Feature Engineering

引言

在某个具体的领域，针对所要求解的问题，往往是数据驱动的。要求我们设计出能够良好适应于该问题的特征，即从原始数据的特征空间，转换到对该问题具有良好解的空间。越有效的转换越是我们追求的目标，一定程度上反映出我们对这一问题本质上的认识。

让我们把视角转到深度学习特征设计上来。目前有关这方面的理论依然是欠缺的，我们更多地是采用端到端的方式来提取特征，中间特征提取的过程对我们来说依旧是黑盒。学界已然有很多有意思的工作在这一问题上进行探究了。举例来说[6][7]，如果我们将训练数据替换为随机噪声数据，或者将标签打乱随机排列，让深度神经网络进行学习，在测试时它会表现出哪些性质呢？测试发现深度神经网络具有记忆功能，能够学习到与数据相关的模式，并且优先学习到简单的模式，同时对传统机器学习中的关于泛化的理论(例如 VC Dimensions)提出了挑战。

既然缺乏理论支持来引导我们，而深度学习的方式提取出的特征又如此有效，那么我们就有必要借鉴学习他人优秀的工作，这些工作中包含了他人对这些问题的洞见和感悟，提供给我们以启发和思考。通过复现别人的工作，除了可以验证论文中提出的方法，也可以深入了解到实现的细节，除了一些技巧方面的掌握之外，更重要的是 “The devils is in the details”。这也是一个双向验证的过程，一方面验证自己的理解是否正确，另一方面则是验证别人的工作是否存在纰漏。

我们将从两个方向来借鉴学习特征设计的过程。 一个方向是网络搭建的过程即是参数确定的过程，网络搭建完成后不再进行参数的更新；另一个方向则是确定好网络结构之后，中间的参数都是需要学习确定的，即标准的神经网络方式。在第一个方向参数确定的过程中，我们也参考了两种方式，第一种是从数据出发采用 PCA 的方式来确定参数，另一种则是利用先验知识确定参数，即 LBP。第一个方向参考了第二个方向中许多结构和技巧，但是又引入了自己独特的特征提取方式，因此在这里总结两个方向的异同。

PCANet[4]

- **特征提取方式**

  假设我们拥有 N 幅训练图片 $\{I_i\}_{i=1}^{N}$ ，尺寸为 $m\times n$，给定提取的图像子区域大小为 $k_1\times k_2$ 。

  1. 在每一幅图像的每一个像素点周围，我们提取 $k_1\times k_2$ 大小的图像子区域，向量化提取出来的图像子区域，这样一幅 $m\times n$ 大小的图像，我们可以能够得到： $x_{i,1},x_{i,2},\ldots,x_{i,mn}\in R^{k_1k_2}$ (实际个数小于 mn)。对范围不足 k_1 或 k_2 的边界像素点不提取子区域。

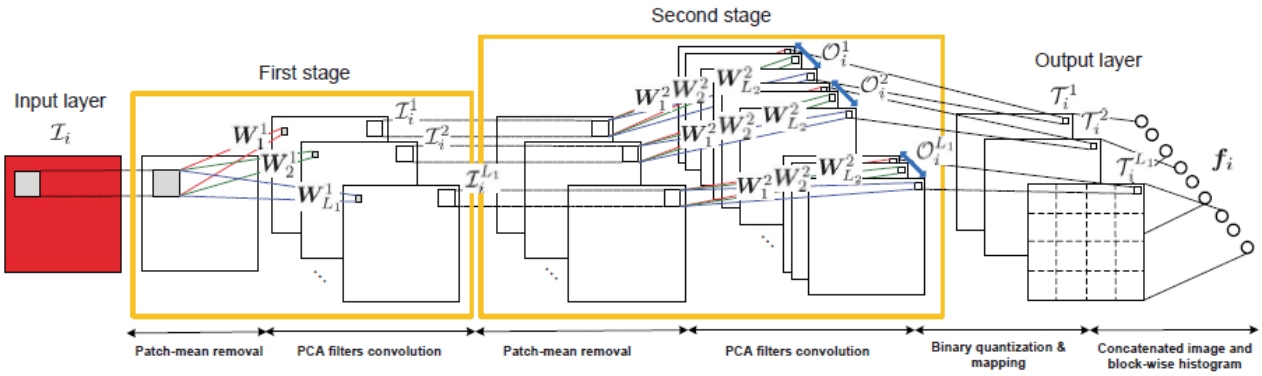
2. 然后我们计算每一个图像子区域上的像素均值，各个图像子区域减去对应均值，得到第 i 幅图像去均值化后的结果：$\bar{X}_i = [\bar{x}_{i,1}, \bar{x}_{i,2}, \ldots, \bar{x}_{i,mn}]$。这里的技巧不同于我们所常知的 PCA 均值化的操作，但是借鉴别人的工作[2]，由于自然图像具有“stationarity”的性质，采用这样的方式作为一种均值化的操作；同时针对自然图像也不采用方差归一化的方式。
3. 针对所有的 N 张训练图片我们采用同样的方式，最终得到：$X = [\bar{X}_1, \bar{X}_2, \ldots, \bar{X}_N \in R^{k_1 k_2 \times Nmn}]$。
4. 对上述获取到的矩阵 X 计算其协方差矩阵 XX^T，保留前 L_1 个特征向量(第一层)，展开为大小 $k_1 \times k_2$ 的矩阵，作为我们利用 PCA 方式获取到的卷积核，即：
$W_l^1 = mat_{k_1,k_2}(q_l(XX^T)) \in R^{k_1 \times k_2}, l = 1,2,3,\ldots,L_1$。
5. 利用第 4 步中提取到的 L_1 个卷积核，对所有的 N 张训练图片中的每个像素进行卷积操作(边界像素点进行零填充，进行卷积的子区域进行均值化处理)，卷积后的图像
$I_i^l = I_i * W_l^1, i = 1,2,3,\ldots,N$ 与原始图像具有相同尺寸。
6. 针对不同卷积核卷积后的每幅图像，我们采用以上相同的方式提取图像子区域和均值化操作，得到
$\bar{Y}_i^l = [\bar{y}_{i,l,1}, \bar{y}_{i,l,2}, \ldots, \bar{y}_{i,l,mn}]$，代表第 i 幅图像利用第 l 个卷积核，在所有 mn 个像素点上的提取结果。
7. 将第 l 个卷积核在原始 N 张每张图像上进行卷积、提取子区域、均值化的结果表示为：$Y^l = [\bar{Y}_1^l, \bar{Y}_2^l, \ldots, \bar{Y}_N^l] \in R^{k_1 k_2 \times Nmn}$。进一步将所有 L_1 个卷积核的结果合并为 $Y = [Y^1, Y^2, \ldots, Y^{L_1} \in R^{k_1 k_2 \times L_1 Nmn}]$。
8. 利用相同的 PCA 方式获得第二层 L_2 个卷积核：
$W_l^2 = mat_{k_1,k_2}(q_l YY^T) \in R^{k_1 \times k_2}, l = 1,2,3,\ldots,L_2$。然后对第一层得到的 $L_1 N$ 个卷积图像进行第二层卷积操作，特别地，在第一层用第 l 个卷积核对第 i 幅图像卷积后的图像 I_i^l 进行第二层 L_2 个通道的卷积操作，得到 $O_i^l = \{I_i^l * W_l^2\}_{l=1}^{L_2}$。此时，原始一幅输入图像经过两层卷积后得到 $L_1 L_2$ 个卷积图像。
9. 对于上一步第一层用第 i 幅、第 $l(l = 1,2,\ldots,L_1)$ 个卷积核得到的图像 I_i^l，进行第二层卷积图像时会得到 L_2 个卷积图像 $\{I_i^l * W_l^2\}_{l=1}^{L_2}$，我们利用 Heaviside step 函数对这 L_2 幅图像进行二值化操作 $\{H(I_i^l * W_l^2)\}_{l=1}^{L_2}$ (图像中的大于 0 的像素点置为 1，小于等于 0 的像素点置为 0)。然后我们将这 L_2 幅图像的每一幅图像上的对应像素点，作为一个长度为 L_2 的二进制字符串中

的对应位的一位，最后将这 L_2 幅图像转化为十进制下的一幅图像（如果我们设定 $L_2=8$，那么这幅图像的每个像素点的值域为 $0\sim255$；否则值域范围为 $0\sim2^{L_2}-1$），

$T_i^l=\sum_{l=1}^{L_2}2^{l-1}H(I_i^l*W_l^2)$。这 L_2 幅图像间的次序是没有影响的。

10. 原始第 i 幅图像，经过第一层卷积操作得到的 L_1 幅图像的每一幅，经过第二层卷积和二值化压缩后得到的图像 $T_i^l(l=1,2,\ldots,L_1)$，划分为 B 个子块（$B=\frac{(width-w)}{s}\times\frac{(height-w)}{s}$，其中 width, height 分别代表图像的宽和高，s 代表相邻子块采样步长，w 代表给定子块大小，除法向下取整）；在每一个子块上计算直方图（拥有 2^{L_2} 个 bins）；然后将这 B 个直方图按照列主导的方式拼接起来，形成第 i 图像，第 l 个卷积核下的特征 $Bhist(T_i^l)$；最后将第 i 幅图像下，所有 L_1 个卷积核对应的特征（列向量）按列拼接为一个列向量 $f_i=[Bhist(T_i^l),\ldots,Bhist(T_i^{L_1})]^T\in R^{2^{L_2}L_1B}$，作为第 i 幅图像最后的特征。注意，在直方图子块进行划分的时候，既可以是互相重叠的，也可以是不重叠的；另外，采用直方图的方式一定程度上使得特征对图像的平移和缩放具有一定的鲁棒性。

- **示意图**



- **分类实验**

1. Test on FERET
   - 实验设置

     卷积核 W 采用在 MultiPIE 数据库上学习得到。

     利用 Whitening PCA 的方式降维到 1000 维。

     距离计算选用 cosine 距离，分类器采用最近邻匹配。

     参数：

     Numstages: 2

     PatchSize: 5×5

     Numfilters: 8×8

HistBlocksize: 15×15
BlkOverLapRatio: 0
Pyramid: 0

- 实验结果
  训练集：fa
  测试集 Accuracy：dup-2: 94.02%

2. Test on MNIST_BASIC
   - 实验设置
     在训练集训练得到卷积核 W 和提取到特征。
     在训练集得到的特征上，采用 SVM Linear 分类器进行训练。
     在测试集上用卷积核 W 提取特征，并用 SVM Linear 测试。
     参数：
     Numstages: 2
     PatchSize: 7×7
     Numfilters: 8×8
     HistBlocksize: 7×7
     BlkOverLapRatio: 0.5
     Pyramid: 0
   - 实验结果
     测试集上错误率: 1.06%
3. Test on LFW
   - 实验设置
     采用经过人脸矫正并裁剪过的数据库。
     距离计算采用 cosine 距离。
     在 View1 上获得阈值之后，在 View2 上进行十折交叉验证，每一折在 View2 的训练集上获取卷积核和降维矩阵，利用 View1 测试集上获取的阈值，在 View2 测试集上进行测试。
     参数：
     Numstages: 2
     PatchSize: 7×7
     Numfilters: 8×8
     HistBlocksize: 15×13
     BlkOverLapRatio: 0
     Pyramid: 0
   - 实验结果
     Verification Accuracy: 83.23%

LBPNet[3]

- **特征提取方式**

  假设总共具有 N 幅图像(包括训练集和测试集)，每幅图像尺寸为 $w \times h$：

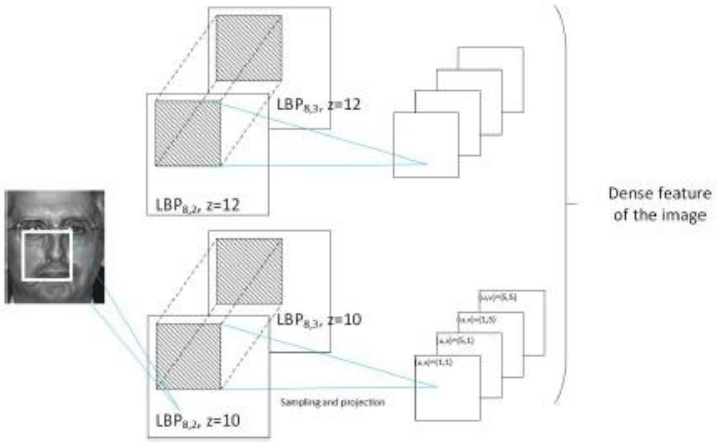
  1. 将 N 幅图像拼接为一个立方体： $ImageCube_{w \times h \times N}$。
  2. 将 N 幅图像的每一幅作一个镜像形成一个立方体：
     $ImageCube\_Flip_{w \times h \times N}$。

3. 将 $ImageCube$ 和 $ImageCube\_Flip$ 中的每一幅图像，利用圆形 LBP 算子(半径为 r，采样点个数为 s)，转化为一幅 LBP 图像(边界像素点无法进行 LBP 采样计算的地方忽略，所以 LBP 图像尺寸要小于原图像)。注意，计算得到 LBP 图像时，我们引入等价模式类的方式，利用圆形 LBP 算子计算得到 LBP 图像每个像素值之后，将该像素值归到第 $n(n=0,2,\ldots,58)$ 类等价类，即 LBP 图像上每个像素值范围为 $0\sim58$。
4. 利用不同半径 $r_1,r_2,\ldots,r_k$ 的圆形 LBP 算子(采样点个数相同)对 $ImageCube$ 和 $ImageCube\_Flip$ 中的每一幅图像计算得到对应 LBP 图像，获得不同半径下的 LBP 图像立方体。
5. 选取不同的直方图子区域大小 $w_1,w_2,\ldots,w_l$，在给定直方图子区域大小 $w_i(i=1,2,\ldots,l)$ 后，循环遍历不同半径 $r_1,r_2,\ldots,r_k$ 下，对 $ImageCube$ 和 $ImageCube\_Flip$ 中的每一幅 LBP 图像进行直方图子区域的划分并在该子区域上计算直方图(bins=59，等价模式类)，存储为四元组 glbph(原图像)、flbph(镜像图像)。四元组 glbph(index1,index2,index3,index4)的存储形式为：由 index1 和 index2 确定存储每一页的大小 $Page_{index1\times index2}$，然后存储 index3 行、index4 列的 Page。具体对应到存储直方图特征时：$index4=N, index3=\dfrac{Width}{w_i}$，其中 index4 代表共有 N 张图片，index3 代表 LBP 图像宽为 width 下，每一行拥有的直方图子区域个数；在每个 Page 里，$index1=k\times59, index2=\dfrac{Height}{w_i}$，其中 index1 表示在每个直方图子区域上计算等价模式类的直方图，然后将 k 个半径下的直方图拼接为一个列向量，index2 代表 LBP 图像高为 Height 下，每一列拥有的直方图子区域个数。注意，在每个直方图子区域上提取得到直方图之后，将该直方图进行开方处理。flbph 同理。这样一幅 LBP 图像经过直方图子区域计算后变为 $index2\times index3\times index1$ 大小的特征图。
6. 给定大小为 $pSize\times pSize$ 的采样块，采样间隔为 $pStride$，在 LBP 图像上进行采样(采样块间可重叠)。将该采样块里面包含的所有直方图按照列主导的方式拼接为一个列向量，作为该图像在该采样块下得到的特征。给定直方图子区域大小 w_i，对于每一个采样块，在 LBP 图像立方体上计算得到每个图像在该采样块下的特征(借助于 lbph 和 flbph 四元组，及其下标操作)，形成大小为 $nums\_of\_feature\times N$ 的特征矩阵，其中 nums_of_feature 代表在该图像在该采样块内拼接所有直方

图获得的特征维数，N 代表 LBP 图像立方体所有图像个数；然后将原始 LBP 图像立方体和对应的镜像图像立方体获得的特征矩阵，按照同样的方式进行划分，按列合并对应的部分作为训练集和测试集。

7. 在训练集上利用 Whitening PCA[2]的方式获取特征向量和均值，对测试集进行均值化和投影操作。获得降维后的训练集和测试集。
8. 对训练集中的每一幅原始图像，计算其原始图像和对应镜像图像，同测试集中每一幅原始图像和对应镜像图像的距离，取最小值，作为原始两幅图像间的距离。得到在该采样块下的距离矩阵。
9. 遍历所有采样块，形成三维距离立方体，第三维度代表每个采样块下形成的距离矩阵。

- **示意图**



- **分类实验**
  - Test on FERET
    - 实验设置

      将三维距离立方体按照第三维求平均值(即各个采样块间距离取平均)，距离计算采用 cosine 距离，分类器采用最近邻分类。

      参数：

      radius: 2,3,4
      samping points: 8
      wlist: 10
      pSize: 50
      pStride: 10
      reduced dimentions: 1800
    - 实验结果

      训练集: fa

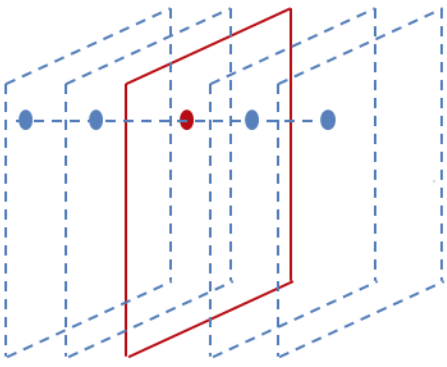
      测试集 Accuracy: fb(98.74%)、fc(94.33%)、dup-1(87.67%)、dup-2(87.18%)
  - Test on LFW

- 实验设置
  距离计算采用 cosine 距离。
  View2 上采用十折交叉验证的方式，6000 对图片对间的距离，作为 6000 个样本的二分类问题(是相同图片或不同图片)，每一个样本的每一个维度代表了在某个采样块上这对图片对间的距离。分类器采用 SVM Linear。
  监督学习：十折交叉验证时，利用每一折上训练集上的距离对训练得到 SVM Linear 分类器，在测试集距离对上进行分类测试。结果取平均。
  非监督学习：根据 LFW 非监督学习部分的协议，绘制 ROC 曲线，得到对应的 AUC 值。十折验证结果取平均。
  参数：
  radius: 1,2,3
  samping points: 8
  wlist: 10
  pSize: 50
  pStride: 10
  reduced dimentions: 800
- 实验结果
  监督学习 Accuracy：82.33%
  非监督学习 AUC：89.01%

AlexNet

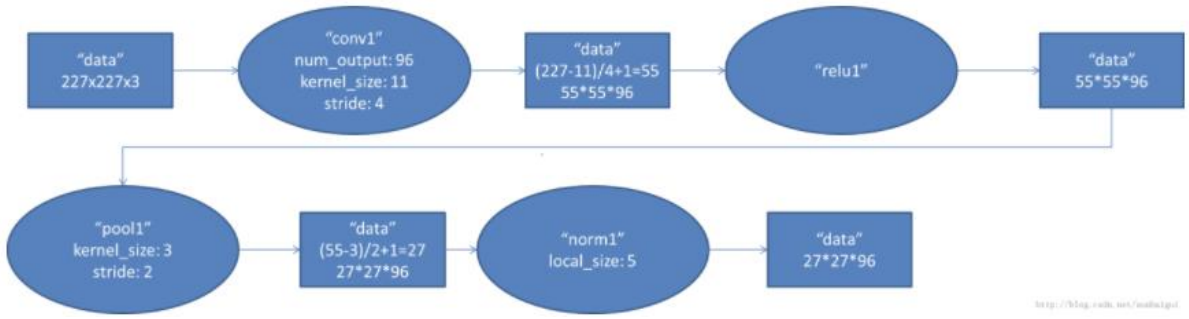
- ## 特征提取方式[5]
  - **LRN 层**
    对当前卷积图像中每个像素点，根据附近几张卷积图像的对应像素点的值进行加权计算做平滑处理。

$$b_{x,y}^{i} = a_{x,y}^{i} / (k + \alpha \sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)} (a_{x,y}^{j})^{2})^{\beta}$$

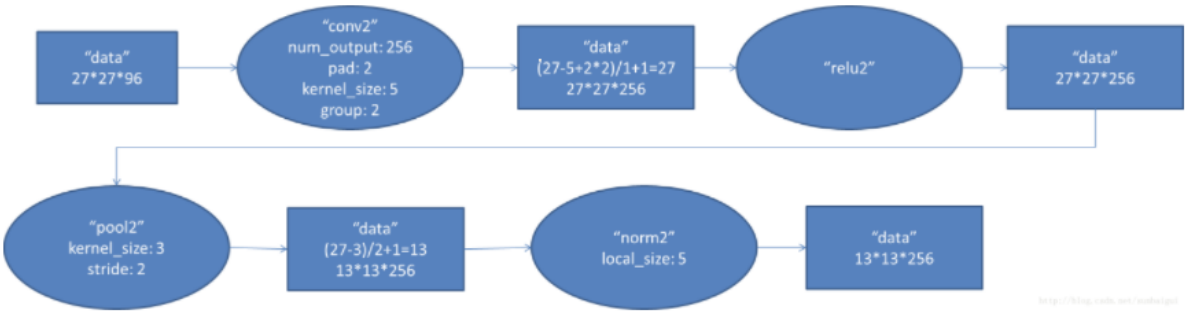


  - ## 网络结构

1. **Conv-Relu-Pooling-LRN**

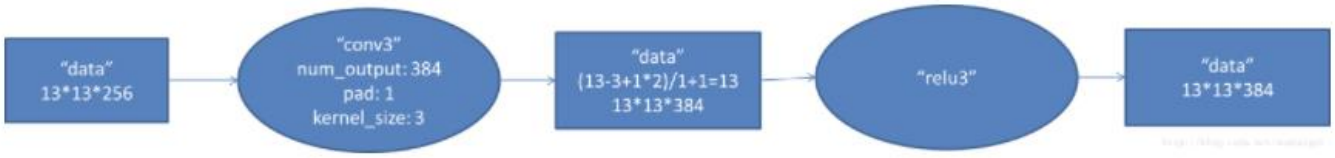


2. **Conv-Relu-Pooling-LRN**

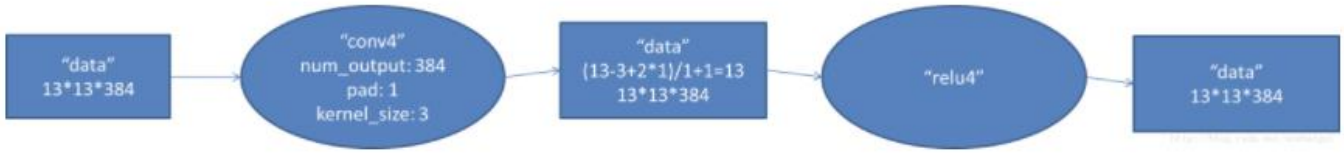


需要注意的是 group=2，这个属性强行把前面结果的 feature map 分开，卷积部分分成两部分做。

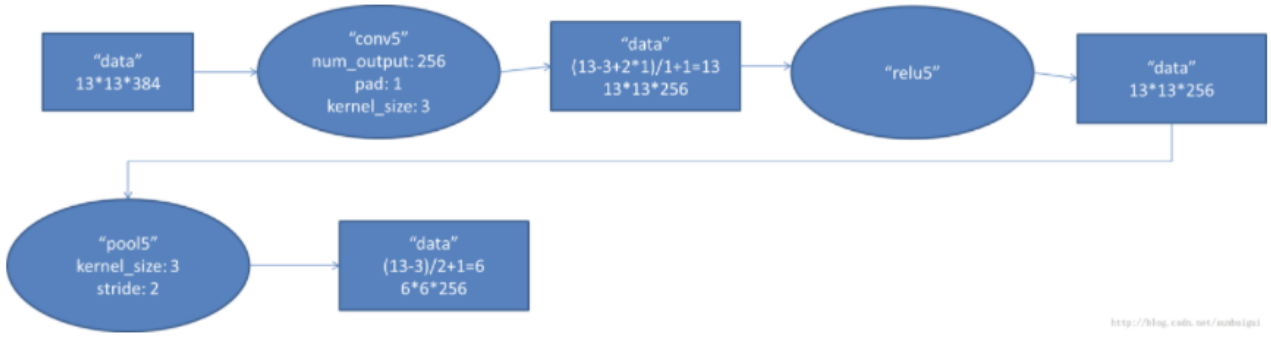
3. **Conv-Relu**



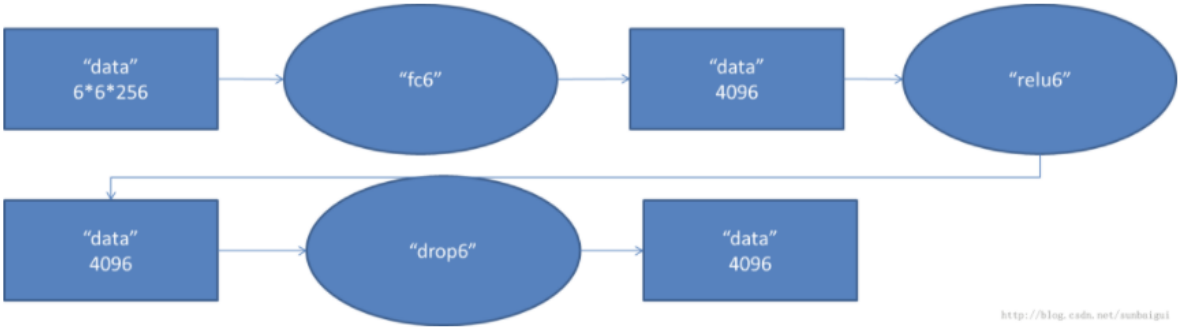
4. **Conv-Relu**



5. **Conv-Relu-Pooling**

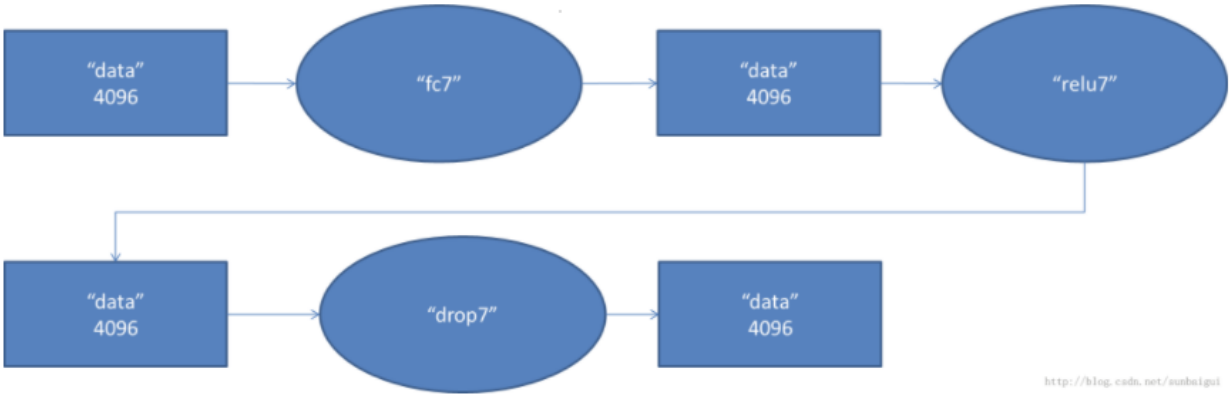


6. **FC-Relu-Dropout**



这里有一层特殊的 dropout 层，在 AlexNet 中是说在训练的以 1/2 概率使得隐藏层的某些 neuron 的输出为 0，这样就丢到了一半节点的输出，BP 的时候也不更新这些节点。

7. **FC-Relu-Dropout**

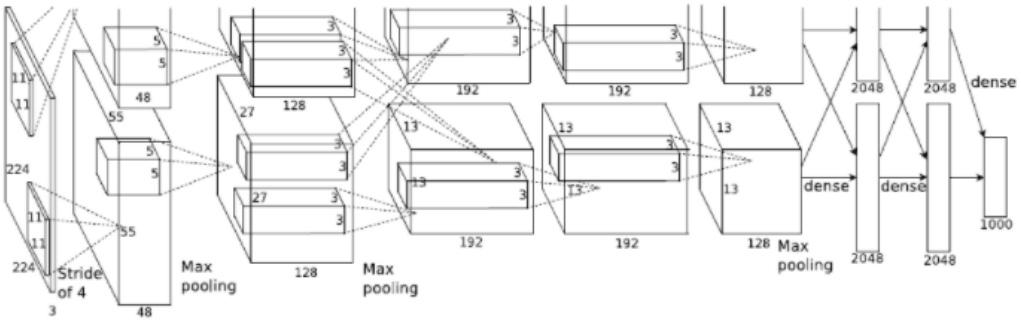


8. **FC-Softmax**

9. **Loss Function: Softmax loss**

$$\mathcal{L} = -\frac{1}{N}[\sum_{i=1}^{N}\sum_{j=1}^{K} 1\{\boldsymbol{y}^{(i)} = j\}\log p_j^{(i)}]$$

- 示意图



- 常用深度学习预处理技巧[1]
  1. **Data Augmentation**
  2. **Pre-Processing**

3. **Initializations**
4. **During Training**
5. **Activation Functions**
6. **Regularizations**
7. **Insights from Figures**
8. **Ensemble**

参考文献

[1] http://lamda.nju.edu.cn/weixs/project/CNNTricks/CNNTricks.html
[2] http://ufldl.stanford.edu/tutorial/unsupervised/PCAWhitening/
[3] Xi M, Chen L, Polajnar D, et al. Local binary pattern network: A deep learning approach for face recognition[C]. international conference on image processing, 2016: 3224-3228.
[4] Chan T, Jia K, Gao S, et al. PCANet: A Simple Deep Learning Baseline for Image Classification?[J]. IEEE Transactions on Image Processing, 2015, 24(12): 5017-5032.
[5] Krizhevsky A, Sutskever I, Hinton G E, et al. ImageNet Classification with Deep Convolutional Neural Networks[C]. neural information processing systems, 2012: 1097-1105.
[6] https://arxiv.org/pdf/1611.03530.pdf
[7] https://arxiv.org/pdf/1706.05394.pdf