|  |  |
| --- | --- |
| MGN算法解析 | 文件类型：技术文档 |
| 文件编号： |
| 面向的部门： 研发中心 |
| 保密等级：高 |
| 作者： 刘凯 |
| 日期：2019-05-09 |
| 版本：1.0V |

MGN算法解析



目 录

[目 录 ii](#_Toc1944209911)

[1 MGN解析 3](#_Toc482708532)

[1.1 简介 3](#_Toc2071596589)

[1.2 MGN架构 3](#_Toc748198150)

[1.3 MGN性能 5](#_Toc452986289)

[1.4 详细网络解析1-Backbone ResNet50 6](#_Toc1653495515)

[1.5 详细网络解析2-MGN特征提取 8](#_Toc1400927480)

[参考文献 1](#_Toc157404419)

# MGN解析

* 1. 简介

多粒度网络(MGN, Multiple Granularity Network)是云从科技提出的一种行人重识别(Person Re-Identification，ReID)算法，目前在Market1501，DukeMTMC和CUHK03等行人重识别数据集上都取得了很好的结果。

MGN算法是一个端到端(end-to-end)的行人特征学习算法，其特征为全局特征和不同粒度的局部特征的结合，如图1所示，其中左边为全局特征，中间和右边则为粒度为2和3的局部特征。

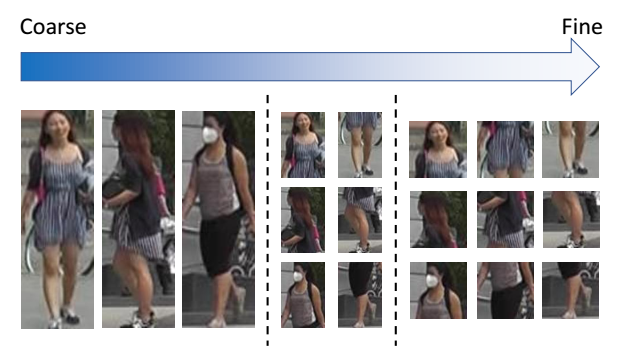


图1：MGN不同粒度划分

* 1. MGN架构

MGN的网络架构以ResNet50为骨干网(backbone)，在res\_conv4\_1之后，根据不同的粒度有三个并行的分支，如图2所示。其中第一个分支为全局特征，在后面接一个stride为2的conv层做下采样；然后经过一个全局最大池化层(GMP, Global Maximum Pooling)得到2048维的特征zg;再通过一个1\*1的卷积层，得到256维的特征fg。特征提取就到此为止，不过训练的话还需接一个全连接层和loss层。后面两个分支的总体结构与全局分支类似，只是有两点不同，其一是骨干网后面接的conv层不做下采样，仍然保持原来的维度；其二为做GMP时，分别将特征图从水平方向上(H维度)切割成2份和3份。

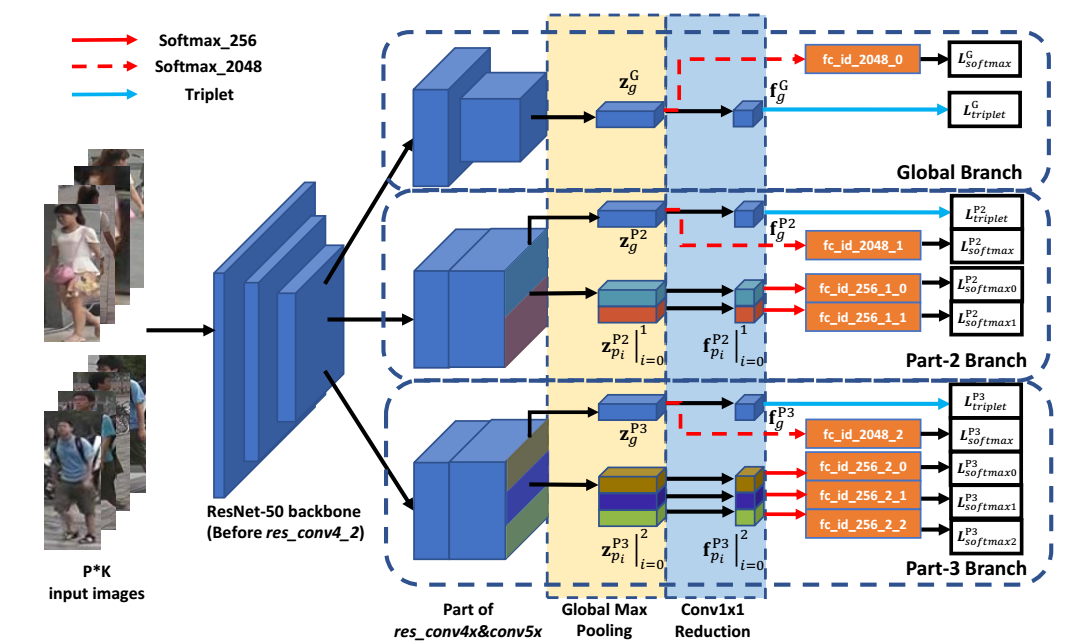


图2：MGN架构

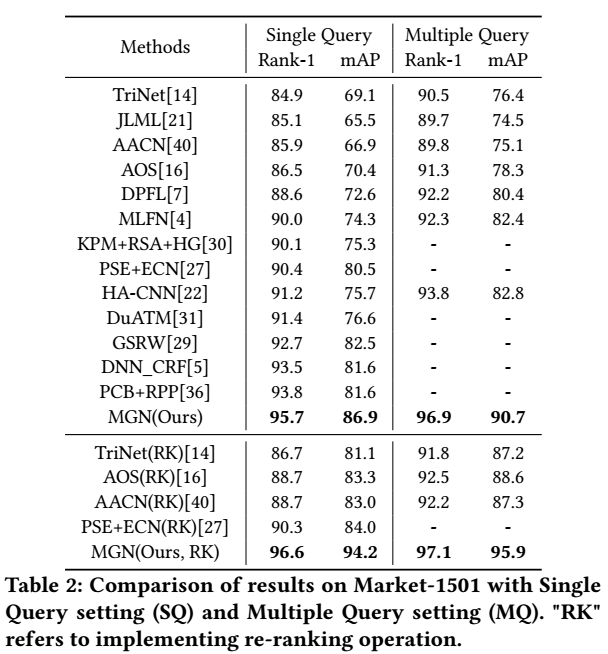


图3：MGN在Market-1501上的性能

* 1. MGN性能

MGN在Market1501，DukeMTMC和CUHK03等行人重识别数据集上的性能如图3-图5所示。

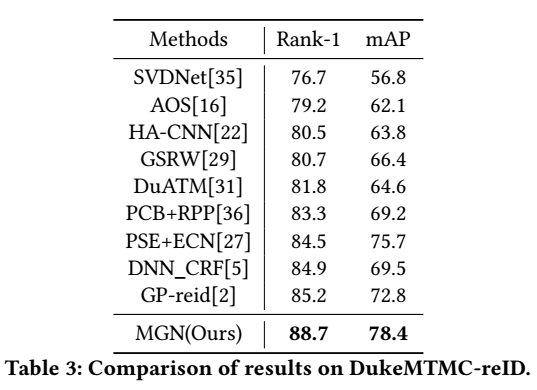


图4：MGN在DukeMTMC上的性能

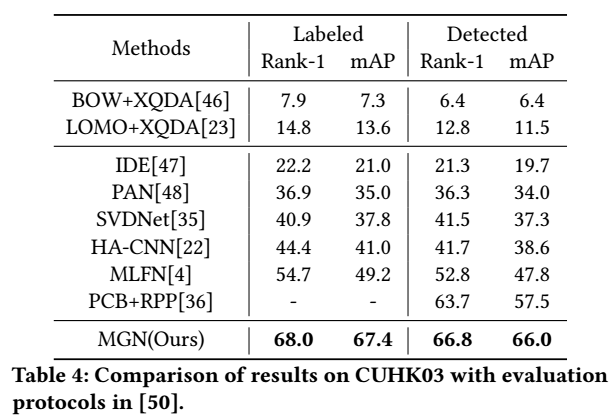


图5：MGN在CUHK03上的性能

* 1. 详细网络解析1-Backbone ResNet50

ResNet50的整体网络结构如图6所示。其中有4个Res模块：Res1到Res4，一个PreRes模块和一个fc模块。下面会一一解析。

PreRes

Res1

Res2

fc

Res4

Res3

图6：ResNet50总体网络结构

PreRes模块由一组conv，bn，relu和maxPooling层组成，其网络结构如图7所示。

Res模块由几个Bottleneck模块组合而成，其中Res1到Res4分别有3,4,6,3个Bottleneck。Bottleneck模块的构成如图8所示，每个Res模块的第一个Bottleneck包含一个downsample层，以保证残差相加时维度一样，后面的Bottleneck的则是直通。Res1模块如图9所示，后面几个Res模块与之类似，不再赘述。

fc模块包含一个averagePooling层和一个fc层，averagePooling层对Res4输出的整个特征做池化，输出2048个通道作为fc层的输入。fc层的维度为2048×1000，输出维度为1000。

Conv0

BN

Relu

MaxPool

图7：PreRes模块结构图

Conv1

BN1+Relu1

Conv2

BN3

Conv3

BN2+Relu2

Conv1-ds

BN1-ds

Sum

Relu3

Conv1

BN1+Relu1

Conv2

BN3

Conv3

BN2+Relu2

Sum

Relu3

(a) (b)

图8：Bottleneck模块示意图。其中(a)为每个Res模块第一个Bottleneck模块网络，输入信号会经过残差和下采样两条路径然后相加; (b)为另一种Bottleneck模块网络，与(a)不同，输入信号会经过残差和直通两条路径，最后相加得到输出。

Bottleneck0

Bottleneck1

Bottleneck2

图9：Res1模块网络结构

* 1. 详细网络解析2-MGN特征提取

MGN的Backbone采用Resnet50的Res3模块之止，之后网络分为三路，根据不同粒度进行不同的处理，最后得到8个全局或局部特征，每个特征256维，合并起来共2048维特征，即为MGN的特征输出。网络结构如图10所示。

1×2048

......

1×256×1×1

1×2048×1×1

backbone

Res4

Reduction

Res4\_p

Res4\_p

GMP

MP\_p3

GMP

MP\_p2

GMP

Reduction

......

Reduction

Concat

图10：MGN网络结构

图10中的backbone=PreRes+Res1+Res2+Res3。MGN输入图片的维度为1×3×384×128，从backbone输出的特征图维度为1×1024×24×8，这个特征图进入三条分支网络。

第一条分支中的Res4即为ResNet50中的Res4模块，该模块有一个stride为2的卷积层，实现2倍下采样，输出维度为1×2048×12×4。全局最大池化GMP会对每个通道做整张特征图的最大池化，输出维度为1×2048×1×1。然后进入Reduction模块，得到256维的特征矢量。Reduction模块由一个1×1的conv层，后面接一共BN层和Relu层，网络结构如图11所示。

Conv1×1

BN

Relu

图11：Reduction模块

第二条分支的Res4\_p模块与Res4模块类似，只是将Res4中做下采样的卷积层的stride变成1，使得特征图的大小仍然为24×8，即输出维度为1×2048×24×8。后面会接两个池化层，MP\_p2是将特征图沿水平切分为两份，即池化核大小为12×8，输出维度为1×2048×2×1。将此输出再分为两个1×2048×1×1的数据分别进入Reduction模块。第二条分支一共有3个256维的特征矢量。

第三条分支与第二条分支类似，只是MP\_p3是将特征图沿水平切分为三份，即池化核大小为8×8，输出维度为1×2048×3×1，然后分为三个1×2048×1×1的数据分别进入Reduction模块。所以第三条分支一共有4个256维的特征矢量。

三条分支共8个256维的特征矢量，拼接成2048维的特征矢量作为最后的输出特征。

表1为MGN所有层的参数表。其中r表示Res模块，b表示Bottleneck，如r1b1conv1表示Res1模块中Bottleneck1中的conv1层，以此类推。权重和输出维度顺序为(N,C,H,W)。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Weight/Kernel | Stride | Padding | 输出维度 |
| input | - | - | - | (1,3,384,128) |
| conv0 | (64,3,7,7) | 2 | 3 | (1,64,192,64) |
| maxpool0 | (7,7) | 2 | 1 | (1,64,96,32) |
| r1b1conv1 | (64,64,1,1) | 1 | 0 | (1,64,96,32) |
| r1b1conv2 | (64,64,3,3) | 1 | 1 | (1,64,96,32) |
| r1b1conv3 | (256,64,1,1) | 1 | 0 | (1,256,96,32) |
| r1b1conv1-ds | (256,64,1,1) | 1 | 0 | (1,256,96,32) |
| r1b2conv1 | (64,256,1,1) | 1 | 0 | (1,64,96,32) |
| r1b2conv2 | (64,64,3,3) | 1 | 1 | (1,64,96,32) |
| r1b2conv3 | (256,64,1,1) | 1 | 0 | (1,256,96,32) |
| r1b3conv1 | (64,256,1,1) | 1 | 0 | (1,64,96,32) |
| r1b3conv2 | (64,64,3,3) | 1 | 1 | (1,64,96,32) |
| r1b3conv3 | (256,64,1,1) | 1 | 0 | (1,256,96,32) |
| r2b1conv1 | (128,256,1,1) | 1 | 0 | (1,128,96,32) |
| r2b1conv2 | (128,128,3,3) | 2 | 1 | (1,128,48,16) |
| r2b1conv3 | (512,128,1,1) | 1 | 0 | (1,512,48,16) |
| r2b1conv1-ds | (512,256,1,1) | 2 | 0 | (1,512,48,16) |
| r2b2conv1 | (128,512,1,1) | 1 | 0 | (1,128,48,16) |
| r2b2conv2 | (128,128,3,3) | 1 | 1 | (1,128,48,16) |
| r2b2conv3 | (512,128,1,1) | 1 | 0 | (1,512,48,16) |
| r2b3conv1 | (128,512,1,1) | 1 | 0 | (1,128,48,16) |
| r2b3conv2 | (128,128,3,3) | 1 | 1 | (1,128,48,16) |
| r2b3conv3 | (512,128,1,1) | 1 | 0 | (1,512,48,16) |
| r2b4conv1 | (128,512,1,1) | 1 | 0 | (1,128,48,16) |
| r2b4conv2 | (128,128,3,3) | 1 | 1 | (1,128,48,16) |
| r2b4conv3 | (512,128,1,1) | 1 | 0 | (1,512,48,16) |
| r3b1conv1 | (256,512,1,1) | 1 | 0 | (1,256,48,16) |
| r3b1conv2 | (256,256,3,3) | 2 | 1 | (1,256,24,8) |
| r3b1conv3 | (1024,256,1,1) | 1 | 0 | (1,1024,24,8) |
| r3b1conv1-ds | (1024,512,1,1) | 2 | 0 | (1,1024,24,8) |
| r3b2conv1 | (256,1024,1,1) | 1 | 0 | (1,256,24,8) |
| r3b2conv2 | (256,256,3,3) | 1 | 1 | (1,256,24,8) |
| r3b2conv3 | (1024,256,1,1) | 1 | 0 | (1,1024,24,8) |
| r3b3conv1 | (256,1024,1,1) | 1 | 0 | (1,256,24,8) |
| r3b3conv2 | (256,256,3,3) | 1 | 1 | (1,256,24,8) |
| r3b3conv3 | (1024,256,1,1) | 1 | 0 | (1,1024,24,8) |
| r3b4conv1 | (256,1024,1,1) | 1 | 0 | (1,256,24,8) |
| r3b4conv2 | (256,256,3,3) | 1 | 1 | (1,256,24,8) |
| r3b4conv3 | (1024,256,1,1) | 1 | 0 | (1,1024,24,8) |
| r3b5conv1 | (256,1024,1,1) | 1 | 0 | (1,256,24,8) |
| r3b5conv2 | (256,256,3,3) | 1 | 1 | (1,256,24,8) |
| r3b5conv3 | (1024,256,1,1) | 1 | 0 | (1,1024,24,8) |
| r3b6conv1 | (256,1024,1,1) | 1 | 0 | (1,256,24,8) |
| r3b6conv2 | (256,256,3,3) | 1 | 1 | (1,256,24,8) |
| r3b6conv3 | (1024,256,1,1) | 1 | 0 | (1,1024,24,8) |
| r4b1conv1 | (512,1024,1,1) | 1 | 0 | (1,512,24,8) |
| r4b1conv2 | (512,512,3,3) | 2 | 1 | (1,512,12,4) |
| r4b1conv3 | (2048,512,1,1) | 1 | 0 | (1,2048,12,4) |
| r4b1conv1-ds | (2048,1024,1,1) | 2 | 0 | (1,2048,12,4) |
| r4b2conv1 | (512,2048,1,1) | 1 | 0 | (1,512,12,4) |
| r4b2conv2 | (512,512,3,3) | 1 | 1 | (1,512,12,4) |
| r4b2conv3 | (2048,512,1,1) | 1 | 0 | (1,2048,12,4) |
| r4b3conv1 | (512,2048,1,1) | 1 | 0 | (1,512,12,4) |
| r4b3conv2 | (512,512,3,3) | 1 | 1 | (1,512,12,4) |
| r4b3conv3 | (2048,512,1,1) | 1 | 0 | (1,2048,12,4) |
| rp4b1conv1 | (512,1024,1,1) | 1 | 0 | (1,512,24,8) |
| rp4b1conv2 | (512,512,3,3) | 1 | 1 | (1,512,24,8) |
| rp4b1conv3 | (2048,512,1,1) | 1 | 0 | (1,2048,24,8) |
| rp4b1conv1-ds | (2048,1024,1,1) | 1 | 0 | (1,2048,24,8) |
| rp4b2conv1 | (512,2048,1,1) | 1 | 0 | (1,512,24,8) |
| rp4b2conv2 | (512,512,3,3) | 1 | 1 | (1,512,24,8) |
| rp4b2conv3 | (2048,512,1,1) | 1 | 0 | (1,2048,24,8) |
| rp4b3conv1 | (512,2048,1,1) | 1 | 0 | (1,512,24,8) |
| rp4b3conv2 | (512,512,3,3) | 1 | 1 | (1,512,24,8) |
| rp4b3conv3 | (2048,512,1,1) | 1 | 0 | (1,2048,24,8) |
| mp\_zg\_p1 | (12,4) | (12,4) | 0 | (1,2048,1,1) |
| mp\_zg\_p2 | (24,8) | (24,8) | 0 | (1,2048,1,1) |
| mp\_zg\_p3 | (24,8) | (24,8) | 0 | (1,2048,1,1) |
| mp\_zp2 | (12,8) | (12,8) | 0 | (1,2048,2,1) |
| mp\_zp3 | (8,8) | (8,8) | 0 | (1,2048,3,1) |
| reduction-conv | (256,2048,1,1) | 1 | 0 | (1,256,1,1) |

参考文献

1. Guanshuo Wang, Yufeng Yuan, Xiong Chen, Jiwei Li, and Xi Zhou. Learning discriminative features with multiple granularities for person re-identification. ACM Multimedia, 2018.