COMS0018: PRACTICAL-Lab4

Dima Damen
Dima.Damen@bristol.ac.uk

Bristol University, Department of Computer Science
Bristol BS8 1UB, UK

October 26, 2018

What else can we do?

- Data Augmentation
- Assessing Dataset Size
- Debugging Strategies

Data Augmentation

- Data augmentation is making the most of the training samples by introducing variations of these samples to accommodate for required invariances
- Why Data Augmentation?
  - Because it's all about the size of your data –> More data for training
  - **More importantly...** to accommodate invariances

Invariances in data

- A *problem* is **invariant** to a *property* when the problem remains unchanged when transformations of a certain type are applied.

Problem	**Invariant to...**
Object Recognition	translation, rotation, scaling,viewpoint
Number plate recognition	translation, scaling
Action Recognition	translation, rotation, scaling, viewpoint, speed

Invariances in data

Problem	Invariant to...
Object Recognition	translation, rotation, scaling, viewpoint

- How to provide invariance? $\rightarrow$ *artificially* augment for:
  - Translation: shifts – luckily CNNs do that for us
  - Rotation: rotations
  - Scaling: croppings
  - Viewpoint: Minor - affine transformations, otherwise :-(collect more data!

Invariances in data

- Other invariances:
  - invariance to random noise
  - invariance to occlusion
  - invariance to lighting conditions
  - invariance to time of year!? — Generative!

Data Augmentation

- Data augmentation for invariances existed before deep learning

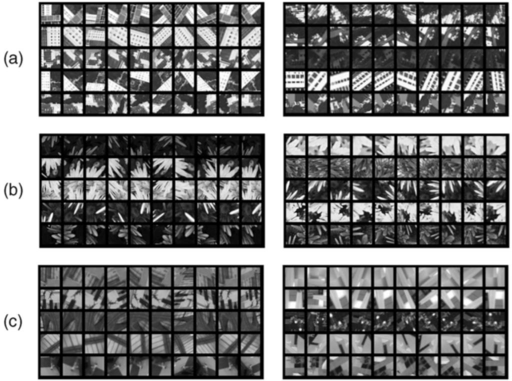


Fig. 7. Warped patches from the images of Fig. 6 show the range of

---

Ozuysal et al (2010). Fast Keypoint Recognition Using Random Ferns. TPAMI.

Data Augmentation

- Why can't current deep learning methods do that automatically for us?

But... Do I need more Data?

- There is a balance between the expense of collecting labelled data and refining the method
- How are you performing on your training data?
  - poorly $\rightarrow$ your algorithm needs work, you are not making the most of the data you have
  - quite well $\rightarrow$ How are you performing on your test data?
    - poorly $\rightarrow$ there is little you can do, collect more data
    - quite well $\rightarrow$ you're done! Find a more interesting problem
- I collected more data but things did not improve?
  - Think about the *quality* of your data
  - Think about the *quality* of your *labels*
  - $\rightarrow$ fix and start over

Debugging Deep Learning Algorithms

- When a general machine learning code performs poorly, including deep learning code, it is very tricky to decide whether that is a bug in the code or a problem in the algorithm
- Compiling correctly and getting numbers out is not an indication of correctness
- We do not know what the "correct" implementation will give in terms of accuracy, that is in fact what we wish to discover
- Careful debugging is thus a must

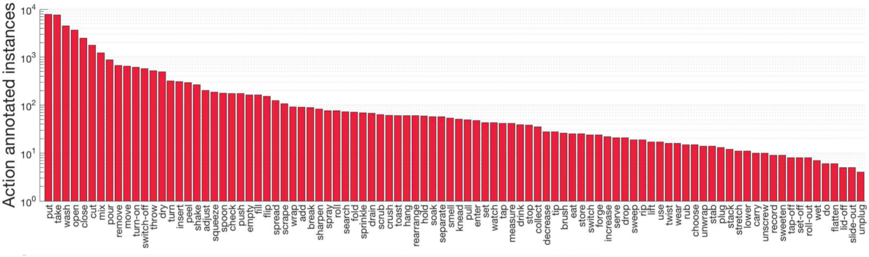
What is your baseline performance?

- Remember: you cannot perform worse than the baseline!
- What can an algorithm that makes decisions by "chance" do?
- For a binary classifier, your baseline is 50%
- For a classifier into N balanced classes, your baseline is $\frac{1}{N}\%$
- For a classifier with unbalanced classes??

Ex. of Baselines - EPIC-Kitchens 2018

- https://epic-kitchens.github.io
- https://youtu.be/Dj6Y3H0ubDw

Ex. of Baselines - EPIC-Kitchens 2018



D Damen, H Doughty, GM Farinella, S Fidler, A Furnari, E Kazakos, D Moltisanti, J Munro, T Perrett, W Price, M Wray (2018). Scaling Egocentric Vision: The EPIC-KITCHENS Dataset. European Conference on Computer Vision (ECCV)

Ex. of Baselines - EPIC-Kitchens 2018

Table 6: Baseline results for the action recognition challenge

	Top-1 Accuracy			Top-5 Accuracy			Avg Class Precision			Avg Class Recall		
	VERB	NOUN	ACTION	VERB	NOUN	ACTION	VERB	NOUN	ACTION	VERB	NOUN	ACTION
Chance/Random	12.62	1.73	00.22	43.39	08.12	03.68	03.67	01.15	00.08	03.67	01.15	00.05
Largest Class	22.41	04.50	01.59	70.20	18.89	14.90	00.86	00.06	00.00	03.84	01.40	00.12
2SCNN (FUSION)	42.16	29.14	13.23	80.58	53.70	30.36	29.39	30.73	5.35	14.83	21.10	04.46
TSN (RGB)	45.68	**36.80**	19.86	**85.56**	**64.19**	**41.89**	**61.64**	34.32	09.96	**23.81**	**31.62**	08.81
TSN (FLOW)	42.75	17.40	09.02	79.52	39.43	21.92	21.42	13.75	02.33	15.58	09.51	02.06
TSN (FUSION)	**48.23**	36.71	**20.54**	84.09	62.32	39.79	47.26	**35.42**	**10.46**	22.33	30.53	**08.83**

D Damen, H Doughty, GM Farinella, S Fidler, A Furnari, E Kazakos, D Moltisanti, J Munro, T Perrett, W Price, M Wray (2018). Scaling Egocentric Vision: The EPIC-KITCHENS Dataset. European Conference on Computer Vision (ECCV)

Debugging Strategies

- *Mickey Mouse Examples* - test your solution on small tests that you know the outcome for
- Evaluate the performance of your building blocks in isolation
- Monitor the model in action
- Look at failure cases (qualitative assessment)

Debugging Strategies

"Directly observing the machine learning model performing its task will help to determine whether the quantitative performance numbers it achieves seem reasonable"

Goodfellow et al, p432

Debugging Strategies

“Evaluation bugs can be some of the most devastating bugs because they can mislead you into believing your system is performing well when it is not”

Goodfellow et al, p432

And now....

READY....

STEADY....

GO...