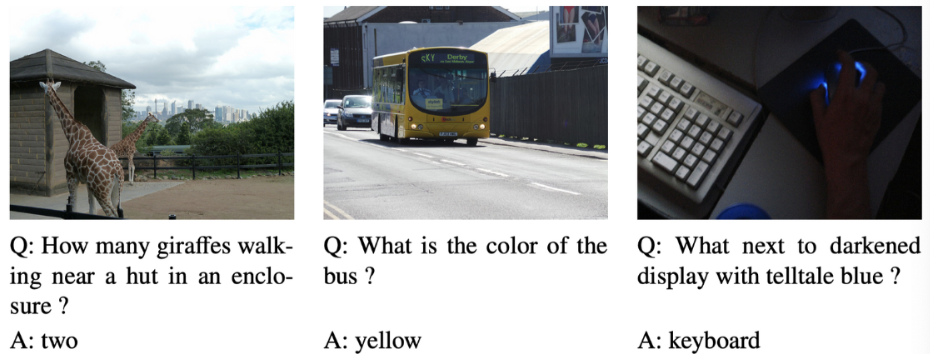
**Visual Question Answering**

|  |  |  |
| --- | --- | --- |
| **学号** | **专业班级** | **姓名** |
| 3190104547 | 计科1902 | 苗昊田 |
| 3190102480 | 计科1903 | 王轶楷 |
| 3190103047 | 混合1902 | 黄志伟 |
| 3190103383 | 计科1902 | 张耀辉 |

## Project Introduction

* 1. 选题

VQA(Visual Question Answering),即视觉问题回答任务，VQA系统将一个图像和一个关于图像的自由而开放式自然语言的问题作为输入，并生成一个自然语言的答案作为输出。



VQA和其他CV任务的区别是，它回答的问题直到run time才确定，因此它更接近general image understanding。VQA比image caption更复杂，因为它往往需要图像以外的信息。另外VQA的答案往往更简短，因此也更容易和ground truth比较。VQA的难点在于图像是高维信息，缺少语法规则和结构，也不能直接使用句法分析、正则表达式等NLP方法。此外，图像捕捉到的信息更接近现实世界，而语言本身就是一种抽象。

* 1. 工作简介

在本任务中，我们使用VQAv2.0数据集的子集进行训练与测试，整体工作包括：数据处理、模型构建、训练与测试。在数据处理中，我们从网站下载数据集，提取train、val、test三部分的image、question、annotation，并将它们进行一一对应，保存在mindrecord中。在模型构建中，我们构建了简单的embedding模型、Clip预训练模型对image和question进行嵌入，并进行了点乘、MFB模型、attention机制等多种方式处理嵌入向量，得到输出。在训练与测试中，我们进行了10个epoch的训练，并调整了超参数如学习率、dropout率等以获得最佳效果。最后，我们记录了准确率以对比实验效果。

* 1. 开发环境及系统运行要求

开发环境：ModelArt Ascend Notebook

系统运行要求：Python3.7, Mindspore1.3.0

## Technical Details

2.1 数据集简介

实验中使用的VQA2.0数据集主要分为image, question以及annotation三个部分，项目要求的数据集其数据量为：train:44375, validation:21435, test:21435。对于一张图片会有若干个问题，对于每一个问题会在annotation中有对应的解答，标明其answer\_type，以及question\_id，给出10个answer,使用answer\_id:1-10进行标号，并给出每个answer的confidence。而test数据集没有对应的annotation文件。

Question文件的JSON格式如下所示：

1. {"image\_id": 131074,
2. "question": "Are the walls done in a summery color?",
3. "question\_id": 131074002}

Annotation文件的JSON格式如下所示：

1. {"answer\_type": "other", *#答案类型*
2. "multiple\_choice\_answer": "looking in fridge", *#答案*
3. "answers": [...], *#包含10个answer的列表*
4. "image\_id": 425226, *#对应的img\_id*
5. "question\_type": "what is the", *#问题类型*
6. "question\_id": 425226003 *#对应question\_id}*
7. {"answer": "looking in fridge",
8. "answer\_confidence": "yes",
9. "answer\_id": 1}

值得注意的是，在给出的数据集中，问题和答案，问题和图片并不是一一对应的，需要人为地去将他对齐，详细代码可查看我们的preprocess\_data.ipynb，其中对不存在于对应image中的question，我们进行了移除。

为了保证question\_id为顺序递增，而非像数据集中离散的，我们对json文件进行了遍历，将答案作为索引，label序号作为键值，创建了两个索引，并创建了trainval\_questionid2label，test\_questionid2label两个字典用于建立question\_id和label序号的映射，下面给出了部分建立train\_val映射关系的伪代码：

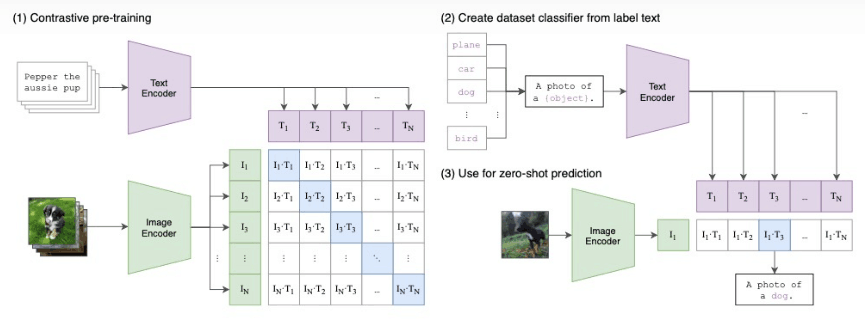
1. *# create labels*
2. label\_dict = {}
3. trainval\_questionid2label = {}
4. i = 0
5. for item in trainval\_json:
6. if(item not in label\_dict):
7. add the item into the label\_dict
8. i+=1
9. add the item into the trainval\_questionid2label, with question\_id as key

为了使得我们的模型能够顺利地被用于预测和评估，我们需要根据华为云的要求创建对应的mindrecord文件，我们的schema\_json文件共有三个成员，分别为问题，图片的embedding嵌入，以及对应的label：

1. schema\_json = {
2. "ques\_emb": {"type": "float32", "shape": [512]},
3. "img\_emb": {"type": "float32", "shape": [512]},
4. "label": {"type": "int32"},
5. }

2.2 CLIP简介

CLIP（Contrastive Language-Image Pre-Training，以下简称 CLIP）模型是 OpenAI 在 2021 年初发布的用于匹配图像和文本的预训练神经网络模型：



该模型直接使用大量的互联网数据进行预训练，在很多任务表现上达到了目前最佳表现（SOTA）。模型采用对比学习，预测N×N对图文数据，将图片分类任务转换成匹配任务：

1. 双流，2个encoder分别处理文本和图片数据，text encoder使用Transformer，image encoder用了2种模型，ResNet和Vision Transformer(ViT)：

a. 5种ResNet：ResNet-50, ResNet-101, EfficientNet-style的ResNet，包括RN50x4, RN50x16, RN50x64；

b. 3种ViT：ViT-B/32, ViT-B/16, ViT-L/14；

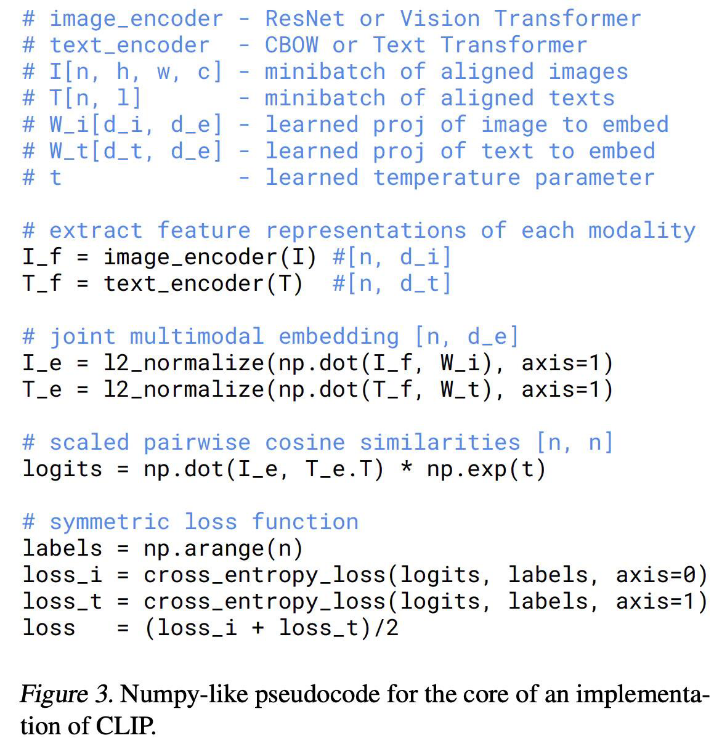
2. encoder representation直接线性投影到multi-modal embedding space；

3. 计算2模态之间的cosine similarity，让N个匹配的图文对相似度最大，不匹配的图文对相似度最小；

4. 对称的cross-entropy loss；

5. 数据增强：对resized图片进行random square crop。

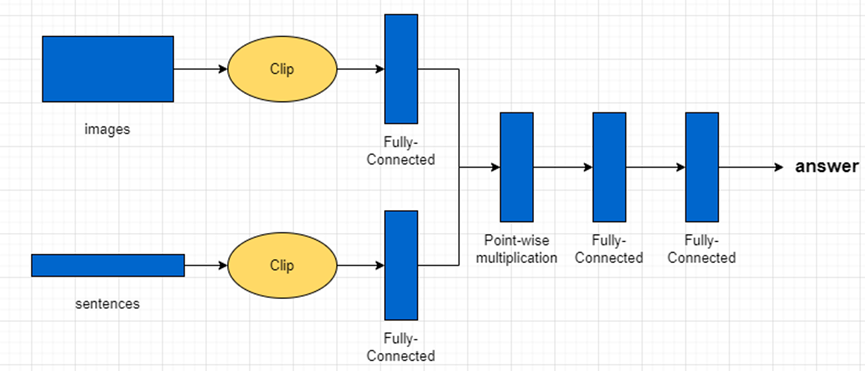
其伪代码如下所示：



使用clip我们可以轻松地提取出image feature，并将其以及question等text转换为embedding，之后我们只需将这两个特征向量融合便可搭建我们的预测网络。

2.3 特征融合与预测网络(BaseLine)

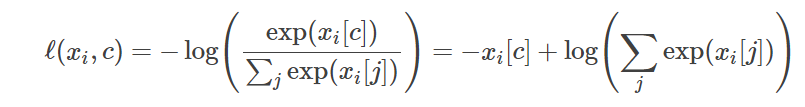
在baseline模型中(如下图所示)，我们选用Clip预训练模型来生成图片的嵌入向量和问题的嵌入向量，然后将二者通过一个全连接层进行映射，之后将两个嵌入向量进行点积操作，并通过两个全连接层进行输出：



其构建算法如下：

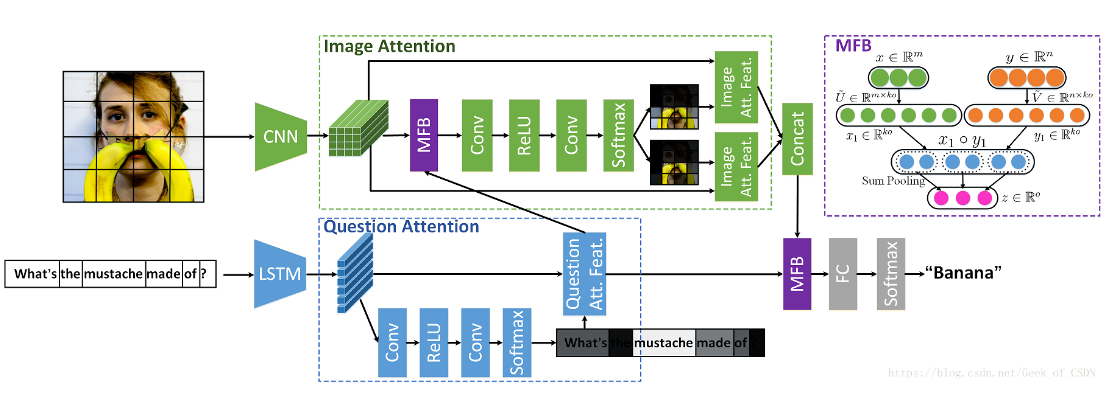
|  |
| --- |
| Baseline Model |
| 1. 利用clip预处理模型做图像嵌入和问题句嵌入，得到两个嵌入向量img\_emb和ques\_emb。 2. 将img\_emb和ques\_emb经过Fully-Connected层和relu激活后得到两个向量：img\_out = relu(nn.Dense(img\_emb), ques\_out = relu(nn.Dense(ques\_emb)。 3. 将img\_out和ques\_emb做点乘操作，并经过两层全连接层进行输出：output = nn.Dense(nn.Dense(img\_out \* ques\_emb)。 4. 利用Softmax交叉熵（nn.SoftmaxCrossEntropyWithLogits）计算损失。 |

在VQA\_baseline中用到的函数说明如下：

* nn.dense(): 这是Mindspore 支持的全连接层API。其接受的两个参数分别是输入向量的维度和输出向量的维度。
* nn.dropout()：这是Mindspore支持的随机丢弃层API。随机丢弃是一种正则化手段，该算子根据丢弃概率 1−keep\_prob1−keep\_prob ，在训练过程中随机将一些神经元输出设置为0，通过阻止神经元节点间的相关性来减少过拟合，在推理过程中，此层返回与 输入维度 相同的Tensor。
* nn.Softmax()：这是Mindspore支持的激活函数API。它是二分类函数 [mindspore.nn.Sigmoid](https://www.mindspore.cn/docs/zh-CN/master/api_python/nn/mindspore.nn.Sigmoid.html#mindspore.nn.Sigmoid) 在多分类上的推广，目的是将多分类的结果以概率的形式展现出来。对输入Tensor在轴 axis 上的元素计算其指数函数值，然后归一化到[0, 1]范围，总和为1。
* nn.ReLU()：这是Mindspore支持的修正线性单元激活函数API。逐元素求max(x, 0) 。特别说明，负数输出值会被修改为0，正数输出不受影响。
* nn.SoftmaxCrossEntrophyWithLogits：这是Mindspore支持的损失函数API。用于计算预测值与真实值之间的交叉熵。使用交叉熵损失函数计算出输入概率（使用softmax函数计算）和真实值之间的误差。对于每个实例 xi ，i的范围为0到N-1，则可得损失为：其中xi是一维的Tensor，cc为one-hot中等于1的位置。

2.4 MFB implementation

我们参考了Multi-modal Factorized Bilinear Pooling with Co-Attention Learning for Visual Question Answering这篇文章中的模型架构：



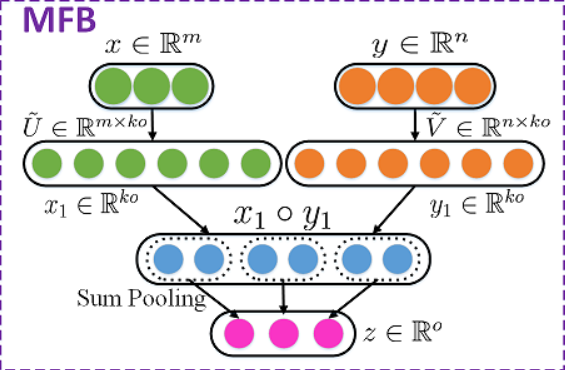
其中提出的MFB(Multi-modal Factorized Bilinear Pooling)假设,就是假设x是图片的特征向量,y属于问题的特征向量。那么bilinear pooling就定义为

上面出现的就是一个投影矩阵，是bilinear模型的输出。上面公式里面的W里面包含了偏置项，所以没有单独写出。需要通过训练来获得相应o维的输出z，但是bilinear pooling有个问题就是因为引入了太多参数，可能会导致运算量过大或过拟合。于是作者使用了本来在uni-model data上的矩阵分解将上面的公式变成了如下(就是将投影矩阵分解成2个低rank的矩阵)：

K就是和的维度，就是Hadmard积，或者另个向量在元素层面上一一对应的乘积，就是全1向量。

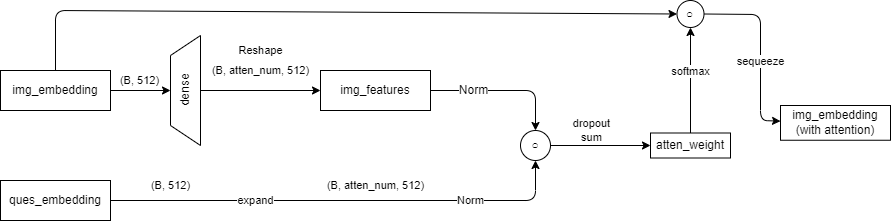
为了得到上面公式的输出向量,y要学习的是两个3阶的向量U,V对应的权重。可以在不影响鲁棒性的前提下将U和V通过简单的reshape直接变形成2维的矩阵和,这样上一条公式就可以重新写成：

上面这条公式里面的SumPooling(x,k)表示用一个一维大小是k的没有重叠的窗口在输入的x上面进行sum pooling。这个模型即被作者称为MFB,上面的内容可以用如下的图片表示



2.5 Attention implementation

我们还针对图像处理开发了一种方法，将图像通过Clip处理后通过一个输出大小为num\*emb\_size的全连接层，然后将其分为num个embedding向量，与ques\_emb做点乘操作，模拟注意力机制，然后将num个结果相加得到最终的预测值。其模型结构如下：



其构建算法如下：

|  |
| --- |
| Attention Model |
| 1. 利用clip预处理模型做图像嵌入和问题句嵌入，得到两个嵌入向量img\_emb和ques\_emb。  2. 通过一个全连接层将img\_emb扩展到num\*img\_embsize大小，将ques\_emb简单repeat 到num\*ques\_embsize大小。  3. 将num个img\_emb和num个ques\_emb进行点乘操作，然后经过softmax层得到输出out  4. 将out的num个结果相加得到预测值。 |

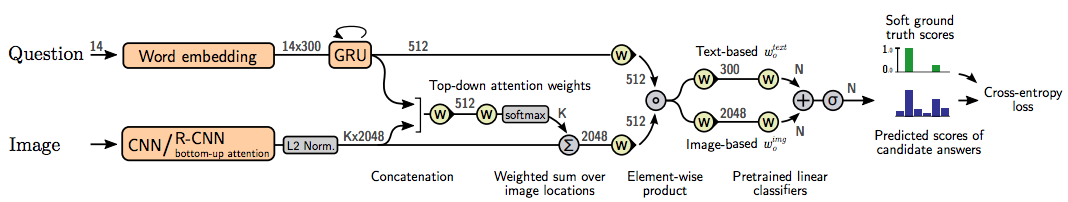
2.6 Nonlinear Layer Implement

参考自Tips and Tricks for Visual Question Answering: Learnings from the 2017 Challenge，其中有一条利用Nonlinear Layer代替单纯的dense来变换向量维度，因为某些特征可能是不是线性变换就可以对齐的。非线性层的数学公式如下：

使用中利用其替代dense进行向量维度的变换即可。

2.7 Faster\_rcnn implementation

根据Tips and tricks for visual question answering: Learnings from the 2017 challenge文章的介绍：

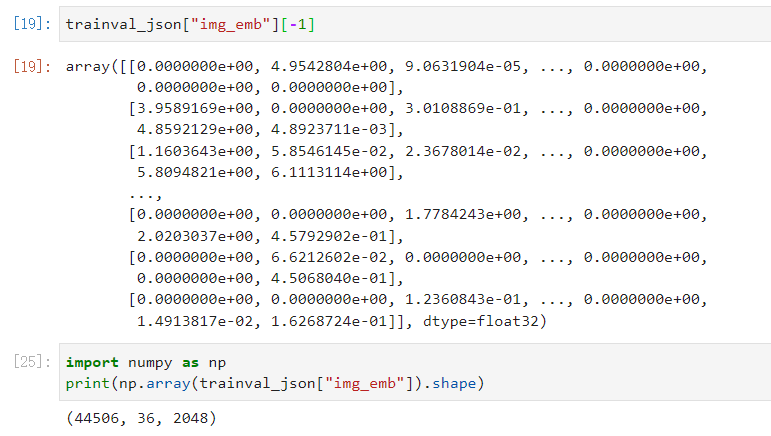


对于Image部分我们可以采用R-CNN对图像进行特征提取，这里使用了trainval\_resnet101\_faster\_rcnn\_genome\_36得到了经过处理后的train,val数据集的bottom-up attention特征，tsv文件中的数据格式如下所示：

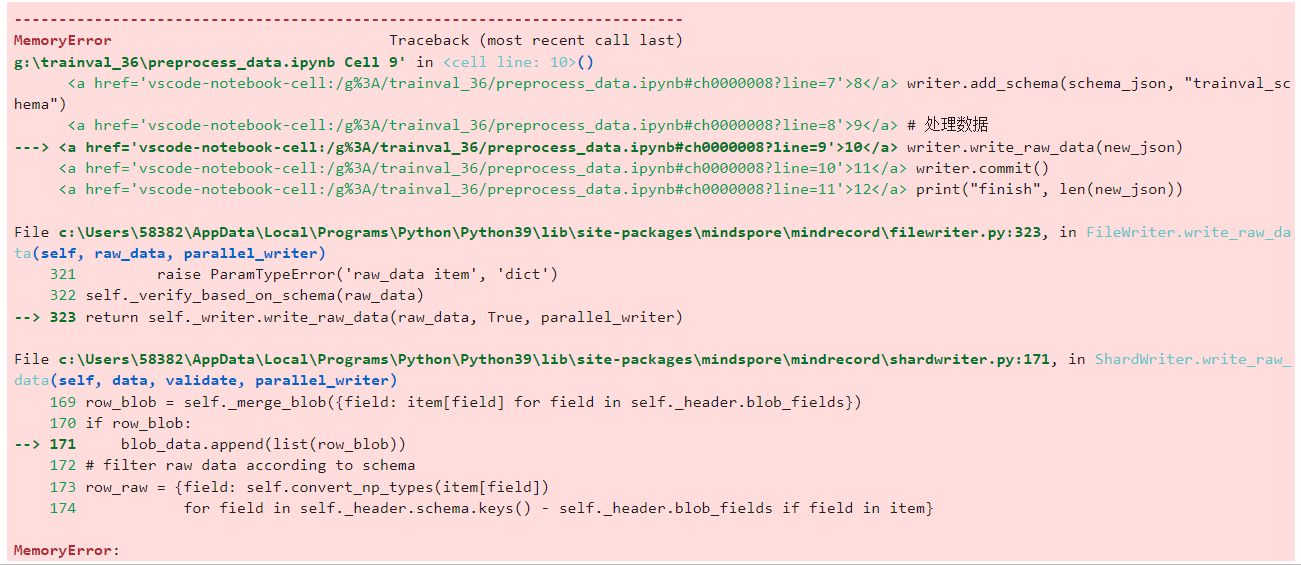
1. FIELDNAMES = ['image\_id', 'image\_w','image\_h','num\_boxes', 'boxes', 'features']

其中item[‘boxes’]为对应检测框的位置信息 x,y,w,h; item[‘features’]为对应检测框， item[‘num\_boxes’]为该张图片对应的检测框数量。

得到的特征向量如下图所示：



但是对于这种方案，我们在尝试生成mindrecord文件以进行预测时遇到了一些障碍，由于我们采用的image feature是36×2048维度的，因此在生成mindrecord文件时出现内存不够用的情况，无论是在本地运行抑或是在华为云的Ascend平台上运行：



同时，如果我们需要将image特征的embedding同样存在JSON文件中，需要将np.array转换为list以支持可序列化，我们在PC端和华为云端都进行了尝试，但最后同样都会得到Memory Error,并且由于华为云端不支持较大的文件mox，trainval的特征读入我们也是在PC端完成的，使用wget命令在华为云获取该文件最终也会卡死。

## Experiment Results

**Baseline model:**



根据同模型结果对比，可以发现，对于baseline模型，当hidden\_dim较大时效果较好，而learning\_rate介于0.0001到0.001之间warm\_up时效果最好，准确率可以达到28.8%左右。

**MFB model:**



根据同模型结果对比，可以发现，对于MFB模型，当learning\_rate为0.005时效果较好，准确率accuracy可以达到27.4%。

**Attention model:**



根据同模型结果对比，可以发现，对于Attention模型，当output\_dim较大时效果较好，而learning\_rate介于0.0001到0.001之间warm\_up时效果最好，同时，模型效果也随着attention\_num在一定程度内的增大而变好，准确率可以达到28.2%左右。

根据以上结果，可以看出参数learning\_rate对结果有一定影响，同时对于attention模型，attention\_num对结果的影响也较大。三种模型中，baseline模型和Attention模型效果相差不大，而MFB相对于baseline模型效果稍差一点，但三种模型的准确率都在相近的范围内，集中在28%左右。

## References:

[1] VQA: Visual Question Answering (ICCV 2015).

[2] Tips and tricks for visual question answering: Learnings from the 2017 challenge (CVPR 2018).

[3] Zhou Yu, Jun Yu, Jianping Fan, Dacheng Tao, 2017. Multi-modal Factorized Bilinear Pooling with Co-Attention Learning for Visual Question Answering.

[4] Tips and Tricks for Visual Question Answering: Learnings from the 2017 Challenge 论文解读: <https://blog.csdn.net/u014248127/article/details/86091518?spm=1001.2014.3001.5502>

[5] Multi-modal Factorized Bilinear Pooling with Co-Attention Learning for Visual Question

Answering笔记: <https://blog.csdn.net/Geek_of_CSDN/article/details/81328487>