论文《Controllable Unsupervised Text Attribute Transfer via Editing Entangled Latent Representation》阅读笔记

论文来源：2019 NIPS
论文主要贡献：提出了一种非监督方式的文本属性转换框架，通过改变文本属性分类器的属性来对latent representation【就是原始风格文本经过encoder部分后得到的表示】进行一些修改。
论文代码：https://github.com/Nrgeup/controllable-text-attribute-transfer

论文主要内容

作者说他们不同于传统的方法，将属性和内容表示分开进行建模，作者直接使用内容和属性缠绕在一起的表示。
文中的模型主要分为两部分，一个是基于transformer的AutoEncoder，一个是Attribute Classifier[属性分类器]。
- 作者首先将AutoEncoder和Attribute Classifier分开来训练
- 然后使用encoder部分去获得source sentence的隐层表示
- 再用FGIM算法去不断编译这个隐层表示，直到这个表示能够被分类器判定为target属性
- 最后在使用decoder从这个隐层表示获取target text.

文中提出的模型结构如下：

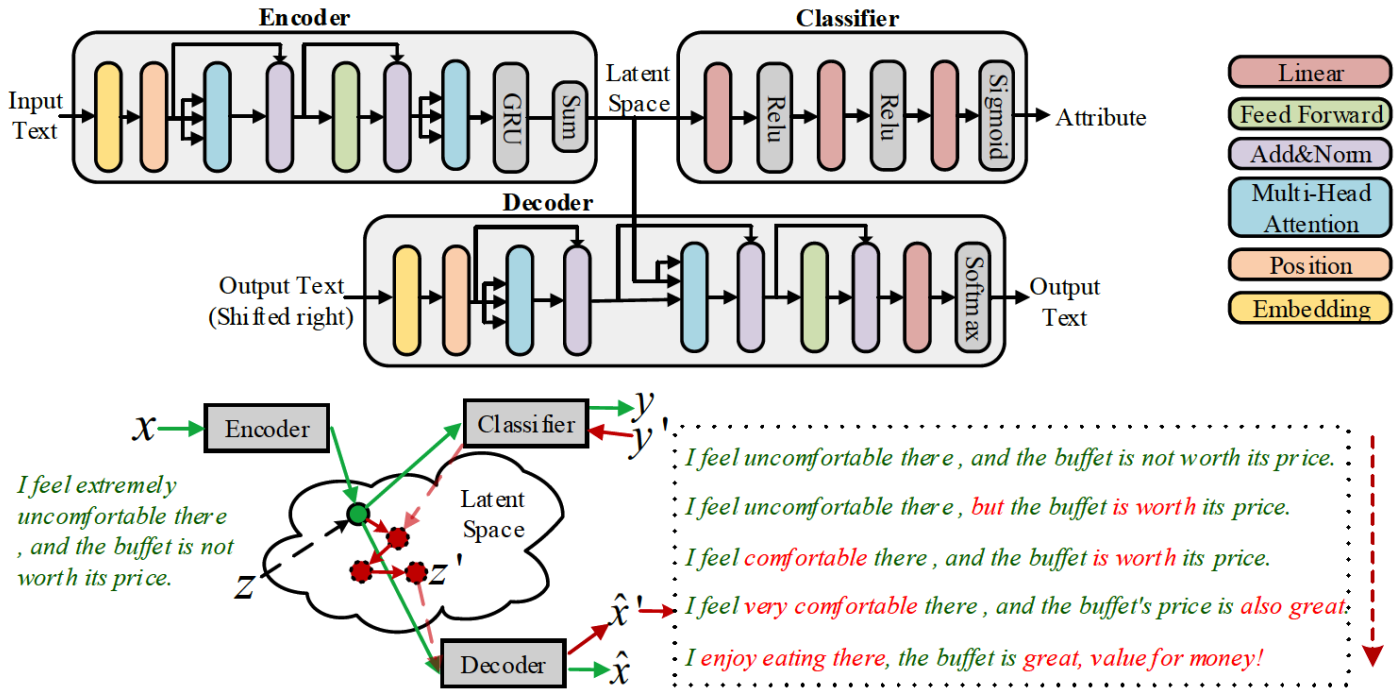

Figure 1: Model architecture.

如图所示，作者对encoder在encoder部分是两层的transformer结构，并在其后接了一层GRU。经过encoder后得到一个latent representation，但是目前的这个latent representation是source风格的，我们希望这个latent representation是target风格的，那么作者是怎么将这个latent representation由source风格转为target风格的呢？**作者提出了一个FGIM算法来修改这个latent representation.**

FGIM算法的伪代码如下：

Algorithm 1 Fast Gradient Iterative Modification Algorithm.

Input: Original latent representation $\boldsymbol{z}$; Well-trained attribute classifier C_{θ_c}; A set of weights $\boldsymbol{w} = \{w_i\}$; Decay coefficient λ; Target attribute $\boldsymbol{y'}$; Threshold t;
Output: An optimal modified latent representation $\boldsymbol{z'}$;

for each $w_i \in \boldsymbol{w}$ **do**
  $\boldsymbol{z}^* = \boldsymbol{z} - w_i \nabla_{\boldsymbol{z}} \mathcal{L}_c(C_{\theta_c}(\boldsymbol{z}), \boldsymbol{y'})$;
  for s-steps **do**
    if $|\boldsymbol{y'} - C_{\theta_c}(\boldsymbol{z}^*)| < t$ **then** $\boldsymbol{z'} = \boldsymbol{z}^*$; Break;
    end if
    $w_i = \lambda w_i$;
    $\boldsymbol{z}^* = \boldsymbol{z}^* - w_i \nabla_{\boldsymbol{z}^*} \mathcal{L}_c(C_{\theta_c}(\boldsymbol{z}^*), \boldsymbol{y'})$;
  end for
end for
return $\boldsymbol{z'}$;

FGIM算法的流程如下：

- 输入包括：
  - 原始的latent representation - z;
  - 一个训练好的Attribute Classifier - C_{θ_c};

    - 一个权重集合 - w = $\{w_i\}$,
    - 一个迭代衰减系数λ，取值为0~1，用来不断降低当前的权重w_i的值，每s步w_i就乘以一次λ；
    - 目标风格属性标签值y'
    - 阈值t，当Attribute Classifier对于当前的latent representation进行判断时，结果与y'的差值小于t就结束迭代；
- 具体步骤
    - 对于权重集合w中的每个权重w_i进行实验，这里的权重是从小到大进行实验的，作者说这样做可以避免我们的优化落入局部最优值；
    - **然后每次计算了损失函数后，都来更新隐层表示z,这里可以看做是本文的亮点，作者固定了Attribute Classifier的参数值，只改变输入z的值，使z来适应分类器**
    - 同时每一个权重w_i还会乘以一个系数λ，来不断降低其值
    - 直到Attriibuter Classifier将当前隐层向量z判定的风格值与y'的差值小于t

实验结果，作者在三个数据集上进行了实验：

Table 2: Automatic evaluation results. ↓ means the smaller the better. We underline the results of our model and bold the best results.

Methods	Yelp			Amazon			Captions		
	Acc	BLEU	PPL ↓	Acc	BLEU	PPL ↓	Acc	BLEU	PPL ↓
CrossAlign [28]	72.3%	9.1	50.8	70.3%	1.9	66.2	78.3%	1.8	69.8
MultiDec [5]	50.2%	14.5	84.5	67.3%	9.1	60.3	68.3%	6.6	60.2
StyleEmb [5]	10.2%	21.1	47.9	43.6%	15.1	60.1	56.2%	8.8	57.1
CycleRL [38]	53.6%	18.8	98.2	52.3%	14.4	183.2	45.2%	5.8	50.3
BackTrans [26]	93.4%	2.5	49.5	84.6%	1.5	48.3	78.3%	1.6	68.3
RuleBase [17]	80.3%	22.6	66.6	67.8%	33.6	52.1	85.3%	**19.2**	35.6
DelRetrGen [17]	88.8%	16.0	49.6	51.2%	29.3	55.4	90.4%	12.0	33.4
UnsupMT [41]	95.2%	22.8	53.9	84.2%	33.9	57.9	**95.5%**	12.7	31.2
Ours	**95.4%**	**24.6**	**46.2**	**85.3%**	**34.1**	**47.4**	92.3%	17.6	**23.7**