机器学习笔记

纳文琪 [1]

2019 年 9 月 11 日

1 学习算法的性能度量 [1]

1.1 错误率和精度

错误率 指的是分类错误的样本占总样本数的比例，主要适用于二分类问题，也可用于多分类问题。

精度 指的是分类正确的样本数占总样本数的比例，同意适用于二分类和多分类问题。

错误率和精度简单、常用，但并不能满足所有需求。

1.2 查准率、查全率和 F1

混淆矩阵 对二分类问题，可以将真实类别与预测类别组合划分成 TP、TN、FP、FN 四种情形，分别表示预测正确的正例和反例、预测错误的正例和反例。分类结果可以使用一个“混淆矩阵”表示：

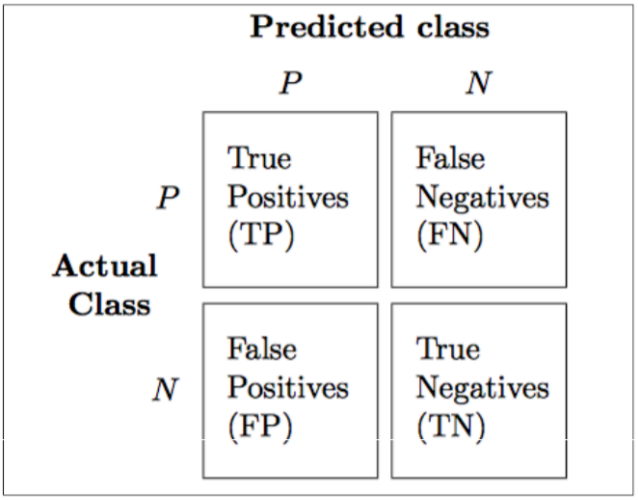

图 1: Confusion Matrix

查准率（准确率，precision） 是从预测结果（其数量作为分母）出发计算的精度，指的是预测为正例的样本中，有多少的预测正确的。其定义是:

$$P = \frac{TP}{TP + FP} \tag{1}$$

查全率（召回率，recall） 是从样本（其数量作为分母）出发计算的精度，指的是所有正例样本中，有多少被预测正确了。其定义是:

$$R = \frac{TP}{TP + FN} \tag{2}$$

查全率与查准率是一对矛盾的度量。

P-R **曲线** 用于直观地显示学习器在样本总体上的查全率、查准率。

F1 **和** F_β F1 是综合考虑查全率和查准率的度量，定义为:

$$F1 = \frac{2 \times P \times R}{P + R} \tag{3}$$

一些应用中，对查准率和查全率的重视程度不同，此时需要用 F_β :

$$F_\beta = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R} \tag{4}$$

F_β 是 F1 的一般形式，当 $\beta = 1$ 时就是 F1；当 $\beta > 1$ 时，查全率有更大影响；当 $\beta < 1$ 时，查全率有更大影响。

1.3 ROC 与 AUC

ROC 全称是“受试者工作特征”，与 P-R 曲线类似，它也有两个坐标，其纵坐标表示的是“真正例率”（TPR），即正例的查全率，横坐标表示的是“假正例率”，及反例被判断错误的比率。两者定义为：

$$TPR = \frac{TP}{TP + FN} \tag{5}$$

$$FPR = \frac{FP}{TN + FP} \tag{6}$$

ROC 曲线的对角线对应的是“随机猜想”模型，而点 (0,1) 则代表“理想模型”。

AUC 进行学习器比较时，与 P-R 曲线类似，若一个学习器的 ROC 曲线被另一个学习器的曲线完全包住，则可断言后者的性能优于前者；若两个学习器的 ROC 曲线有交叉，则需要比较 ROC 曲线下的面积，即 AUC 来进行判断。

2 线性模型

2.1 LMNN

kNN 成功的关键是对距离度量方法的选择，一般我们都会选欧氏距离作为距离度量方法，但这并不一定都会有效。理想情况下，我们应该根据具体的问题来选择距离度量方法，基于特定样本，学习一个特定的距离度量方法可以使得 kNN 的性能得到改善。

LMNN 就是通过样本学习一个线性变换 M，而样本之间的距离则通过 M 进行度量。如下图：

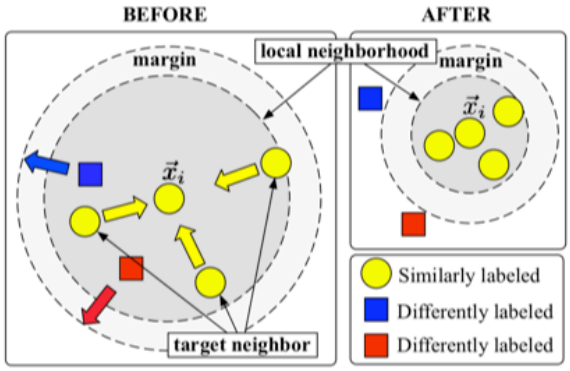

Figure 1: Schematic illustration of one input's neighborhood $\vec{x}_i$ before training (*left*) versus after training (*right*). The distance metric is optimized so that: (i) its $k=3$ target neighbors lie within a smaller radius after training; (ii) differently labeled inputs lie outside this smaller radius, with a margin of at least one unit distance. Arrows indicate the gradients on distances arising from the optimization of the cost function.

图 2: Confusion Matrix

之前是默认的欧式距离的度量，正样本与负样本之间的距离差不多，之后是通过学习得到的新的度量方法所获得的效果。

3 神经网络

隐层参数 $W_j \in \mathbb{R}^{d\times n}$ 表示 j-th 层的参数，此层有 n 个神经元，接收 d 个来自的 (j-1)-th 层的输入。W_j 的第 k 列，就是输入到第 k 个神经元的数据对应的参数。

隐层输出 $y_j \in \mathbb{R}^n$ 表示 j-th 层的输出，n 个神经元有 n 个输出，计算公式为：$y_j = W_j^T y_{j-1} + b_j$

4 Deep Neural Networks for Bot Detection[2]

Motivation　现有的系统都是在 account-level 进行 bots 的发现，需要根据特定帐号的一系列历史活动记录来确认帐号是否是 bot。这在进行检测的时候代价非常昂贵。论文希望通过仅仅一条 tweet 来判断是否是 bot。

论文将 bot 发现的方法分为用户级和 tweet 级，进行分类；tweet 级根据

数据集　论文使用的数据集存在不平衡问题，作者分别使用 SMOTE+ENN 和 SOMTE+TOMEK 的方法平衡数据。数据分为用户级和 tweet 级两类，用户级数据（用户元数据）包括 statuses count、followers count 等，tweet 级数据包括 retweet count、number of hashtags 等。

5 损失函数

5.1 softmax

下溢 (underflow) 当接近零的数被四舍五入为零后发生下溢。

上溢 (overflow) 当大数量级的数被近似为 ∞ 或 $-\infty$ 时，发生上溢。

softmax **函数** 可对下溢和上溢进行数值稳定，定义为:

$$softmax(\mathrm{x})_i = \frac{exp(x_i)}{\sum_{j=1}^{n} exp(x_j)} \tag{7}$$

这个式子同样会产生溢出，例如，当 x 是很小的负数时，分母会变成零；当 x 是很大的数时，一样会发送上溢。这个问题可以通过计算 $softmax((z), z = x - \max_i x_i$ 同时解决。

6 A Discriminative Feature Learning Approach for Deep Face Recognition [3]

6.1 Introduction

动机 一般的物体识别，主要是闭集识别，利用 softmax loss 即可处理，但人脸识别不单需要特征可分离 (separable)，也需要可区分 (discriminative)，构建一个高效的 loss 可提升可区别性，为此，本文提出 center loss。

Center loss 为每一个 class 的特征维护一个 center，每次进行 SGD 的时候同时更新 center，并最小化 class 中样本特征到中心的距离。

6.2 The Proposed Approach

toy example 本文使用一个 toy example 来演示算法的效果。它最后一个隐层的维度是 2，以便于我们使用二维图像来显示特征分布。toy example 使用 softmax loss，最后输出的特征的分布表示为图4 ：

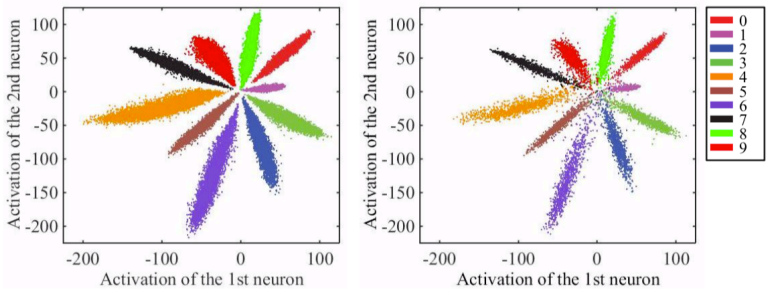

Fig. 2. The distribution of deeply learned features in (a) training set (b) testing set, both under the supervision of softmax loss, where we use 50K/10K train/test splits. The points with different colors denote features from different classes. **Best viewed in color.**

图 3:

Center loss 为增强可区分度，需在保持特征分类的同时，最小化内部 class 的方差。center loss 定义为:

$$\mathfrak{L}_C = \frac{1}{2}\sum_{i=1}^{m} \|\mathrm{x}_i - \mathrm{c}_{y_i}\|_2^2 \tag{8}$$

理论上，c_{y_j} 应当随着特征的变化而更新，因此，在每次更新更新参数的时候都要对它进行更新。

综上，总的损失函数可以表示为：

$$\mathfrak{L} = \mathfrak{L}_S + \lambda \mathfrak{L}_C \tag{9}$$

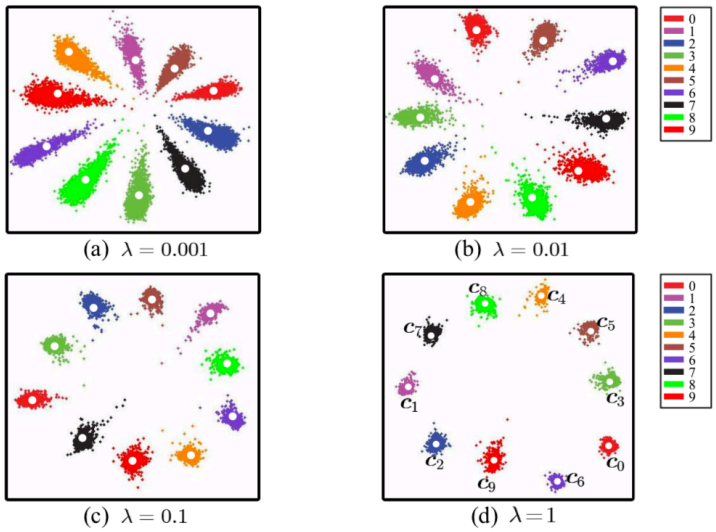

Fig. 3. The distribution of deeply learned features under the joint supervision of softmax loss and center loss. The points with different colors denote features from different classes. Different λ lead to different deep feature distributions ($\alpha = 0.5$). The white dots ($\boldsymbol{c}_0$, $\boldsymbol{c}_1$,...,$\boldsymbol{c}_9$) denote 10 class centers of deep features. **Best viewed in color.**

图 4:

7 Large-Margin Softmax Loss for Convolutional Neural Networks[4]

为通过可区分的信息来增强 CNN 性能，本文提出一种新的损失函数 L-Softmax。它通过用特征化的方法来将样本与参数划分开。如图所示，第一列是普通的 softmax，后面的是 l-softmax，使用 l-softmax 会使学习到的特征更加紧凑和区分良好（well separated）：

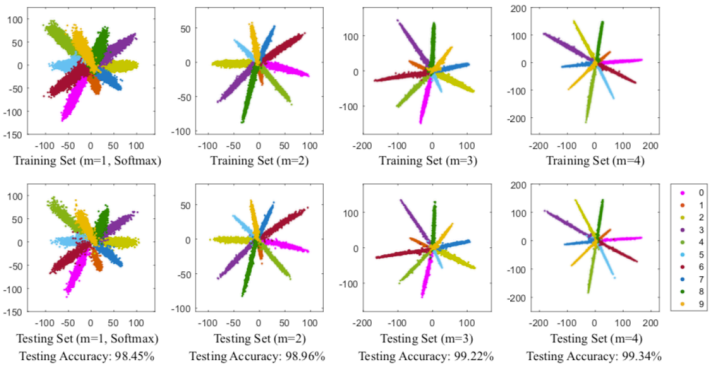

Figure 2. CNN-leanmed features visualization (Softmax Loss (m=1) vs. L-Softmax loss (m=2,3,4)) in MNIST dataset. Specifically, we set the feature (input of the L-Softmax loss) dimension as 2, and then plot them by class. We omit the constant term in the fully connected layer, since it just complicates our analysis and nearly does not affect the performance. Note that, the reason why the testing accuracy is not as good as in Fig.2 is that we only use 2D features to classify the digits here.

图 5:

7.1 基础

由于矩阵的积可以表示为：$W_j^T x_i = \|W_j\| \|x_i\| \cos(\theta_j)$，因此，softmax 可以表示为：

$$softmax(\mathrm{x})_i = \frac{\exp(\|W_{y_i}\| \|x_i\| \cos(\theta_{y_i}))}{\sum_{j=1}^{n} \exp(\|W_j\| \|x_i\| \cos(\theta_j))} \tag{10}$$

假设一个 2 分类问题，如果 x 输入分类 1，则 softmax 肯定会希望属于分类 1 的概率比分类 2 大，也就是 $W_1^T x > W_2^T x$，用上面的式子替换就是：$\|W_1\| \|x\| \cos(\theta_1) > \|W_2\| \|x\| \cos(\theta_2)$。然而，我们为了获得更加严格的分类边界，可以要求：$\|W_1\| \|x\| \cos(m\theta_1) > \|W_2\| \|x\| \cos(\theta_2) (0 \le \theta_1 \le \frac{\pi}{m})$，这里 m 是一个正整数。

7.2 定义

根据以上信息，我们可以定义 L-softmax 为：

$$L_i = -\log(\frac{\exp(\|W_{y_i}\| \|x_i\| \cos(\theta_{y_i}))}{\exp(\|W_{y_i}\| \|x_i\| \psi(\theta_{y_i})) + \sum_{j \neq y_i} \exp(\|W_j\| \|x_i\| \cos(\theta_j))}) \tag{11}$$

上式中，$\psi(\theta)$ 应该是一个单调递减函数，而 $\cos(m\theta)$ 函数仅在 $0 \leq \theta_1 \leq \frac{\pi}{m}$ 范围内是单调递减的。因此，这里需要引入一个新的单调递减函数 $D(\theta)$，并且 $D(\frac{\pi}{m}) = \cos(\frac{\pi}{m})$。这样的话，$\psi(\theta)$ 就可以定义为：

$$\psi(\theta) = \begin{cases} \cos(m\theta), 0 \leq \theta \leq \frac{\pi}{m} \\ D(\theta), \frac{\pi}{m} \leq \theta \leq \pi \end{cases} \tag{12}$$

7.3 几何解释

从以下图中可以直观看出决策边界的变化：

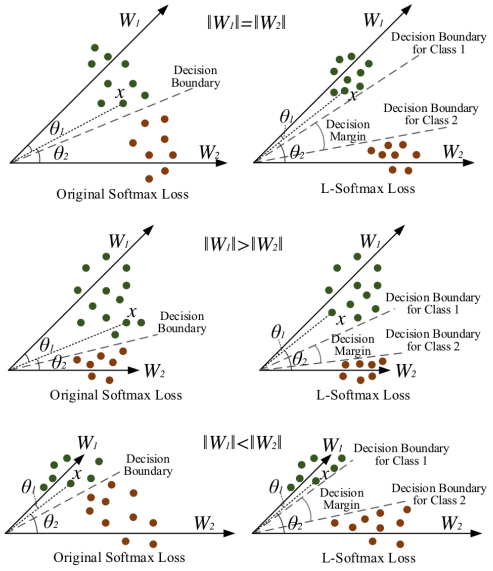

Figure 4. Examples of Geometric Interpretation.

图 6:

8 FaceNet: A Unified Embedding for Face Recognition and Clustering[5]

本文提出一种可同时用于人脸验证（是否是同一个人）、识别（这是谁）和聚类（找出相同的脸）的系统，其主要基于 CNN 为每个图像学习一个欧几里得嵌入，越相似的脸，他们之间的距离就越小。一旦学习到嵌入，验证问题就是比较两个嵌入的距离，识别问题就是一个 kNN 分类问题，聚类就是普通聚类问题。

FaceNet　基于 LMNN 方法，使用一个三元组（triplet）将图像训练输出为一个 128 维的嵌入。triplet 由两个匹配的人脸图片和一个不匹配的人脸图片组成，损失函数的目的是让一个样本与他的正样本尽可能靠近，与他的负样本尽可能分离。

模型结构　把 CNN 看成一个黑盒子，模型可以表示为：

Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

图 7:

该模型使用的损失函数是 triplet loss，最终目的是训练一个从图像 x 到特征空间 $\mathbb{R}^d$ 的嵌入 $f(x)$，使得相同的脸之间的距离较小，不同脸之间的距离较大。

Triplet loss　上述嵌入表示为 $f(x) \in \mathbb{R}^d$，且限制在一个单位超球体内，即要求 $\|f(x)\|_2 = 1$。通过这个损失函数，我们希望确保一个人的图像 x_i^a(anchor) 与其他同是这个人的图像 x_i^p(positive) 距离更加，与其他人的图像 x_i^n(negative) 距离更远。如图所示：

Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

图 8:

可以表示为：

$$\|x_i^a - x_i^p\|_2^2 + \alpha < \|x_i^a - x_i^n\|_2^2, \forall (x_i^a, x_i^p, x_i^n) \in \tau \tag{13}$$

其中，α 是一个边界值。总体的损失函数即可定义为最小化：

$$L = \sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ \tag{14}$$

学习的过程就是生成三元组，然后最小化 L。

Triplet **的选择**　如果生成全部可能的三元组用于学习，将会导致有很多三元组很容易满足公式 (13)，这些无用的三元组会导致收敛速度变慢，因此，我们需要选择那些违反了公式 (13) 的三元组来进行训练。也就是说，给定 x_i^a，我们应该选择距离它最远的 x_i^p(hard positive)(argmax)，以及距离它最近的 x_i^n(hard negitive)(argmin)。然而，这种方式并不容易处理，因为会有一些标记错误的样本严重影响训练过程。我们有两种方法避免这个问题：

离线生成　每隔 n 次训练之后，使用最近的网络参数计算 argmax 和 argmin；

在线生成　在 mini-batch 中选择。

在线选择三元组　为了获得有意义的距离表示，在每一个 mini-batch 中必须保证有一定数量的 positive 样本。本文在数千个样本的 mini-batch 中要求 40 个左右的 positive 样本。选择 positive 是，我们选最远的那个，而选择 negitive 时，我们选所有的 negitive 样本。

参考文献

[1] 周志华, 机器学习, Qing hua da xue chu ban she, 2016. 1

[2] Sneha Kudugunta and Emilio Ferrara, "Deep neural networks for bot detection", arXiv preprint arXiv:1802.04289, 2018. 6

[3] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao, "A discriminative feature learning approach for deep face recognition", in European conference on computer vision. Springer, 2016, pp. 499–515. 8

[4] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang, "Large-margin softmax loss for convolutional neural networks.", in ICML, 2016, vol. 2, p. 7. 10

[5] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering", in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 815–823. 12