

Why Accuracy Is Not Enough: The Need for Consistency in Object Detection

Caleb Tung, Abhinav Goel, Fischer Bordwell, Nick Eliopoulos, Xiao Hu, Yung-Hsiang Lu
{tung3, goel39, fbordwel, neliopou, hu440, yunglu}@purdue.edu - Purdue University

George K. Thiruvathukal
gkt@cs.luc.edu - Loyola University Chicago

Abstract—Object detectors are vital to many modern computer vision applications. However, even state-of-the-art object detectors are not perfect. On two images that look similar to human eyes, the same detector can make different predictions because of small image distortions like camera sensor noise and lighting changes. This problem is called inconsistency. Existing accuracy metrics do not properly account for inconsistency, and similar work in this area only targets improvements on artificial image distortions. Therefore, we propose a method to use non-artificial video frames to measure object detection consistency over time, across frames. Using this method, we show that the consistency of modern object detectors ranges from 83.2% to 97.1% on different video datasets from the Multiple Object Tracking Challenge. We conclude by showing that applying image distortion corrections like .WEBP Image Compression and Unsharp Masking can improve consistency by as much as 5.1%, with no loss in accuracy.

■ **MUCH OF MODERN COMPUTER VISION RELIES ON OBJECT DETECTORS.** An object detector is a Deep Neural Network (DNN) that takes an image as the input and then identifies the locations and types of objects found in that image. Across scientific disciplines, object detectors are increasingly ubiquitous. From electronic package sorting in e-commerce to collision detection in traffic monitoring, from remote sensing in low-orbit satellites to automated MRI screening in the fight against cancer: object detectors are driving an entire frontier of technology. With so many critical applications, object detectors need to be consistently accurate. Modern object detectors use different architectures (e.g., single-shot, R-CNN, etc.) and training methods (e.g., multitask loss, neural architecture search, etc.) to achieve state-of-the-art accuracy on popular

image datasets like Microsoft COCO (Common Objects in Context) [1].

The Consistency Problem

Even though object detectors are carefully tested for accuracy, this article observes that *consistency* is also a valuable metric that receives less attention in literature. As we discuss later, common image datasets make it challenging to test for consistency.

Accuracy typically measures how often an object detector is correct *on average*. This information is partially deficient because it does not capture the variation in an object detector's performance when input images are similar. Since accuracy is reported as an average, there could be multiple ways an object detector achieves a given accuracy, some of which may be less

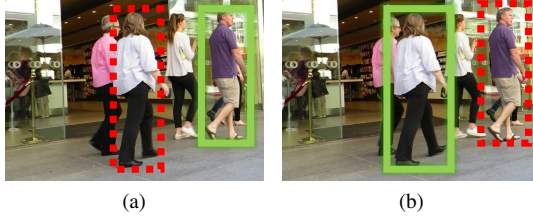


Figure 1: State-of-the-art object detector Mask-RCNN is inconsistent on two images taken 0.03s apart, even though both images look alike. In (a), the woman is missed (red dashed-line box) while the man is detected (green solid box). In (b), the reverse is true. (All other people are detected correctly in both images, giving an average 3/4 accuracy in both images.)

desirable. For example, in Fig. 1, a state-of-the-art object detector (Mask-RCNN [2]) detects three out of four people per image. The accuracy is the same *on average* (3/4), yet the detector behaves inconsistently: it misses a different person in each image.

Inconsistent behavior becomes cause for concern in vision applications that demand strict performance guarantees. For example, a collision prevention algorithm with 95% accuracy that misses the same 5% of objects allows one to investigate the cause of the 5% error more easily, because the defective behavior is *consistent*. However, an algorithm that behaves inconsistently, missing different objects across each image in the test, is much harder to troubleshoot.

This article investigates object detector *consistency* as a method to augment existing accuracy metrics. Consistency measures the difference in predictions from an object detector across *similar* images. We explore different methods to quantify consistency, ultimately choosing a metric that tracks a detector’s behavior on time-series images from the MOT Challenge [3]—a dataset originally intended to benchmark object tracking. As shown in Fig. 2, we find that state-of-the-art detectors (Mask-RCNN, Faster-RCNN [4], RetinaNet [5], SSD [6]) exhibit inconsistent behavior. In our experiments, we observe up to an average of 17% inconsistent detections. We evaluate methods to improve consistency and present a selection of methods (lossy image compression, gamma boosting, etc.) that successfully raise consistency by up to 5%.

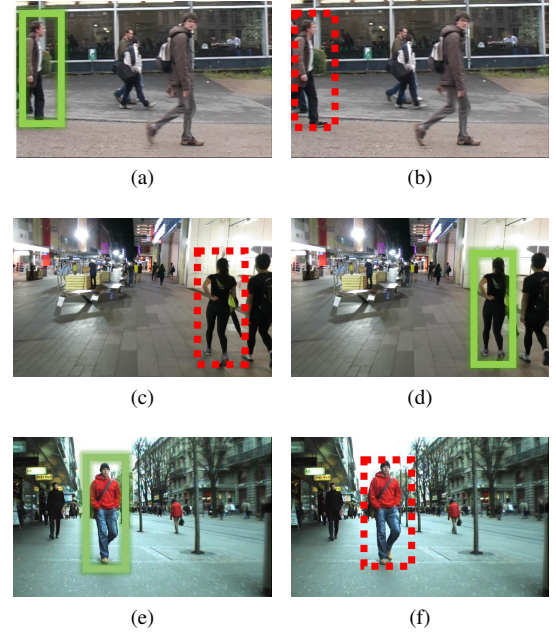


Figure 2: Different object detectors behave inconsistently on images that look visually similar. Left images are taken 0.03s before the right. Red, dashed-line boxes are missed detections, green solid boxes are correct detections. (a)-(b) Faster-RCNN. (c)-(d) RetinaNet. (e)-(f) SSD.

What accuracy measures

Accuracy is determined by comparing the predictions of an object detector against the ground truth in a dataset of images. Standard accuracy metrics like mean Average Precision (mAP) usually consider two factors simultaneously: (1) how much of the ground truth the detector successfully predicts, and (2) how many of the detector’s predictions were incorrect. To get perfect accuracy, each of the predictions an object detector makes must be correct: it cannot blanket the image with guesses.

The mAP accuracy metric is defined in terms of *precision* and *recall* using true/false positives and negatives (see Tab. 1). Precision and recall are defined in Eqn. 1 and 2, respectively. mAP is calculated for a given image dataset by integrating the area under the precision-versus-recall curve with respect to recall (Eqn. 3).

$$\text{precision } p = \frac{TP}{TP + FP} \quad (1)$$

$$\text{recall } r = \frac{TP}{TP + FN} \quad (2)$$

Table 1: Terminology used for calculating object detection accuracy.

	Object detected	Object not detected
Object exists	True Positive (TP)	False Negative (FN)
Object does not exist	False Positive (FP)	True Negative (TN)

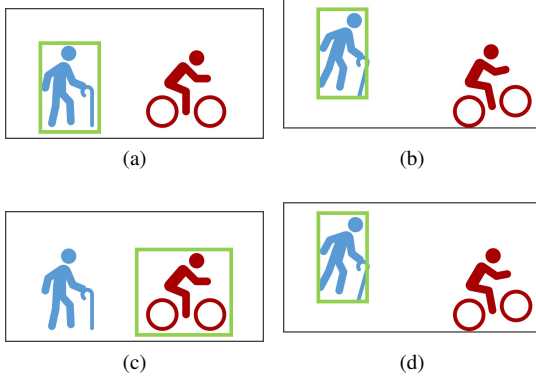


Figure 3: Accuracy does not capture consistency. For object detections shown in similar-looking images, (a)-(b) has the same average accuracy (i.e. 1/2 correct) as (c)-(d). (a)-(b) has consistent detections (same object detected in both images), but (c)-(d) is inconsistent (different objects detected in both images).

$$\text{accuracy} = \int_0^1 p(r) dr \quad (3)$$

Accuracy does not measure consistency

When accuracy is not 100%, there may be more to the story (for context, the state-of-the-art Mask-RCNN reaches 63% mAP on the COCO dataset). Accuracy does not describe whether the detector is consistent. Consider an object detector that achieves 50% accuracy on a set of images, where each image is slightly different but still contains the same objects. One might hope that the detector would predict consistently as shown in Fig. 3a-b. However, it is still possible for other detector predictions (Fig. 3c-d) to achieve the same 50% accuracy, albeit inconsistently.

Additionally, simply reporting fine-grained accuracy statistics does not capture consistency. It is true that we can infer more information about the neural network’s consistency by reporting additional statistics like the standard deviation or variance of per-image accuracy across a dataset of

similar images. However, reporting the standard deviation still does not reveal the Fig. 3c-d case discussed above, where the detector inconsistently detects and misses *different* objects in each frame even though the *number* of objects remains the same.

As we showed earlier in Fig. 2, inconsistent behavior exists even in popular object detectors. Thus, accuracy does not always communicate the full picture of a detector’s performance because it does not explicitly describe consistency. Inconsistency may have severe consequences in applications that involve safety.

Related work

There is a growing number of studies to improve computer vision, but they do not focus on consistency. These efforts can largely be grouped into two categories: (1) adversarial attacks and (2) synthetic image distortions.

Adversarial attacks

Adversarial attacks present a significant challenge to neural networks. Goodfellow, et al. [7] demonstrate that a well-trained image classifier can be tricked into misclassifying an image by slightly perturbing the values of the pixels. In a typical adversarial attack, an algorithm makes minute, calculated adjustments to the pixels of a correctly predicted image until the neural network makes an incorrect prediction. The final image, known as the adversarial sample, appears to human eyes as very similar to the original image. Several different adversarial samples are shown in Fig. 4.

Existing methods to defend against adversarial attacks often involve some combination of (1) including adversarial samples during network training [7], (2) transforming input images into a lower-dimensional space before feeding the neural networks [10], and (3) filtering adversarial samples through a custom neural network before they reach the main network [11].

Synthetic image distortions and data augmentation

Computer vision models can make incorrect predictions on images from natural sources (i.e., not manipulated for adversarial attacks) [12]. Two images of the same scene can be captured by

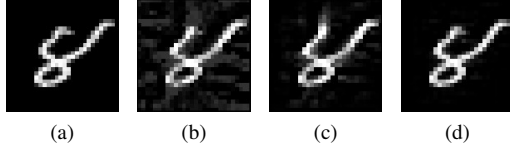


Figure 4: Similar-looking adversarial samples generated from an image in the MNIST handwriting dataset. (a) Original image. Correctly classified as “8” with 0.9 confidence. (b) Fast-Gradient Sign Method adversarial sample. Wrongly classified as “4” with 0.9 confidence [7]. (c) Jacobian Saliency Map Attack adversarial sample. Wrongly classified as “9” with 0.6 confidence [8]. (d) DeepFool adversarial sample. Wrongly classified as “4” with 0.83 confidence [9].

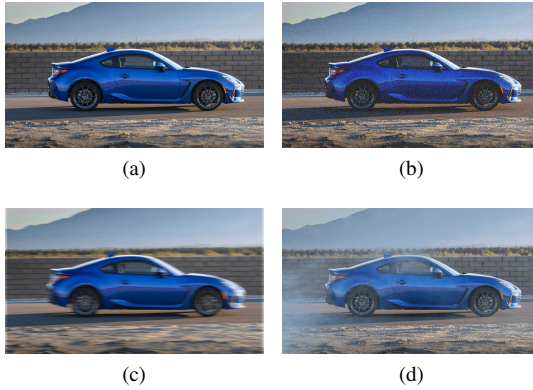


Figure 5: Examples of synthetic image distortions. (a) Original image. (b) Increased brightness. (c) Motion blur. (d) Artificial fog.

the same camera less than a second apart, yet the predictions of a neural network on those two visually similar images can differ dramatically. The small differences between the two images are called *image distortions*, and they can be caused by a range of factors, including ambient light level and camera sensor noise [13].

To make neural networks more robust against image distortions, data augmentation is widely used. The networks are trained on datasets that are modified, or synthetically distorted, to emulate the natural distortions. Common synthetic image distortions (shown in Fig. 5) include perturbing the pixels with Gaussian noise, adding artificial motion blur, adjusting color saturation and brightness, and even adding computer-generated fog, snow, and rain. [12]

The discussed previous work uses only accu-



Figure 6: Consistency is only meaningful when measured across images that look similar (a), (b) (where (b) is taken a second later and camera autofocus is slightly blurred). Consistency is meaningless on images that look different (a), (c). Popular datasets use images that look different, making them appropriate for measuring accuracy, but less suited for measuring consistency.

racy as the metrics, without explicitly measuring consistency. Although some publications [14], [15] explore ways to measure the robustness of a neural network beyond accuracy, they neither concretely define consistency nor propose solutions. This article aims to do both.

Considerations when measuring consistency

In this article, we define consistency as a metric of how differently an object detector behaves on similar images:

If images appear similar to the human eye, an object detector should consistently detect the same objects.

Since accuracy does not always quantify consistency, we now discuss the considerations needed to properly capture consistency.

Use Video/Time-Series Data: consistency measurements require similar test images

A meaningful metric for consistency should use input images that are already consistent. Consider Fig. 6a, b. Those pictures appear similar to the human eye, so we would expect consistent performance from an object detector. However, if the images are significantly different (Fig. 6a, c), it is meaningless to use them to make claims about consistency.

As Tung, et al. [13] observe, popular image datasets (e.g., ImageNet, Microsoft COCO) are filled with visually dissimilar images. Thus, those datasets are largely unsuitable for consistency testing. Instead, those authors recommend consecutive frames from videos as a better source

Quantifying image similarity

This article emphasizes the importance of using similar-looking images to test for consistency. We use consecutive frames from videos in the MOT Challenge because such frames are taken up to 30 times per second, ensuring that adjacent frames look similar.

Beyond the scope of this article, other popular image similarity measurement methods attempt to relate two images beyond raw pixel value differences. The Structural Similarity Index (SSIM) [16] identifies structural details about the images and then compares the details. This means that SSIM identifies a noisy version of an image as similar to the original, even though raw pixel values are very different. Beyond SSIM, techniques such as the Feature Similarity Index (FSIM) [17] and deep-learning driven comparison extract low-level features to better approximate the way humans compare images.

of visually similar images. Although adversarial attacks and artificial image transformations can also be used to generate consistency test data from such popular datasets, Gu, et al. [15] report that such techniques do not well represent image distortions that would occur naturally. Instead, this paper uses consecutive frames from video to test consistency - this way, any inconsistencies would be caused by natural image distortions.

Use MOT Ground Truth: Consistency measurements need additional labels

When benchmarking an object detector, bounding box and class ground truth labels are sufficient to report accuracy, but they cannot reveal all inconsistencies. In particular, that ground truth cannot show whether the same objects were detected between two similar images (the above Fig. 3c-d problem): the ground truth is identifier-agnostic. Thus, we need a way to check whether objects from two images are the same, so that consistency can be measured.

We choose to use per-image object identifier

(Object ID) ground truth labels to keep track of objects during measurement. Each unique object is assigned the same Object ID across the dataset.

Proposed method of measuring consistency

Based on our prior discussion of accuracy vs. consistency, we present a method that specifically tracks whether an object detector detects the same objects, given visually similar, time-series images. Our source of visually similar images is the Multi-Object Tracking (MOT) Challenge [3], consisting of high-quality videos from various datasets.

The *pairwise consistency* $C_{i,j}$ refers to the object detector's consistency on a pair of images I_i and I_j . It is calculated as shown in Eqn. 4. If the detector is perfectly consistent, $C_{i,j}$ is 1. If it is entirely inconsistent, then $C_{i,j}$ is 0. As shown in Fig. 7, consistency looks to capture whether objects were detected in one image and missed in another.

$$C_{i,j} = \frac{|G_i \cap G_j| - |M_{i,j}| - |M_{j,i}|}{|G_i \cap G_j|} \quad (4)$$

Eqn. 4 is explained using the example in Fig. 8. Fig. 8a is image I_i , and Fig. 8b is image I_j . G_i is the set of I_i 's ground truth Object IDs {A, B, C}, and G_j is the set of I_j 's ground truth Object IDs {D, A, B}. $M_{i,j}, M_{j,i}$ captures the objects that were inconsistently detected as follows: $M_{i,j}$ is the set of ground truth Object IDs that satisfy the following conditions: (1) the ground truth box is present in both images I_i, I_j (i.e. in $G_i \cap G_j$), (2) the object detector detected the object in frame I_i , and (3) the object detector missed the object in frame I_j . So $M_{i,j}$ is object B, while $M_{j,i}$ is empty.

If an object is present in both images, yet is detected in only one of the images, then consistency should decrease. Taken together, $M_{i,j}, M_{j,i}$ captures consistency decreases in I_i, I_j . This also implies that if both $|M_{i,j}|, |M_{j,i}|=0$, the detector can be said to be consistent.

Bounding boxes predicted by the object detector are eligible for consideration in $M_{i,j}, M_{j,i}$ calculations only after being filtered through non-maximum suppression (we use the common IoU threshold = 0.5) and a confidence threshold of

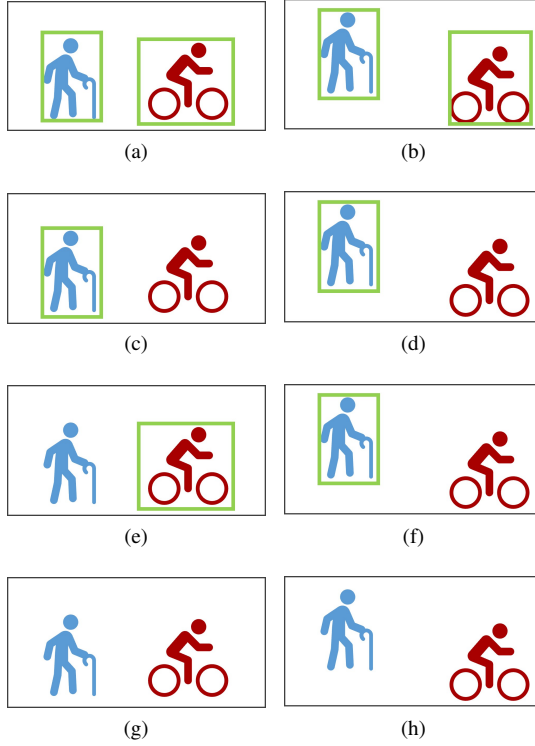


Figure 7: Consistency tracks whether an object is detected in one image and missed in another similar-looking image. This complements accuracy measurements. (a)-(b) 100% accurate (all objects correctly detected in both images), 100% consistent (nothing was missed in one image and detected in another). (c)-(d) 50% accurate (only one of two objects detected in each image), 100% consistent (nothing was missed in one image and detected in another). (e)-(f) 50% accurate (only one of two objects detected in each image), 0% consistent (any objects detected in one image was missed in the other). (g)-(h) 0% accurate (nothing detected), 100% consistent (nothing was missed in one image and detected in another). Improving accuracy does not necessarily imply improving consistency and vice versa.

0.7 (see sidebar: “IoU and non-maximum suppression”).

We measure consistency across a given video V with N frames by measuring pairwise consistency between each pair of adjacent frames in the video, and then averaging the results across all $N - 1$ pairs. This is expressed in Eqn. 5.

$$C_V = \frac{1}{N-1} \sum_{i=1}^{N-1} C_{i,i+1} \quad (5)$$

As shown earlier in Fig. 7, consistency and

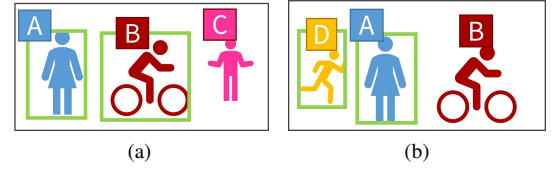


Figure 8: Visualization of Eq. 4, where (a) is I_i , (b) is I_j and the green boxes indicate object detections. $G_i \cap G_j$ contains objects A and B since they appear in both I_i, I_j . Objects C and D are not included in calculations because they do not appear in both images. Because object B is detected in I_i but missed in I_j , so $M_{i,j}$ contains object B. No shared boxes are detected in I_j and missed in I_i , so $M_{j,i} = \emptyset$. Thus, consistency $C_{i,j} = (2 - 1 - 0)/2 = 0.5$.

IoU and non-maximum suppression

IoU (Intersection-over-Union) is a common measurement in object detection, used to determine if two bounding boxes overlap sufficiently to be counted as the same object. It is calculated by dividing the area of two bounding boxes’ overlap by the area of the union of the two boxes. If $\text{IoU} = 1$, then the boxes perfectly overlap. If $\text{IoU} = 0$, the boxes have no overlap. In literature, an IoU threshold of 0.5 is typically used [13] to decide if two bounding boxes sufficiently overlap.

Non-maximum suppression is a common application of IoU to filter an object detector’s predictions so that only the “best” ones remain. It finds all overlapping predicted bounding boxes (determined by the IoU threshold) and then filters out the ones that have the same object class and lower confidence scores.

accuracy can work together to supply more information about object detection performance than either could on its own. Improving accuracy does not necessarily imply improved consistency and vice versa. Colloquially, one might say that consistency indicates how similarly an object detector behaves on two similar images, while accuracy indicates whether that behavior is desirable (detects everything consistently) or undesirable (misses

everything consistently).

Finally, note that we choose to compare bounding box object IDs instead of output feature maps to calculate consistency. This is because comparing feature maps requires arbitrary similarity metrics - selecting an appropriate metric is itself an open problem. Additionally, bounding boxes are already used for accuracy measurements; re-using boxes for consistency will make it more convenient for researchers to take consistency measurements.

Consistency of modern object detectors

We present the consistency of several state-of-the-art object detectors, showing that they exhibit inconsistent behavior. We demonstrate using highly accurate two-stage object detectors (Faster-RCNN and Mask-RCNN) as well as the faster, but less accurate, single-shot detectors (RetinaNet and SSD). We use Facebook’s official, pretrained models from their `torchvision` Python package. Measurements are taken on the videos found in the MOT Challenge.

As shown in Fig. 9, all object detectors exhibit some inconsistent behavior, ranging from 83.2% to 97.1% consistency (C_V as calculated in Eqn. 5). We also see that the two-stage models are more accurate and more consistent.

Toward consistent object detection

Object detector inconsistency is caused by the detector missing an object. As described in Related Work, missed detections can be caused by adversarial attacks and image distortions. Because the MOT Challenge is not an adversarial dataset, we expect the inconsistencies to be caused by image distortions naturally present in the dataset. Therefore, we apply different image distortion corrections from literature to compare their efficacies at improving consistency.

1) Gaussian Denoise (GD). Random sensor noise and shot noise can decrease detection performance. As demonstrated by Kang et al. [20], we apply a normalized Gaussian filter to all images in the dataset in an attempt to reduce the noise.

2) Horizontal Flip (HF). Zhang, et al. [14] note that the slight translation of an object in an image could cause a previously misdetected object to become correctly detected. Further, Yin,

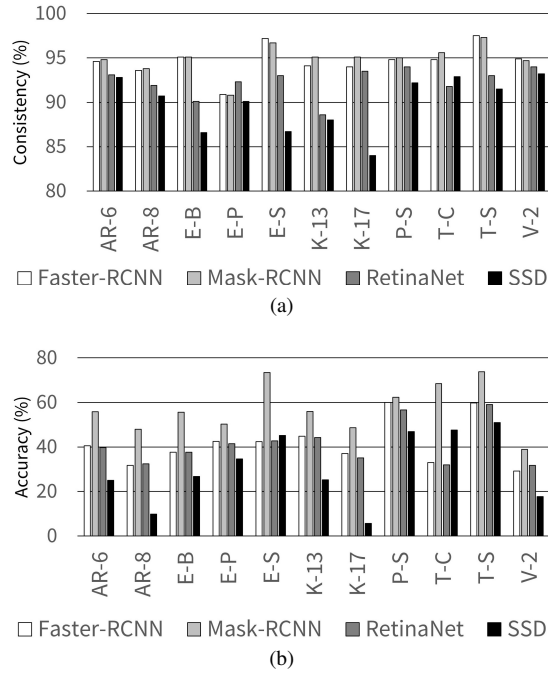


Figure 9: Object detector consistency (our method, (a)) and accuracy (mAP, (b)) measured on the different videos in the MOT Challenge. Videos are: AR-6: ADL-Rundle-6, AR-8: ADL-Rundle-8, E-B: ETH-Bahnhof, E-P: ETH-Pedcross2, E-S: ETH-Sunnyday, K-13: KITTI-13, K-17: KITTI-17, P-S: PETS09-S2L1, T-C: TUD-Campus, T-S: TUD-Stadtmitte, V-2: Venice-2. State-of-the-art object detectors exhibit inconsistent behavior, ranging from 83.2% to 97.1% consistency. The two-stage models are both more consistent (a) and more accurate (b) than their single-shot counterparts.

et al. [21] observe that horizontally flipping an image can mitigate the detrimental effects of noise on a neural network. Thus, we horizontally flip all images.

3) WEBP Compression (WC). Compressing an image using the lossy JPEG format is already known to defeat adversarial attacks [12]. Yin, et al. [21] find that because WEBP compression introduced loop filtering, it is even better suited to breaking down the structures in an image that result from synthetic image distortions. Thus, we apply WEBP compression to the dataset images using a compression quality factor of 30 (Yin, et al. find that lower quality factors allow neural networks to perform better).

4) Unsharp Mask (UM). Tung, et al. [13] explain that if an object is moving, the motion blur can make it harder for a network to extract

Improving consistency via training

Common training techniques such as *data augmentation* and *dropout* are known to improve a model’s robustness to image transformations and distortions. Despite these techniques being employed to produce our pretrained models, we find that inconsistencies still persist. To improve consistency, we use post-training image distortion corrections because of their accessibility. However, emerging training methods appear promising for improving consistency. Zhang et al. propose weakly-supervised, context-based techniques [18], [19] that gather context for the scene (e.g., optical flow and prior knowledge) to provide additional information to an object detector - this information could stabilize the detector and improve consistency over time-series data from videos. Other techniques project neural network features into low-dimensional representations during training [10] - this could improve consistency on images of similar objects taken from different angles, lighting, etc.

features along the object’s edges. The Unsharp Mask is a linear image processing technique commonly used to remove blur. The technique first identifies a set of blurry details by subtracting a further-blurred image from the original. The details are then emphasized in the original image. We apply the Unsharp Mask to the dataset to reduce the blur on object edges.

5) Gamma Correction (GC). Yeu, et al. [22] show that artificially increasing an image’s contrast and perceived brightness can help object detectors like Faster-RCNN perform better, particularly when the detectors are trained on day-time images. Gamma Correction is a common brighten/contrast technique that is driven by the Power Law expression; we use it on the dataset as well.

Tab. 2 shows the average improvement across the MOT Challenge in terms of consistency percentage points (i.e. a table entry of Y% means that it raises consistency from X% to X+Y%), when

Table 2: Avg. Consistency Improvements

	Faster-RCNN	Mask-RCNN	RetinaNet	SSD
GD	0%	-0.3%	0%	-0.6%
HF	-5.3%	-5.4%	-7.3%	-10.1%
WC	0.6%	0.5%	0.7%	0.4%
UM	3.6%	2.6%	3.0%	1.1%
WC+UM	5.1%	3.0%	3.2%	1.3%
GC	0.1%	0.1%	0.4%	0.1%

Table 3: Avg. Accuracy Improvements

	Faster-RCNN	Mask-RCNN	RetinaNet	SSD
GD	2.1%	2.4%	-0.6%	-1.1%
HF	-19.3%	-19.4%	-25.5%	-28.4%
WC	1.5%	1.8%	0.5%	0.5%
UM	2.0%	3.2%	8.3%	3.6%
WC+UM	3.2%	4.1%	8.6%	3.9%
GC	0.1%	-0.5%	-0.7%	-0.1%

the different distortion corrections are applied. Similarly, Tab. 3 shows the accuracy improvement.

We see that both WEBP Compression (WC) and Unsharp Mask (UM) improve both consistency and accuracy for all object detectors. Applying both effects at the same time (WC+UM) gives a further overall improvement. In fact, the example inconsistencies in Fig. 1 and 2 are resolved using WC+UM. Gaussian Denoise (GD) and Horizontal Flip (HF) both degrade consistency and accuracy (likely because applying these effects on relatively un-noisy images degrades the feature structure of the images [17]).

Finally, we note that improvements in consistency do not always equate to improvements in accuracy. Gamma Correction (GC) improves consistency, but degrades accuracy: in other words, the detector is consistently worse on the GC data, as described earlier in Figure 7.

Conclusion and Future Work

Object detectors are vital to many modern computer vision applications. However, even state-of-the-art object detectors exhibit inconsistent behavior when the input undergoes small changes. This inconsistent behavior is not fully captured by existing measurement tools; accuracy metrics and popular image datasets cannot measure whether the same objects are detected consistently. We devise a consistency measurement method that uses images from videos and object ID labels. Our method compliments accu-

racy measurement. Using this method, we show that object detectors have consistency ranging from 83.2% up to 97.1%, depending on the input data. Additionally, applying image distortion corrections like WEBP Compression and Unsharp Masking can improve consistency by as much as 5.1%. There is still room for improvement by the community. We only explore post-training methods to raise consistency. Future exploration should explore training-aware consistency improvements and further investigate the relationship between accuracy and consistency.

Acknowledgments

This project is supported in part by the National Science Foundation OAC-2104709. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Authors

Caleb Tung is a doctoral student in the Elmore Family School of Electrical and Computer Engineering at Purdue University. Caleb's research is on designing energy-efficient neural network operations for computer vision on embedded devices. Outside the lab, Caleb can be found at the piano bench or somewhere in a good book.

Abhinav Goel is a doctoral candidate in the Elmore Family School of Electrical and Computer Engineering at Purdue University. Abhinav's research interests are in the area of energy-efficient artificial intelligence systems.

Fischer Bordwell is a Master's student in the Elmore Family School of Electrical and Computer Engineering at Purdue University. Fischer's research is on utilizing computer vision to reduce misinformation in the news.

Nick Eliopoulos is currently is a doctoral student the Elmore Family School of Electrical and Computer Engineering at Purdue University. Outside of academics, Nick enjoys drawing and cooking.

Xiao Hu is an M.S. student in the Elmore Family School of Electrical and Computer Engineering at Purdue University. His research broadly connects to the fields of human-computer interaction, low-power computer vision, fairness

in artificial intelligence, machine learning, and full-stack web development.

George K. Thiruvathukal is a professor and chairperson in the Department of Computer Science of Loyola University Chicago and visiting computer scientist at Argonne National Laboratory in the Leadership Computing Facility. His research interests include parallel computing, software engineering, and computer vision.

Yung-Hsiang Lu is a professor in the Elmore Family School of Electrical and Computer Engineering at Purdue University. He is also the director of Purdue's John Martinson Engineering Entrepreneurial Center. His research topics include computer systems, computer vision, and embedded systems. He is a Fellow of the IEEE and Distinguished Scientist of the ACM.

REFERENCES

1. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), Lecture Notes in Computer Science, (Cham), pp. 740–755, Springer International Publishing, 2014.
2. K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018.
3. L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking," *arXiv:1504.01942 [cs]*, Apr. 2015. arXiv: 1504.01942.
4. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems 28* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 91–99, Curran Associates, Inc., 2015.
5. T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," pp. 2980–2988, 2017.
6. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *Computer Vision – ECCV 2016* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), Lecture Notes in Computer Science, (Cham), pp. 21–37, Springer International Publishing, 2016.
7. I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in *2015 Inter-*

- national Conference on Learning Representations*, Mar. 2015.
8. S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal Adversarial Perturbations," pp. 1765–1773, 2017.
 9. S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks," pp. 2574–2582, 2016.
 10. T. d'Orsi, P. K. Kothari, G. Novikov, and D. Steurer, "Sparse PCA: Algorithms, Adversarial Perturbations and Certificates," in *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 553–564, Nov. 2020. ISSN: 2575-8454.
 11. S. Gu and L. Rigazio, "Towards Deep Neural Network Architectures Robust to Adversarial Examples," *arXiv:1412.5068 [cs]*, Apr. 2015. arXiv: 1412.5068.
 12. S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–6, June 2016.
 13. C. Tung, M. R. Kelleher, R. J. Schlueter, B. Xu, Y. Lu, G. K. Thiruvathukal, Y. Chen, and Y. Lu, "Large-Scale Object Detection of Images from Network Cameras in Variable Ambient Lighting Conditions," in *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pp. 393–398, Mar. 2019.
 14. R. Zhang, "Making Convolutional Networks Shift-Invariant Again," in *International Conference on Machine Learning*, pp. 7324–7334, May 2019.
 15. K. Gu, B. Yang, J. Ngiam, Q. Le, and J. Shlens, "Using Videos to Evaluate Image Model Robustness," *arXiv:1904.10076 [cs]*, Aug. 2019. arXiv: 1904.10076.
 16. Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, Apr. 2004. Conference Name: IEEE Transactions on Image Processing.
 17. L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: A Feature Similarity Index for Image Quality Assessment," *IEEE Transactions on Image Processing*, vol. 20, pp. 2378–2386, Aug. 2011. Conference Name: IEEE Transactions on Image Processing.
 18. D. Zhang, J. Han, L. Yang, and D. Xu, "SPFTN: A Joint Learning Framework for Localizing and Segmenting Objects in Weakly Labeled Videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 475–489, Feb. 2020. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.
 19. D. Zhang, J. Han, L. Zhao, and D. Meng, "Leveraging Prior-Knowledge for Weakly Supervised Object Detection Under a Collaborative Self-Paced Curriculum Learning Framework," *International Journal of Computer Vision*, vol. 127, pp. 363–380, Apr. 2019.
 20. D. Kang, Y. Sun, D. Hendrycks, T. Brown, and J. Steinhardt, "Testing Robustness Against Unforeseen Adversaries," *arXiv:1908.08016 [cs, stat]*, June 2020. arXiv: 1908.08016.
 21. Z. Yin, H. Wang, J. Wang, J. Tang, and W. Wang, "Defense against adversarial attacks by low-level image transformations," *International Journal of Intelligent Systems*, vol. 35, no. 10, pp. 1453–1466, 2020. _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/int.22258>.
 22. Y. H. Yeu, M. I. Shapiai, Z. H. Ismail, and H. Fauzi, "Investigation on Different Color Spaces on Faster RCNN for Night-Time Human Occupancy Modelling," in *2019 IEEE 7th Conference on Systems, Process and Control (ICSPC)*, pp. 118–121, Dec. 2019.