Learning with Noise: Enhance Distantly Supervised Relation Extractionwith Dynamic Transition Matrix

1 Motivation

- **显式化噪音**

这篇文章还是基于那两篇经典的递进模型, 即 PCNN+ATT.

但是再次之上, 本文作者为了减少 distant supervision (DS)带来的噪声, 使用神经网络来对噪音进行了描述, 这里就是和其他方案不同的地方, 之前的方案并没有尝试去显式化噪音.

> **注意 : 这里的神经网络的作用不是去消除噪音, 而是去近似噪音的模式**

- **训练中的迷惑**

  迷惑其实就是说, 我们在面对一个句子时, 依据以往的经验, 我们并不能指明说他属于哪个模式, 比如说:

  - 在之前的训练过程中, 训练过两个句子A和B.
  - A和B 中都有一部分相同的词汇 W , 但是 A和B 的所属关系(label) 是不同的.
  - 那么对于一个新的句子C, 若其中含有 W, 我们就会产生迷惑.
  - 接下来我们要做的是, 降低对于 W 词汇的信任度, 因为它是墙头草.

- **目的**

  通过这个神经网络可以:

  - 数值化某一个句子的迷惑度(用来描述<不能确定是哪种关系>的信息).
  - 指出噪音的模式.

2 How to quantify Noise?

上面说道是用神经网络来显式化 **噪声的模式**, 首先看应该用什么方法去表达 **噪声**.

用上面举的例子比较容易理解, 如果, 我们用的是朴素贝叶斯模型的话, 设定下面的场景:

- 如果含有关系 R1 的句子中出现过的词汇有 W1, W2, W3 , R1 可表示为 [1,1,1].
- 而另外的, 同样出现 W1 的句子所属关系是 R2. 那么 R2 可表示为 [1,0,0].
- 对于一个新的句子, 假设这个句子由 W1+W2+实体 组成.
- 按照朴素贝叶斯的观点, 这个句子有0.75的概率属于关系R1, 有0.25的概率属于关系R2.
- 正因为 W1 这样的词汇的出现, 才使得各种 relation 不是独立的, 而是相互关联的, 而这个就造成了噪声.

根据上面的描述, 我们发现需要三个变量才能描述一个句子中的迷惑度:

- 首先当然是句子向量S
- 其次是, 关于关系R1的向量, 和R2的向量.

因此我们知道了对于一个句子S, 我们需要一个矩阵 T 去描述关于他的迷惑度信息, 其中 T_{ij} 代表, 在句子S的真实关系为 i 的情况下, 被识别为 j 的条件概率. 那么, 很明显, 最佳的情况是T为单位矩阵,

接下来, 看论文是如何实现的:

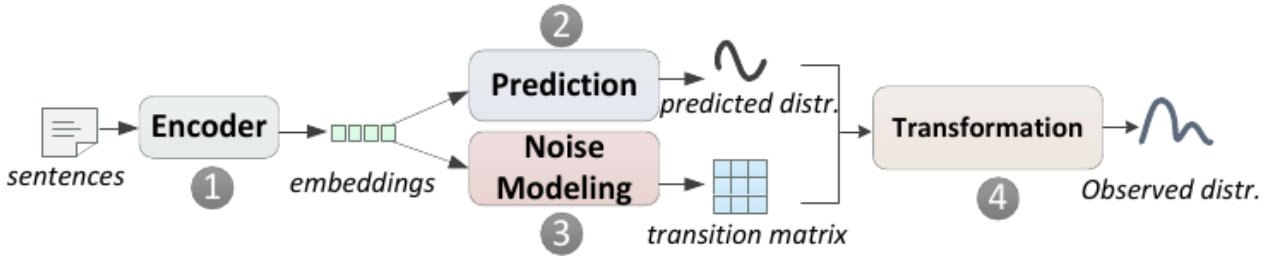
$$T_{ij} = \frac{exp(\mathbf{w}_{ij}^T \mathbf{x}_n + b)}{\sum_{j=1}^{|\mathbb{C}|} exp(\mathbf{w}_{ij}^T \mathbf{x}_n + b)}$$

其中, w_{ij} 是根据全局信息获得的关系i和关系j之间的相关信息. x_n 是句子的特征向量.

这个T矩阵就是一个每行的和都为1的矩阵.

3 How to incorporate noise in model?

模型如下:



- 上面的 1->2, 是之前介绍过的最基础的CNN方法, 一模一样. 其中1结束后得到的 embeddings 就是 x_n. 2 结束后得到的 predicted distr(ibution) 是 p .
- 重点在于3. 这里就是上一节中最后的那个公式, 得到的 transition matrix 为 T.
- 4 : 结合2和3

  $$\mathbf{o} = \mathbf{T}^T \cdot \mathbf{p}$$

  这里的 T 相当于是一个权重, 比如, $T_1 = [0.5, 0.3, 0.2]$, $p = [0.6, 0.1, 0.3]$

  那么, $o_1 = 0.39$, 而如果 $T_1 = [1, 0, 0]$, $o_1 = 0.6$,

  可以看出, T的确信值对最后的结果产生了很大的影响.

  这里, o需要在经过这一步计算后在进行归一化.

4. Incorporate with PCNN-ATT

第三节是为了过渡, 对每一个句子进行了编码. 虽说这里的研究已经可以去除噪声, 但是还是要从源头尽可能地减少噪声.

相比于上一节, 1(计算句向量)和3(计算过渡矩阵)的步骤发生了变化.

- **计算句向量**

$$\mathbf{s}_j = \sum_i^n a_{ij}\mathbf{x}_i;\ a_{ij} = \frac{exp(\mathbf{x}_i^T\mathbf{r}_j)}{\sum_{i'}^n exp(\mathbf{x}_{i'}^T\mathbf{r}_j)}$$

注意, 由于这里要用attention方法, 因此必须将目标relation的信息加入句向量, 这是与上面不同的地方.

- **计算过渡矩阵**

  由于上面的 s_j 已经含有了relation j 的信息, 因此按照第2节中的思想, 我们只需要三者(S, R1, R2)的融合. 那么这里就只需要增加另外的那个 relation 的信息即可:

$$T_{ij} = \frac{exp(\mathbf{s}_i^T \mathbf{r'}_j + b_i)}{\sum_{j=1}^{|\mathbb{C}|} exp(\mathbf{s}_i^T \mathbf{r'}_j + b_i)}$$

5. Forward

在前向计算我们只需要使用1->2得出的分布 p , 不使用 o.

6. Training

6.1 迹正则化

这里是为了能够使得 T 接近单位向量, 单位向量的特点就是迹最大.

另外, 由于是要最小化损失函数 L, 因此需要在 L 中减去 迹正则化项.

$$L = \sum_{i=1}^{N} -((1-\alpha)log(o_{iy_i}) + \alpha log(p_{iy_i})) \\ - \beta trace(\mathbf{T}^i)$$

上面的部分是 预测分布 o 和 p 的加权结合与真实分布 [0,0,...,1,...,0] 的交叉熵.

> 交叉熵公式: $\sum_i p(i) log(\frac{1}{q(i)})$, 其中, p(i) 是真实分布, q(i) 是预测分布.

y_i 是预测的 relation.

6.2 Curriculum Learning

即使已经精明到这个地步, 这个模型还是忘了很重要的一点, 即没有加入关系数据的先验, 如果办不到这一点的话, 事态可能会向奇怪的方向发展, 放在机器学习中就是陷入局部最小值.

这里加入关于数据先验的方法是使用 **课程学习**.

- 数据层的课程学习

  课程学习的思想, 先使用可靠的数据集进行训练, 在模型靠谱后, 再放入更多的数据.

  即, 一开始就可以依靠可靠数据在下面的目标函数下训练:

  $$L = \sum_{m=1}^{M} \sum_{i=1}^{N_m} -log(o_{mi,y_{mi}}) - \beta_m trace(\mathbf{T}^{mi})$$

  这里的 β_m 指的是针对不同的数据集要用不同的参数. 其值越大以为着我们对数据越信赖.

- 模型层的课程学习

  但是我们在这里并无法分离出我们可以确保完美的数据, 或者高质量的数据. 但是我们可以控制模型进行课程学习, 即从采取可靠的数据转化为使用可靠的模型. 及调整上面的权重 α

  反映在上面的式子就是, 一开始我们要 disable 引入误差的预测值对训练的影响. 因为误差的近似矩阵中含有大量参数, 这些随机的参数一开始说不定会把结果引向错误的方向. 之后再慢慢地提升 α 值.

这一节还有很多内容, 具体见论文4.2.

下面是效果:

Method	P@R_10	P@R_20	P@R_30
Mintz	39.88	28.55	16.81
MultiR	60.94	36.41	-
MIML	60.75	33.82	-
avg	58.04	51.25	42.45
avg_TM	58.56	52.35	43.59
att	61.51	56.36	**45.63**
att_TM	**67.24**	**57.61**	44.90

感觉比co-training的那一篇靠谱.

7. Intuition

个人感觉, 这篇文章的出发点虽说是对noise建模, 其实是用到了贝叶斯的思想.

T 代表的条件概率正是代表了, **参数遵循分布** 的思想. 虽然这里不是直接从分布采样取参数, 而是将参数的不确定性反映在了出来的结果上. 这样简化了这个采样分布的过程, 从而便于训练.

确实很棒!