Implicit Semantic Data Augmentation for Deep Networks

文末有关于MetaSAug: Meta Semantic Augmentation for Long-Tailed Visual Recognition的分析

Background

看标题就可以知道这篇文章讲得是隐式的语义上的数据增强：

- 隐式和显式：显示数据增强就是直接的生成样本来增强，比如说GAN这种generate样本的数据增强；隐式数据增强就是没有实际产生样本但是带来了数据增强的效果，一般常见比如说有调整loss。
    - 我认为隐式的增强最大的好处就是降低了计算cost，也比较容易部署在各个网络中。
- 语义：非语义增强，举个例子，对于图像数据增强，torch内部自带的有随机crop、翻转这种，这些就是和图片内容不相关的数据增强；而从语义出发自然是进一步考虑图像内容做数据增强，比如说在人脸识别中替换背景。论文也提到：
    - ```
      Recent work has shown that data augmentation can be more powerful if (class identity preserving) semantic transformations are allowed
      ```
    - 我认为语义数据增强的优点在于它扩大了数据增强的范围，还是拿人脸识别说，在非语义增强中我们可能只能用随机crop这些增强，但是在语义增强中我们可以在保持人脸identity不变的同时改变背景、改变表情、改变发色等等，这极大的扩大了数据增强的范围。

Related work

论文提了下面三个方面，其中第一类数据增强基本上都是显示的（也有隐式的），第二类改loss就基本都是隐式的了，第三类是从语义上出发。

- Data augmentation: 这类基本都是显式的方法，里面我觉得主要包括语义还是非语义两大类。
    - 非语义的有代表性的是AutoAugment（就是自动选一个合适的增强方法） 和 learning with marginalized corrupted features（论文说是隐式的 没看过不是很了解）；
    - 语义的有代表性的就是GAN了，比如DAGAN。这类方法上面有说了，显式的，计算cost比较大。
- Robust loss function: 这里面都是改loss，可以算在隐式类别里面，我了解的有Focal loss、调整margin出发的loss（这两类loss都是用在长尾问题、类别不均衡等问题里面的）。
- Semantic transformations in deep feature space：这个是本篇paper主要follow的方向，主要包含了deep feature interpolation（通过对预训练产生的高阶特征做简单扰动实现数据增强）、Variational Autoencoder(VAE)和GAN （找到图像对应的隐藏表征）等方法。这些方法应该都还是显式的。

Motivation

论文提到出发点是：
      ```

- ```
  Our work is motivated by the fact that high-level representations learned by deep convolutional networks can potentially capture abstractions with semantics. In fact, translating deep features along certain directions is shown to be corresponding to performing meaningful semantic transformations on the input images.
  ```

- ```
  Our approach is motivated by the intriguing observation made by recent work showing that the features deep in a network are usually linearized.
  ```

- 也就是说通过深度网络最后得到的高维特征embedding其实是能抽象代表图像的语义层面上的特征的，同时这个高位特征往往是线性化的，改变最后这个高位embedding的线性化方向就代表着修改了输入图片的语义特征。

从上面思路出发，论文的切入点是：

- 如果直接去随机扰动最后的这个embedding来获取数据增强会得到很多没用的新数据，同时如果找人去标注出其中哪些是有用的语义增强数据又会很耗费人力。
  - 从有效性出发：为了确保最后对高维embedding的修改能确实带来有用的语义增强，论文提出了一个计算每个类中的embedding的协方差阵，依据这个获取合适的embedding的修改方向。这其中协方差阵其实就是衡量了同一个类中，embedding不同维度所表征的特征间的关系，这也是语义的来源。
  - 从效率出发：论文是直接把上面的思路进一步抽象成了CE loss的改进版。原来修改embbding来显式的做语义增强计算cost比较大，改成隐式增强后计算cost低且容易部署。

Method

从代码部署上来看，论文提出的方法其实就包含两部分：

- Online estimation of class-conditional covariance matrices：第一部分就是利用类内embedding的协方差阵，找到embedding的不同维度特征间的关系，利用这个类内协方差阵，可以确定一个包含合适语义数据增强方向的数据分布。
- Optimization with a robust loss function：第二部分就是把上面这个显式获取语义增强的过程抽象成隐式数据增强，即提出一个改进版的 CE loss。

Semantic Transformations in Deep Feature Space

针对上面的第一部分，对于某一类别y_i，首先计算出其类间的高维embedding的协方差阵$\sum_{y_i}$，然后构造正态分布$N(0, \lambda \sum_{y_i})$,从这个正态分布中随机抽取向量作为对高维embedding的修改。换句话收，如果要对 y_i中的某一样本做语义增强，如果它的高维embedding是a_i，那么该样本语义增强后的数据分布为$N(a_i, \lambda \sum_{y_i})$。

这个第一部分实际上只需获得每个类别的协方差阵$\sum_{y_i}$即可，因为前面也说到在第二部分时候已经无需显式生成。获取协方差阵$\sum_{y_i}$是通过一个动态机制实现的（其实就是和sgd里面计算momentum那样子差不多的，这里直接贴图展示一下）：

Dynamic estimation of covariance matrices. During the training process using $\overline{\mathcal{L}}_\infty$, covariance matrices are estimated by:

$$\boldsymbol{\mu}_j^{(t)} = \frac{n_j^{(t-1)}\boldsymbol{\mu}_j^{(t-1)} + m_j^{(t)}\boldsymbol{\mu'}_j^{(t)}}{n_j^{(t-1)} + m_j^{(t)}}, \tag{9}$$

$$\Sigma_j^{(t)} = \frac{n_j^{(t-1)}\Sigma_j^{(t-1)} + m_j^{(t)}\Sigma'_j{}^{(t)}}{n_j^{(t-1)} + m_j^{(t)}} + \frac{n_j^{(t-1)}m_j^{(t)}(\boldsymbol{\mu}_j^{(t-1)} - \boldsymbol{\mu'}_j^{(t)})(\boldsymbol{\mu}_j^{(t-1)} - \boldsymbol{\mu'}_j^{(t)})^T}{(n_j^{(t-1)} + m_j^{(t)})^2}, \tag{10}$$

$$n_j^{(t)} = n_j^{(t-1)} + m_j^{(t)} \tag{11}$$

where $\boldsymbol{\mu}_j^{(t)}$ and $\Sigma_j^{(t)}$ are the estimates of average values and covariance matrices of the features of j^{th} class at t^{th} step. $\boldsymbol{\mu'}_j^{(t)}$ and $\Sigma'_j{}^{(t)}$ are the average values and covariance matrices of the features of j^{th} class in t^{th} mini-batch. $n_j^{(t)}$ denotes the total number of training samples belonging to j^{th} class in all t mini-batches, and $m_j^{(t)}$ denotes the number of training samples belonging to j^{th} class only in t^{th} mini-batch.

对于上面这种动态机制，我暂时没在论文看到初始的协方差阵怎么初始化的，但是毫无疑问最开始的几个epoch肯定没有后面的epoch来得稳定，所以上面公式里的$\lambda = (t/T)\lambda_0$ 其实就是用来缓解这个问题的，这里的λ_0 就是一个需要通过验证集确定的超参。

Implicit Semantic Data Augmentation (ISDA)

- conclusion:

针对第二部分，在上面第一部分的基础上，已经获得了每个类别的$\sum_{y_i}$，这一步就是要把显式变隐式，通过引入$\sum_{y_i}$改loss实现。

从最简单直接的显式增强想法出发，我们有了第一部分的基础，假设总共有N 个样本，对于任意一个样本a_i，我们从$N(a_i, \lambda\sum_{y_i})$可以任意取出无数个对应的增强数据样本$a_i$，不妨假设为M个，这样子的显式增强后的CE loss为：

$$\mathcal{L}_M(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{M}\sum_{k=1}^{M} -log(\frac{e^{\boldsymbol{w}_{y_i}^T\boldsymbol{a}_i^k + b_{y_i}}}{\sum_{j=1}^{C} e^{\boldsymbol{w}_j^T\boldsymbol{a}_i^k + b_j}}),$$

显然，如果M变得大一些，计算损失会特别大。那么在这里论文给出了改进的loss：

$$\frac{1}{N}\sum_{i=1}^{N} -log(\frac{e^{\boldsymbol{w}_{y_i}^T\boldsymbol{a}_i + b_{y_i}}}{\sum_{j=1}^{C} e^{\boldsymbol{w}_j^T\boldsymbol{a}_i + b_j + \frac{\lambda}{2}(\boldsymbol{w}_j^T - \boldsymbol{w}_{y_i}^T)\Sigma_{y_i}(\boldsymbol{w}_j - \boldsymbol{w}_{y_i})}}) \triangleq \overline{\mathcal{L}}_\infty$$

这个loss可以看出来，相较于最开始的CE loss, 其实就是分母多了一项关于$\sum_{y_i}$的，且没有新增其它参数，十分容易部署，下面说一下是怎么得出来的。

- proof

首先，看一下显式增强后的 CE loss，因为M 变大影响计算cost， 那就假设 M 为无穷，这样由于M个新样本都是从$N(a_i, \lambda\sum_{y_i})$随机采样出来的，所以后面那部分其实就是均值：

$$\mathcal{L}_\infty(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\Theta}|\boldsymbol{\Sigma}) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{E}_{\tilde{\boldsymbol{a}}_i}[-log(\frac{e^{\boldsymbol{w}_{y_i}^T\tilde{\boldsymbol{a}}_i + b_{y_i}}}{\sum_{j=1}^{C} e^{\boldsymbol{w}_j^T\tilde{\boldsymbol{a}}_i + b_j}})].$$

很自然地，根据詹森不等式 $E(log(x)) <= log(E(x))$，可以进一步处理成：

$$\frac{1}{N}\sum_{i=1}^{N} log(\sum_{j=1}^{C} \mathrm{E}_{\tilde{\boldsymbol{a}}_i}[e^{(\boldsymbol{w}_j^T-\boldsymbol{w}_{y_i}^T)\tilde{\boldsymbol{a}}_i+(b_j-b_{y_i})}])$$

接着要把均值$E(\cdot)$换成可以计算的，引入$\sum_{y_i}$，这一步通过矩母函数地性质实现，考虑到我们地新样本是从$N(a_i, \lambda\sum_{y_i})$中sample出来的，那么就可以用下面的公式进一步处理：

$$\mathrm{E}[e^{tX}] = e^{t\mu+\frac{1}{2}\sigma^2t^2},\ \ X\sim\mathcal{N}(\mu,\sigma^2)$$

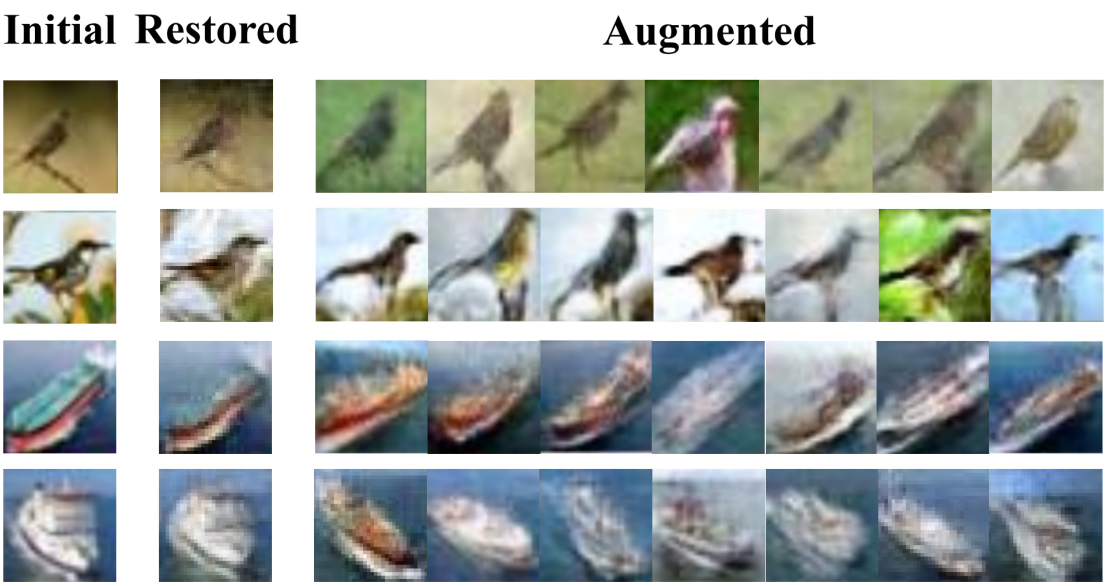
$$(\boldsymbol{w}_j^T-\boldsymbol{w}_{y_i}^T)\tilde{\boldsymbol{a}}_i+(b_j-b_{y_i}) \sim \mathcal{N}\left((\boldsymbol{w}_j^T-\boldsymbol{w}_{y_i}^T)\boldsymbol{a}_i+(b_j-b_{y_i}), \lambda(\boldsymbol{w}_j^T-\boldsymbol{w}_{y_i}^T)\Sigma_{y_i}(\boldsymbol{w}_j-\boldsymbol{w}_{y_i})\right)$$

处理完带回去就是：

$$\mathcal{L}_\infty(\boldsymbol{W},\boldsymbol{b},\boldsymbol{\Theta}|\boldsymbol{\Sigma}) \le \frac{1}{N}\sum_{i=1}^{N} -log(\frac{e^{\boldsymbol{w}_{y_i}^T\boldsymbol{a}_i+b_{y_i}}}{\sum_{j=1}^{C} e^{\boldsymbol{w}_j^T\boldsymbol{a}_i+b_j+\frac{\lambda}{2}(\boldsymbol{w}_j^T-\boldsymbol{w}_{y_i}^T)\Sigma_{y_i}(\boldsymbol{w}_j-\boldsymbol{w}_{y_i})}}) \triangleq \overline{\mathcal{L}}_\infty$$

到目前为止，上面的proof证明出了上面这个结论，那回到最开始，如果是显示增强的话，我们是要最小化不等式左边，不等式右边其实是一个上界，那么如果最小化右侧这个上界，不等式左边自然也就最小化了，也就是实现了显示增强的一个效果。

Experiments



看最后的可视化结果，可以看到比较明显的语义特征转变是背景，比如背景颜色这种。在几个数据集上实验效果都不错，这里就不放出来了。

我还没有复现这篇paper，这篇paper的motivation比较重要，代码思路其实到最后不复杂，就是用动态的机制根据epoch的前后去计算每个类别的类内特征协方差阵，再放入改进的ISDA loss中就好了。

My Opinion

这篇文章的motivation比较好，第一个motivation是，文章里说的那个线性化的deep feature其实就是deep learning model提取出来的高维线性空间，回想当时高代中提出特征空间这个概念本身就是为了把所有事物抽象到一个共同的概念（也就是特征空间）一起讨论，deep learning model其实就是在做这样一个映射，把一堆图像映射、抽象成最后的高维线性空间。对于最后那个高维的线性空间，利用这个空间是图像特征的映射出的特征空间这一想法，自然我们修改最后的高维的线性空间来做数据增强。

第二个motivation是，修改高维特征空间时，不是随便修改，往哪个方向改、幅度多少都是要考虑的，不管怎么说，我们去做修改肯定是符合一定规律的，也就是修改出来的新的特征embedding应该是要符合一定的数据分布的。考虑到语义这一因素，不难想到一个类别中的样本的高维embedding中因该有很多是一样的，也有少部分是各自特有的，那这其实就是语义，要考虑不同特征之间的关系，协方差阵可以做到这一点，所以通过协方差阵引入语义。另一方面，基于上面的分析，既然一个类别中的样本有很多高维embedding的特征是一样的，我们自然想到就是以原来的样本的embedding做为新样本分布的均值。现在有一个均值和一个协方差阵，最简单直接的就是一个正态分布结合二者。

第二个motivation是把显式增强变成隐式增强，利用了一系列的理论证明，把M趋向无穷变成均值以消除这个实际的显式增强数量这一变量，最后推导出一个loss上界，这两点是证明中比较有想法的两步。

Others

后面21年应该也是同一个团队的，把ISDA这个技术用在了长尾分布问题里面，也就是MetaSAug: Meta Semantic Augmentation for Long-Tailed Visual Recognition这篇paper，在那里面可以看到单独按照这篇文章提出来的情况主要还是面向平衡数据集的，因为像在这里：

$$\mathcal{L}_M(\boldsymbol{W},\boldsymbol{b},\boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{M}\sum_{k=1}^{M} -log(\frac{e^{\boldsymbol{w}_{y_i}^T\boldsymbol{a}_i^k+b_{y_i}}}{\sum_{j=1}^{C}e^{\boldsymbol{w}_j^T\boldsymbol{a}_i^k+b_j}}),$$

默认所有样本都给它增强M个样本，这样在长尾问题里反而会加大头部类和尾部类之间的gap。MetaSAug就是用了一个meta learning和re-weighting的方法进行改进，在最后训练时还加入了一个两阶段的trick。re-weighting就是每个类别loss加权求和了一下；meta learning是指由于直接用原来ISDA的策略有风险，而ISDA策略主要通过协方差阵来影响loss，所以论文干脆就选了一个小的均衡的validation dataset来先求一下最优的协方差阵，而原本是直接动态计算出来了无需在这里先optimization一下，得到协方差阵后再放到整个数据集上按照re-weighting的方式去迭代。总体来说MetaSAug这篇paper感觉起来还是偏调整model的方向。