resnet” model had about 16 million parameters. On the other hand, we also inferred that a ”vgg-12” model with 21 million parameters performed almost the same as a ”vgg-16” model that had 33 million parameters The loss curves of ”resnet”, ”resnet_aug” and ”vgg-16” models are shown in figure 3 The presentation video is given here.

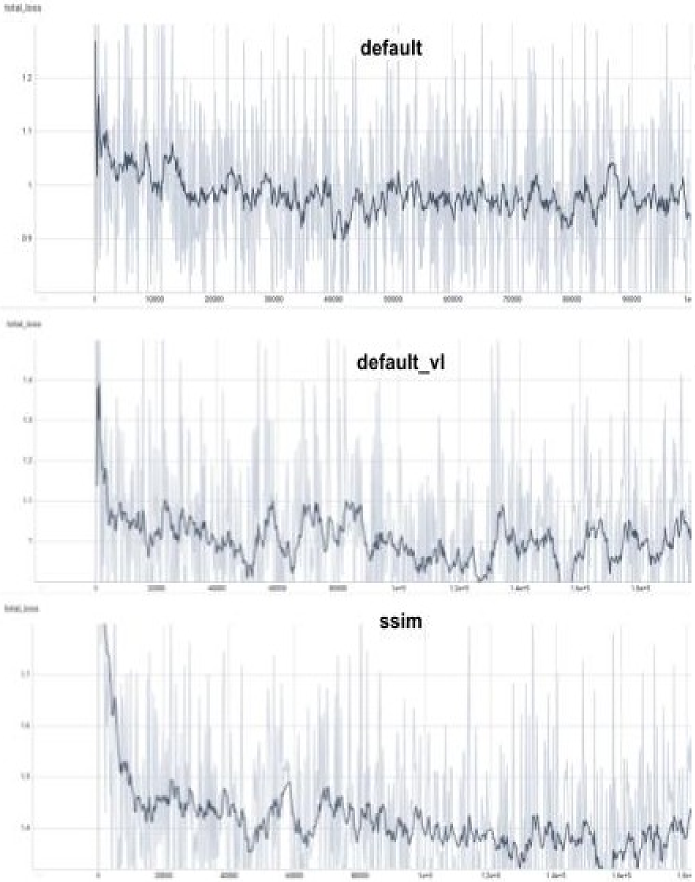

Fig. 3. Loss curves of failed experiments (0.9 smoothness)

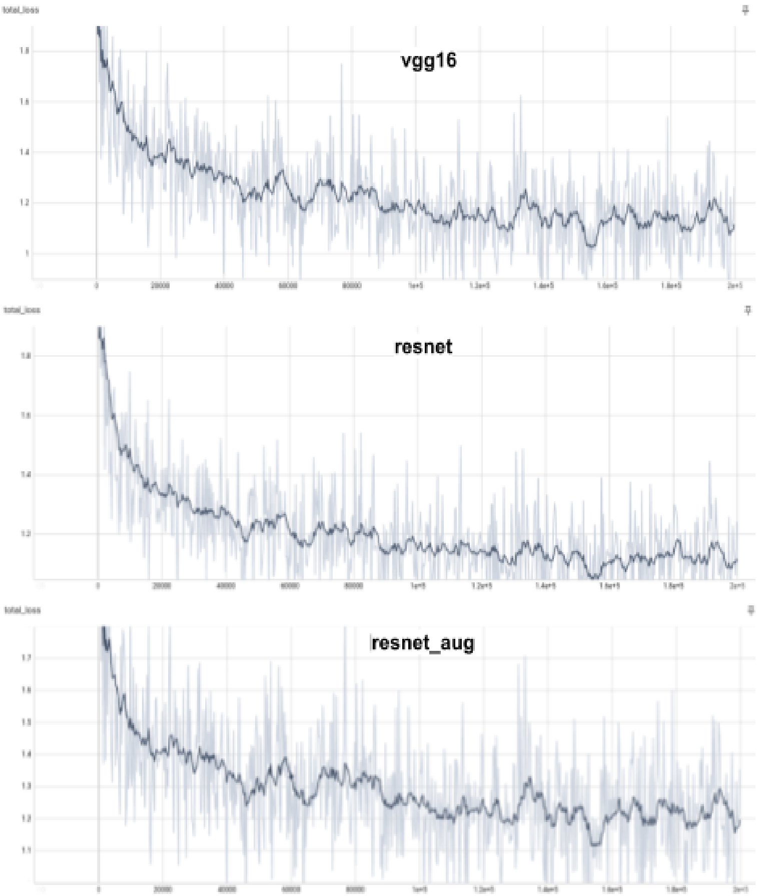

Fig. 4. Loss curves of successful experiments (0.9 smoothness)

References

[1] T. Zhou, M. Brown, N. Snavely and D. Lowe, ”Unsupervised Learning of Depth and Ego-Motion from Video,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017 pp. 6612-6619. doi: 10.1109/CVPR.2017.700

[2] Z. Yin and J. Shi, ”GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose,” 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 1983-1992, doi: 10.1109/CVPR.2018.00212.

[3] Zhou Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, ”Image quality assessment: from error visibility to structural similarity,” in IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, April 2004, doi: 10.1109/TIP.2003.819861.

[4] https://papers.nips.cc/paper/2014/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf

[5] https://github.com/tinghuiz/SfMLearner

[6] https://github.com/yzcjtr/GeoNet