文章编号：2096-1618(2021)02-0154-05

口罩佩戴识别中的 Tiny-YOLOv3 模型算法优化

曹远杰[1,2]， 高瑜翔[1,2]， 杜鑫昌[1,2]， 王亚飞[1,2]

(1. 成都信息工程大学通信工程学院，四川 成都 610225；2. 气象信息与信号处理四川省高校重点实验室，四川 成都 610225)

摘要：针对深度学习网络(Tiny-YOLOv3)算法准确率不高以及更改网络模型后实时性的问题，提出一种网络改进方案和基于BN层剪枝的优化算法。将Tiny-YOLOv3的前四层池化层改为两步长的卷积层进行下采样以及增加特征的提取，将后两层池化层和第六个卷积层改为一个残差结构层，再利用BN层剪枝算法，将网络进行压缩和BN层合并来加速网络。改进优化后的模型算法相比原始Tiny-YOLOv3网络，在口罩佩戴识别的平均精确率(mAP)提升了14%，模型体积只有19.2 MB，压缩了42%；平均每秒传输帧数(FPS)增加了17%。实验结果表明，改进优化后的模型有更好的精确性和实时性。

关 键 词：深度学习；BN层合并；口罩识别；模型剪枝；卷积神经网络

中图分类号：TP183 **文献标志码**：A

doi：10.16836/j.cnki.jcuit.2021.02.005

0 引言

2020年暴发的新型冠状病毒传播的主要途径是呼吸道飞沫传播，戴口罩是防止病毒传播最重要的一个环节，也是人们健康最基本的保障。因此，在公共场合放置检测系统来检测人们是否佩戴口罩是很有必要的，不仅可以提高检测效率，还可以减少人员流动。目前在车站、景点、小区等都需要专门的检查人员站岗检测是否佩戴口罩[1]。在非常时期乘坐公交车时，司机为了全车人的安全还需要目不转睛地盯着上车的乘客是否佩戴口罩。因此，在公交车上部署检测口罩是否佩戴系统可以减少司机的工作量，也能更好地保护全车人的安全。同样，在医院和高危工作岗位等也有必要部署口罩佩戴的检测系统。

目前在目标检测方面，以基于深度学习的算法为主流。在YOLO算法发表前，主流算法都以R-CNN和Faster-RCNN为主[2]。Faster-RCNN[3]虽然比R-CNN有着更高的精确度和速度，但是对于有速度要求的项目而言，该算法检测速度仍然不能达到实时标准。YOLOv[4-5](you only look once)作为现在的主流目标检测算法，把目标区域预测和目标类别预测合二为一[6-7]，本文将目标检测任务看作目标区域预测和类别预测的回归问题，并且采用kmeans聚类多尺度预测，极大地改善了算法的精确度[8]。

由于某些特殊场合和嵌入式平台的需要，YOLOv3算法在资源较少的设备上无法满足实时性，而Tiny-YOLOv3正适合部署在嵌入式等资源较少的平台。针对Tiny-YOLOv3剪掉大部分的网络，精度较低的问题，董晓等[9]提出在网络部分层加入残差网络来提高模型精度。马立等[10]提出用步长为2的卷积替代池化层以及增加预测尺度来提高网络准确率。以上这些改进都可以增加精度但是牺牲了部分速度。针对加速网络，姚巍巍等[11]利用稀疏化训练，根据BN层[12]γ系数来进行剪枝，模型压缩率可以达到原来的两倍；段杰等[13]提出将批处理归一化和以前的线性层相集成的方法来加速神经网络。这些加速方案提升了前向推理速度，但是没有增加识别精度。

本文针对Tiny-YOLOv3模型精度低以及改进网络结构后的模型(GAI-Tiny-YOLOv3以下简称G-Tiny-YOLOv3)实时性问题，提出一种改进网络结构后再进行剪枝和BN层合并的网络(GAI batch normalization-tiny-YOLOv3，以下简称GB-Tiny-YOLOv3)方案。该方案在原网络的基础上增加了残差结构层，去掉了原来的池化层，改为步长为2的卷积层，来提高检测精度，并使用剪枝和BN层合并的算法来优化网络，同时改进网络的实时性。

1 Tiny-YOLOv3 网络结构改进

1.1 Tiny-YOLOv3 网络

Tiny-YOLOv3有13层卷积层，6层池化层以及

收稿日期：2020-10-15
基金项目：四川省教育厅高校创新团队资助项目(15TD0022)

1 层上采样层,输出层有 2 个尺度进行预测,分别是 13×13和26×26两种尺度[14],输出结果进行非极大值抑制获取有效的预测框,其结构如图 1 所示。

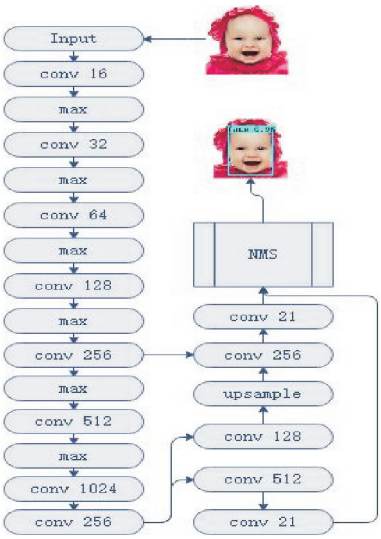


图 1　Tiny-YOLOv3 网络结构图

本文训练数据集采用开源的口罩识别数据集,从 8000 多张数据集中随机采取 3000 张作为训练数据集。训练环境如下:硬件环境为 AMD Ryzen 5 3500X 3.6 GHz处理器和 NVIDIA 显卡(GTX1060),软件环境为在 pycharm 使用python3.6和 Keras tensorflow 框架,训练 batchsize 选择为 4。

1.2　Tiny-YOLOv3 改进网络

为解决原始 Tiny-YOLOv3 网络精度低的问题,在原来网络的基础上改进网络的结构,改进结构后的网络简称为 G-Tiny-YOLOv3。本次改进主要是将 Tiny-YOLOv3 网络的前 4 个 MAXPOOL 层改为了步长为 2 的卷积层来下采样,并将第 10 层的卷积层改为 256 个卷积核,第 5 个和第 6 个池化层分别改为步长为 2 和 1 的、卷积核个数为 512 的卷积层。因为检测目标是人脸口罩,所以保留了原来网络的 26×26 和 13×13 的两个预测网络。其中,conv2 代表步长为 2 的卷积层。更改部分用红色替代,第 9、10、11 层则采用残差结构,改进模型如图 2 所示。网络改进后的模型大小为 39.0 MB,比 Tiny-YOLOv3 网络增加了5.8 MB,但是精确度有很大的提升。Tiny-YOLOv3 网络在口罩测试集取得的 mAP = 57.99%,而改进后的网络 G-Tiny-YOLOv3 在相同测试集取得的 mAP = 72.24%。检测速度上,Tiny-YOLOv3 网络检测 700 张图片用了 38.81 s,由于增加了网络复杂性,所以 G-Tiny-YOLOv3 模型检测 700 张数据集用了41.35 s,两种网络对比如表 1 所示。

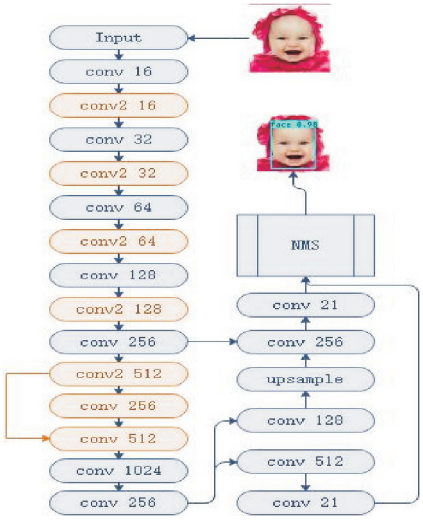


图 2　G-Tiny-YOLOv3 网络结构图

表 1　Tiny-YOLO 网络对比

模型	Tiny-YOLOv3	G-Tiny-YOLOv3
精度/%	57.99	72.24
模型体积/MB	33.2	39.0
检测速度/s	46	42

原模型和改进网络后的模型训练的 loss 曲线如图 3 所示,蓝色曲线为改进网络模型训练 loss 曲线,红色曲线为原模型训练 loss 曲线。为了增加训练效率,训练采用 3 种学习率训练。经过多次实验,发现前 15 个 epoch 使用0.01 的学习率,中间 30 个 epoch 使用 0.001,最后再用 5 个 epoch 使用0.0001进行训练可以比较快地达到收敛。

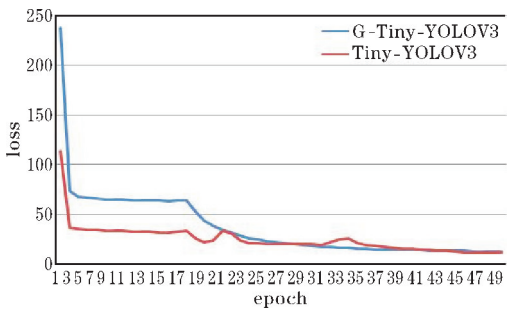


图 3　训练 loss 曲线图

2　G-Tiny-YOLOv3 网络压缩优化

2.1　Batch Normalization

训练深度学习的网络不仅需要好的硬件设备,还需要一些技巧,Batch Normalization 是 Google 提出来的一个训练技巧。这个算法不仅解决了训练模型收敛慢的问题,还在一定程度上帮助控制梯度消失和梯度爆

炸的问题。经过实验验证,加入 BN 层后可以使得深度学习网络模型变得更加稳定。

以前的网络中,只对输入数据进行归一化处理,没有在每一层都进行归一化。而输入数据经过与权重矩阵相乘后,数据的分布很有可能发生变换。随着层数的增加变化越来越大,随之也很难训练。在每层加入 BN 层,可以将这些数据在网络每层进行批次归一化到均值为 0 方差为 1 的分布处理。BN 层还添加了可学习的缩放和平移参数 γ、β,完美地解决了因为归一化而导致学习的特征被破坏的问题,这 2 个参数可以恢复出在某层学习到的特征。Batch Normalization 当前向传导公式如下:

$$\mu_\beta \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \tag{1}$$

$$\sigma_\beta^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_\beta)^2 \tag{2}$$

$$\hat{x}_i \leftarrow \frac{x_i-\mu_\beta}{\sqrt{\sigma_\beta^2+\varepsilon}} \tag{3}$$

$$y_i \leftarrow \gamma\hat{x}_i+\beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \tag{4}$$

BN 层加入到每层网络中的位置如图 4 所示,在 Relu 层之前。

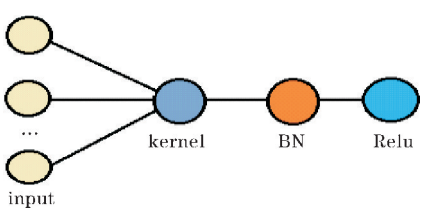


图 4 卷积通道流程

2.2 模型剪枝

神经网络参数众多,但并不是所有的参数都对网络有用,其中有一些参数对最终结果的贡献值太小而显得冗余。剪枝就是要将这些多余的参数减去,可以减少参数量和计算量,特别是对于移动设备、嵌入式系统之类的对速度要求比较高的环境,就需要剪枝压缩模型,减小内存开销。

对于网络剪枝可以追溯到 1990 年 YanLeCun 的研究,至今,剪枝的方式各种各样,根据贡献度的指标来减去一些神经元,可以是权重参数 L1,BN 层参数 γ 等。根据每个神经卷积通道提出一种基于 BN 层参数 $a=\gamma/\sqrt{\sigma_\beta^2+\varepsilon}$ 和 $b=\beta-\mu_\beta\gamma/\sqrt{\sigma_\beta^2+\varepsilon}$ 的剪枝算法,如图 4 所示,kernel 层输出的每一个值都需要经过 BN 层。假设 kernel 输出值为 X,则 BN 层的输出为

$$Y_{\mathrm{BN}}=\frac{\gamma(X-\mu_\beta)}{\sqrt{\sigma_\beta^2+\varepsilon}}+\beta \tag{5}$$

$$Y_{\mathrm{BN}}=\frac{\gamma}{\sqrt{\sigma_\beta^2+\varepsilon}}\cdot X+\beta-\mu_\beta\frac{\gamma}{\sqrt{\sigma_\beta^2+\varepsilon}} \tag{6}$$

其中,γ、β 为 BN 层的缩放因子和平移参数,μ_β、σ_β^2 为均值和方差,ε 的作用是防止分母为 0,根据训练网络设置为0.001。Y_{BN} 输出值需要经过 Relu 激活函数后输出到下一层,当 Y_{BN} 层的输出值很小时,经 Relu 层后对网络的贡献很小。根据参数 $a=\gamma/\sqrt{\sigma_\beta^2+\varepsilon}$ 的贡献度来对模型进行剪枝无疑比 γ 参数更精确,当参数 a 很小,参数 b 很大时,也会有比较大的贡献度,所以剪枝需要判断 2 个参数的贡献度来进行修剪。

本剪枝算法的参数阈值设置为动态参数阈值。由于每个卷积层的大小不一样,如果设置固定参数,当通道参数量足够大,就算参数 a 比较小,最后的结果也会有比较大的贡献度。而 a 比较小容易被剪掉,这样会降低模型的精确度。所以,本算法根据每个卷积层的输入通道数(S)宽(W)高(H)来确定阈值。根据多次实验,设阈值为当参数 a 的绝对值小于$1/(S\cdot W\cdot H)$并且 b 小于一个很小的数,这里设置为0.001时,判断为低贡献度的通道,将本层通道以及下一层与之对应的输入通道剪掉。BN 层剪枝示意图如图 5 所示。

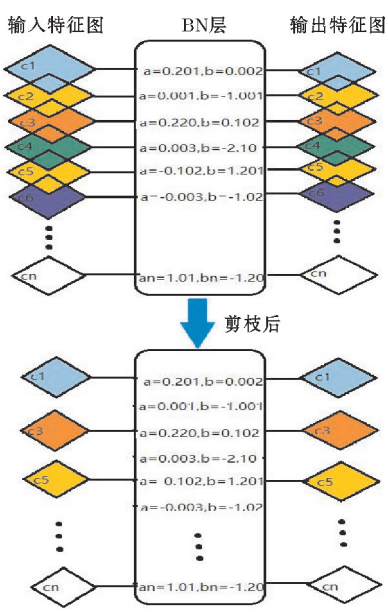


图 5 BN 剪枝示图

改进网络结构后的模型(G-Tiny-YOLOv3)各卷积层剪枝如表 2 所示,网络总共有 19 层卷积层,其中第 16 层和第 19 层为输出层,浅层网络通道尺度较小,提取特征信息比较重要,剪枝较少。深层网络剪枝较多。网络中的 shortcut 也进行了通道剪枝,第 10 ~ 12 层为 shortcut 层,因此将第 10 和第 12 层需要剪枝的通道进行对比,通道相同的进行剪枝。此剪枝算法动态阈值剪枝的好处就是不需要稀疏训练,也能将大量的冗余

通道剪掉。

表 2　G-Tiny-YOLOv3 剪枝前后卷积层大小

卷积层	剪枝前	剪枝后
Conv1	16	15
Conv2	16	14
Conv3	32	28
Conv4	32	32
Conv5	64	59
Conv6	64	61
Conv7	128	110
Conv8	128	109
Conv9	256	206
Conv10	512	488
Conv11	256	237
Conv12	512	488
Conv13	1024	211
Conv14	256	199
Conv15	512	492
Conv17	128	119
Conv18	256	224

2.3　BN 层合并

训练模型时,加入 BN 层非常有必要,可以加速训练等。但是,预测时却会多很多层,从而影响模型的性能。训练好后的 BN 层是一些固定的值,其中有 4 个参数,均值:μ_β,方差:σ_β^2,缩放因子:γ,偏移:β。由于 Tiny-YOLOv3 加入 BN 层的卷积没有加偏置参数,所以卷积计算结果为

$$x_{\text{conv}}=\sum_{i=0}^{n}x_i\cdot w_i \tag{7}$$

将式(7)代入到式(6)得到 BN 层输出与卷积关系:

$$Y_{\text{BN}}=\sum_{i=0}^{n}\left(x_i\cdot\frac{\gamma\cdot w_i}{\sqrt{\sigma_\beta^2+\varepsilon}}\right)+\beta-\frac{\gamma\cdot\mu_\beta}{\sqrt{\sigma_\beta^2+\varepsilon}} \tag{8}$$

合并后的权重参数为

$$W_i'=\frac{\gamma\cdot w_i}{\sqrt{\sigma_\beta^2+\varepsilon}} \tag{9}$$

合并后的偏置参数为

$$\beta'=\beta-\frac{\gamma\cdot\mu_\beta}{\sqrt{\sigma_\beta^2+\varepsilon}} \tag{10}$$

合并后的通道输出为

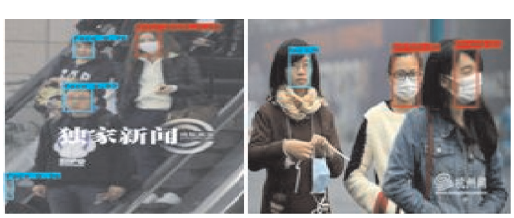
$$Y=\text{Relu}\left[\sum_{i=0}^{n}\left(x_i\cdot W_i'\right)+\beta'\right] \tag{11}$$

剪枝合并 BN 层后的网络简称为 GB-Tiny-YOLOv3,平均 FPS 可以达到 54,而 Tiny-YOLOv3 的平均 FPS 只有 46。参数总量这里定义的是模型所有参数所占大小,单位为 MB,常用于评估占用资源的大小。GB-Tiny-YOLOv3 网络参数量只有19.2 MB,是 G-Tiny-YOLOv3 的 49%,是 Tiny-YOLOv3 的58%。计算量量度为 BFLOPs,在这里代表某次卷积运算需要多少个十亿次浮点数运算,将每层卷积运算所消耗的 BFLOPs 加起来代表这个算法的时间复杂度,在一定条件下用于评估这个算法所需消耗的时间。GB-Tiny-YOLOv3 网络检测 700 张图片只用了35 s,比 Tiny-YOLOv3 要快3 s左右。其中,T-Y 表示 Tiny-YOLOv3,G-T-Y 表示 G-Tiny-YOLOv3,GB-T-Y 表示 GB-Tiny-YOLOv3. 网络对比如表 3 所示。

表 3　各网络对比

模型	T-Y	G-T-Y	GB-T-Y
精度/%	57.99	72.24	72.09
GTX1060 检测速度/s	46	42	54
MX130 检测速度/s	17	14	23
参数总量/MB	33.2	39.0	19.2
计算量 BFLOPs/s	5.567	6.684	4.297

经过剪枝后的模型在精度方面基本没有降低,但是时间复杂度和空间复杂度大大降低,这对于资源内存较小的设备是非常有利的。检测速度分别在 GTX1060GPU 和 MX130GPU 上进行测试,都能看出剪枝后的模型比原始的 Tiny-YOLOv3 在精度和速度上都要高很多。Tiny-YOLOv3 和 GB-Tiny-YOLOv3 模型识别效果对比如图 6 所示。分别识别了戴口罩和没带口罩的图片,图 6(a)为 GB-Tiny-YOLOv3 识别的结果,图 6(b)为 Tiny-YOLOv3 识别的结果,可以看出,GB-Tiny-YOLOv3 识别是否佩戴口罩要比原始网络 Tiny-YOLOv3 效果更好。



(a) GB-Tiny-YOLOv3 识别结果图

(b)Tiny-YOLOv3 识别结果图

图 6　算法改进前后预测结果

3 结束语

本文所改进的网络算法 GB-Tiny-YOLOv3 比原始 Tiny-YOLOv3 在计算量减少几个量级的同时,参数量也减少了很多。计算量代表时间复杂度,参数量代表空间复杂度。改进后的网络在资源占用和计算时间上都优越于 Tiny-YOLOv3 网络,在 GTX1060GPU 上检测平均 FPS 达到了 54,后续硬件实现量化后,可以更好地在移动设备上达到实时性与准确性的要求。

参考文献:

[1] 肖俊杰. 基于 YOLOv3 和 YCrCb 的人脸口罩检测与规范佩戴识别[J]. 软件,2020,41(7):164-169.

[2] He K M, Zhang X Y, Ren S Q, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(9): 1904-1916.

[3] Girshick R. Fast R-CNN [C]. IEEE International Conference on Computer Vision. Santiago: IEEE, 2015:1440-1448.

[4] Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection[C]. IEEE Conference on Computer Vision and Pattern Recognition(CVPR), IEEE, 2016:779-788.

[5] J Farhadi A. YOLO9000: Better, Faster, Stronger [C]. IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2017:6517-6525.

[6] Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation[C]. CVPR. IEEE, 2014: 580-587.

[7] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[J]. Lecture Notes in Computer Science, 2016(1):21-27.

[8] Redmon J, Farhadi A. An Incremental Improvement [J]. arXiv e-prints, 2018(3).

[9] Xiao D, Shan F, Li Z, et al. A Target Detection Model Based on Improved Tiny-yolov3 Under the Environment of Mining Truck[J]. IEEE Access, 2019, (99):1.

[10] 马立,巩笑天,欧阳航空. Tiny YOLOV3 目标检测改进[J]. 光学精密工程,2020,28(4):988-995.

[11] 姚巍巍,张洁. 基于模型剪枝和半精度加速改进 YOLOv3-tiny 算法的实时司机违章行为检测[J]. 计算机系统应用,2020,29(4):41-47.

[12] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift[J]. arxiv, 2015(2).

[13] Duan J, Zhang R X, Huang J, et al. The Speed Improvement by Merging Batch Normalization into Previously Linear Layer in CNN[C]. 2018 International Conference on Audio, Language and Image Processing(ICALIP). 2018.

[14] Xu Z F, Jia R S, Liu Y B, etal. Fast Method of Detecting Tomatoes in a Complex Scene for Picking Robots[J]. IEEE Access, 2020, (99):1.

Tiny-YOLOv3 Model Algorithm is Optimized for Mask Wearing Recognition

CAOYuanjie[1,2], GAO Yuxiang[1,2], DU Xinchang[1,2], WANG Yafei[1,2]

(1. College of Communication Engineering, Chengdu University of Information Technology, Chengdu 610225, China; 2. Meteorological Information and Signal Processing Key Laboratory of Sichuan Education Institutes, Chengdu 610225, China)

Abstract: Aiming at the low accuracy of deep learning network (Tiny-YOLOv3) algorithm and the instantaneity after changing the network model, a network improvement scheme and an optimization algorithm based on BN layer pruning are proposed. In this method, the first four pooling layers of Tiny-Yolov3 are replaced by a two-step convolutional layer for down-sampling and feature extraction, and the latter two pooling layers and the sixth convolutional layer are changed into a residual structure layer. Then the BN layer pruning algorithm is used to compress the network and combine the BN layer to accelerate the network. Compared with the original Tiny-YOLOv3 network, the improved and optimized model algorithm improves the mean accuracy rate (mAP) of mask wearing recognition by 14%. The model volume is only 19.2 MB, which is compressed by 42%. The average number of frames per second (FPS) increased by 17%. The experimental results show that the improved and optimized model has better accuracy and real-time performance.

Keywords: deep learning; BN merge; mask recognition; model pruning; convolutional neural network