RICE UNIVERSITY, DEPARTMENT OF COMPUTER SCIENCE

Assignment 5, COMP 540

Chen Zeng(cz39), Zhihui Xie(zx18)

March 20, 2018

1 Deep neural networks

Answer1 In deep learning, part of the architecture of a deep network or part of the training process is typically devoted to unsupervised feature extraction. It can contain a great deal of information. Such data is clean enough and rich enough so that the nonlinear features learned by deep neural networks represent real phenomena within the data and are not just artifacts of over training. Shallow networks like single-layer network cannot own this feature due to the limitation of it's layer and parameters. Therefore, deep networks typically outperform shallow networks.
ref: https://www.quora.com/When-should-I-prefer-deep-learning-algorithms-over-shallow-machine-learning-algorithms

Answer2 Instead of the function being zero when x < 0, a leaky ReLU will instead have a small negative slope (of 0.01, or so). That is, the function computes $f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } otherwise \end{cases}$ where α is a small constant.
Leaky ReLUs are one attempt to fix the "dying ReLU" problem. ReLU units can be fragile during training and can "die". For example, a large gradient flowing through a ReLU neuron could cause the weights to update in such a way that the neuron will never activate on any datapoint again. If this happens, then the gradient flowing through the unit will forever be zero from that point on. That is, the ReLU units can irreversibly die during training since they can get knocked off the data manifold.
ref1: http://cs231n.github.io/neural-networks-1/
ref2: https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

Answer3 Here are the contrast:

- AlexNet
  In 2012, AlexNet significantly outperformed all the prior competitors and won the challenge by reducing the top-5 error to 15.3%. The second place top-5 error rate, which was not a CNN variation, was around 26.2%.
  The network had a very similar architecture as LeNet by Yann LeCun et al but was deeper, with more filters per layer, and with stacked convolutional layers. AlexNet was trained simultaneously on two Nvidia Geforce GTX 580 GPUs which is the reason for why their network is split into two pipelines. AlexNet was designed by the SuperVision group, consisting of Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever.

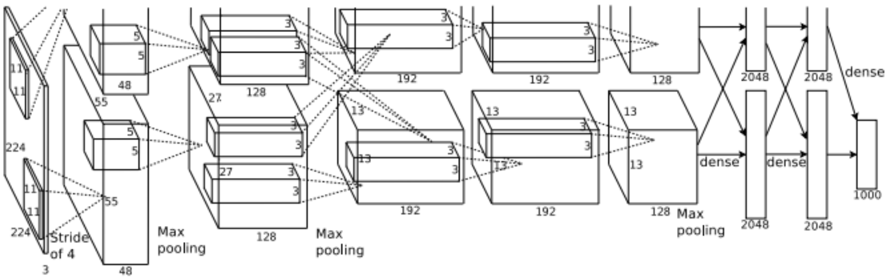

Figure 1.1: AlexNet

- VGG-Net
  The runner-up at the ILSVRC 2014 competition is dubbed VGGNet by the community and was developed by Simonyan and Zisserman . VGGNet consists of 16 convolutional layers and is very appealing because of its very uniform architecture. It only performs 3x33 times 33x3 convolutions and 2x22 times 22x2 pooling all the way through. It is currently the most preferred choice in the community for extracting features from images. The weight configuration of the VGGNet is publicly available and has been used in many other applications and challenges as a baseline feature extractor. However, VGGNet consists of 140 million parameters, which can be a bit challenging to handle.

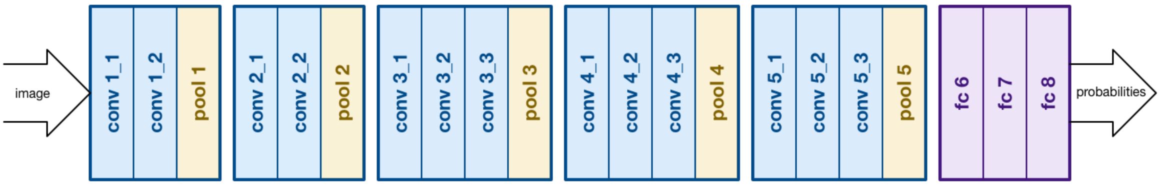

Figure 1.2: VGG-Net

- GoogleNet
  The winner of the ILSVRC 2014 competition was GoogleNet(Inception) from Google. It

achieved a top-5 error rate of 6.67%! This was very close to human level performance which the organisers of the challenge were now forced to evaluate. As it turns out, this was actually rather hard to do and required some human training in order to beat GoogLeNets accuracy. After a few days of training, the human expert (Andrej Karpathy) was able to achieve a top-5 error rate of 5.1%. The network used a CNN inspired by LeNet but implemented a novel element which is dubbed an inception module. This module is based on several very small convolutions in order to drastically reduce the number of parameters. Their architecture consisted of a 22 layer deep CNN but reduced the number of parameters from 60 million (AlexNet) to 4 million.

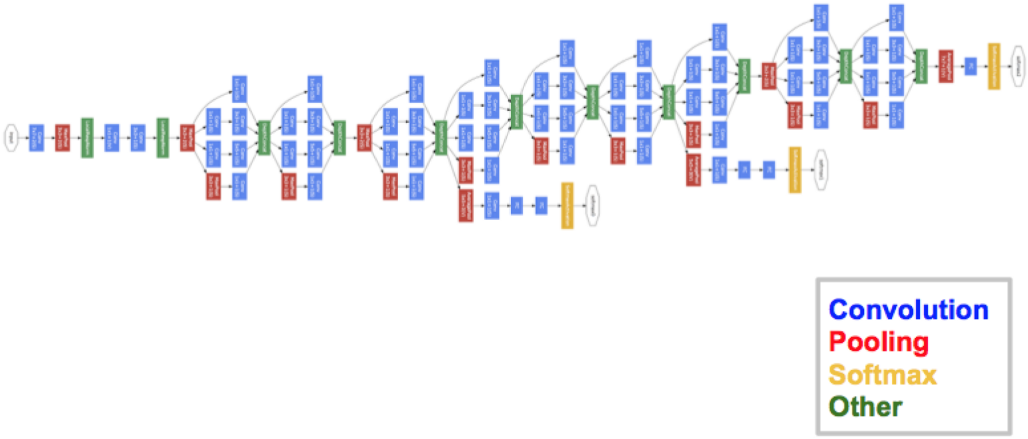

Figure 1.3: GoogleNet

- ResNet
  At last, at the ILSVRC 2015, the so-called Residual Neural Network (ResNet) by Kaiming He et al introduced anovel architecture with "skip connections" and features heavy batch normalization. Such skip connections are also known as gated units or gated recurrent units and have a strong similarity to recent successful elements applied in RNNs. Thanks to this technique they were able to train a NN with 152 layers while still having lower complexity than VGGNet. It achieves a top-5 error rate of 3.57% which beats human-level performance on this dataset.

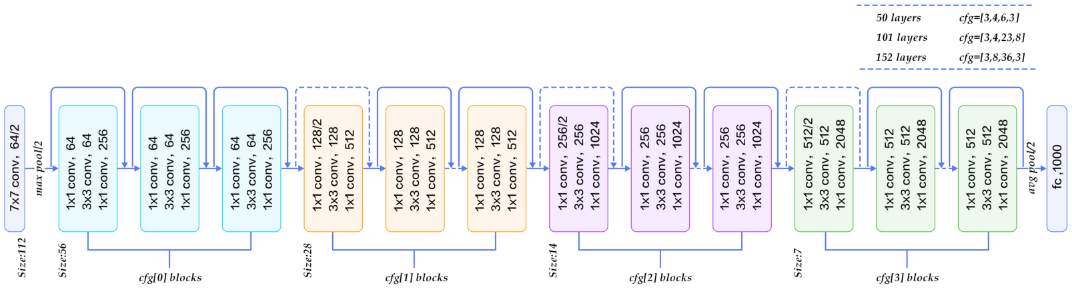
50 layers cfg=[3,4,6,3]
101 layers cfg=[3,4,23,8]
152 layers cfg=[3,8,36,3]
7x7 conv, 64/2
max pool/2
1x1 conv, 64 3x3 conv, 64 1x1 conv, 256
1x1 conv, 128/2 3x3 conv, 128 1x1 conv, 512
1x1 conv, 128 3x3 conv, 128 1x1 conv, 512
1x1 conv, 256/2 3x3 conv, 256 1x1 conv, 1024
1x1 conv, 256 3x3 conv, 256 1x1 conv, 1024
1x1 conv, 512/2 3x3 conv, 512 1x1 conv, 2048
1x1 conv, 512 3x3 conv, 512 1x1 conv, 2048
avg pool/2
fc, 1000
Size:112
Size:56
cfg[0] blocks
Size:28
cfg[1] blocks
Size:14
cfg[2] blocks
Size:7
cfg[3] blocks

Figure 1.4: ResNet

Here are the one defining characteristic of each network:

- AlexNet: the architecture is very similar to LeNet by Yann LeCun et al but was deeper
- VGG-Net: large amount of parameters which is about 140 million
- GoogleNet: reduced the number of parameters from 60 million (AlexNet)
- ResNet: "skip connections" which is known as gated units or gated recurrent units

ref: https://medium.com/@siddharthdas_32104/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

2 Decision trees, entropy and information gain

Answer1 According to the definition of entropy, we have:

$$H(S) = -Slog(S) - (1-S)\,log(1-S)$$

Take the derivation of it and then we can get:

$$H'(S) = \big(-1 - log(S) + log(1-S) + 1\big)\big(log(e)\big) = log\left(\frac{1-S}{S}\right) log(e)$$

At the same time, we know that $S = \frac{p}{p+n}$, which means it's range is $(0,1)$. Therefore, we have the range of $H'(S)$ with the relationship of q as below:

$$H'(S) = \begin{cases} >0 & \text{if } 0 < S < \frac{1}{2} \\ =0 & \text{if } S = \frac{1}{2} \\ <0 & \text{if } \frac{1}{2} < S < 1 \end{cases}$$

It proves that $H(S)$ increase as S increases from 0 to $\frac{1}{2}$ and will decrease as S increases from $\frac{1}{2}$ to 1, which means $H(S)$ will get the maximum as $S = \frac{1}{2}$. Therefore we can get:

$$H(S) \le H\left(\frac{1}{2}\right) = -\frac{1}{2} log\left(\frac{1}{2}\right) - \left(1-\frac{1}{2}\right) log\left(1-\frac{1}{2}\right) = 1$$

Thus $H(S) \leq 1$ is proved.
At the same time, according to the above discussion, $H = 1$ happens when and only when $S = \frac{1}{2}$, which means $\frac{p}{p+n} = \frac{1}{2}$. As we solve it, we find $p = n$.
Thus $p = n$ when $H(S) = 1$ is proved.

Answer2 Here are the calculation of the reduction in cost using misclassification rate, entropy, and Gini index for models A and B:

- Using misclassification rate:

$$cost(D) = \frac{1}{|d|} \sum_{(x,y)\in D} I\left(y \neq y'\right)$$

$$cost(D) = \frac{1}{800} * 400 = \frac{1}{2}$$

  As for model A:

$$cost\left(D_{left}\right) = \frac{1}{400} * 100 = \frac{1}{4}$$

$$cost\left(D_{right}\right) = \frac{1}{400} * 100 = \frac{1}{4}$$

$$reduction_A = \frac{1}{2} - \frac{1}{2} * \frac{1}{4} - \frac{1}{2} * \frac{1}{4} = \frac{1}{4}$$

  As for model B:

$$cost\left(D_{left}\right) = \frac{1}{800} * 400 = \frac{1}{2}$$

$$cost\left(D_{right}\right) = \frac{1}{600} * 200 = \frac{1}{3}$$

$$reduction_B = \frac{1}{2} - \frac{3}{4} * \frac{1}{3} = \frac{1}{4}$$

- Using entropy:

$$cost(D) = -\frac{400}{400+400} log\left(\frac{400}{400+400}\right) - \frac{400}{400+400} log\left(\frac{400}{400+400}\right) = 1$$

  As for the model A:

$$cost\left(D_{left}\right) = -\frac{300}{300+100} log\left(\frac{300}{300+100}\right) - \frac{100}{300+100} log\left(\frac{100}{300+100}\right) = 0.8113$$

$$cost\left(D_{right}\right) = -\frac{100}{300+100} log\left(\frac{100}{300+100}\right) - \frac{300}{300+100} log\left(\frac{300}{300+100}\right) = 0.8113$$

$$\begin{aligned} reduction_A &= cost(D) - \frac{\left|D_{left}\right|}{|D|} cost\left(D_{left}\right) - \frac{\left|D_{right}\right|}{|D|} cost\left(D_{right}\right) \\ &= 1 - \frac{400}{800} * 0.8113 - \frac{400}{800} * 0.8113 \\ &= 0.1887 \end{aligned}$$

As for the model B:

$$cost\left(D_{left}\right) = -\frac{200}{200+400}log\left(\frac{200}{200+400}\right) - \frac{400}{200+400}log\left(\frac{400}{200+400}\right) = 0.9183$$

$$cost\left(D_{right}\right) = -\frac{200}{200+0}log\left(\frac{200}{200+0}\right) - \frac{0}{200+0}log\left(\frac{0}{200+0}\right) = 0$$

$$\begin{aligned} reduction_B &= cost\left(D\right) - \frac{\left|D_{left}\right|}{|D|}cost\left(D_{left}\right) - \frac{\left|D_{right}\right|}{|D|}cost\left(D_{right}\right) \\ &= 1 - \frac{600}{800} * 0.9183 - \frac{200}{800} * 0 \\ &= 0.311275 \end{aligned}$$

- Using Gini index:

$$cost\left(D\right) = 2 * \frac{400}{400+400} * \frac{400}{400+400} = \frac{1}{2}$$

As for the model A:

$$cost\left(D_{left}\right) = 2 * \frac{300}{300+100} * \frac{100}{300+100} = \frac{3}{8}$$

$$cost\left(D_{left}\right) = 2 * \frac{100}{300+100} * \frac{300}{300+100} = \frac{3}{8}$$

$$\begin{aligned} reduction_A &= cost\left(D\right) - \frac{\left|D_{left}\right|}{|D|}cost\left(D_{left}\right) - \frac{\left|D_{right}\right|}{|D|}cost\left(D_{right}\right) \\ &= \frac{1}{2} - \frac{400}{800} * \frac{3}{8} - \frac{400}{800} * \frac{3}{8} \\ &= \frac{1}{8} \end{aligned}$$

As for the model B:

$$cost\left(D_{left}\right) = 2 * \frac{200}{200+400} * \frac{400}{200+400} = \frac{4}{9}$$

$$cost\left(D_{right}\right) = 2 * \frac{200}{200+0} * \frac{0}{200+0} = 0$$

$$\begin{aligned} reduction_B &= cost\left(D\right) - \frac{\left|D_{left}\right|}{|D|}cost\left(D_{left}\right) - \frac{\left|D_{right}\right|}{|D|}cost\left(D_{right}\right) \\ &= \frac{1}{2} - \frac{600}{800} * \frac{4}{9} - \frac{200}{800} * 0 \\ &= \frac{1}{6} \end{aligned}$$

Because the reduction of cost of model B is bigger than the model A, model B is the preferred split.

Answer3 No, the misclassification rate won't ever increase when splitting on a feature. Here is the provement:
Suppose we have M positive labels and N negative labels. And we can suppose $M < N$, then we have:

$$mis\_rate = \frac{M}{M+N}$$

Let's suppose that after the splitting, the left node has t_1 positive labels and t_2 negative labels. Therefore, the right node has $M - t_1$ positive labels and $N - t_2$ negative labels. Then, let's discuss it into the following situations:

- if $t_1 > t_2$, then due to $M < N$ we have $M - t_1 < N - t_2$

  $$mis\_rate_1 = \frac{t_2}{t_1 + t_2}$$

  $$mis\_rate_2 = \frac{M - t_1}{M + N - t_1 - t_2}$$

  $$\begin{aligned} mis\_rate\_split &= \frac{|D_1|}{|D|} mis\_rate_1 + \frac{|D_2|}{|D|} mis\_rate_2 \\ &= \frac{t_1 + t_2}{M+N}\frac{t_2}{t_1 + t_2} + \frac{M+N-t_1-t_2}{M+N}\frac{M-t_1}{M+N-t_1-t_2} \\ &= \frac{M - t_1 + t_2}{M+N} \\ &< \frac{M}{M+N} \\ &= mis\_rate \end{aligned}$$

  Therefore, we get:

  $$mis\_rate\_split < mis\_rate$$

- if $t_1 < t_2$ and $M - t_1 < N - t_2$

  $$mis\_rate_1 = \frac{t_1}{t_1 + t_2}$$

  $$mis\_rate_2 = \frac{M - t_1}{M + N - t_1 - t_2}$$

  $$\begin{aligned} mis\_rate\_split &= \frac{|D_1|}{|D|} mis\_rate_1 + \frac{|D_2|}{|D|} mis\_rate_2 \\ &= \frac{t_1 + t_2}{M+N}\frac{t_1}{t_1 + t_2} + \frac{M+N-t_1-t_2}{M+N}\frac{M-t_1}{M+N-t_1-t_2} \\ &= \frac{M}{M+N} \\ &= mis\_rate \end{aligned}$$

  Therefore, we get:

  $$mis\_rate\_split = mis\_rate$$

- if $t_1 < t_2$ and $M - t_1 > N - t_2$, which is equal to $M > t_1 + N - t_2$

$$mis\_rate_1 = \frac{t_1}{t_1 + t_2}$$

$$mis\_rate_2 = \frac{N - t_2}{M + N - t_1 - t_2}$$

$$\begin{aligned} mis\_rate\_split &= \frac{|D_1|}{|D|} mis\_rate_1 + \frac{|D_2|}{|D|} mis\_rate_2 \\ &= \frac{t_1 + t_2}{M + N} \frac{t_1}{t_1 + t_2} + \frac{M + N - t_1 - t_2}{M + N} \frac{N - t_2}{M + N - t_1 - t_2} \\ &= \frac{N + t_1 - t_2}{M + N} \\ &< \frac{M}{M + N} \\ &= mis\_rate \end{aligned}$$

Therefore, we get:

$$mis\_rate\_split < mis\_rate$$

To conclude, we get $mis\_rate\_split \leq mis\_rate$. Therefore, the misclassification rate won't ever increase when splitting on a feature.

3 Bagging

Answer1 According to the definition of $\epsilon_{bag}(x)$, we have:

$$\epsilon_{bag}(x) = \left(\frac{1}{L} \sum_{l=1}^{L} \left(f(x) + \epsilon_l(x)\right)\right) - f(x) = \frac{1}{L} \sum_{l=1}^{L} \epsilon_l(x)$$

According to the description of problem, when $m \neq l$, we have:

$$E_X[\epsilon_m(x)\epsilon_l(x)] = 0$$

We can have the following transformation:

$$\begin{aligned}
E_{bag} &= E_X\left[\epsilon_{bag}(x)^2\right] \\
&= E_X\left[\left(\frac{1}{L}\sum_{l=1}^{L}\epsilon_l(x)\right)^2\right] \\
&= \frac{1}{L^2}E_X\left[\sum_{l=1}^{L}\epsilon_l(x)^2 + 2\sum_{m\neq l}^{L}\epsilon_m(x)\epsilon_l(x)\right] \\
&= \frac{1}{L^2}\sum_{l=1}^{L}E_X\left[\epsilon_l(x)^2\right] + \frac{2}{L^2}\sum_{m\neq l}^{L}E_X\left[\epsilon_m(x)\epsilon_l(x)\right] \\
&= \frac{1}{L^2}\sum_{l=1}^{L}E_X\left[\epsilon_l(x)^2\right] \\
&= \frac{1}{L}\left(\frac{1}{L}\sum_{l=1}^{L}E_X\left[\epsilon_l(x)^2\right]\right) \\
&= \frac{1}{L}E_{av}
\end{aligned}$$

Therefore, the conclusion is proved.

Answer2 According to the last problem, we've got:

$$\epsilon_{bag}(x) = \frac{1}{L}\sum_{l=1}^{L}\epsilon_l(x)$$

According to Jensen's equality states that for any convex function, we have:

$$\left[\sum_{l=1}^{L}\frac{1}{L}\epsilon_l(x)\right]^2 \leq \sum_{l=1}^{L}\frac{1}{L}\epsilon_l(x)^2$$

At the same time, we have:

$$E_{bag} = E_X\left[\epsilon_{bag}(x)^2\right] = E_X\left[\frac{1}{L}\sum_{l=1}^{L}\epsilon_l(x)\right]^2$$

$$E_{av} = \frac{1}{L}\sum_{l=1}^{L}E_X\left[\epsilon_l(x)^2\right] = E_X\left[\sum_{l=1}^{L}\frac{1}{L}\epsilon_l(x)^2\right]$$

Therefore, we can get:

$$E_{bag} \leq E_{av}$$

The conclusion is proved.

4 Fully connected neural networks and convolutional neural networks