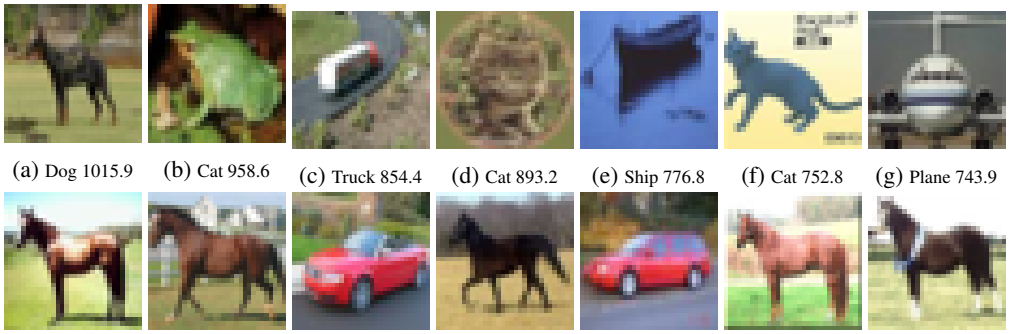
Unsupervised Difficulty Estimation with Action Scores

Octavio Arriaga
Department of Computer Science
University of Bremen
28359 Bremen, Germany
arriagac@uni-bremen.de

Matias Valdenegro-Toro
Robotics Innovation Center
German Research Center for Artificial Intelligence
28359 Bremen, Germany
matias.valdenegro@dfki.de

Abstract

Evaluating difficulty and biases in machine learning models has become of extreme importance as current models are now being applied in real-world situations. In this paper we present a simple method for calculating a difficulty score based on the accumulation of losses for each sample during training. We call this the action score. Our proposed method does not require any modification of the model neither any external supervision, as it can be implemented as callback that gathers information from the training process. We test and analyze our approach in two different settings: image classification, and object detection, and we show that in both settings the action score can provide insights about model and dataset biases.



(a) Dog 1015.9 (b) Cat 958.6 (c) Truck 854.4 (d) Cat 893.2 (e) Ship 776.8 (f) Cat 752.8 (g) Plane 743.9

(h) Horse 0.073 (i) Horse 0.0280 (j) Car 0.288 (k) Horse 0.291 (l) Car 0.305 (m) Horse 0.322 (n) Horse 0.335

Figure 1: Most difficult (top-row) and easiest examples (bottom-row) in CIFAR10. Our proposed *action score* is displayed below each image as well as the true label.

1 Introduction

Current state-of-the-art models in computer vision tasks rely on the use of convolutional neural networks (CNNs). However, modern CNN architectures contain sufficient structural-priors to reduce the solution space to a computable and generalisable one, but not restricted enough to prevent them from learning unstructured data nuances [10, 7, 2, 1]. In this paper we present a simple method to assess the difficulty and possible biases of machine learning models by tracking the loss of each sample during training. This method does not rely in any external supervision nor model modification as opposed to similar methods [8, 6, 4, 9]. Specifically, we test it in a simple image classification scenario and a more complex setting with a multi-objective loss used in object detection.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

The use of per-sample loss values is widespread in the literature. [8] uses the per-sample loss to mine for hard negative examples while training an object detector. [6] proposes a way to sample mini-batches using the loss as a criteria, where training samples with higher loss will be chosen more frequently. This has the effect of speeding up training by $5\times$. The focal loss [4] introduces a similar concept where an object detector focuses on harder samples. Difficulty estimation is an emerging topic in this field. [9] proposes an additional output branch and a related loss function in order to learn to estimate sample difficulty. This method has learning difficulties and cannot be trained end-to-end.

2 Unsupervised Difficulty Estimation

Given a loss function $\mathscr{L}$ and a model m with free parameters θ_n, we define the action $\mathscr{A}$ of a sample $x \in X$ with labels $y \in Y$ as

$$\mathscr{A}(x) = \sum_{n=0}^{N} \mathscr{L}(y, m(x; \theta_n)) \tag{1}$$

where n represents epochs. Consequently, the action[1] of a sample is the accumulated loss over all epochs. Our method characterizes the action of each sample as a measurement of its difficulty. Therefore, samples with a higher accumulated loss represent samples that are more difficult to learn. Specifically, we argue that the action is directly proportional to its difficulty i.e. $D(x) \propto \mathscr{A}(x)$. Within this framework we can also recover sample pairs that accumulate the least amount of loss during optimization. These samples reflect which elements are easier to learn as well as possible biases that might be present in the data. We would like to emphasize that the method presented here can be applied to any learning algorithm that is optimized iteratively and is not limited to artificial neural networks nor supervised methods.

3 Results

We first tested our method in simple classification task in which we train a VGG-like CNN[2] on CIFAR10 using the cross-entropy loss. At every epoch we calculated and stored the loss of each sample in the test set. After the conclusion of the training phase we calculated the action of each sample by summing up the stored losses. In Figure 1 we display the samples with the most and least action scores. From Figure 1 we can observe that model learns to distinguish with the least action two specific set of samples: brown horses and red cars. For our second experiment we calculate the action scores of a multi-objective loss function used for training the single-shot object detector SSD300 [5]. The total loss of this model consist of the combination of three different losses: positive classification, negative classification and bounding box regression. For the localization loss the samples with the most and least action are shown in Figure 2

We can observe that the most difficult samples for the box regression loss correspond to images that contain undistinguishable small objects. Moreover, easier samples for the same loss are determined by single centered objects.

We provide additional examples of object detection on PASCAL VOC 2007 in the supplementary material.

4 Conclusions and Future Work

In this work we presented a method for calculating the difficultly and possible biases of a model. Our method requires no external supervision nor a modification of the original model and it can be easily integrated in any learning framework. We test our method in two different settings. We displayed the samples with the highest and lowest *actions scores*. Our obtained results indicate that the maximum and minimum *action scores* do qualitatively correspond to difficult or biased samples. For future work we propose to apply our method in unsupervised settings, as well as to test its variability along different models.

[1] We adopt this name due to its similarity of a physical system following the path of stationary action [3].

[2] We used the Keras CIFAR10 example CNN available at keras-examples

References

[1] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[2] Jason Jo and Yoshua Bengio. Measuring the tendency of cnns to learn surface statistical... *arXiv preprint arXiv:1711.11561*, 2017.

[3] LD Landau and EM Lifshitz. *Course of theoretical physics. vol. 1: Mechanics*. Oxford, 1960.

[4] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[6] Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*, 2015.

[7] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled. pages 427–436, 2015.

[8] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016.

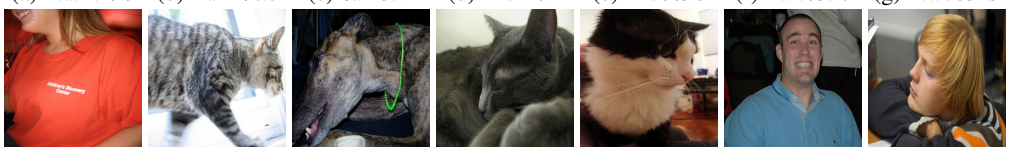
[9] Pei Wang and Nuno Vasconcelos. Towards realistic predictors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 36–51, 2018.

[10] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

A Object Detection Results on PASCAL VOC 2007 with SSD

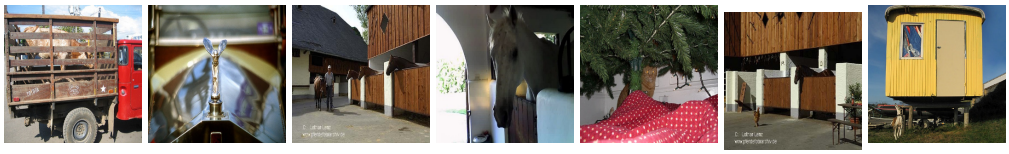
In this section we show results on the PASCAL VOC 2007 validation set using the Single Shot Multibox detector [5]. SSD uses a multi-task loss, a localization loss for bounding box regression, and a cross-entropy loss for class predictions. The cross-entropy loss can be divided into loss for the positive examples (target objects), and loss for the negative examples (background). We show results in each components of the multi-task loss, namely localization, positive, and negative losses.

(a) Boat 1749.3 (b) Train 496.6 (c) Car 457.1 (d) Bird 413.1 (e) Bird 385.8 (f) Plant 357.6 (g) Bottle 354.5



(h) Person 3.9 (i) Cat 4.1 (j) Dog 4.2 (k) Cat 4.7 (l) Cat 5.0 (m) Person 5.2 (n) Person 5.3

Figure 2: Most difficult (top-row) and easiest examples (bottom-row) in the VOC 2007-VAL with the SSD localization loss. The *action scores* are displayed below each image as well as the true label.



(a) Cow 596.6 (b) Car 568.5 (c) Person and Horse 565.0 (d) Horse 544.6 (e) Cat 544.3 (f) Plant, Bottle and Horse 527.8 (g) Dog 524.7

Figure 3: Hardest Examples on PASCAL VOC 2007 with SSD validation positive loss. Action score is included in each caption.

(a) Person 5.7 (b) Person 6.1 (c) Person 6.2 (d) Cat 6.2 (e) Person 6.3 (f) Cat 6.4 (g) Person 6.7

Figure 4: Easiest Examples on PASCAL VOC 2007 with SSD validation positive loss. Action score is included in each caption.

(a) Person 1086.9 (b) Boat 626.9 (c) Table 548.1 (d) Person 476.6 (e) Table 452.9 (f) Chair 416.2 (g) Bicycle 401.5

Figure 5: Hardest Examples on PASCAL VOC 2007 with SSD validation negative loss. Action score is included in each caption.

(a) Cat 6.5 (b) Cat 8.2 (c) Cat 8.5 (d) Dog 8.6 (e) Cat 8.7 (f) Person 9.6 (g) Cat 10.0

Figure 6: Easiest Examples on PASCAL VOC 2007 with SSD validation negative loss. Action score is included in each caption.