ReactCNN项目报告

丁霄汉

2017312365

黄锐泓

2016311939

郭高扬

2016311946

**摘要**

本项目实现了一个卷积神经网络可视化工具——ReactCNN，用来辅助卷积神经网络的调试与优化。这个可视化工具拥有与Tensorboard相似的工作原理，它在后台运行一个VGG-16，然后在前端将VGG-16运行产生的实时数据进行可视化。具体地，ReactCNN可以动态地将VGG-16中每层的filter输出均值、每层的feature map以及每层的filter聚类结果进行可视化，基于这些可视化结果，我们可以更好地窥视到VGG-16的内部工作原理，并对其进行异常检测和启发式冗余检测等，从而在一定程度上指导我们进一步优化VGG-16。在实验部分，我们对ReactCNN的可视化结果进行了详细的展示，实验结果说明该工具是可用且有效的。

# 引言

随着人类社会数据量和计算能力的不断提升，卷积神经网络（CNN）在视觉、NLP等领域的应用日益广泛。然而，到目前为止，CNN的本质还是一个黑箱。一方面，CNN缺乏可解释性使得其难以调试；另一方面，CNN的设计过于经验主义，难以评价设计的好坏。

我们认为，CNN的使用者和研究者期望得到以下重要信息：一张图进入网络后，网络的各个filter是怎样响应的；产生的输出（feature map）是怎样的；当网络的输出不正常时，是哪一部分出了问题；网络的设计是不是冗余的。

于是，我们寻求可视化技术的帮助，试图将CNN的输入、中间结果和输出可视化，以帮助使用者直观地看到网络的工作过程，辅助错误定位和调试；另一方面，我们希望使用者能够定性的、直观地评价网络的冗余性，为设计紧凑的、高效的CNN提供帮助。

# 功能与实现

ReactCNN主要分后台支撑和前台渲染两部分，其系统架构图如图2.1所示。后端负责运行VGG-16并存储中间数据。前端负责实时读取中间数据并可视化。

../../../Desktop/演示文稿1.pdf

图2.1 ReactCNN系统架构图

具体地，ReactCNN可以动态地将VGG-16中每层的filter输出均值、每层的feature map以及每层的filter聚类结果进行可视化，并支持启发式冗余检测，下边介绍ReactCNN一些重要部分的实现细节。

## 后台支撑

本项目使用Tensorflow作为深度学习框架。在实验中，我们使用了在CIFAR-10数据集上训练得到的VGG-16模型作为示例。

后端程序运行的过程本质上是前向传播的过程。我们设置batch size=1，每次向网络中输入（feed）1幅图片，取出（fetch）每层的输出。

然而，将VGG-16中每一层的全部feature map都保存起来是不可行的。例如，对于每一幅输入图像，第一和第二卷积层各产生64个尺寸为32×32的矩阵。

在实践中我们发现，一个filter所输出的feature map矩阵的全局平均值（global average pooling）可以很好地代表这一个filter的行为。因此，我们从网络中fetch的实际上是feature map的全局平均值（GAP值），这就大大减少了时间和空间开销。

由于filter本身数学形式缺乏可解释性（3阶张量），我们通过两个filter的行为的相似性来估计两个filter本身的相似性。具体来说，当本系统采集到了两个filter在M幅图片上的输出（GAP值）后，得到两个M维向量，我们用这两个向量的Pearson相关系数评价这两个filter的相似程度。随着系统运行时间变长，采集到的数据越来越多，这一估计会趋于稳定，最终很好地反映出filter之间的相似性。

经过必要的预处理后，我们把采集到的GAP值和由此计算出的相关系数写入缓存文件中，以备前端使用。

## filter全局平均值可视化

1）创建最外层容器svg、图片标签image、用于显示各层filter的g标签以及用于显示分类结果和概率的g标签；

2）使用函数d3.scaleLinear()和d3.interpolate()创建根据数值大小选择颜色的取色器函数；

3）使用函数window.setInterval()创建定时器；

4）实现定时器调用的函数timer，该函数的主要功能是定时读取当前后端处理完的图片、后端输出的包含对应各层filter全局平均值的文件以及包含所有feature map数值的文件，并更新前端界面显示；

5）关于filter全局平均值可视化的具体细节如下：每个filter全局平均值以一个由其数值决定颜色的小矩形（rect标签）来表示，颜色较深的表示数值较大，颜色较浅的表示数值较小，一共有13层网络，第1-2层分别有64个小矩形，3-4层分别有128个小矩形，5-7层分别有256个小矩形，8-13层分别有512个小矩形，首先根据每层不同的数据个数通过绑定data和enter()函数添加不同数量的rect标签，然后以外层svg的中轴线为中心，每层网络每行显示8个小矩形的形式来定位各个矩形的位置。当后台输出新的包含filter全局平均值的文件和包含所有feature map数值的文件时，由于数据的数量没有发生改变，因此小矩形的位置和数量也不会发生改变，只有小矩形的颜色会随着图片的更新发生改变；

6）关于分类结果和概率的显示：概率最大的一类即为分类结果，颜色会标红。

## feature map可视化

1）在最外层容器svg内创建用于显示单个filter详细展开对应feature map的内层svg标签；

2）在2.2中代表每个filter全局平均值的小矩形上绑定mouseover事件，用于显示对应filter展开的feature map，以及mouseout事件，用于清空当前显示的feature map；

3）关于feature map的显示，同样地，每个数值以一个由其数值大小决定颜色的小矩形来表示，颜色深浅反映了数值的大小，不同网络层中filter对应的feature map分别会显示在该层的正上方，具体每个小矩形的位置由其所在网络层数、对应的filter位置、feature map中数值的个数等综合计算得出。

## filter聚类可视化

为了对filter进行冗余分析，我们基于filter的行为特征对filter进行聚类，属于相同类的filter我们认为在一定程度上具有相似的行为特征，从而有较大的概率是冗余的。

首先，我们对filter进行行为特征的提取。基于经验，filter输出值的均值可以很好地刻画该filter的特征，所以我们使用该均值来构建filter的特征向量。对于某个filter，VGG-16每处理一张图片，则其产生一个均值，则其特征向量增加一个维度，随着图片处理数量的增加，该特征向量的维度也会增加，直觉上对filter的刻画精度也会提高，聚类结果也会趋向于稳定。

接着，基于filter的行为特征，我们使用Louvain算法对VGG-16任意一层的所有filter进行聚类，Louvain算法是一个基于图的社区发现算法，我们将每个filter看作一个点，将filter特征向量之间的协方差系数看作边，从而构造一个完全图，然后使用Louvain算法对该图进行聚类。

最后，我们使用d3的力导向图对filter的聚类结果进行可视化，相同类的filter颜色相同，距离相近，不同类的filter颜色不同，距离较远。

## 启发式冗余检测

基于filter的聚类结果，我们可以对VGG-16进行启发式冗余检测。点击力导向图中的某个filter（表现为一个点），矩形图（即filter输出均值的可视化图）中对应的所有与该filter属于同一类（包括该filter本身）的filter会发生颜色变化，以告诉用户这些属于同一类的filter在VGG-16中的具体位置，从而指导用户进行进一步的冗余分析。

具体在实现过程中，对力导向图中的每个点增加监听事件，监听到点击事件后，获取该filter所属的类别和属于该类别的所有filter的id，然后通知矩形图对相应id的filter进行颜色刷新。

# 结果展示

## filter全局平均值可视化展示

随着后端程序对不同的图片进行处理,前端会展示不同图片的各层网络中各个filter全局平均值的可视化结果以及对于该图片的分类结果,展示效果如图3.1所示:

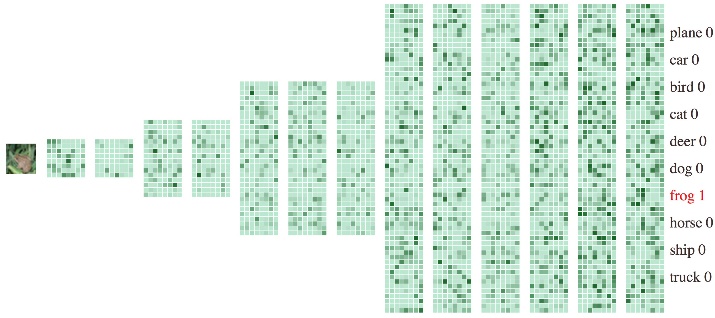


图3.1 filter全局平均值可视化展示

该图中颜色的深浅表示了filter全局平均值的大小，右边的文字表示模型认为该图片有极大概率是frog。

## feature map可视化展示

将鼠标滑动到任意一个小矩形块上时，在该层矩形组的正上方会显示该小矩形块对应的filter的具体的feature map的可视化展示，鼠标滑出小矩形范围时，对应的feature map会消失。如图3.2所示，鼠标滑动到了第三层网络的某个小矩形块上，在第三层矩形组的上方就显示了对应的feature map的可视化结果。

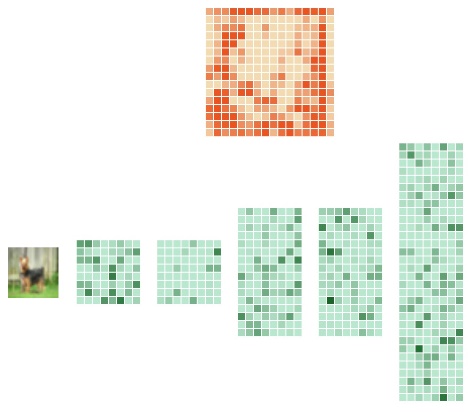


图3.2 feature map可视化展示

## filter聚类可视化展示

点击矩形图中的任意层，我们对该层的所有filter进行聚类并可视化。VGG-16第四层某一时刻的filter聚类可视化结果如图3.3所示：

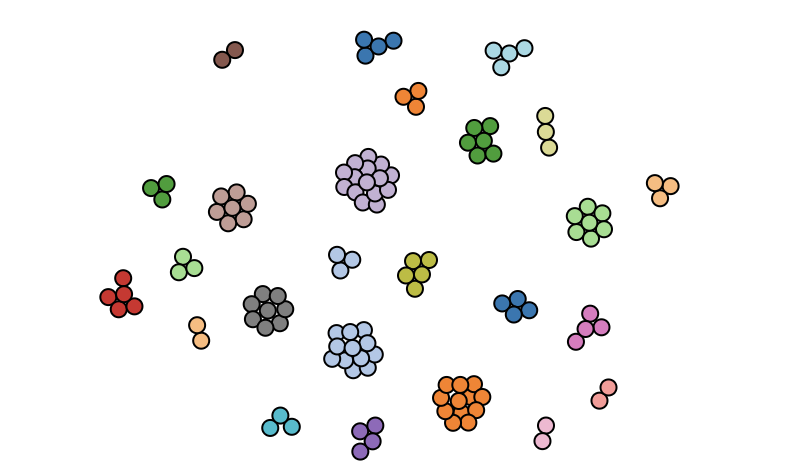


图3.3 filter聚类可视化结果

该图展示了VGG-16的第四层的128个filter某一时刻的聚类结果。图中共有128个点，代表128个filter，共由25个点簇，代表25个类别。属于同一类别的filter拥有相似的行为特征，从而有较大的概率是冗余的。

## 启发式冗余检测展示

启发式冗余检测的可视化效果如图3.4所示：

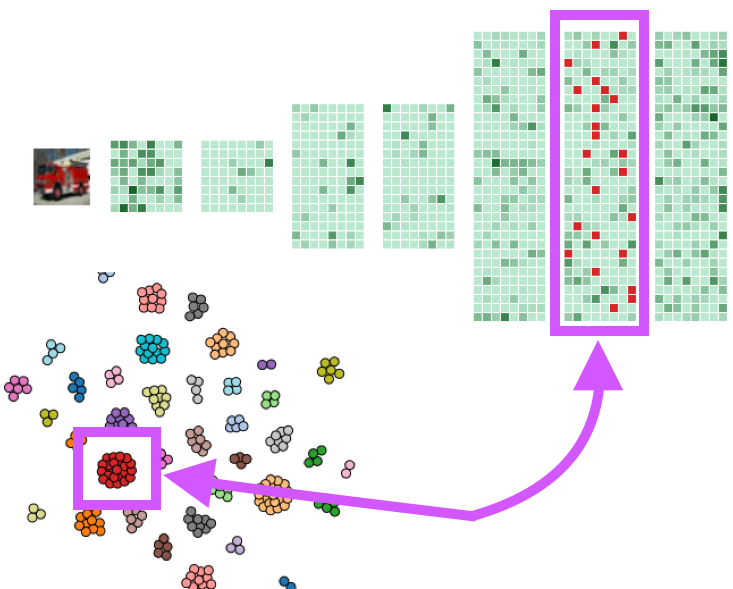


图3.4 启发式冗余检测

图中使用紫色箭头和方框标注出了力导向图和矩形图的对应关系，在力导向图中，红色的簇代表属于同一类的若干filter，点击该簇中的任意一点，矩形图中所有与该簇对应的矩形也会显示为红色，以指导用户进行进一步的冗余检测。

# 总结

一方面，作为一个CNN辅助调试和优化工具，本系统成功的实现了预期的功能。另一方面，在本系统的实现过程中，我们对CNN的基本理论和运作过程有了更多直观的感受。在实际使用中，我们发现较低层次的卷积层还是具有一定的可解释性的。例如，我们发现图4中当前显示的feature map中的深色部分和输入图片中的绿色部分是高度重叠的，由此我们有理由认为其对应的filter的功能是提取绿色区域。

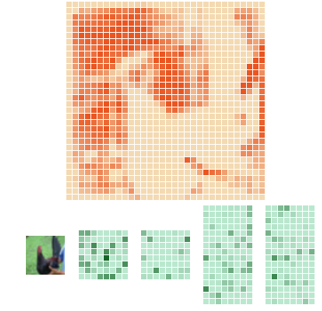


图4 具有一定的可解释性的feature map

# 分工

丁霄汉：后端逻辑设计和实现，模型设计、预训练、数据结构设计和预处理。

黄锐泓：负责不同层网络的filter全局平均值可视化展示和feature map可视化展示,同时参与启发式冗余检测功能的交互实现。

郭高扬：负责神经网络不同层filter聚类结果的可视化展示，参与完成启发式冗余检测功能的实现。