COMS0018: PRACTICAL - Lab5

Dima Damen
Dima.Damen@bristol.ac.uk

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

November 2, 2018

Previous tutorial

- Data Augmentation - Making the most of your training data

This tutorial

- Beyond test data...
- Why do we have a training/testing split?
- So after we perform well on the test split, can we do anything more?
- Articles where "Deep Neural Networks" outperform humans!
- Fooling DNNs through adding noise

Adversarial Training

- Szegedy et al (2014) constructed a method to optimise the search for a different input x' that is as close as possible to input x but the network's output y' is different than that of y
- This assumes that y is the correct output for x', and attempts to find one optimal x', through moving the input point in an optimal direction.
- By keeping x' as close as possible to x, you hope that a human observer won't be able to see the difference and thus won't be 'fooled'.
- This attempt was not made to prove that DNNs can be fooled, but to increase their robustness by adding x' to the training data, and training for it using the correct label y. This is referred to as adversarial training
- In this lab, you'll attempt to implement and evaluate adversarial training.

Why is it too easy?

- Deep Neural Networks are *"excessively linear"*
- Do you remember all our attempts to make them non-linear?
- They are still made up of *primarily* linear blocks
- The output of a linear function $y = f(\mathbf{x})$ can change very rapidly with minor changes of $\mathbf{x}$ for high-dimensional input $\mathbf{x}$.

The essence of adversarial training

- discourages highly sensitive locally linear behaviour.
- by encouraging the network to be locally consistent in the neighbourhood of each training sample.
- DNNs can represent functions that can range from ‘highly linear’ to ‘locally consistent’. Adversarial training encourages the latter weights.
- Does not guarantee that the trained adversarial network indeed belongs to the right class y.
- The main assumption is that small ‘independent’ perturbations to the training input $\mathbf{x}$ do not cause a jump from one class to another.

Independent Perturbations to $\mathbf{x}$

- We call these perturbations independent because we search for these for each dimension independently
- In an image, we search for changes to each pixel in the image, towards the optimal x'

Adversarial Training in Today's Lab

	training	**testing**
1	normal training	normal testing
2	normal training	adversarial testing
3	normal + adversarial training	normal testing
4	normal + adversarial training	adversarial testing

By the end of these lab sessions, you should be able to...

- Define a Fully-Connected Deep Neural Network (DNN) architecture
- Define a Convolutional Neural Network (CNN) architecture, as well as understand a pre-implemented one
- Train and validate a CNN, and monitor its progress and results using Tensorboard
- Understand and estimate the effect of changing hyper-paramters on your results
- Implement and evaluate a variety of data augmentation techniques
- Implement and evaluate the effect of adversarial training (today)

By the end of these lab sessions, you should be able to...

- By the end of this lab, you can upload all your zip files to SAFE:
  - Lab_1_username.zip
  - Lab_2_username.zip
  - Lab_3_username.zip
  - Lab_4_username.zip
  - Lab_5_username.zip
- Deadline is 28th of November - **but you can do it asap**
- These will be marked for completion and originality - no judgement on any choices you made.
- Labs should be individual work. You can use your code (any or all the group members) to start your project.
- Project will be released end of this week.
- Remember to select your topic for the talk by **9 Nov**

And now....

READY....

STEADY....

GO...