Efficient Distributed Training of Vehicle Vision Systems

Sung-Li Chiang
Xinlei Pan

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2016-195
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-195.html

December 11, 2016

Efficient Distributed Training of Vehicle Vision Systems

Sung-Li Chiang

Abstract

Self-driving vehicle vision systems must deal with an extremely broad and challenging set of scenes. We propose a distributed training regimen for a CNN vision system whereby vehicles in the field continually collect images of objects that are incorrectly or weakly classified. These images are then used to retrain the vehicle's object detection system offline, so that accuracy on difficult images continues to improve over time. In this report we show the feasibility of this approach in several steps. First, we note that an optimal subset (relative to all the objects encountered) of images can be obtained by importance sampling using gradients of the recognition network. Next we show that these gradients can be approximated with very low error using just the last layer gradient, which is already available when the CNN is running inference. Then, we generalize these results to objects in a larger scene using an object detection system. Finally, we describe a self-labelling scheme using object tracking. Objects are tracked back in time (near-to-far) and labels of near objects are used to check accuracy of those objects in the far field. Finally we present some experiments and show the data reductions that are possible.

1 Introduction

Autonomous driving has recently become a popular topic with various enormous challenges [22]. One of the most important features of autonomous driving vehicles is the ability to interpret the surroundings and perform complex perception task such as the detection and recognition of lanes, roads, pedestrians, vehicles, and traffic signs [17]. Laser, radar and cameras are common sensors on autonomous driving cars to sense the driving environment. Due to the low cost and portability, camera has now become an indispensable equipment for the system, and is critical to provide a reliable solution to the facing challenges. Recently, with the appearance of Convolutional Neural Networks (CNNs), significant progress has been made in the field of object detection and recognition [10, 11, 16]. It is now possible to detect objects with high accuracy [16].

A useful and reliable visual system on an autonomous driving vehicle is used to mimic the human visual system so as to accurately recognize objects while driving. Recording and storing of driving videos and images is inevitable in order to train the autonomous driving visual system with rich scene information. Due to the intrinsic temporal continuity, videos are more powerful data than independent images in terms of training and labeling [14]. Such kind of data can be of several hours long [19, 9] and and a tremendous amount of space [4] is

often required to store these data for subsequent training purposes. It is also an unrealistic amount of work to label these video data frame-by-frame manually [25].

However, considering the characteristic of driving video data, on one hand, most part of the image and video data such as the background objects are usually static and are not relevant to the decision making of vehicle actions. Thus ignoring these image content causes little information loss but saves the amount of space and time from storing and processing the entire image. On the other hand, the detection and tracking of critical objects in these video and image data is sometimes not accurate enough, especially when these objects are in the far field or blurred by other objects. In this case, fine-tuning the visual system's parameters is required. Therefore, it is beneficial to sample a subset from the entire recorded data pool and emphasize training on those error-prone video frames or object regions. Selected training data set is also useful for improving the accuracy of a deep neural network and saving training time as we have less data to work on. In addition, considering the temporal continuity of driving video, one thing being specific to driving video is that objects near the camera are usually larger, clearer and easier to be correctly classified, while objects in the far field are usually smaller, and the classification is often prone to be contaminated by other objects. Therefore, while creating the training data, it is reasonable to trust the detection results of nearby objects and it can be used to check the identity of the same objects in the far field.

Traditional uniform sampling approach samples a subset of training data with equal weight of individual data sample. While it does help to reduce the total data volume, it provides little help of an efficient use of other information provided by data itself and does not represent the data very well. Importance sampling has been well known for solving this problem by providing a data-driven sampling scheme [5]. Recently, importance sampling has been successfully applied in the field of machine learning and optimization [28, 5]. In [2], it is shown that the learning efficiency and accuracy of a neural network is improved by selecting useful message with importance sampling, we follow the idea and apply the idea to the visual system in order to find out the most important and informative video frames and regions from the driving video data. We define the heuristic of the importance of images as following: a image with higher magnitude of gradients is the more important images. The reason is that the larger gradient magnitude corresponds to larger decreasing of loss function in stochastic gradient descent and thus improves the model more than images with lower magnitude of gradient.

In this report, we propose a practical approach to use importance sampling for visual training data reduction, and use the temporal continuity of driving video to generate labels of video frames in semi-supervised fashion. First of all, we show that an optimal subset of images can be collected by using importance sampling with the probability distribution defined by the Frobenius norm of gradients with respect to model parameters given input video frames. The model we are going to use is the Faster-RCNN [16] object detection network. In addition, we show that it is possible to use the Frobenius norm of last layer gradients of the object detection network to approximate the gradients of all model parameters with low error. In this way, we can save time by avoiding a large amount of computation doing back-propagation. Finally, we use object tracking framework to label video frames to automatically generate final training

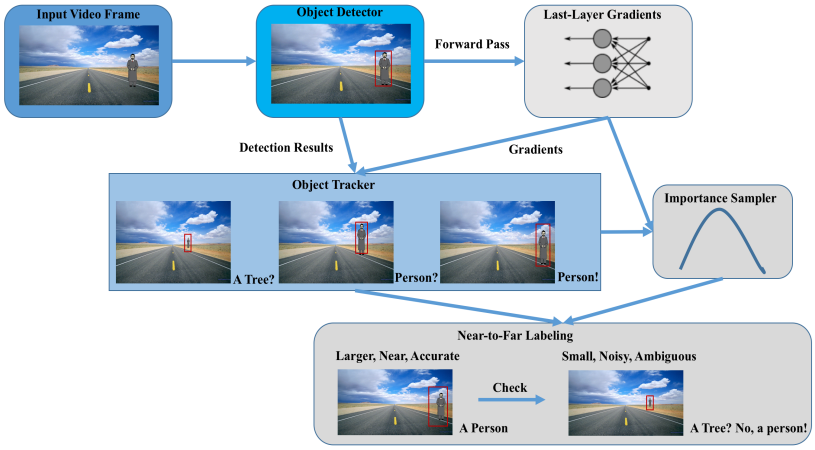

Figure 1: Overview of our method. Given a sequence of video frame inputs, the object detection network first detects objects in a frame, the forward pass step. Then the last layer gradient is computed, doing one step backward pass. Both detection results and gradient will be sent to object tracker which keeps a list of active trackers. The importance sampler determines whether to save the detection or not based on the gradients. Furthermore, the near-to-far labeling step check the accuracy of objects that are in the far field using the classification result of near field objects, where we believe near field objects are larger and the classification is accurate.

data. The same object in different frames are associated via visual tracking. Then we tracked back in time from near to far. Identities of far field objects are checked according to their near field object counterpart labels.

The contributions of this report are:

- We proposed to use importance sampling for data reduction for large volume autonomous driving visual data.
- We proposed to use near-to-far labeling scheme to perform semi-supervised labeling of unlabeled data, which improves the labeling efficiency.
- We performed both near-to-far labeling and importance sampling to select data that the pre-trained object detection network tends to make mistake on.
- We evaluated our method using relative variance ratio and labeling accuracy.

2 Related Work

Importance Sampling: Importance sampling is a well-known technique used to estimate properties of a particular distribution while only having samples generated from another distribution [15, 18]. The work of [28] studied the problem of improving traditional stochastic optimization with importance sampling, where they improved the convergence rate of prox-SMD [6, 7] and prox-SDCA [20] by reducing the stochastic variance using importance sampling. The work of [2] improves over [28] by incorporating stochastic gradient descent and deep neural networks. Also there are some work in using importance sampling for

minibatch SGD [5], where they proposed to use importance sampling to do data sampling in minibatch SGD and this can accelerate the converge of SGD.

As for self driving vehicles' vision system training, we typically do not know the ground truth distribution of the data, which is the images or video data captured by cameras. Thus, importance sampling will be very useful in estimating the properties of the data from an data-driven sampling scheme. Some exciting work of importance sampling for self-driving vehicle are already there: [29, 24, 27, 8]. The work of [24] and [8] proposed to use importance sampling for visual tracking, but their focus was not on reducing the training data amount and creating labeled data using visual tracking.

Object Tracking and Labeling. Object tracking is a fundamental problem in computer vision, and there has been much progress for past recent years [23, 12, 13]. In the scenario of autonomous driving, we are interested in tracking critical objects and identifying their categories. For example, when the vehicle is driving on a highway, it is useful to detect and track other vehicles to keep an appropriate distance with them, and when the vehicle is driving through a busy road, it is important to track pedestrians to avoid traffic accidents.

Typically, it is relatively hard to detect and identify an object when it is very far away and easier otherwise. Intuitively, when a object detection neural network such as Faster-RCNN [16] takes an image that is hard to perform object detection, the output loss will be larger comparing to an image that is much easier to detect objects. Therefore, the gradients of the loss function with respect to the harder-to-detect image will be larger than the easier-to-detect one. Given the large volume of video data captured by cameras on vehicles, it is useful to track an object when it is not very clear what it is and save the corresponding image and discard the image frames that we are already able to tell what's in the scene. That is to say, we should focus more training on those difficult images.

3 Methods

The pipeline of our entire algorithm includes two parts. First of all, a object detector module is use to find out object targets in a scene as well as to compute the gradients from the object detector network for each object of interest. The gradients will be regarded as sampling weights according to the importance sampling mentioned in [2].The second part is a self-labeling module. An object tracker matches and creates a relation of a same object in video frames coming from the real driving recordings on-line by incorporating Kalman-filter algorithm. Then, the same object in the same stream will be labeled from near to far according to the label of the closer object.

The system architecture is shown in figure 1. Here, we first describe the importance sampling framework followed by the discussion of details about object tracking, self-labeling.

3.1 Sampling An Optimal Subset of Images

The training of a deep neural network for vision system of self-driving car typically entails a set of forward propagation to calculate the loss function and backward propagation to evaluate the gradients of the loss function with respect to all parameters of the model. Nowadays, stochastic gradient descent

(SGD) is a common way in the training procedure.

Inspired by the idea of importance sampling [2], we can select an optimal subset of the data by sampling the data according to importance sampling probability distribution so that the variance of the sampled data is minimized under an expected size of sampled data. Here, the sampling distribution is proportional to the gradients magnitude of each image.

Consider the calculation of gradients as an estimation problem, the goal is to estimate the expected gradients $f(x)$ based on a data distribution $p(x)$, where x is the input data instance. However, usually we do not know the ground truth distribution of the data $p(x)$, and we can rely on a sampling proposal $q(x)$ to unbiasedly estimate this expectation, with the requirement that $q(x) > 0$ whenever $p(x) > 0$. This is commonly known as importance sampling:

$$\int p(x)f(x)dx = \mathbb{E}_{p(x)}[f(x)] = \mathbb{E}_{q(x)}[\frac{p(x)}{q(x)}f(x)] \tag{1}$$

It has been proved in [2] that the variance of this estimation can be minimized when we have,

$$q^*(x) \propto p(x)||f(x)||_F \tag{2}$$

Defining $\tilde{q^*}(x_i)$ as the unnormalized optimal probability weight of image x_i, it is obvious that images with a larger gradient norm value should have a larger weight. Since we have access to a dataset $\mathcal{D} = \{x_n\}_{n=1}^N$ sampled from $p(x)$, we can obtain $q^*(x)$ by associating the probability weight $\tilde{q^*}(x_n) = ||f(x_n)||_F$ to every $x_n \in \mathcal{D}$, and to sample from $q^*(x)$ we just need to normalize these weights:

$$q^*(x_n) = \frac{\tilde{q^*}(x_n)}{\sum_{i=1}^N \tilde{q^*}(x_1)} = \frac{||f(x_n)||_F}{\sum_{i=1}^N ||f(x_i)||_F} \tag{3}$$

where $f(x_i)$ is the gradients of the loss function of input x_i with respect to all model parameters, which can be either a scalar or a matrix, and is transformed into a scalar after taking the Frobenius Norm. To reduce the total number of data instances used for estimating $\mathbb{E}_{p(x)}[f(x)]$, we draw M samples from the whole N data instances ($M << N$) based on a multinomial distribution where $(q^*(x_1), ..., q^*(x_N))$ are the parameters of this multinomial distribution. According to the discussion above, we obtained an estimation of $\mathbb{E}_{p(x)}[f(x)]$ which has least variance compared to all cases where we draw M samples from the entire N data set.

3.2 Object Detection Network Gradient Extraction

As the first step of looking for candidates in the images that we may be interested in collecting, we have to find out where is the objects of interest. Here we focus on pedestrians, any kinds of vehicles or signs, which are entities that an autonomous driving vehicle may have to pay attention to while it is making a control decision from the scenes. The object detection network used in this study is Faster RCNN [16], which is a state-of-the-art recognition network bundling the region proposal and object recognition mechanisms together. The advantage of this architecture is that it is able to detect the objects in an image and propose bounding boxes information faster in time. Therefore, it is attractable for on-line image object detection.

Furthermore, as mentioned in previous section, we need gradients for each object to define the sampling weight. We extract gradients from the object detection network for every given input x_i, here it represents an image. We do forward propagation with an input image x_i to obtains object candidates, and then we do backward propagation to compute gradients. Ideally we are supposed to propagate backward to the very beginning to see how importance the image is, since we are interested in how does the image improve the network to do better. However, computing gradients of the very beginning layer is very expensive, we instead use the last layer gradients to approximate the value of the early layers to avoid the cost of computation time.

We propose to use the Frobenius norm of the last layer gradients as an approximation to represent the corresponding gradient magnitude of the image, the gradient of loss with respect to all parameters. This simplifies the procedure of calculating gradients and saves a remarkable time from avoid propagating backward. The reason for the feasibility of this approach is that the calculation of gradients follows the chain rule and since operations in chain rule are multiplications of a scalar, which are linear operations. As a result, the magnitude of the last layer gradients is approximately linearly correlated with the magnitude of previous layers' gradients.

$$||g_{i,last}||_F \approx c||g_{i,j}||_F \tag{4}$$

where $g_{i,last}$ is the gradient of the loss function with respect to the parameters of the last layer, and $g_{i,j}$ is the gradient of the loss function with respect to parameters of the j^{th} layer, and i here denotes data i^{th} instance .

Moerover, since the calculation of probability weight $q^*(x_i)$ only depends on the normalized gradients norm, so this approximation is reasonable. To show this, we evaluated gradients at several layers of the network and all are then taken Frobenius norm for comparison.

In figure 2, we show that the magnitude of Frobenius-normed gradients of the former layers are with high linearity with respect to the last layer. We can see that the r square value of the linear regression can be even higher than 0.9.

In table 1, we show the sampling efficiency comparing the sample estimator with magnitude of gradients from former layers to the estimator with magnitude of the last layer of the Faster RCNN. Apparently, the ratio is very close to 1, which means that the magnitude of gradients from former layers are interchangeable with the last layer gradients.

	FC7	FC6	Conv5_1	Conv4_1	Conv3_1	Conv2_1	Conv1_1
Last Layer	1.01	1.03	1.07	1.10	1.12	1.13	1.14

Table 1: Sampling efficiency of using former layers' gradients in Faster RCNN with respect to using the last layer gradients

3.3 Measure the Efficiency of Variance Reduction

Once we get the sampling distribution $q^*(x_i)$, we then perform the importance sampling. Images with a higher gradient norm will get higher likelihood to be sampled, which is reasonable since a larger gradient norm means that the loss function has not reached its minimum value. We, further, measure how

efficient that we estimate the gradient norm distribution. Since the goal of using importance sampling approach here is to reduce the variance while estimating properties of the data (in this case is the total gradient norm given individual sample video frame input) from a subset of the data.

To show that the gradients estimated from the sampled images have close variance with gradients variance estimated from all images, we computed a relative variance value. This value is the ratio of whole data set gradient norm variance over sampled images' gradient norm variance.

Suppose the data set is $\mathcal{D} = \{x_n\}_{n=1}^N$, and we extracted last-layer gradient $g(x_i)$ given individual input x_i. To calculate the sampling probability, we take the Frobenius norm of $g(x_i)$ and get $||g(x_i)||_F$, and define the sampling probability of image x_i when we expect to sample M out of N images ($M < N$) as,

$$q(x_i) = \min\left[1, \frac{M||g(x_i)||_F}{\sum_{i=1}^N ||g(x_i)||_F}\right] \tag{5}$$

taking the minimum compared with 1 is to ensure that the probability of sampling image x_i can not be larger than 1, which happens when $\frac{M||g(x_i)||_F}{\sum_{i=1}^N ||g(x_i)||_F}$ is saturated. Note that, when the sampling probability is 1, we should sample this image. With the scaled sampling weight $\frac{M||g(x_i)||_F}{\sum_{i=1}^N ||g(x_i)||_F}$, we change M so that we can get different numbers of images out of the entire image date. Typically, choosing a M such that the sample gradient norm variance is close to whole data gradient norm variance. Since the data are in the discrete space, the relative variance is defined as,

$$\mathrm{R} = \frac{\sum_{i=1}^N ||g(x_i)||_F^2}{\sum_{i=1}^N ||g(x_i)||_F^2/q(x_i)} \tag{6}$$

where $\sum_{i=1}^N ||g(x_i)||_F^2/q(x_i)$ can be expressed as,

$$\begin{aligned} &\sum_{i=1}^N ||g(x_i)||_F^2/q(x_i) \\ &= \sum_{j=1}^k \frac{\sum_{i=1}^N ||g(x_i)||_F}{M||g(x_j)||_F}||g(x_j)||_F^2 + \sum_{j=k+1}^N ||g(x_j)||_F^2 \\ &= \frac{\sum_{i=1}^N ||g(x_i)||_F}{M}(\sum_{j=1}^k ||g(x_j)||_F) + \sum_{j=k+1}^N ||g(x_j)||_F^2 \end{aligned} \tag{7}$$

where $q(x_1)$, $q(x_2)$, $\cdots$, $q(x_k)$ are smaller than 1 and $q(x_{k+1})$, $\cdots$, $q(x_N)$ are equal to 1.

3.4 Object Tracking Module-Self Labeling

After we have objects and gradient values for each object candidate, we then wish to mark the objects in a sequence of images as in the same group if they are supposed to be the same object across the these sequence of images. In [3], it shows a simple tracking algorithm incorporating with the Kalman filter and taking only present image frame and a few previous image frames into

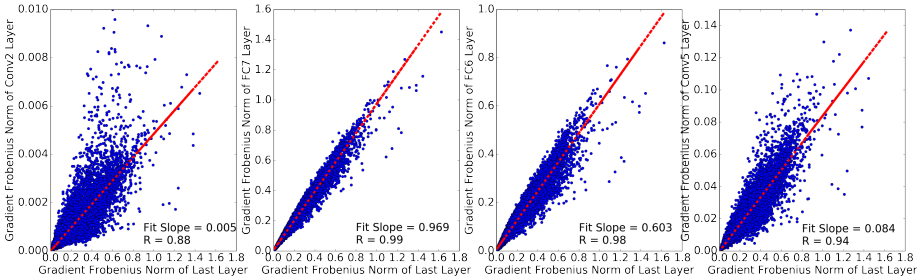

Figure 2: Plot of gradient Forbenius-norm of last layer in VGG 16 versus the gradient Forbenius-norm of fully-connected layer 7 (FC7), fully-connected layer 6(FC6), Convolutional Layer 5 (Conv5) and Convolutional Layer 1 (Conv1)

consideration. This algorithm works as long as we are able to provide bounding box information of objects in frames. This simple algorithm is specially fit to the on-line tracking for autonomous driving [1], since we are not able to get future images in real time. In this paper, we have Faster RCNN as our information provider.

The tracker object of the tracking algorithm will try to give each object in the frame a tracking box, if the object in the current frame is the same object in the previous frame, the object in the current frame will be assign to the same tracking box as in the previous frame. From previous frame, we can infer the location of the tracking box with the help of Kalman filter. Ideally, the movement of the same object should not have a large difference in consecutive image frames. The assumption makes a lot of sense, since the recording rate for cameras are high enough such as 60fps, even a cheap device on a autonomous driving can achieve this. Otherwise, if there is no match for existing tracking box, the object will be assigned a new tracking box and be referred to as a new object.

Near-to-Far Labeling Scheme. With the help of object entity identification from the tracker mentioned above, we then proposed to use near field image detection results to check the identify of far field detection results. Our proposed method does not require the data to be labeled in advance. Therefore, this is a method of automatic labeling the training data with the help of a pre-trained model. Since our purpose is to select training images for better performance of visual system, we will label images as we track them for a certain period. Considering of the case that we are tracking an object from far to near field. When the image is in a far distance away from our current location, the object could be very small or blurred in the image, which makes it very difficult to be correctly classified. In this scenario, if the object is classified incorrectly according to the near field object information and the gradient norm of this image is relatively large, then we should have a high probability to sample this image and save it. As the object approaches the vehicle, the detection network has a higher confidence to correctly classify this object, and the gradients of this image will become smaller and thus we have a relatively low probability to sample this image and save it. We label the objects as we track them, if the norm of gradients of an input frame exceeds a certain threshold, we need to save this image for subsequent training. Otherwise, we treat the object detection network as a good enough model to detect objects in this frame and discard

Input: *dets*(detections information of a single frame);
trks (keep a list of active trackers);
min_age (delete a tracker if it has not been updated for more than this time);
grad_th (save a detection if the gradient norm is larger than this threshold)
Result: *ret* (collections of detections to be saved)
for $\forall$ *trk* $\in$ *trks* **do**
| Predict next state of trk using Kalman Filter
end
for $\forall$ *det, trk* $\in$ *dets, trks* **do**
| Match *det*, *trk*
| get matched (det, trk) pairs,
| get unmatched *det*
| get unmatched *trk*
end
retain = {}
for $\forall$ matched (det, trk) pairs **do**
| Update *trk* using *det*;
| Using confidence score of *det* to check the accuracy of history scores of *det* in *trk*;
| retain.append(All history of *det* $\in$ *trk*)
end
for $\forall$ unmatched *det* **do**
| Add a new *trk* based on *det*
| retain.append(All history of *det* $\in$ *trk*)
end
for $\forall$ unmatched *trk* **do**
| **if** *trk* has not been updated for more than *min_age* times **then**
| | Remove *trk* from *trks*
| **end**
end
for $\forall$ ret $\in$ retain **do**
| **if** *ret* has history length > h **then**
| | mark the first n frames in ret as needed to be saved if they have different classification form the latest frames.
| **end**
end
return *retain*

Algorithm 1: Object Tracking and Labeling Algorithm

the frame. The threshold value is ad-hoc and depends on the detection result from experiments.

The object tracking and labeling algorithm is described in this algorithm 1.

4 Experiments

Data Set and Pre-trained Model. In the experiment, we use the dataset from Stanford autonomous driving car video recording. The dataset includes real scenes the vehicle met while the vehicle is driving on the road. We use this dataset to simulate the real word situation we may encounter in order to test the feasibility of our algorithm in the real word task. We use the KITTI benchmark data set which is a challenging autonomous driving data set [9]. This data set is fairly rich as it contains high-resolution color and grayscale images and videos captured in rural areas and on highways. The raw data has several categories covering many aspects of transportation: city, residential, road, campus, person, etc.

Since our goal is to perform importance sampling and near-to-far labeling to select images that are hard to train, we used pre-trained Faster-RCNN model [26, 21] for object detection on images selected from the KITTI data set. The pre-trained model we used is the one from [16]. At the same time, since the object labels from the pre-train model may not fit the object classes we will actually meet on the road such as "horse" or "tvmonitor", we also fine-tune the Faster RCNN with CityScapes data set. We used around 2900 images from CityScapes with fine ground truth annotations to prepare our training data and fine tune some convolution layers and fully connected layers.

Check the Feasibility of Using Last Layer Gradient. As mentioned before, we did not do a entire back-propagation to get the gradient value of the loss function with respect to model parameters given sampling image as input. Instead, we just use the last layer gradient. As the data is not labeled, we first do a forward pass to get the label of the input image, namely, get the bounding box and corresponding scores. Then we do a backward propagation to the last layer to get the gradient of the last layer. In this step, we assume that the class the the object decided from the highest value of output score from the network is correct.

To show that importance sampling can optimize the variance with selecting images with high gradients more, we calculated gradient Frobenius norm at every layer for every input frame. For each layer, we compute the variance of keeping all data and keeping part of the data sampled with the importance sampling. The ratio of variances is calculated according to equation 6.

Labeling Accuracy. We perform near to far labeling and check the accuracy of labeling. The accuracy is calculated as the ratio of number of proposals that are incorrectly labeled by the original model but are then corrected by our labeling scheme over the total number of incorrectly labeled proposals.

5 Results

Gradient Extraction Results As mentioned in section 3.2, we use the last layer gradient norm to approximate the total gradient norm. We calculated

the last layer gradients' norm and several previous layers' gradients' norm and we show that these values are highly linear correlated. It can be seen from the image that the last several fully connected layers' gradient norm follow a more linear correlation with the last layer gradient norm while the first several convolutional layers' gradient norm has larger variance.

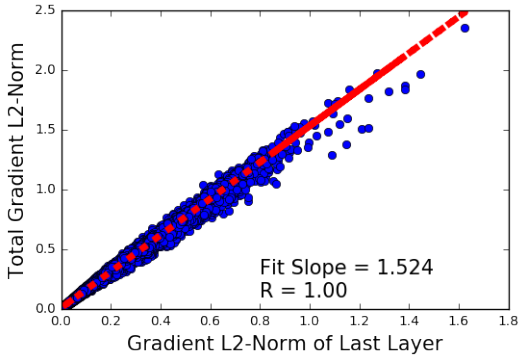

Figure 3: The linear fit of last layer gradient Frobenius norm and total gradient Frobenius norm

Relative Variance Results We use the relative variance mentioned in section 3.3 to measure how good we estimate the gradient norms. To do this, we calculated the curve of number of samples we draw (M) versus the relative variance ratio. The result is shown below. From the plot, we can see that by scaling the importance sampling weight as mentioned in 5, we are able to keep high sampling efficiency compare to the use of original gradients norm magnitude while only retaining half number of the entire dataset.

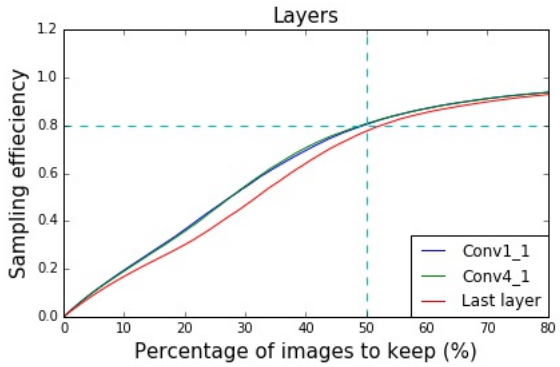

Figure 4: Relative Variance Evaluation Results

Implicit Labeling Results Implicit labeling means we use near field object detection results to check the accuracy of objects that are in the far field. To check the accuracy, we performed sampling and labeling on the KITTI data set, the accuracy calculation method was mentioned before, and the accuracy is 0.93.The accuracy without relabeling is 0.90. An example sequence of implicit labeling for correcting annotations is shown in 5.

6 Conclusion

We proposed the idea of using importance sampling to select the most important images for training and online processing purposes and have successfully extracted gradients from the last fully connected layer of faster R-CNN as the probability of weight. More work should be done to better demonstrate the advantage and ability of importance sampling to reduce the storage space for visual data generated by autonomous vehicles, including comparing the learning quality from importance sampling selected images with that of the entire dataset.

We also realized that there is some intrinsic problems in labeling poorly detectedhave any influence in training the model since it will not saved for later retraining. We think our method is able to improve the situation where we are able to detect objects with pre-trained model but not able to correctly classify.

Our proposed implicit labeling scheme helps to unsupervisedly labeling the data using pre-trained model, and the labeled data can be used to retrain the model and fine-tune the parameters.

Labeling results from left to right: motorbike, car, car (ground truth). As the vehicle approaches the object, it becomes clearer and no longer hidden by the pole.

Labeling results from left to right: bus, car, train (ground truth). The object looks like a bus in the far field, but is classified as train when it's in the near field considering it's on railroad.

Labeling results from left to right: train, car, car (ground truth). The object looks like a train in the far field, but is classified as car when it's in the near field.

Labeling results from left to right: bus, car, car (ground truth). At first sight, the car is blurred and hidden by other objects, then it became more clearer that this is a car.

Figure 5: Examples of implicit labeling. These images are from the KITTI Benchmark data set [9]. The labeling results are obtained from pre-trained Faster-RCNN model. The bounding box shows the detected objects being tracked. Near field object detection results are used to check the accuracy of the detection results of objects in the far field.

References

[1] Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, Sept 2016.

[2] Guillaume Alain, Alex Lamb, Chinnadhurai Sankar, Aaron Courville, and Yoshua Bengio. Variance reduction in sgd by distributed importance sampling. *arXiv preprint arXiv:1511.06481*, 2015.

[3] Alex Bewley, ZongYuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. *CoRR*, abs/1602.00763, 2016.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *arXiv preprint arXiv:1604.01685*, 2016.

[5] Dominik Csiba and Peter Richtárik. Importance sampling for minibatches. *arXiv preprint arXiv:1602.02283*, 2016.

[6] John Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10(Dec):2899–2934, 2009.

[7] John C Duchi, Shai Shalev-Shwartz, Yoram Singer, and Ambuj Tewari. Composite objective mirror descent. In *COLT*, pages 14–26, 2010.

[8] Rana Farah, Qifeng Gan, JM Pierre Langlois, Guillaume-Alexandre Bilodeau, and Yvon Savaria. A computationally efficient importance sampling tracking algorithm. *Machine Vision and Applications*, 25(7):1761–1777, 2014.

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

[11] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[12] Seunghoon Hong, Tackgeun You, Suha Kwak, and Bohyung Han. Online tracking by learning discriminative saliency map with convolutional neural network. *arXiv preprint arXiv:1502.06796*, 2015.

[13] Hanxi Li, Yi Li, and Fatih Porikli. Deeptrack: Learning discriminative feature representations online for robust visual tracking. *IEEE Transactions on Image Processing*, 25(4):1834–1848, 2016.

[14] Ishan Misra, Abhinav Shrivastava, and Martial Hebert. Watch and learn: Semi-supervised learning of object detectors from videos. *CoRR*, abs/1505.05769, 2015.

[15] Yi Zhou Owen, Art; Associate. Safe and effective importance sampling. *Journal of the American Statistical Association*, 449:135 – 143, 2000.

[16] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[17] Eduardo Romera, Luis Miguel Bergasa, and Roberto Arroyo. Can we unify monocular detectors for autonomous driving by using the pixel-wise semantic segmentation of cnns? *CoRR*, abs/1607.00971, 2016.

[18] Reuven Y Rubinstein and Dirk P Kroese. *Simulation and the Monte Carlo method*, volume 707. John Wiley & Sons, 2011.

[19] Eder Santana and George Hotz. Learning a driving simulator. *CoRR*, abs/1608.01230, 2016.

[20] Shai Shalev-Shwartz and Tong Zhang. Proximal stochastic dual coordinate ascent. *arXiv preprint arXiv:1211.2717*, 2012.

[21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[22] Alex Teichman and Sebastian Thrun. Practical object recognition in autonomous driving and beyond. *Advanced Robotics and its Social Impacts (ARSO), 2011 IEEE Workshop on*, pages 35–38, 10 2011.

[23] Alex Teichman and Sebastian Thrun. Tracking-based semi-supervised learning. *The International Journal of Robotics Research*, 31(7):804–818, 2012.

[24] Naiyan Wang and Dit yan Yeung. Learning a deep compact image representation for visual tracking. In C.j.c. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 809–817. 2013.

[25] Jenny Yuen, Bryan Russell, Ce Liu, and Antonio Torralba. Labelme video: Building a video database with human annotations. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1451–1458. IEEE, 2009.

[26] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.

[27] Yan Zhai, Mark Yeary, Joseph P Havlicek, J-C Noyer, and Patrick Lanvin. Visual tracking using sequential importance sampling with a state partition technique. In *IEEE International Conference on Image Processing 2005*, volume 3, pages III–876. IEEE, 2005.

[28] Tong Zhang and RUTGERS EDU. Stochastic optimization with importance sampling for regularized loss minimization. 2014.

[29] Ding Zhao, Henry Lam, Huei Peng, Shan Bao, David J. LeBlanc, Kazutoshi Nobukawa, and Christopher S. Pan. Accelerated evaluation of automated vehicles based on importance sampling techniques. *CoRR*, abs/1605.04965, 2016.