基于概率图的属性推理检测

此题目是根据受大家启发，想到的一个方案，其中包括部分推导，鉴于水平有限，如有错误，请各位指正，谢谢！

摘要：**图片属性检测**是一个多标签的问题，在一张图片中，可能存在多个属性。传统方法上，在预测图片属性时，往往会忽略其他属性本文利用属性间的相关关系，仅使用单一的物体属性进行预测。下面提到的方法，利用了图像中其他属性的信息来预测当前需要预测的属性。**在本文中，使用概率图描述属性间的相关关系，并求解最大后验概率。并通过多层神经网络的方式进行建模，与CNN结合，从而达到端到端的模型。**

关键词：**mutil-task**，**mutil-label**，**deep-learning**，**probability graph**

属性检测，一般即用分类器做属性分类，下文用到的符号有

$attribute_i$：最终预测的标签结果。

$AttributePredict_j$：分类器输出的各属性分类结果。

1.背景和动机

对于一张服装图片，当我们需要预测其中的某一种属性的时候，我们希望能**借用其他的属性辅助作出更准确**的决策。

即以下公式（N为需要预测的属性的个数）：

$$p(attribute_j|AttributePredict_i, Image) i \neq j \quad , \ j = 1...N$$

而不是

$$p(attribute_j|Image) \ j = 1...N$$

2.方案与想法

假设我们已经有M个softmax分类器，来预测N个属性（假设每个属性的类别数相等），此时输入图片，通过CNN，我们会得到标签的预测，即：

$$p(attribute_j|Image) \ \ j = 1...N$$

因此我们**要解决的问题**有：

1.如何利用$p(attribute_j|Image)$ 建立与

$$p(attribute_j|AttributePredict_i, Image)$$

的联系，即怎么找到合适的模型来建模。

2.怎样找到最大的

$$k = argmax_k p(attribute_j = k|AttributePredict_i, Image)$$

第一个问题，我们可以引入属性的相关矩阵A，利用A做辅助判断。根据这样的情况，我们可以做以下3件事情：

1.对原公式进行以下变换

$$p(attribute_j|image, AttributePredict_i) = p(attribute_j|attribute_k) * p(attribute_k|image, AttributePredict_i)$$

所以，我们有，

$$k = argmax_k p(attribute_j = k|AttributePredict_i, Image)$$
$$= argmax_k p(attribute_j|attribute_k) * p(attribute_k|image, AttributePredict_i)$$

$= argmax_k p(attribute_k | image, AttributePredict_i)(k = 1...N)$ (1)
（因为$p(attribute_j | attribute_k) = 1$）

2.为

$$p(attribute_k | image, AttributePredict_i) \quad (2)$$

建模。采用概率图方式，在此，**建立一个概率图**G ={ V，E}，其中$V : [v_i]$ $(i = 1...N)$ E：$[e_{ij} = a_{ij}]$ $(i = 1...N, j = 1...N)$

3.然后添加节点$v_{N+1} = Image$,添加边$e_{N+1,j} = p(AttributePredict_j | Image)$ $(j = 1...N)$

第二个问题，根据第一个问题的内容，采用概率图计算概率，其实在于设计特征函数。

在此采用的特征函数表达公式(2):

特征函数：$xAx + x * diag(AttributePredict) * x$

所以

$$p(attribute_k | image, AttributePredict_i) = \frac{e^{xAx+x*diag(AttributePredict)*x}}{\sum_x e^{xAx+x*diag(AttributePredict)*x}}$$

所以（1）公式可变为

$$argmax_k p(attribute_k | image, AttributePredict_i)(k = 1...N) = argmax_x(xAx + x * diag(AttributePredict) * x) \quad (3)$$
$$s.t. x_i = 1 \ or \ 0 (i = 1...N)$$
$$\sum_i x_i = 1 \ if \ x_i \ is \ the \ attribute \ group$$

这是个整数规划问题，np-hard问题，所以**我们可以通过穷举所有可能的x，并选取目标函数最大的结果。**

3.使用神经网络进行端到端的训练

假设$g(x) = argmax_x(xAx + x * diag(AttributePredict) * x)$

所以，

我们可以**用一个多层神经网络**$f(x, A, AttributePredict)$来拟合它

即$f(x, A, AttributePredict) = argmax_x(xAx + x * diag(AttributePredict) * x)$

损失函数为

$$loss = -(xAx + \lambda_1 * x * diag(AttributePredict) * x + \lambda_2 * \sum_j (1 - \sum_i x_i)^2 + \lambda_3 * softmax\_loss(x, label)) \quad (4)$$

所以整个结构网络图如下所示

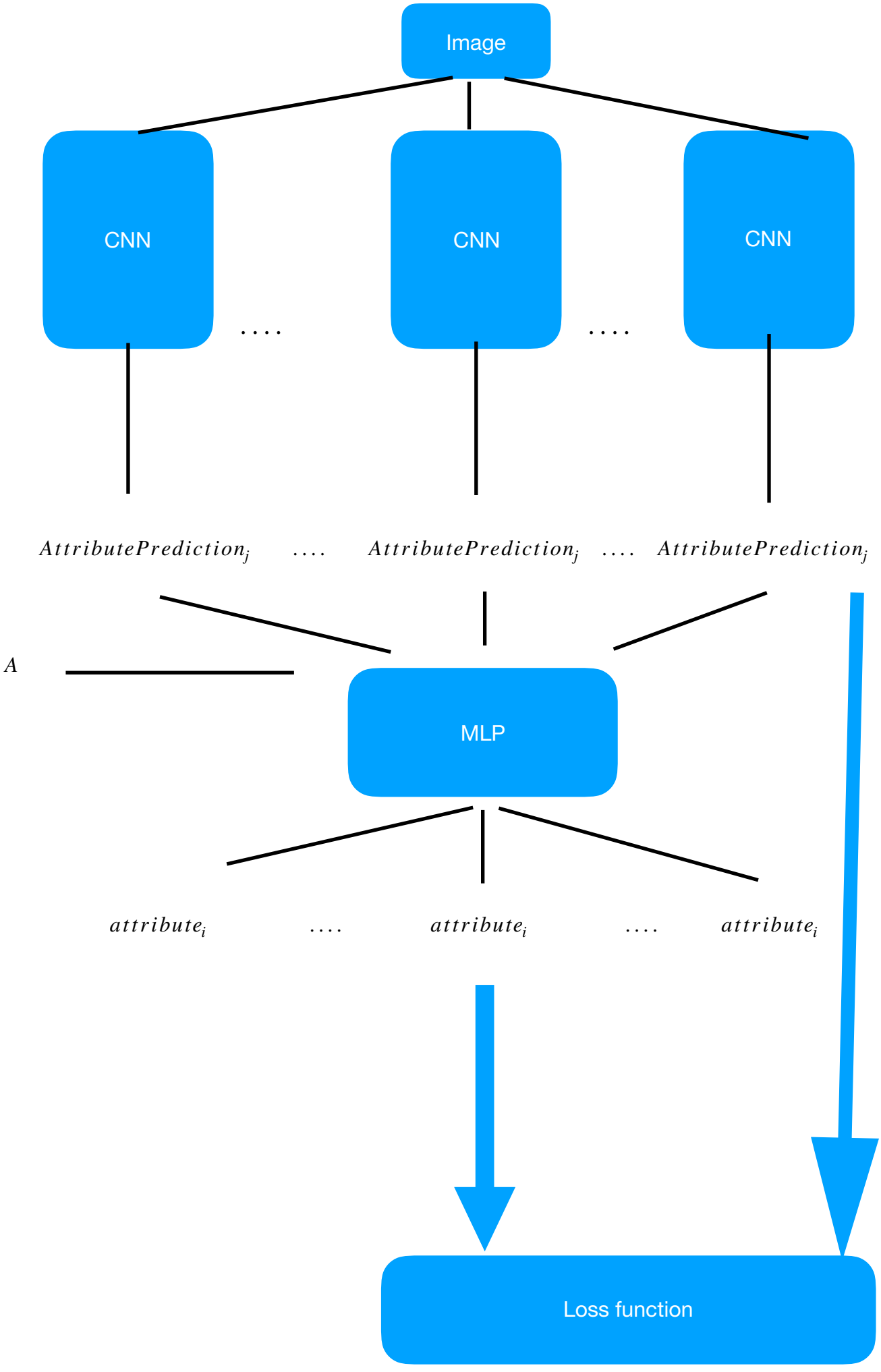
Image
CNN
CNN
CNN
$AttributePrediction_j$ .... $AttributePrediction_j$ .... $AttributePrediction_j$
A
MLP
$attribute_i$ .... $attribute_i$ .... $attribute_i$
Loss function

4.实验设计

根据我们设计的算法，我们有三个要回答的问题。
1.使用属性检测+推理的方法是否比仅用属性检测的准确率高，若不高，为什么，有什么其他的好处。
2.使用概率图的方法建模是否是准确描述了推理过程。
3.使用神经网络拟合优化目标函数的方式合不合理，能否达到2的效果。
4.端到端的优化有没有优势（即精度是否比没有端到端的精度高）。

针对以上的问题，我们可以做以下的实验，收集相应的数据：
1.训练CNN，使其在属性检测上达到最好的结果，作为我们的基线模型。
2.在1的基础上加上概率图模块，对属性进行推理，并收集推理后的结果，对比1中的结果，可回答问题1，2。
3.建立概率图神经网络，并优化概率图模块，与实验2比较，可回答问题3。
4.在3的基础上优化概率图+CNN，与1,2,3实验对比，可回答问题4。

针对以上实验，我们可以设计以下实验流程:（5.20启动）
1.我们统一在python，tensorflow的环境下做实验。
2.准备服装数据集，图片的标签由已有的分类器得到。收集神经网络，概率图模块pyhon库。（5.23前）
3.对于实验1，使用一个或多个现有的CNN网络得到基线准确度。（由一位组员负责，上交实验结果表，代码和模型）(可跟小朋拿数据）（6.1号前）
4.在流程3的模型、代码基础上，根据《算法一》利用python概率图模块添加概率图，并完成实验2（由一位组员负责，上交实验结果表，代码）(6.10前）
5.同时，根据《算法二》，在tensorflow平台上设计概率图神经网络，在流程3的基础上，添加代码，并训练，统计实验结果。（由一位组员负责，上交实验结果表，代码和模型。）(6.10前）
6.在5的流程基础上，用端到端的方法训练，并统计结果。（由流程5组员负责，上交实验结果。）（6.15前）
7.汇总以上流程实验结果和代码，纂写文章，上传代码。(6.20前)

算法一：使用概率图的方法推理属性。

Algorithm 1: Obtaining $Attribute_j$ by probability graph

Input: $AttributePredict_j$ are the results of attribute detector by CNN.A is the coefficient matrix of all attribute compute offline from the label where $a_{i,i} = 0$

Output：$Attribute_j$ are the result of probability graph

Step1: build a graph $G = \{V, E\}$where $V = \{v_i\}$where $i = 1...N, E = A$

Step1.1:using a matrix G_{n*n} to represent the graph and let $G_{n*n} = A$

Step2: Add a vertex V_{N+1} to $G = \{V, E\}$,and connect the vertex to $Attribute_j$.

Step2.1: update G_{n*n} to $G_{n+1*n+1}$,where $g_{n+1,j} = g_{j,n+1} = AttributePredict_j$ and $g_{n+1,n+1} = 0$.

Step3:assume $Attribute_j = x$,then optimize with (4)

Step3.1:try all x satisfying the constrain ,compute loss and compare until find the x has minimum loss.

算法二：使用神经网络方法模拟概率图推理并设置相应损失函数并训练。

Algorithm 2: Training MLP

Input: A is the coefficient matrix of all attribute compute offline from the label where $a_{i,i} = 0$

Output：$Attribute_j$ are the result of MLP

For batch = 1,...,K **do**
    For image =1,...,P **do**
        use CNN attribute detector to extract $AttributePredict_j$
        concatenate [A,$AttributePredict_j$] as MLP input
        forward computing MLP
        compute the loss according to (4)
        loss_batch += loss
    Back propagation with gradient decent()
    end
end