RLDM

Bingjie Yan

July 23, 2021

Learning Data Manipulation for Augmentation and Weighting

用强化学习的方法学习数据操作，进行数据增强与数据加权，这里同时也是模型和数据操作方法一起学习[1]

Zhiting Hu, Bowen Tan, Ruslan Salakhutdinov, Tom Mitchell, Eric P. Xing
Carnegie Mellon University, Petuum Inc

[1]这篇文章涉及到挺多强化学习和信息论中的内容，掌握不是很好，中间夹杂着一些我自己的理解，可能会有一些错误

Data 和 Reward 的等效性

输入 x，输出 y，模型 $p_\theta(y|x)$ 可以看做是一种 policy，y 被看做是 action，x 是 state，policy(model) 根据 state x 给出 action y
$R(x, y|\mathcal{D})$ 作为 Reward 函数，$\mathcal{D}$ 为 supervised 的训练数据，$p(x)$ 是根据经验 $\mathcal{D}$ 得到的一个数据分布。
策略优化函数 (RL-as-inference, MPO)

$$\mathcal{L}(q, \theta) = \mathbb{E}_{q(x,y)}[R(x, y|\mathcal{D})] - \alpha \mathrm{KL}(q(x, y) \parallel p(x)p_\theta(y|x)) + \beta \mathrm{H}(q) \tag{1}$$

$\mathrm{KL}(\cdot \parallel \cdot)$ is the Kullback–Leibler divergence KL 散度，$\mathrm{H}(\cdot)$ 是 Shannon entropy; 香农熵，均来自信息论
$\alpha, \beta > 0$ 是用于平衡权重的参数

KL 所谓 KL 散度，是指当某分布 $\mathrm{q(x)}$ 被用于近似 $\mathrm{p(x)}$ 时的信息损失。也就是说，$\mathrm{q(x)}$ 能在多大程度上表达 $\mathrm{p(x)}$ 所包含的信息，KL 散度越大，表达效果越差

$$D(p \parallel q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)} \tag{2}$$

在这里也就是让 $p(x)p_\theta(y|x)$ 与 $q(x,y)$ 接近，$q(x,y)$ 是真正的对应关系，$p_\theta(y|x)$ 是 policy 控制的对应关系，这个部分也就是想让 $p_\theta(y|x)$ 学习到真正的数据分布，也就是 dataset 中的分布（因为 dataset 中的分布通常认为是当前任务的分布）
香农熵也就是信息熵

$$\mathrm{H} = -\sum_{i=1}^{N} p(x_i) \cdot \log p(x_i) \tag{3}$$

是一个正则项，也是一个期望，这里是一个用于平滑的正则项

MPO

$$
\begin{aligned}
\log p_\theta(O=1) &= \log \int p(x,y)p(O=1|x,y)\mathrm{d}(x,y) \\
&\geqslant q(x,y)[\log p(O=1|x,y) + \log \tfrac{p(x,y)}{q(x,y)}]\mathrm{d}(x,y) \\
&= \mathbb{E}_{q(x,y)}[R(x,y|\mathcal{D})/Z] - \mathrm{KL}(q(x,y) \parallel p(x,y))
\end{aligned} \tag{4}
$$

后续的求解涉及到一个拉格朗日乘子法与 KKT 条件优化，后面推导出 E-step

$$
q'(x,y) \propto \exp\left\{\frac{\alpha \log p(x) p_\theta(y|x) + R(x,y|\mathcal{D})}{\alpha+\beta}\right\}/Z \tag{5}
$$

优化 θ

EM 优化，最大化期望 (Expectation Maximization)，这个就是用来不断迭代更新 θ 的方法

$$\begin{aligned} E-step: \quad & q'(x,y) = \exp\left\{\frac{\alpha \log p(x) p_\theta(y|x) + R(x,y|\mathcal{D})}{\alpha+\beta}\right\}/Z \\ M-step: \quad & \theta' = \arg\max_\theta \mathbb{E}_{q'(x,y)}[\log p_\theta(y|x)] \end{aligned} \tag{6}$$

这是一个不断迭代更新 $q(x,y)$ 和 θ 的过程，首先得到一个分布 $q(x,y)$，然后认为数据服从该分布，(x,y) 服从 $q'(x,y)$，最大化 $p_\theta(y|x)$ 对数似然估计的期望（这里选取对数似然估计就是要将每个元素的条件概率连成变成求和的形式方便计算），也就是从 q 中 sample 出一对数据，让这个模型预测出来的概率最大。这里 Z 是一个正则项

EM 优化

让 $\alpha \to 0, \beta = 1$，然后 Reward 长这个形式

$$R_\delta(x, y|\mathcal{D}) = \begin{cases} 1 & \text{if}(x, y) \in D \\ -\infty & \text{otherwise} \end{cases} \tag{7}$$

这样的话 M-step 就变成了

$$\theta' = \arg\max_\theta \mathbb{E}_{p(x)\exp\{R_\delta\}/Z}[\log p_\theta(y|x)] \tag{8}$$

也就是只有当 x, y 在数据集 $\mathcal{D}$ 中产生，这样就不能生成新的数据了，所有后面就要去优化 Reward，让 q 产生服从分布的新数据

优化 Reward

强化学习中基于梯度下降的求 Reward 的方法很多，这里比较相关的就是内部的 intrinsic reward 和 extrinsic reward，Reward 函数的参数为 ϕ，θ 是由 ϕ 进行更新的，所以 θ 可以看做是一个关于 ϕ 的函数，然后朝着让 reward 最大的方向做梯度上升

$$\theta' = \theta + \gamma \nabla_\theta \mathcal{L}^{ex+in}(\theta, \phi) \tag{9}$$

$\mathcal{L}^{ex+in}$ 就是内部和外部 reward 的和，更新 ϕ 的时候

$$\phi' = \phi + \gamma \nabla_\phi \mathcal{L}^{ex}(\theta'(\phi)) \tag{10}$$

只用到了外部的 reward，自己的理解：
这所谓的内部 reward 和外部 reward 其实就是在训练集上的和验证集上的，有点像元学习中的 support set 和 query set，但是在 θ 更新的时候是两者都用到了，而元学习是每次用一个
所以在这种情况下的更新就是先找一个 p，优化 θ，得到一个 ϕ，优化 ϕ

流程

Algorithm 1 Joint Learning of Model and Data Manipulation

Input: The target model $p_\theta(y|\boldsymbol{x})$
The data manipulation function $R_\phi(\boldsymbol{x}, y|\mathcal{D})$
Training set $\mathcal{D}$, validation set $\mathcal{D}^v$

1: Initialize model parameter $\boldsymbol{\theta}$ and manipulation parameter $\boldsymbol{\phi}$
2: **repeat**
3: Optimize $\boldsymbol{\theta}$ on $\mathcal{D}$ enriched with data manipulation through Eq.(7) $\boldsymbol{\theta}' = \arg\max_\theta \mathbb{E}_{p(\boldsymbol{x}) \exp\{R_\phi(\boldsymbol{x}, y|\mathcal{D})\}/Z} \left[\log p_\theta(y|\boldsymbol{x})\right]$.
4: Optimize $\boldsymbol{\phi}$ by maximizing data log-likelihood on $\mathcal{D}^v$ through Eq.(8) $\boldsymbol{\phi}' = \arg\max_\phi \mathbb{E}_{p(\boldsymbol{x}) \exp\{R_\phi(\boldsymbol{x}, y|\mathcal{D}^v)\}/Z} \left[\log p_{\theta'}(y|\boldsymbol{x})\right] = \arg\max_\phi \mathbb{E}_{(\boldsymbol{x}, y) \sim \mathcal{D}^v} \left[\log p_{\theta'}(y|\boldsymbol{x})\right]$,
5: **until** convergence

Output: Learned model $p_{\theta^*}(y|\boldsymbol{x})$ and manipulation $R_{\phi^*}(y, \boldsymbol{x}|\mathcal{D})$

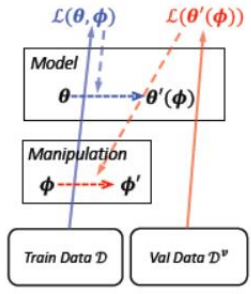

Figure 1: Algorithm Computation. Blue arrows denote learning model $\boldsymbol{\theta}$. Red arrows denote learning manipulation $\boldsymbol{\phi}$. Solid arrows denote forward pass. Dashed arrows denote backward pass and parameter updates.

实验 1

作者在两个方面进行了实验，一个是数据增强，一个是数据加权
在 NLP 领域的文本分类任务做的实验

	Model	SST-5 (40+2)	IMDB (40+5)	TREC (40+5)
	Base model: BERT [7]	33.32 ± 4.04	63.55 ± 5.35	88.25 ± 2.81
	Base model + val-data	35.86 ± 3.03	63.65 ± 3.32	88.42 ± 4.90
Augment	Synonym	32.45 ± 4.59	62.68 ± 3.94	88.26 ± 2.76
Augment	Fixed augmentation [49]	34.84 ± 2.76	63.65 ± 3.21	88.28 ± 4.50
Augment	**Ours**: Fine-tuned augmentation	$\mathbf{37.03 \pm 2.05}$	$\mathbf{65.62 \pm 3.32}$	$\mathbf{89.15 \pm 2.41}$
Weight	Ren et al. [39]	36.09 ± 2.26	63.01 ± 3.33	88.60 ± 2.85
Weight	**Ours**	$\mathbf{36.51 \pm 2.54}$	$\mathbf{64.78 \pm 2.72}$	$\mathbf{89.01 \pm 2.39}$

Table 1: Accuracy of Data Manipulation on Text Classification. All results are averaged over 15 runs ± one standard deviation. The numbers in parentheses next to the dataset names indicate the size of the datasets. For example, (40+2) denotes 40 training instances and 2 validation instances *per class*.

两种方法，数据增强与同义词替换和固定标签的两种方法进行对比，数据加权与每轮重新估计权重的一个方法进行对比

实验 1 方法

$$R_{\phi}^{aug}(x, y|\mathcal{D}) = \begin{cases} 1 & \text{if } x \sim_{\phi} (x|x^*, y), (x^*, y) \in D \\ -\infty & \text{otherwise} \end{cases} \tag{11}$$

这里 ϕ 其实就是 LM（Language Model）的参数，用作数据增广，只有当 y 为真实标签，且 x 是用 LM 中增广得到的句子时，才为 1，其他情况都为 $-\infty$，然后更新模型的 θ，再更新 ϕ

实验 2

第二个实验是在小样本任务上的

Model	20 : 1000	50 : 1000	100 : 1000
Base model: BERT [7]	54.91 ± 5.98	67.73 ± 9.20	75.04 ± 4.51
Base model + val-data	52.58 ± 4.58	55.90 ± 4.18	68.21 ± 5.28
Proportion	57.42 ± 7.91	71.14 ± 6.71	76.14 ± 5.8
Ren et al. [39]	74.61 ± 3.54	76.89 ± 5.07	80.73 ± 2.19
Ours	$\mathbf{75.08 \pm 4.98}$	$\mathbf{79.35 \pm 2.59}$	$\mathbf{81.82 \pm 1.88}$

Table 3: Accuracy of Data Weighting on Imbalanced SST-2. The first row shows the number of training examples in each of the two classes.

Model	20 : 1000	50 : 1000	100 : 1000
Base model: ResNet [14]	72.20 ± 4.70	81.65 ± 2.93	86.42 ± 3.15
Base model + val-data	64.66 ± 4.81	69.51 ± 2.90	79.38 ± 2.92
Proportion	72.29 ± 5.67	81.49 ± 3.83	84.26 ± 4.58
Ren et al. [39]	74.35 ± 6.37	82.25 ± 2.08	86.54 ± 2.69
Ours	$\mathbf{75.32 \pm 6.36}$	$\mathbf{83.11 \pm 2.08}$	$\mathbf{86.99 \pm 3.47}$

Table 4: Accuracy of Data Weighting on Imbalanced CIFAR10. The first row shows the number of training examples in each of the two classes.

主要解决的是类别不平衡的问题，一个是文本分类一个是图片分类任务，可以看到都达到了不错的效果

实验 2

$$R_\phi^w(x, y|\mathcal{D}) = \begin{cases} \phi_i & \text{if } (x, y) = (x_i^*, y_i^*), (x_i^*, y_i^*) \in \mathcal{D} \\ -\infty & \text{otherwise} \end{cases} \tag{12}$$

这里直接让 ϕ 为数据的权重，ϕ_i 就是第 i 条数据的权重[2]

[2]因为自己之前关于相关领域的内容了解并不深刻，这篇论文并未进行复现

Thanks!