Homework 05 – Design

Arthur J. Redfern
arthur.redfern@utdallas.edu

0 Outline

1 Reading
2 Theory
3 Practice

1 Reading

1. Design

    Motivation: understand xNN design
    https://github.com/arthurredfern/UT-Dallas-CS-6301-CNNs/blob/master/Lectures/xNNs_050_Design.pdf

Complete

2. Understanding LSTM networks

    Motivation: an alternative presentation of RNNs and variants
    https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Complete

3. Attention and augmented recurrent neural networks

    Motivation: an alternative presentation of attention
    https://distill.pub/2016/augmented-rnns/

Complete

4. The illustrated transformer

    Motivation: an alternative presentation of self attention
    http://jalammar.github.io/illustrated-transformer/

Complete

5. The annotated transformer

    Motivation: a code walk through of self attention
    http://nlp.seas.harvard.edu/2018/04/03/attention.html

Complete

6. [**Optional**] ResNet / ResNeXt

    Motivation: residual connections are used throughout all types of network designs so it's worthwhile to read the initial sequence of papers that introduced these. A suggestion: set aside an hour of time and read all 3 of these together in 1 sitting.

Deep residual learning for image recognition
https://arxiv.org/abs/1512.03385
Identity mappings in deep residual networks
https://arxiv.org/abs/1603.05027
Aggregated residual transformations for deep neural networks https://arxiv.org/abs/1611.05431

Complete

7. [**Optional**] Neural architecture search: a survey

Motivation: neural architecture search is used throughout all types of network designs so it's worthwhile to read some more on this topic (see the lecture slides for many additional references).
https://arxiv.org/abs/1808.05377

Complete

2 Theory

8. Using pencil and paper, compute the receptive field size at the input to the global average pooling layer for ResNet 50.

Bottom Up

			Receptive field size
Start value			1
Level 5 standard	(+0, +2, +0)	repeat 2 times	5
Level 4 – 5 down sample	(+0, +2, *2 – 1, +0)		13
Level 4 standard	(+0, +2, +0)	repeat 5 times	23
Level 3 – 4 down sample	(+0, +2, *2 – 1, +0)		49
Level 3 standard	(+0, +2, +0)	repeat 3 times	55
Level 2 – 3 down sample	(+0, +2, *2 – 1, +0)		113
Level 2 standard	(+0, +2, +0)	repeat 3 times	119
Level 1 – 2 down sample	(*2 – 1, +2)		239
Level 0 – 1 down sample	(*2 – 1, +6)		483

Some comments

- If the input image is smaller than 483 pixels on each side then effectively this is saying that the receptive field covers the full image
- The energy aggregation is not uniform over the full receptive field; the shape looks ~ Gaussian like as a consequence of the central limit theorem and convolving filters, so more energy is concentrated in a narrower lobe within the 483 pixel window, maybe 483 / 3 ~ 161 pixels
- When working on vision problems that involve localization and don't include a global average pooling operation ask yourself: how big is the receptive field or receptive field / 3 relative to objects of interest in the image? If I was looking at the image through a window of this size, could I easily identify the underlying object?
- To compute the receptive field for other places in the network, start with a value

of 1 and work backwards using a subset of the same + and * operations as above

3 Practice

9. Understand all lines of code in the following example (https://github.com/arthurredfern/UT-Dallas-CS-6301-CNNs/blob/master/Code/xNNs_Code_031_CIFAR_ResNetV2b.py) and run it in Google Colab. Note that it skips levels 0, 1 (tail levels) and 2 (initial body level) in a standard ResNetV2 implementation and includes original levels 3, 4 and 5, a result of the input being 3 x 32 x 32 instead of 3 x 256 x 256. An alternative would have been to also include levels 0, 1 and 2 but skip the down sampling operations in these levels.
Complete

10. Similar to the above ResNetV2 example for CIFAR-10, use pencil and paper to plan out your own version of a popular network for CIFAR-10 by doing the following:

- Choose 1 of the following networks: MobileNet V2, MobileNet V3 (small or large), EfficientNet-B0, RegNetX-(200MF, 400MF, 600MF or 800MF) or RegNetY-(200MF, 400MF, 600MF or 800MF).
  Choice = MobileNet V2

- Draw out the network structure and each of the basic building blocks.
  - These networks were originally designed for ~ 3 x 256 x 256 images in ImageNet, so you'll likely replace portions of the network until ~ after the original 3rd level of down sampling with a simple tail.
  - Skipping these initial levels will also make the network less "wide" than the original ImageNet optimized version.
  - The final feature map before global average pooling should be ~ N x 8 x 8 where N is > the number of classes (maybe >>).
  - Pay careful attention to any places where multiple paths add together and make sure that the ranges of both paths is compatible.
  - Take inspiration from the ResNet example.

3×224×224

32×112×112

16×112×112

24×56×56

32×28×28

64×14×14

96×14×14

160×7×7

320×7×7

1280×7×7 avg. pool

conv2d 1×1

1280 x 1 x 1

- Compute the receptive field size at the feature map before the global average pooling layer. How does this compare to the original image size?

Receptive field size = <network choice dependent> = 1280

- Compute the feature map size and feature map memory required for each of the linear transforms. What is the maximum feature map size (this will set the optimal on device memory size)?

Max feature map size = < towards the beginning> = 3x224x224

- Compute the filter coefficient size and filter coefficient memory required for a complete block in each of the levels. Which level has the largest filter memory in a block? Which level has the smallest filter memory in a block?

Max filter memory size = <towards the end> = 320x7x7

- Compute the MACs required for a complete block in each of the levels. How do the number of required MACs change through the network?

585M

- From the perspective of increasing receptive field size, minimizing filter memory and minimizing MACs, which level is best to repeat blocks within?

Level = <network choice dependent; typically the level with 14x14 feature maps for a standard classifier applied to ImageNet with 3 x 224 x 224 inputs>

64x14x14

11. In software, implement and train the pencil and paper designed network from above. Ideally, create the network using a generator such that blocks can be repeated different numbers of times to build larger or smaller versions of the network. Note that you may need to modify the training hyper parameters. What is the accuracy that you achieve?

https://colab.research.google.com/github/pytorch/pytorch.github.io/blob/master/assets/hub/pytorch_vision_mobilenet_v2.ipynb#scrollTo=azevHfm0O1qw

12. [**Optional**] The following is a laundry list of additional items to consider trying

- Modify the network to add squeeze and excite style feature map re weighting (note that some of the above network choices already do this)
- Repeat blocks at different levels different numbers of times and record the accuracy of the trained network; create an optimal frontier of accuracy vs MACs and filter memory