Effective Inter-Clause Modeling for End-to-End Emotion-Cause Pair Extraction.

In Proc. of ACL 2020: The 58th Annual Meeting of the Association for Computational Linguistics, pages 3171—3181

摘要

ECPE（Emotion-cause pair extraction）任务：直接提取文档中所有潜在的情感-原因对。

ECPE 任务是由南京科技大学夏睿教授以及他的学生在 2019ACL 的一篇论文《Emotion-Cause Pair Extraction: A New Task》中首次提出，并给出了 pipeline 方式的解决方法，方法分为两步，具体如下：
步骤一，把 ECPE 任务通过多任务学习网络拆成两个独立的任务：情感抽取和原因抽取，目标是取出情感子句集和原因子句集。
步骤二，执行情感-原因匹配并且过滤没有因果关系的情感-原因对。
但由于 pipeline 方法存在错误传递等问题，而且两个步骤可能不能相互适应，于是本文提出了新的模型来解决 ECPE 任务。

本文提出了一种新的方法，从排名的角度（即在文档中对子句对的候选进行排名）着手处理 ECPE 任务，并提出了一步到位的神经网络方法，该方法注重子句间的建模以执行端到端抽取：
1. 使用图注意力网络对文档中子句之间的相互关系进行建模，以学习子句表示；
2. 通过基于内核的相对位置嵌入来增强子句对表示，以进行有效排名；
3. 实验结果表明，该方法明显优于当前的两步系统，特别是在一个文档中提取多个情感-原因对的情况下。

这是一个端到端的联合神经网络模型，能够直接提取情感-原因对。所以就不存在《Emotion-Cause Pair Extraction: A New Task》论文中 two-steps 方法的错误传递问题和不能直接抽取情感-原因对的问题。

目录

1 说明

近年来，情感原因分析已经在情感分析和文本挖掘领域获得持续关注。它的目标是是检测某种情绪原因或对文本中特定情绪表达的刺激。了解一种情感为什么会产生具有广泛的应用，如消费者评论挖掘和舆论监测。

之前的研究主要关注于 ECE 任务（emotion cause extraction），旨在于给定文本中的情感，然后提取相对应的原因子句。

南京科技大学夏睿教授团队指出，ECE 任务忽略了情感与原因之间的相互指示，而且对于情感的预先需求也限制了其应用范围。为了克服这种限制，他们提出了 ECPE 任务，旨在从给定文档提取所有情感表达子句及其原因。

例如：情感子句 c3 和其对应的原因子句 c2 构成情感-原因对(c3，c2)

He told us that since his illness (c1), his classmates and advisors have given him much help about the schoolwork (c2). He has been touched (c3), and said that he will repay them (c4).

相比于 ECE 任务，ECPE 任务更具有挑战性，因为我们需要对文档内容和结构有综合的理解，以实现情感原因子句共抽取、过滤无因果关系的情绪-原因子句对。

夏睿团队提出了两步式解决方案来提取情感-原因对。在第一步，使用多任务 LSTM 网络分别提取情感子句和原因子句。然后在第二步，使用二分类器从所有可能的情感-原因对中过滤出无因果关系的情感-原因对。尽管两步式解决方案已显示出其有效性，这种流水线式的系统对于情感原因对提取任务是次优的，因为它存在错误传递等问题，而且两个步骤可能不能相互适应。

连贯的文档具有潜在的结构，情感原因对的两个子句之间存在因果关系，从而将其与文档中其他非情感原因对（无因果关系）区别开来。因此，有关文档中子句间相互关系的知识对于提取潜在的情感原因对是有益的。进一步，根据话语的凝聚性和连贯性，两个遥远子句存在因果的概率相对较小。因此，子句对的两个子句之间的相对位置信息可以被视为情感原因对提取的有效特征。

基于以上两个考虑，在本文中，从排名的角度(即在给定文档中对子句对候选进行排名)解决情感原因对提取问题，并提出了一步到位的方法，该方法注重子句间的建模以执行端到端抽取。我们的方法首先通过利用图注意力来学习子句表示，建模子句间关系，并通过捕获两个子句之间的潜在关系来促进情感-原因对提取。然后，它学习子句对表示并将这些对排序以提取情感原因对。提出了一种基于内核的相对位置嵌入方案，以对相对位置之间的相互影响进行建模，并增强子句对表示以进行有效排名。将这两个组件集成到一个统一的神经网络中，并进行了端到端的优化。与之前的两步解决方案不同，我们的方法可以直接从文档中提取情感原因对。

本文工作主要贡献如下：

- 提出了第一个端到端的情感原因配对提取方法，一个从排名的角度来解决这一任务的统一模型；
- 通过集成子句间关系建模和基于内核的相对位置子句对排名增强来强调子句间建模；
- 实验结果表明，该方法明显优于当前性能最佳的系统，尤其是在一个文档中提取多对数据的情况下。

2 问题定义

ECPE 任务的定义：从文档中抽取情感-原因对

给定文档 $D = [c_1, c_2 \dots c_{|D|}]$，|D|是文章的长度，也是子句数；

$c_i = [w_1^i, w_2^i, \dots w_{|c_i|}^i]$，是第 i 句话的单词序列

我们的目标是得到 D 中的全部情感-原因对：

$$\mathcal{P} = \{(c^{\text{emo}_1}, c^{\text{cau}_1}), (c^{\text{emo}_2}, c^{\text{cau}_2}), \dots\}$$

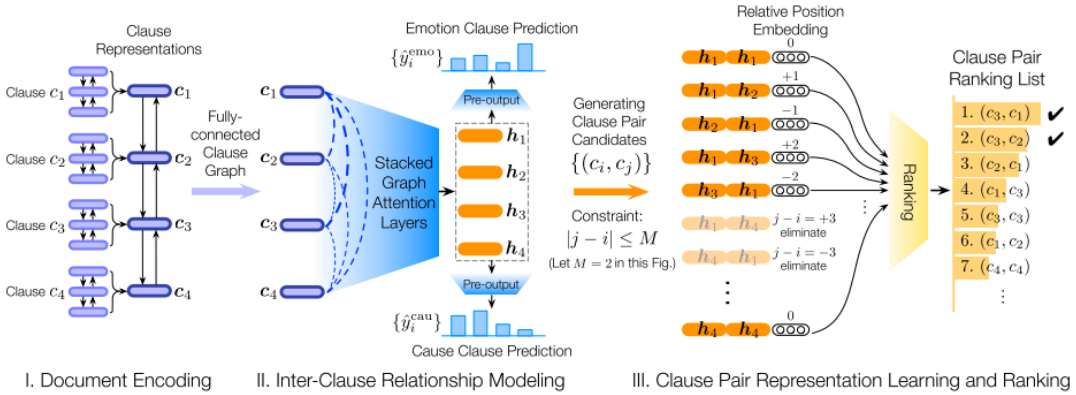
其中 (c_{emoj}, c_{cauj}) *是第 j 对；*

$c_{emoj} \in D$ *是一个情感子句，* $c_{cau\,j} \in D$ *是其对应的原因子句*

特别注意，一个情感子句可能对应多个原因子句；同样，一个原因子句可能对应多个情感子句。

3 方法

本文提出一步方法 RANKCP，对文档中的候选原因子句进行排序，进而抽取情感原因对。模型整体架构如下图所示：



RANKCP 模型有三个模块。
第一个模块进行词级别的建模和句子级别的建模，**得到子句的向量表示**；
第二个模块构建子句间的完全图（包括自己连接自己），使用图注意力网络对子句进一步建模，**得到更好的子句表示**，并且完成两个子任务——情感预测，原因预测。然后作者认为两

个相隔较远的句子构成情感-原因对的概率极小，所以|j-i|>M 的对直接不要；
第三个模块对情感-原因对的表示加入相对位置信息，得到对的最终表示：(子句 1，子句 2，距离向量)三者的拼接，然后进行排名。

3.1 Document Encoding

用分层 RNN 对文本内容进行编码并学习子句表示。(使用预训练的 BERT 效果会更好)

1. 对于每一个子句 $c_i = [w_1^i, w_2^i, \dots w_{|c_i|}^i]$，使用**词级 Bi-RNN** 来编码内容，获得子句的隐藏层表示 $(h_1^i, h_2^i, \dots h_{|c_i|}^i)$；
2. 使用注意力机制更新每个子句的隐藏层表示
   对于每一个子句 c_i 返回一个状态向量 h_i

   $$\boldsymbol{h}^i = \sum_{j=1}^{|c_i|} \alpha_j \boldsymbol{h}_j^i$$

   其中 α_j 是子句 c_i 中第 j 个词的 attention weight

   W_a, b_a , w_a 是多层感知机 MLP 的参数

   $$\alpha_j = \text{Softmax}\left(\boldsymbol{w}_a^\top \tanh(\boldsymbol{W}_a \boldsymbol{h}_j^i + \boldsymbol{b}_a)\right)$$

3. 将获得的文档的子句表示序列 $(h1, h2, \dots, h|D|)$ 输入到**子句级 Bi-RNN** 中
   产生子句表示 $(c1, c2, \dots, c|D|)$。

3.2 使用图注意力网络建模子句间关系

子句间的关系对于提取情感-原因对很有用，所以在得到一个文档的子句表示之后，采用图注意力网络来建模子句间的关系。把一个文档中的每个子句作为图的节点，所有节点都相互连接构成**全连接图(任意两个节点都由一条边连接)**。特别地，**节点自身也和自身相连**，因为存在一个情感句子对应的原因子句可能就是它自身。

图注意网络通过堆叠(stacking)多个图注意层来在子句之间传播信息，其中每一层将通过使用自注意(self-attention)聚合相邻子句的信息来学习更新的子句表示。

在第 t 个图注意力层，输入子句表示为：

$$\{\boldsymbol{h}_1^{(t-1)}, \boldsymbol{h}_2^{(t-1)}, \dots, \boldsymbol{h}_{|D|}^{(t-1)}\}$$

其中，子句 c_i 的表示为

$$\boldsymbol{h}_i^{(t-1)} \in \mathbb{R}^{d_{t-1}}$$

图注意机制通过以下聚合方案对文档中的每个子句进行操作：

$$\boldsymbol{h}_i^{(t)} = \mathsf{ReLU}\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{(t)} \boldsymbol{W}^{(t)} \boldsymbol{h}_j^{(t-1)} + \boldsymbol{b}^{(t)}\right)$$

其中，$h_i^{(t)}$是该注意力层的输出，$W^{(t)}, b^{(t)}$ 是可学习参数

N(i)表示子句c_i 的直接近邻（即文档中的全部子句）

$\alpha_{ij}^{(t)}$ 是子句c_i 和子句c_j 之间的注意力权重，反应了子句间的聚集度，通过一个多层感知机 MLP 进行学习，MLP 的参数为$w^{(t)}$，[;]是连接(concatenation)操作

$$e_{ij}^{(t)} = \boldsymbol{w}^{(t)\top} \tanh\left(\left[\boldsymbol{W}^{(t)}\boldsymbol{h}_i^{(t-1)}; \boldsymbol{W}^{(t)}\boldsymbol{h}_j^{(t-1)}\right]\right)$$

$$\alpha_{ij}^{(t)} = \frac{\exp\left(\mathsf{LeakyReLU}(e_{ij}^{(t)})\right)}{\sum_{k\in\mathcal{N}(i)} \exp\left(\mathsf{LeakyReLU}(e_{ik}^{(t)})\right)}$$

用矩阵表示公式:

$$\boldsymbol{H}^{(t)} = \mathsf{ReLU}\left(\boldsymbol{A}^{(t)}\boldsymbol{H}^{(t-1)}\boldsymbol{W}^{(t)\top} + \boldsymbol{b}^{(t)}\right)$$

其中:

$$[\boldsymbol{A}^{(t)}]_{ij} = \alpha_{ij}^{(t)}$$

第一层的输入即为前文 document encoder 的输出

$$\boldsymbol{H}^{(0)} = \left(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{|D|}\right)^\top$$

通过堆叠 T 个图注意力层来建模子句间关系，最终得到的更新的子句表示为

$$\boldsymbol{H}^{(T)} = \left(\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_{|D|}\right)^\top$$

若使用 multi-head attention 则可以捕获更多的特定信息，每一个 head 都可以捕捉一个基于图注意有序属性的全局模式。在实践中，在每两个相邻层之间添加 highway connection 以控制信息流。
因此，每个子句表示 h_i 通过自适应地融合其他子句的信息而产生，并且可以充分了解文档中的子句间关系。

此时得到的子句表示:

$$\{\boldsymbol{h}_i\}_{i=1}^{|D|}$$

两个预测任务：（emo，cau）

将该子句表示作为两个 pre-output 层的输入，来预测子句是否是情感/原因子句。特别地，一个多层感知机 MLP（参数w_{emo}, b_{emo}）及 sigmoid 函数用来预测子句c_i 为情感

子句的概率$\hat{y}_i^{emo}$。

$$\hat{y}_i^{\text{emo}} = \sigma\left(\boldsymbol{w}_{\text{emo}}^{\top}\boldsymbol{h}_i + b_{\text{emo}}\right)$$

同理。预测子句c_i为原因子句的概率 $\hat{y}_i^{cau}$，使用相同的公式。

3.3 子句对排序与基于核的相对位置嵌入

为了实现端对端抽取情感-原因对，需要学习子句对表示并对其排序以获得情感-原因对。而两个子句间的相对位置对抽取情感-原因对非常关键。因此，通过相对位置嵌入学习（relative position embedding learning)，将相对位置信息注入到子句对表示的学习过程中。

假设两个距离很远的句子是情感-原因对的概率极其小。基于此假设设置阈值 M，对于子句对（c_i, c_j）,当两个子句的相对距离 $|i-j|$ 大于 M 时，则舍弃该子句对。所以对于给定文档 D，候选子句对集合为：

$$\mathcal{P}' = \{(c_i, c_j) \mid -M \le j - i \le +M\}$$

学习子句对表示

对于每一个候选子句对，它的初始表示由三部分拼接而成：子句 c_i 的表示 h_i，

子句 c_j 的表示 h_j，相对位置的嵌入 r_{j-i} 。

$$p_{ij} = (c_i, c_j) \in \mathcal{P}'$$

使用一个单层 MLP 来学习候选子句对的表示：（参数w_p, b_p 可学习）

$$\boldsymbol{p}_{ij} = \text{ReLU}(\boldsymbol{W}_p[\boldsymbol{h}_i; \boldsymbol{h}_j; \boldsymbol{r}_{j-i}] + \boldsymbol{b}_p)$$

下面介绍如何建立相对位置嵌入。

Vanilla 相对位置嵌入

对每一个相对位置 $m \in \{-M, \ldots, -1, 0, +1, \ldots, +M\}$，通过采样均匀分布来随机初始化嵌入$r_m$。然后，每一个相对位置嵌入都随着模型训练一起学习。

基于核的相对位置嵌入

Vanilla 嵌入模式认为每个相对位置之间相互独立。为了获得更好的相对位置嵌入，可以认为每个相对位置之间也有关系，所以对不同相对位置之间的相互作用进行了建模。

对每一个相对位置 $m \in \{-M, \ldots, -1, 0, +1, \ldots, +M\}$，使用 RBF 核函数$Km(\cdot)$ 对相对位置 m 和其他相对位置之间的影响进行建模：

$$\text{K}_m(j) = \exp\left(-\frac{(j-m)^2}{\sigma_{\text{K}}{}^2}\right)$$

其中，$j \in \{-M, \ldots, -1, 0, +1, \ldots, +M\}$ 是可能的相对位置值；σ_k 限制了核函数的形状。

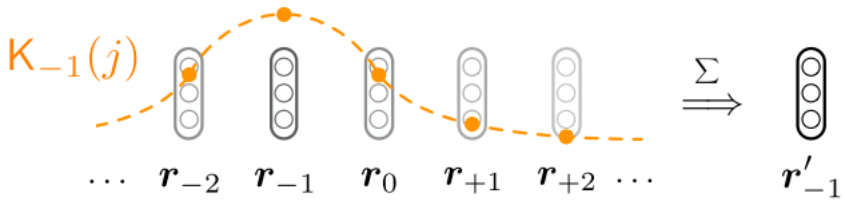
注：RBF：径向基核，也叫高斯核，是常用的一种核函数。公式为：

$$K_{||x-y||} = exp(-\frac{(x-y)^2}{2\sigma^2})$$

然后，通过聚集其他相对位置的影响来增强 vanilla 嵌入r_m ：

$$r'_m = \sum_{j=-M}^{+M} K_m(j) \cdot r_j$$

如果 j 更接近 m，那么 r_j 相较于其他远处的相对位置，会对r'_m 有更大的影响。
Eg. 当 m = -1 时，示例如下：



随着 $\sigma_K \to 0$，基于核的嵌入就退化为 vanilla 嵌入。所以基于核的嵌入模式可以看成是 vanilla 嵌入模式的正则化版本。

子句对排序

用一个带有激活函数的排序层，来给每个子句对产生排序得分$\hat{y}_{ij}$ ：

$$\hat{y}_{ij} = f_{\text{act}}\left(\boldsymbol{w}_r^\top \boldsymbol{p}_{ij} + b_r\right)$$

其中，参数w_r, b_r，激活函数$fact(\cdot)$，$p_{ij} \in P'$

3.4 Optimization

RANKCP 是端到端优化的。对于给定文档 D，损失函数包含以下两个部分：

第一部分：衡量子句对的排序得分

1. Pointwise ranking loss 交叉熵损失

$$\mathcal{L}_{\text{pair}} = \sum_{p_{ij} \in \mathcal{P}'} -(y_{ij} \log \hat{y}_{ij} + (1-y_{ij}) \log(1-\hat{y}_{ij}))$$

2. Pairwise ranking loss

$$\mathcal{L}_{\text{pair}} = \sum_{\{p^+, p^-\} \in \mathcal{P}'_{p^+ \succ p^-}} \max\{0, -(y^+ - y^-) + \gamma\}$$

正例得分减去负例得分，如果大于 margin 了，loss 就是 0，不更新了，反之 loss 不为 0

第二部分：衡量情感子句预测和原因子句预测
衡量 loss 函数还可以用来衡量前面图注意网络的情感子句预测和原因子句预测。
由子句对的基本事实可知一个子句对是否为情感/原因子句，使用两个交叉熵损失函 $\mathcal{L}_{emo}, \mathcal{L}_{cau}$ 监督情感子句预测和原因子句预测。

将以上两个部分的和作为给定文档 D 的损失函数 $\mathcal{L}$

$$\mathcal{L} = \mathcal{L}_{\text{pair}} + (\mathcal{L}_{\text{emo}} + \mathcal{L}_{\text{cau}})$$

这形成了对**子句表示学习**和**子句对排名**的两级监督。

3.5 基于情感极性词典（lexicon)的抽取

在测试阶段，关键问题是根据所有候选子句对的排序分数抽取潜在的情感-原因子句对。但是很难确定一个总体阈值，以对所有文档的候选子句对进行分类（情感-原因、非情感-原因)。因为如果我们设置一个分数阈值，高于这个分数的就是情感-原因子句对，低于的不是。这种方法显然不具有普适性，有的文档可能所有对得分都不高，有的文档可能所有对得分都很高。

针对这个问题，使用情感极性词典(lexicon)来辅助抽取。
对于一个测试文档，得到 top-N 子句对排序$\{p1, p2, \ldots, pN\}$
首先，抽取得分最高的$p1$，默认作为情感-原因对。因为给定的数据集中，文档里最少的也有一个情感-原因对；
然后，对于剩下的候选子句对 $pi = (c^{i,1}, c^{i,2}) \in \{p2, \ldots, pN\}$，如果子句 $c^{i,1}$ 中包含了情感极性词典中的词，那么我们就把子句对 pi 认定为情感-原因对。

因此，RANKCP 模型可以从一个给定文档中了，提取多个情感-原因对。

4 实验

4.1 实验设置

数据集与评估标准
选用的数据集参考了 2019 年夏睿教授团队的论文《Emotion-Cause Pair Extraction: A New Task》，该数据集是基于一个由新浪新闻 1,945 份中文文档组成的情感原因语料库。

语料库数据统计如下表:

# Doc. having one emotion-cause pair	1,746
# Doc. having two emotion-cause pairs	177
# Doc. having three or more emotion-cause pairs	22
Avg. of # Clause per document	14.77
Max. of # Clause per document	73

跟随前人的工作，采用 10-fold cross-validation 的数据分割。
衡量标准是 P，R，F1：

$$P = \frac{\#\text{correctly predicted pairs}}{\#\text{predicted pairs}},$$
$$R = \frac{\#\text{correctly predicted pairs}}{\#\text{ground-truth pairs}},$$
$$F_1 = \frac{2 \cdot P \cdot R}{P + R}.$$

此外，还分别评估了模型提取情感子句和提取原因子句的性能。将情绪-原因对分解为一组情绪子句和一组原因子句，然后分别评估这两组的性能。P，R，F1 的定义与上文的公式的类似（将公式中的 pairs 换为 emotion clauses 或者 cause clauses 即可）。

比较的方法

主要与 2019 年夏睿教授团队的论文《Emotion-Cause Pair Extraction: A New Task》中的方法进行对比。
夏睿教授团队论文中因流水线第一步的不同，分为三种方法：
INDEP：使用 BILSTM 对子句进行编码，后使用两个独立的 BILSTM 分别抽取情感子句和原因子句；
INTER-CE：不同于 INDEP，首先抽取原因子句，然后将其预测分布作为额外特征辅助抽取情感子句；
INTER-EC：与 INTER-EC 类似，不同在于使用情感子句辅助抽取原因子句；
三种方法的第二步均为使用二分类过滤无因果关系的子句对。

实现细节

为了对比的公平，使用和 INTER-EC 相同的词嵌入。

使用 LSTM 作为 RNN 单元，子句表示的维度为 200。使用两个图注意层，以搭建图注意网络，每层添加 rate = 0.1 的 dropout。相对位置的阈值 M 设为 12，相对位置嵌入的维度为 50。RBF 核函数中 $\sigma_k = 1$ 。

使用 Adam 优化器，学习率 0.001，4 个 mini-batch size，l_2 正则化系数设为 1e-5。使用 pointwise ranking loss，因为其训练速度比 pairwise loss 快。使用 ANTUSD 作为情感极性词典，超参数 N 设置为 3。

4.2 实验结果

ECPE 的结果

Approach	Emotion-Cause Pair Extraction			Emotion Clause Extraction			Cause Clause Extraction		
	F_1	P	R	F_1	P	R	F_1	P	R
INDEP	0.5818	0.6832	0.5082	0.8210	0.8375	0.8071	0.6205	0.6902	0.5673
INTER-CE	0.5901	0.6902	0.5135	0.8300	0.8494	0.8122	0.6151	0.6809	0.5634
INTER-EC	0.6128	0.6721	0.5705	0.8230	0.8364	0.8107	0.6507	0.7041	0.6083
RANKCP (top-1)	0.6562	**0.6910**	0.6254	0.8428	**0.8735**	0.8146	0.6790	**0.7130**	0.6468
RANKCP	**0.6610**	0.6698	**0.6546**	**0.8548**	0.8703	**0.8406**	**0.6824**	0.6927	**0.6743**

上表说明了情感原因子句对提取和两个子任务（情感子句提取和原因子句提取）的比较结果。一步方法 RANKCP 在所有三个任务上均比其他 baseline 具有明显优势。在三个任务上，F1 相对于性能最佳的 baseline（INTER-EC）分别提高了 4.82%，3.18%和 3.17%。

更具体地，观察到其优势主要来自召回率 R 的显著改善。与 INTER-EC 相比，RANKCP 在情感原因对提取和原因子句提取方面分别实现了 8.43%和 6.60%的改进，这表明 RANKCP 可以有效地抽取更正确的情感-原因子句对，而不会影响精度 P。

表中最后两行的比较展示了基于词典的提取的有效性。可以看到，添加基于词典的提取方案可以提高召回率 R，这表明它确实获得了更多正确的情绪原因对。尽管精度 P 略有下降，F1 仍然优于仅提取文档中的前 1 对。因此，基于词典的提取是有效的。

需要说明的是，RANKCP(top-1)的意思是不用情感极性词典，直接排名后取得分最高的子句对。可以看到，RANKCP(top-1)的 precision 值是最高的，这是因为排名后，只要排名没错误，那第一名肯定是情感-原因对。而 recall 值不如 RANKCP，说明它只取 top-1，这就导致很多其他的情感-原因子句对被丢弃。

抽取多个情感-原因对的结果

进一步比较了在一个文档中提取多个情感-原因子句对的结果。将每个 fold 的测试集分为两个子集：一个子集包含仅具有一对情感-原因子句对的文档，另一个子集包含具有两个或多个情感-原因子句对的文档。

比较结果如下表：

# Pairs	Approach	F_1	P	R
One per doc.	INTER-EC	0.6288	**0.6734**	0.5939
	RANKCP	**0.6780**	0.6625	**0.6966**
Two or more per doc.	INTER-EC	0.4206	0.5912	0.3302
	RANKCP	**0.5531**	**0.7508**	**0.4390**

可以看出，RANKCP 在两个子集上始终优于 INTER-EC。对于具有一对以上情感原因的文档，RANKCP 方法相对更有效（F1 改进超过 13%）。

ECE 的结果

Emotion Cause Extraction	F_1	P	R
RB	0.5243	0.6747	0.4287
MULTI-KERNEL	0.6752	0.6588	0.6927
CONVMS-MEMNET	0.6955	0.7076	0.6838
CANN	0.7266	0.7721	0.6891
RTHN	**0.7677**	0.7697	0.7662
Cause Clause Extraction	F_1	P	R
CANN – E	0.3797	0.4826	0.3160
RTHN-APE	0.5694	0.5800	0.5618
INTER-EC	0.6507	0.7041	0.6083
RANKCP	**0.6824**	0.6927	0.6743

表中上面部分，是标准的 ECE 任务；表中下面部分，则没有把标注好的情感子句作为输入，而是直接抽取原因子句。可以看到，RANKCP 模型取得了最好的效果，甚至相比有情感标注的 ECE 任务，也可以和 CONVMS-MEMNET 模型相提并论。
因此，RANKCP 模型受益于子句间建模，且在原因子句抽取上表现了其有效性。

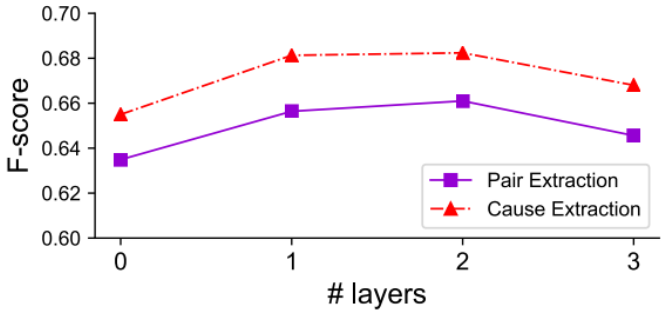
4.3 进一步探究

进行消融实验(ablation studies)，以证明 RANKCP 的每个部分均对提升 F1 值有帮助。

Two-level 监督的影响
RANKCP 模型是通过两级监督的混合来训练的：图注意力网络输出的子句表示学习的子任务监督 $\mathcal{L}_{emo} + \mathcal{L}_{cau}$，以及子句对表示学习和排名的主任务监督$\mathcal{L}_{pair}$。 为了验证对子任务监督的效果，对有无子任务（emo，cau）监督的效果进行了比较：

Loss Function	F_1	P	R
$\mathcal{L}_{\text{pair}}$	0.6241	0.6412	0.6090
$\mathcal{L}_{\text{pair}} + (\mathcal{L}_{\text{emo}} + \mathcal{L}_{\text{cau}})$	**0.6610**	**0.6698**	**0.6546**

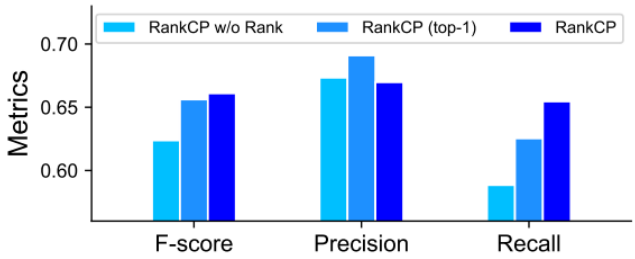
结果表明，采用两级监督进行训练可以提高提取性能。加入子任务（emo，cau）的监督有助于更好地学习子句表示，并最终促进子句对表示的学习和排名过程。
图注意力层的影响
用图注意力网络来建模子句间的潜在关系是模型的关键部分。将图注意力网络的图注意力层数设置为 0~3 分别进行了实验。实验结果如图：



结果表明：1.图注意力网络是有用的；2.有两个图注意力层时，效果最好，即两层能最好的建模子句间的关系。

子句对表示学习的影响

进一步研究是否可以通过直接使用子句表示来预测情绪子句和原因子句来获得理想的性能，即**删除了子句对表示学习和排名部分**(移除了 3.3 节的内容)，并利用图注意力网络的预测（3.2 内容）生成情感-原因子句对。在预测了文档中的情感子句和原因子句之后，将预测的情感与原因的所有组合都视为提取的情感-因果对，该变体模型与完整 RANKCP 模型的比较结果如下图所示。



完整 RANKCP 模型的表现要比变体模型好得多，尤其是在召回率 Recall 上。由此证明**子句对表示学习对模型性能具有积极作用。**

相对位置嵌入的影响

删除 RANKCP 中的相对位置嵌入部分以验证其效果，并比较了基于 vanilla 和基于内核的相对位置嵌入模式。

结果如下表所示：

Relative Position Scheme		F_1	P	R
No	(top-1 ext.)	0.6267	0.6600	0.5973
No	(lexicon-based ext.)	0.6260	0.6378	0.6160
Vanilla	(top-1 ext.)	0.6468	0.6810	0.6164
Vanilla	(lexicon-based ext.)	0.6582	0.6669	0.6510
Kernel	(top-1 ext.)	0.6562	**0.6910**	0.6254
Kernel	(lexicon-based ext.)	**0.6610**	0.6698	**0.6546**

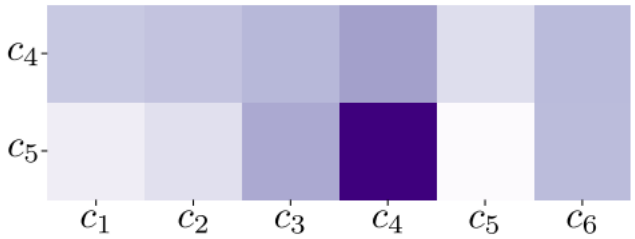
去除相对位置嵌入会导致性能下降，表明**子句对之间的相对位置确实对预测有用**。

表中前两行的另一个观察结果是，基于词典的提取不能胜过 top-1 提取，这进一步证明了**没有相对位置嵌入的模型不能提供理想的排名列表**。

在 top-1 和基于词典的提取中，基于内核的嵌入的性能均优于基于 vanilla 的嵌入，因此考虑**相对位置之间的相互影响有助于获得更强大的子句对表示**，并进一步提高情感-原因子句对抽取的性能。

4.4 案例分析

下面的例子 INTER-EC 处理会出现错误，但是 RANKCP 模型能够正确的处理。
4 月 11 日（c1），长沙网友洛丽塔在网上发帖吐槽（c2），她有一个极品男友（c3），如果要去的餐馆没有团购就要求换地方（c4），这让她感觉很不爽（c5），也很没面子（c6）。
对子句 c4 和子句 c5 的注意力权重进行了可视化，如下图所示：



情感子句 c5 对于对应的原因子句 c4 有最高的权重，说明图注意有效的捕捉了这两个子句间的关系。

5 相关工作

作者总结了 ECE 任务和 ECPE 任务的发展。
ECE：
2010 年，情感原因提取首次被研究，基于语言规则的系统来检测原因事件。早期的工作尝试基于规则、基于常识和基于传统机器学习的方法，以提取某些情绪表达原因。 2016 年，一种事件驱动的多内核 SVM 方法并发布了基准测试。基于特征的方法和神经方法。2019 年，采用了带有位置信息和集成全局预测嵌入的 Transformer 编码器，以提高性能。结合了情绪和位置调节器来限制参数学习。利用外部情感分类语料库对模型进行了预训练。
在其他研究领域，一些工作在具有多用户结构的微博中提取了情感原因。此外，将情感修饰为结构化现象，并研究了情感的语义作用，包括触发短语、体验者、目标和原因以及读者的感知。
ECPE：
以前所有关于情感原因分析的研究都需要采用已知的情感条子句作为模型输入。 2019 年南京科技大学夏睿教授团队首先提出了情感原因配对提取任务。他们提出了一种分两步的方法，分别提取情感和原因子句，然后训练分类器来过滤否定对。与他们的工作不同，我们的工作是通过有效的子句间建模来进行端到端情感原因对提取的一步式解决方案，从而显著提高了性能。

6 结论与未来工作

本文提出了第一个单步神经网络方法 RANKCP，以解决情感-原因对抽取的问题，该方法从排名的角度强调了子句间建模。RANKCP 方法有效地建模了子句间的关系，以学习子句的表示，并将**相对位置增强子句对排名**集成到一个统一的神经网络中，以端对端的方式提取情感-原因子句对。

在基准数据集上的实验结果表明，RANKCP 的性能明显优于以前的系统，并且进一步的分析验证了模型中每个组件的有效性。

未来工作：

1. 现在对情感原因的分析主要是 clause-level 的抽取，是粗粒度，设计出细粒度的 span-level 或者 phrase-level 是不错的方向；
2. 设计有效的模型来把合适的语言知识注入到神经模型中，对情感分析任务是很有价值的；
3. 研究情感的语义角色会很有趣，这需要考虑到情感表达的全面结构并且更具有挑战性。