How We Should Evaluate Vision Algorithms?

Alan Yuille

What Have Deep Nets done to Computer Vision?

- Compared to human observers, Deep Nets are brittle and rely heavily on large annotated datasets. Unlike humans, Deep Nets have difficulty learning from small numbers of examples, are oversensitive to context, have problems transferring between different domains, and lack interpretability.
- Ho well do Deep Networks really perform? Are our current methods for testing them adequate? Can they deal with the combinatorial complexity of real world stimuli?
- *Alan Yuille and Chenxi Liu. "Deep Networks: What have they ever done for Vision?". Arxiv. 2018.*

The Fundamental Problem

- Current methods for evaluating algorithms are highly problematic.
- They are based on machine learning concepts – balanced finite-size training and test sets – which are often not appropriate for real world situations. They favor regression algorithms – e.g., deep networks – which risk overfitting the datasets (both the testing and training sets).
- They can succeed by exploiting biases in the datasets. All datasets have some biases. The dataset should be sufficient to represent the complexity of the real world and natural images. But this complexity is so big that finite-sized datasets may be insufficient to capture it.

Two Issues

- How to learn models that generalize from one dataset to another with, if possibly, limited training?
- How to test models for performance that works in real world conditions, i.e. not just on the datasets it is trained on?

- These are two separate questions. This lecture will mainly focus on the second – but these issues has big implications for what types of algorithms computer vision researchers should use.

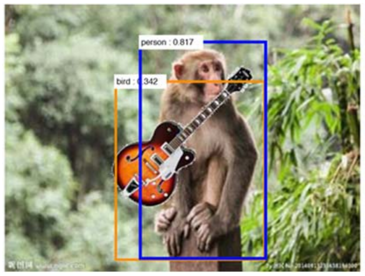
Examples: from meetings last week.

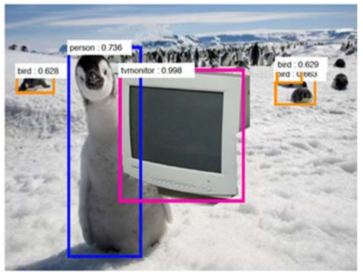
- GBO exam. Counting the number of people in images. Three datasets. Fairly good performance on all datasets. But algorithms trained on one dataset do not transfer to other datasets.
- MIT Technology Article: The AI developed by Google Health can identify signs of diabetic retinopathy from an eye scan with more than 90% accuracy—which the team calls “human specialist level”—and, in principle, give a result in less than 10 minutes. The system analyzes images for telltale indicators of the condition, such as blocked or leaking blood vessels. *The system worked well on datasets in Google, but mostly failed on real patients in Thailand.*

Why don't results transfer?

- Case Study: Edge Detection – Sowerby and South Florida datasets.
- Sowerby Images – English Country Scenes: significant texture in background, edges not very sharp. (figure top left: left panel)
- South Florida dataset – Mostly indoor images: texture regions removed, edges very sharp (step like). (figure top left: right panel).
- Different types of image statistics.
- But transfer is possible with generative methods (Konishi et al.) P(f|on-edge) P(f|off-edge). The statistics P(f|off-edge) differ between datasets but can assume that P(f|on-edge) is similar. Exploit fact that most image pixels are off-edge.
- See Konishi et al. 2003. Handout: edge detection lecture in this course.

Why Don’t Results Transfer?

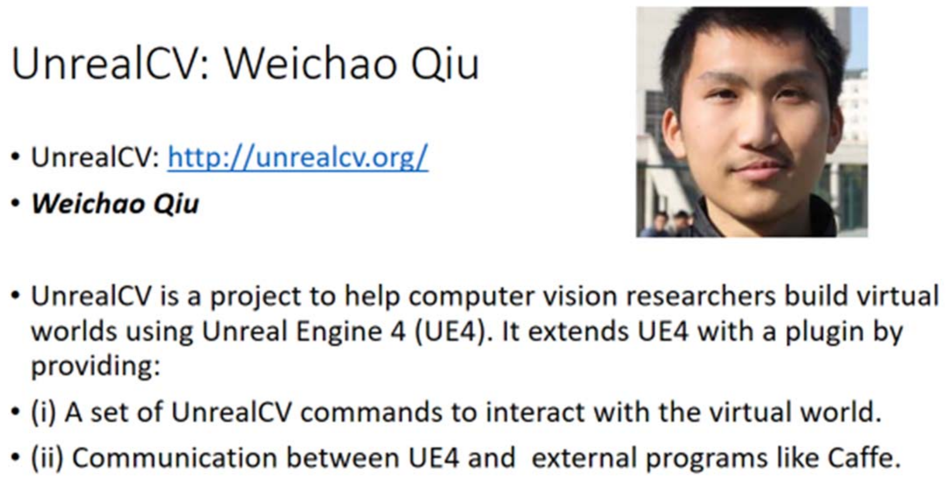




- Bias of context, viewing conditions.
- Rare events. Corner cases. “In the real world everything is a corner case” (Anonymous annotator at a DARPA presentation).
- Background/context bias (see figure top right).
- Caltech 101: fish are the only objects which occur in water (i.e. can detect fish by detecting water).
- UFC activity classification dataset: boxing occurs in a boxing ring – detecting the boxing ring detects boxing (see later).
- Many rockstars are photographed with guitars. But a guitar is not a Rockstar (see presentation later in this lecture).
- A. Torrabla & A. Efros. Unbiased Look at Dataset Bias. CVPR. 2011.

Virtual Data: Making Controlled Datasets

- Tools like UnrealCV enable us to generate datasets which have many annotations and which test algorithms systematically.
- This enables us to stress test algorithms in challenging conditions.



UnrealCV: Weichao Qiu

- UnrealCV: http://unrealcv.org/
- ***Weichao Qiu***

- UnrealCV is a project to help computer vision researchers build virtual worlds using Unreal Engine 4 (UE4). It extends UE4 with a plugin by providing:
- (i) A set of UnrealCV commands to interact with the virtual world.
- (ii) Communication between UE4 and external programs like Caffe.

Using Virtual Stimuli to Stress-Test Algorithms.

- Object detection algorithms (W. Qiu & A.L. Yuille. ECCV workshop 2016).
- E.g., Sofa detectors trained on ImageNet may not work on other data.

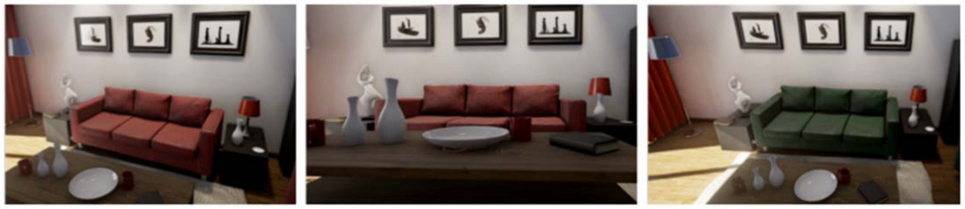
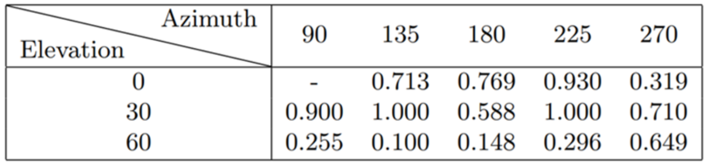


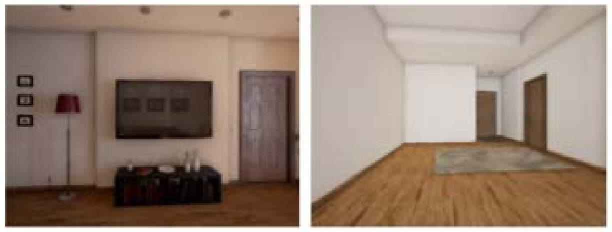
Fig. 4. Images with different camera height and different sofa color.



Elevation \ Azimuth	90	135	180	225	270
0	-	0.713	0.769	0.930	0.319
30	0.900	1.000	0.588	1.000	0.710
60	0.255	0.100	0.148	0.296	0.649

Table 1. The Average Precision (AP) when viewing the sofa from different viewpoints. Observe the AP varies from 0.1 to 1.0 showing the sensitivity to viewpoint. This is perhaps because the biases in the training cause Faster-RCNN to favor specific viewpoints.

- Stress-test binocular stereo. Yi Zhang et al. UnrealStereo. 3DV. 2018.



(a) Specularity (b) No texture

(c) Disparity jumps (d) Transparency

Synthetic Data: Activity Recognition

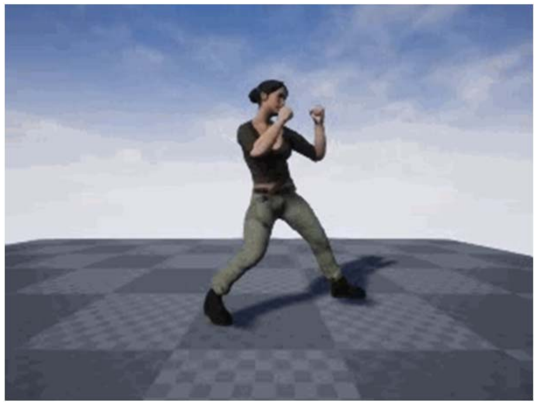
- Activity Recognition is a visual task which is at big risk for combinatorial complexity. Synthetic Data can be used to explore this.
- We render some synthetic videos of humans punching. Train state-of-the art activity recognition methods (TSN and I3D) on these tasks using the USC101 activity dataset.

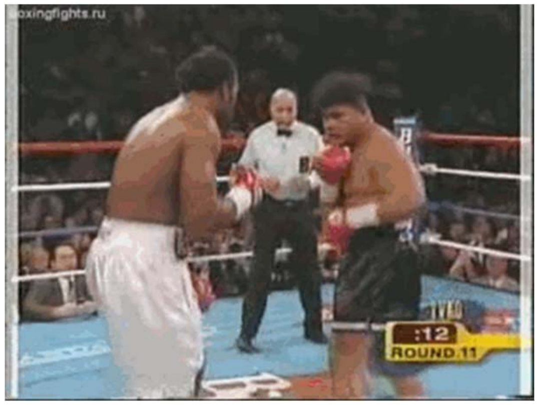
Model	Class Name	Top-1 accuracy	Top-5 accuracy
TSN	Punching	0.00	0.00
I3D	Punching bag	6.25	41.67
I3D	Punching person	6.25	31.25

- Why are the Deep Nets (TSN and I3D) so bad at generalizing to the synthetic data?
- (There are problems for algorithms trained on real to generalize to synthetic, but they are not usually as bad as this).

Why TSN fail to recognize synthetic punching ?

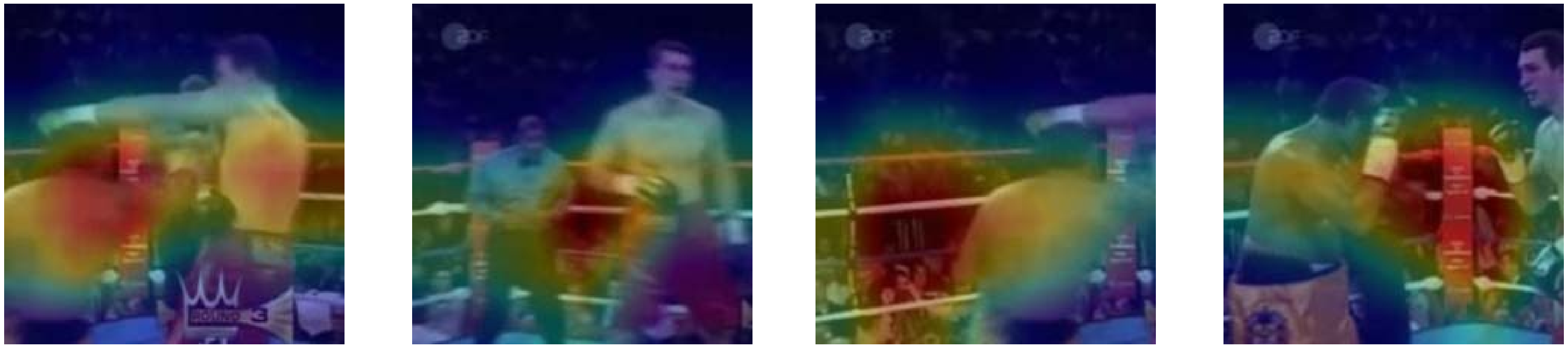
- Conjecture: TSN model trained on UCF101 (right) may have overfit to background and are unable to localize punching action. Synthetic data consists of a single boxer (left).
- Videos from this class in UCF101 are mostly boxing games and punching sandbags.





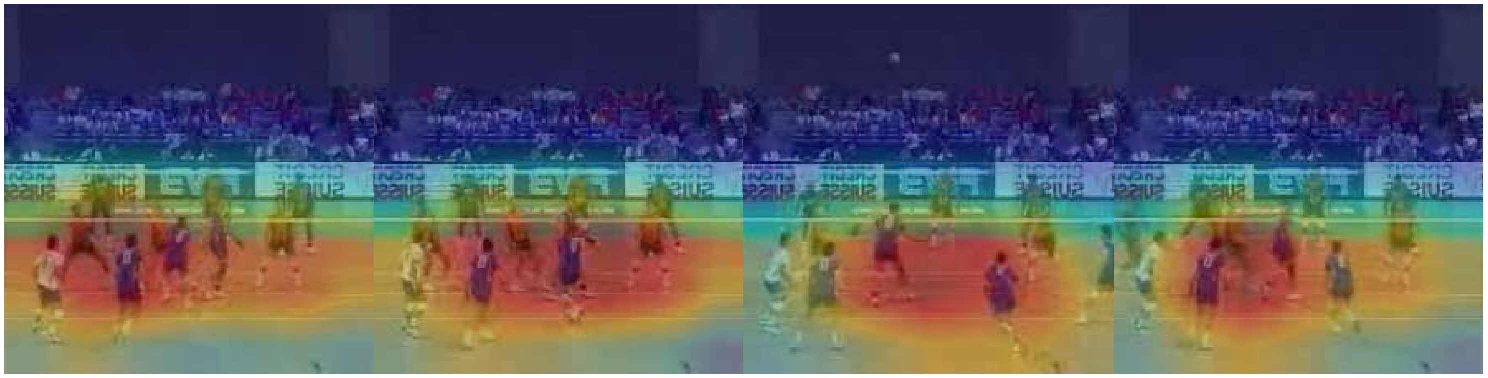
Can the TSN correctly localize the punching action ?

- Class Activation Maps (CAM) are a standard technique to detect the discriminative image regions used by a CNN to identify a specific activity class.
- CAMs of punching videos from UCF101 test set – detecting ropes.



Understanding Scenes

- Volleyball Spiking:



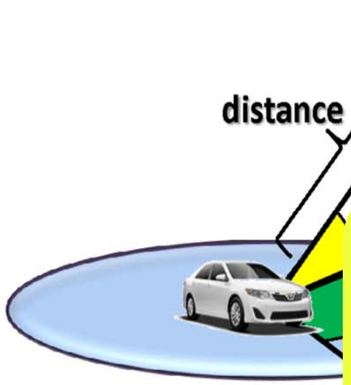
- The model is unable to localize the spiking action in time. It relies on context, i.e. the ability to recognize the scene of volleyball games.

Can you simply make the dataset bigger?

- Thought experiment: for object recognition (unoccluded) the dataset should take into account all the possible viewing conditions (lighting, viewpoint, material, local background).
- From a computer graphic perspective. A model for rendering a 3D virtual scene into an image will have several parameters: e.g.,. camera pose, lighting, texture, material and scene layout. If we have 13 parameters, see next slide, and they take 1,000 values each then we have a dataset of 10^39 images.

- This is a very large number. (Need a model that understands how the data was generated -- i.e. the underlying 13 dimensional space).

Images from synthesized computer graphics model.



Sythesized data: INFINITE image space

Camera Pose(4):
azimuth
elevation
tilt(in-plane rotation)
distance

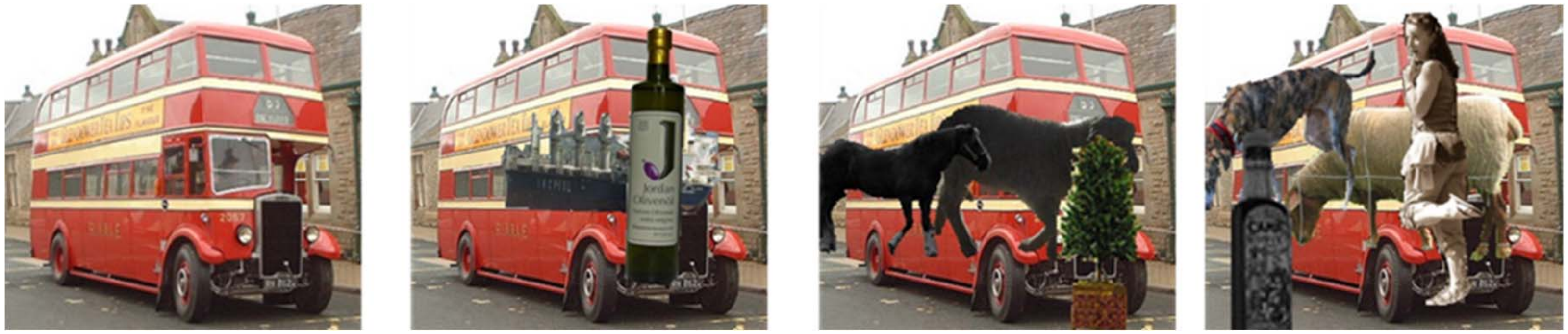
#light source
type(point, dire
omni)
position
color
...

cene Layout(3):
Background
Foreground
Position(Occlusion)

Suppose we simply sample 10^3 possibilities of each parameter listed...

Occlusion

- If you allow occlusions, then the number of possibilities increases exponentially.



- This is a problem if we are testing by random sampling: what is the probability that our samples occur in the problematic situations.

More generally

- What happens if you want to classify a visual scene? There are combinatorial number of ways of constructing visual scenes.
- How can evaluate tasks like image captioning which, in principle, could depend on all the objects present in the image and their precise configurations? How to distinguish between “man kicks dog” and “dog bites man”?
- How can you evaluate activity recognition, which sequences, which involves image sequences over time?

What to do?

- Ignore the problem – publishing papers requires good performance on benchmarked datasets (Hinton's tables). This has some value, it shows that your algorithms work within the domain represented by the dataset, but you should not fool yourself into thinking you have solved the problem!
- Acknowledge the problem – but do nothing, what is there to do? I need publish papers to graduate (if student) or to get new grants (if professor).
- Create challenges that test the algorithms in different ways. Identify abilities which you want to vision algorithms to have (examples later).
- Create alternative ways to testing algorithms – e.g., sequential tests which probe the algorithms to detect weak points, instead of testing with random samples.

Create challenges: motivated by Human Vision

- The human visual system is far superior to current state-of-the-art computer vision algorithms and, in particular, outperforms them in several dimensions:
- (i) Generality: the ability to address multiple visual tasks (e.g., detect objects, parse them into parts, estimate their 3D configurations, find their boundaries)
- (ii) Efficiency: learning from few examples by exploiting prior knowledge and physical properties of the world,
- (iii) Robustness: to viewpoint and challenging viewing conditions, to novel context, occlusion (including several objects overlapping as in Captchas), and to small changes in images which confuse deep networks but do not fool humans,
- (iv) Doman Transfer: deep networks typically perform well only on datasets they have been trained on, but humans can learn on one image domain (e.g., real images, line drawings, computer graphics stimuli, and visual art) and perform well on others.

Make the Datasets bigger and adaptive.

- Classic example – hard negative mining. This was very important for detecting faces and text in real images. Easy to get realistic finite-sized training set of positives examples (faces and text) but how to get negative examples? There are so many.
- Solution: keep expanding the dataset. Initialize with positive and negative training examples. Train your algorithms. Run them over large datasets. Indentity the false positives, put them into your set of negative examples. Retrain your algorithm. Repeat.

- Arguably, “robust” methods for dealing with imperceptible adversarial attacks are following this strategy.

But to go further.

- Let your worst enemy test your algorithm!
- Allow the examiner to ask a sequence of questions, where each question depends on the answer to the previous questions. This enable the examiner to test over an exponential space (like decision trees can test 2^n possibilities with only n questions).
- This is a type of Turing test. Where the examiner can alter the image to find weaknesses of the algorithms.
- Note: testing on random samples (as in machine learning) is not a smart strategy. No teacher would use it to test the knowledge of his students. No engineer would use it to test an advanced machine, like a car or an airplane.